Université de Montréal

Les logiciels libres sous l'angle de la responsabilité civile

par
Pierre-Paul Lemyre

Faculté de droit

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de maîtrise
en droit des technologies de l'information

novembre 2002

Université de Montréal
Faculté des études supérieures

Ce mémoire intitulé :
Les logiciels libres sous l'angle de la responsabilité civile

présenté par :
Pierre-Paul Lemyre

a été évalué par un jury composé des personnes suivantes :

Claude Fabien
président-rapporteur

Daniel Poulin
directeur de recherche

Karim Benyekhlef
codirecteur

Vincent Gautrais
membre du jury

Résumé

Distribués sous des licences permissives qui assurent des droits d'utilisation, de modification et de redistribution aux licenciés, l'élaboration des logiciels libres est fondée sur un modèle de développement décentralisé. Ces caractéristiques posent de nombreux défis au milieu juridique, particulièrement en ce qui a trait à la responsabilité civile. Ainsi, les développeurs se demandent dans quelles circonstances leur responsabilité civile peut être engagée suite à la défaillance de leur logiciel libre. De la même façon, ils questionnent la possibilité d'appliquer cette responsabilité à un nombre important de développeurs dispersés aux quatre coins du globe.

L'analyse présentée montre que le droit, tel qu'il existe actuellement, est en mesure de résoudre la majorité des problèmes relatifs à la détermination et l'application de la responsabilité civile en matière de logiciels libres. Les règles de la responsabilité civile représentent donc un risque potentiel pour les développeurs de logiciels libres, même s'ils sont relativement bien protégés par les contextes juridiques et factuels.

Mots clés : programme, informatique, développeur, programmeur, ouvert, faute, contractuelle, extra-contractuelle, dommage, causalité

Summary

Distributed under permissive licenses that guarantee the users use, modification and redistribution rights, the development of free software is decentralized. Numerous legal challenges flow from this, particularly respecting civil liability matthers. In consequence, developers are concerned regarding the circumstances in which they could be liable based on the faillure of their free software. They are equally questionning the possibility that numerous developers scattered around the world could be jointly liable.

The analysis show that the law, in its actual form, is able to answer the majority of the issues resulting from the establishment and the application of civil liability regarding free software. In this way, civil liability rules are representing a potential risk to free software developers, even if they are relatively well protected by the legal and factual contexts.

Key words : free, software, open, source, programmer, civil, liability, negligence, damage, causation

Table des matières

Liste des abréviations

ACM	Association for Computing Machinery
APSL	Apple Public Source License
ATICA	Agence pour les Technologies de l'information et de la Communication dans l'administration
BSD	Berkeley Software Distribution
Convention de Berne	Convention de Berne pour la protection des oeuvres littéraires et artistiques
Convention de Bruxelles	Convention de Bruxelles concernant la compétence judiciaire et l'exécution des décisions en matière civile et commerciale
Convention de La Haye	Avant-projet de convention sur la compétence et les jugements étrangers en matière civile et commerciale
CVS	Concurrent Versions System
DMCA	Digital Millennium Copyright Act
FSF	Free Software Foundation
GNU	Gnu's Not Unix
GPL	GNU General Public License
ICCP	Institute for Certification of Computer Professionals
IETF	Internet Engineering Task Force
ISO	International Standards Organization
LAMDI	Linux Anesthesia Modular Devices Interface
LGPL	Lesser General Public License
MPL	Mozilla Public licence
NAS	National Academy of Sciences
NCSA	National Center for Supercomputing Applications

Traité de l'OMPI	Traité de l'OMPI sur le droit d'auteur
UCC	Uniform Commercial Code
UCITA	Uniform Computer Information Transactions Act

Les logiciels libres sous l'angle de la responsabilité civile

Introduction

[1] À la fin de l'année 2000, *IBM* divulguait son intention d'investir un milliard de dollars américains dans le développement de *Linux* en 2001[1]. Lors de cette annonce, le président de l'entreprise aborda la question du marché des systèmes d'exploitation en affirmant que « The movement to standards-based computing is so inexorable, I believe *Sun* --and *EMC* and *Microsoft* for that matter-- is running the last big proprietary play we'll see in this industry for a good long while »[2].

[2] Ces propos peuvent étonner, dans la mesure où il émanent d'une multinationale ayant constitué une partie de sa fortune grâce à ses logiciels. Pourtant, *IBM* a tenu ses promesses et, aujourd'hui, l'entreprise est grandement impliquée dans le développement des logiciels libres. *IBM* est même devenu le plus grand employeur de programmeurs travaillant au développement de *GNU/Linux*[3]. L'entreprise a également adopté sa propre licence libre sous laquelle elle divulgue le code source de logiciels qui jusqu'ici était précieusement gardé secret[4]. Et cette nouvelle orientation porte déjà ses fruits puisque les ventes de super-ordinateurs *IBM* équipés de

[1] Jay LYMAN, « IBM To Pump $1 Billion Into Linux », (2000) *E-Commerce Times*, source : <http://www.ecommercetimes.com/perl/story/6027.html>.

[2] Louis GERSTNER, cité dans Joe WILCOX, « IBM to Spend $1 Billion on Linux in 2001 », (2000) *Cnet News*, source : <http://news.cnet.com/news/0-1003-200-4111945.html>.

[3] Pierre FRICKE, « Linux Strategies and Solutions : Linux Server Suppliers Contend for Leadership », (2002) *D.H. Brown Associates*, source : <http://www-1.ibm.com/linux/news/pb020405.pdf>.

[4] IBM, « IBM Public License Version 1.0 », (2001) *IBM* source : <http://oss.software.ibm.com/developerworks/opensource/license10.html>.

GNU/Linux ne cessent de croître, entre autres dans les domaines de l'exploration minière[5] et de la défense[6].

[3] *IBM* n'est pas la seule grande entreprise à percevoir le potentiel des logiciels libres. En fait, mis à part *Microsoft*, les principaux fournisseurs d'infrastructures informatiques ont tous commencé à adopter ce nouveau modèle de développement à plus ou moins grande échelle. Par exemple, la nouvelle compagnie formée par la fusion de *Hewlett-Packard* et *Compaq* supporte activement l'implémentation de *GNU/Linux* sur sa plate-forme *Itanium* ainsi que plusieurs projets libres visant à améliorer la gestion des imprimantes[7]. Même *Sun Microsystems* a fait passer la majeure partie de sa suite de traitement texte *Staroffice* sous une licence libre[8] et travaille actuellement à porter sa structure d'applications *Sun ONE* sous *GNU/Linux*[9].

[4] Toutefois, ces géants de l'informatique ne font que suivre l'exemple qui leur est donné depuis quelques années par un nombre toujours grandissant d'individus et d'entreprises. En effet, de plus en plus de programmeurs s'impliquent dans le développement de logiciels dont le code source est disponible gratuitement sur Internet. En adoptant une forme de développement distribuée, ceux-ci mettent leurs efforts en commun afin que chacun puisse bénéficier du travail des autres. Ils ont réalisé que la valeur d'un logiciel augmente considérablement lorsque tous ses utilisateurs ont la capacité de l'adapter à leur propres besoins et de participer à son évolution. C'est, entre autres, le modèle choisi par la compagnie *Red Hat,* qui distribue

[5] Stephen SHANKLAND, « IBM Drills Linux into Oil Industry », (2002) *CNET News*, source : <http://news.zdnet.co.uk/story/0,,t289-s2110816,00.html>.

[6] ASSOCIATED PRESS, « The Penguin Continues Its March », (2002) *Wired News*, source : <http://www.wired.com/news/linux/0,1411,52863,00.html>.

[7] HEWLETT-PACKARD, « Linux and HP », (2002) *HP*, source : <http://www.hp.com/united-states/linux/>.

[8] OPENOFFICE, « About Us: OpenOffice.org », (2002) *OpenOffice*, source : <http://www.openoffice.org/about.html>.

[9] P. FRICKE, *loc. cit.*, note 3.

gratuitement sa propre version de *GNU/Linux* tout en monnayant son expertise auprès des entreprises en fournissant des services de soutien et de dépannage.

[5] Aujourd'hui, le mouvement amorcé atteint même les administrations étatiques qui voient dans les logiciels libres une chance de réduire leurs dépenses tout en amoindrissant l'emprise de leurs prestataires de services. C'est notamment le cas en France où un rapport présenté au premier-ministre qualifient les logiciels libres d' « outils naturels pour les administrations »[10]. D'ailleurs, l'*Agence pour les Technologies de l'information et de la Communication dans l'administration* (*ATICA*) abonde dans ce sens en privilégiant l'utilisation des logiciels libres au sein de l'État[11]. La même tendance se perçoit aux quatre coins du monde puisque de nombreux gouvernements ont déjà adoptés des politiques allant dans ce sens[12].

[6] L'ampleur du phénomène ne laisse évidemment personne indifférent, particulièrement chez *Microsoft* où l'on craint la popularité croissante de l'idéologie du libre. En 1998, un mémorandum interne y notait déjà que les logiciels libres « pose[s] a direct, short-term revenue and platform threat to *Microsoft* »[13]. Depuis, les dirigeants de la multinationale font régulièrement des commentaires publics visant à miner la crédibilité de *GNU/Linux* et des logiciels libres. Ainsi, en 2001, le vice-président Jim Allchin déclarait que le législateur américain devait prendre conscience de la menace que représente

---

[10] Thierry CARCENAC, « Pour une administration électronique citoyenne », (2001) *Internet.gouv.fr*, source : <http://www.internet.gouv.fr/francais/textesref/rapcarcenac/sommaire.htm>.

[11] AGENCE POUR LES TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION DANS L'ADMINISTRATION, « L'approche en matière de logiciels et de logiciels libres », (2001) *ATICA*, source : <http://www.atica.pm.gouv.fr/bouquet-libre/ll_fr.pdf>.

[12] Eugene LIU, « Governments Embrace Open Source », (2002) *OsOpinion*, source : <http://www.osopinion.com/perl/story/18157.html>.

[13] Vinod VALLOPPILLIL, « Open Source Software A (New?) Development Methodology », (1998) *Open Source Initiative*, source : <http://www.opensource.org/halloween/halloween1.html>.

les logiciels libres pour l'innovation : « Open source is an intellectual-property destroyer.... I can't imagine something that could be worse than this for the software business and the intellectual-property business »[14].

[7] Malgré les craintes émises par *Microsoft*, la quantité de logiciels libres disponibles augmente actuellement à un rythme soutenu. En fait, bon nombre d'entre eux offrent aujourd'hui des solutions supérieures ou équivalente à celles proposées par les logiciels propriétaires de la même catégorie. Pour cette raison, de plus en plus d'entreprises et de programmeurs migrent vers les logiciels libres. Pourtant, cette transition ne s'effectue pas toujours à la vitesse souhaitée par leurs partisans. Ceci s'explique principalement par l'incertitude qui accompagne l'adoption de ce modèle de développement. En effet, certaines entreprises ont encore des doutes quant à la disponibilité d'un support technique et à la viabilité à long terme du mouvement[15].

[8] Cette incertitude se retrouve également sur le plan juridique car jusqu'à maintenant le milieu juridique n'a pas porté beaucoup d'attention au phénomène des logiciels libres. Pourtant, celui-ci soulève des questions légales d'importance. Ceci est particulièrement vrai en ce qui a trait à la responsabilité civile. Ainsi, alors que certains programmeurs se croient exempts de toute responsabilité, d'autres hésitent à s'investir dans le développement de logiciels libres par crainte d'être tenu responsables des dommages que ceux-ci pourraient éventuellement causer. Cette dernière réalité est particulièrement constatée par ceux qui oeuvrent dans le domaine médical, où une erreur de programmation peut entraîner la mort[16]. C'est notamment le cas du projet *Linux Anesthesia Modular Devices Interface*

---

[14] BLOOMBERG NEWS, « Microsoft Executive Says Linux Threatens Innovation », (2001) *News.com*, source : <http://news.cnet.com/investor/news/newsitem/0-9900-1028-4825719-RHAT.html?tag=ltnc>.

[15] Alan RADDING, « IT Managers Become More 'Open'-Minded », (2001) *InformationWeek*, source : <http://www.informationweek.com/833/opensource.htm>.

(*LAMDI*) qui vise à mettre au point un logiciel libre devant s'intégrer à de l'équipement d'anesthésie[17]. Bien que les enjeux ne soient pas toujours aussi dramatiques, la détermination des droits et obligations des développeurs de logiciels libres demeure importante pour les milliers d'individus et d'entreprises qui y sont déjà impliqués.

[9] Par conséquent, le premier objectif de cette étude est de déterminer dans quelle circonstances la responsabilité civile des développeurs peut être engagée suite à la défaillance d'un logiciel libre. Une partie de la réponse se trouve dans l'étude de la relation contractuelle qui prend forme lors de l'acceptation de la licence du logiciel. Quels sont les droits et obligations de chaque partie relativement à cette entente? Quelle est la validité des clauses d'exclusion de responsabilité qui s'y retrouvent? Est-ce que le droit de la protection des consommateurs s'y applique? Cependant, toutes ces questions s'avèrent non-pertinentes lorsque la victime est un tiers. Pour cette raison, une autre partie de la réponse doit être recherchée au niveau extra-contractuel. Quel sont les critères permettant de déterminer la faute ou la négligence des développeurs? Est-ce que ces derniers sont soumis aux règles de la responsabilité stricte du fabricant? Existe-t-il des textes législatifs assujettissant les développeurs à des normes spécifiques?

[10] Une fois cette première difficulté résolue, l'application de la responsabilité civile entraîne une problématique supplémentaire. Quels développeurs doivent répondre aux conséquences légales résultant de l'élaboration du logiciel libre? L'entreprise *SSH Communications Security* fait actuellement face à un problème de ce genre[18]. Possédant une marque de

---

[16] Andrew LEONARD, « Life or Death Software », (1999) *Salon*, source : <http://www.salon.com/tech/feature/1999/08/05/anesthesia/index.html>.

[17] Stefan HARMS, « Linux Anesthesia Modular Devices Interface », (2000) *LAMDI*, source : <http://gasnet.med.yale.edu/lamdi>.

[18] Evan LEIBOVITCH, « Open Source's Quiet Revenge », (2001) *ZDNet News*, source : <http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2688215,00.html>.

commerce sur le terme « SSH », ses dirigeants ont exigé que le logiciel *OpenSSH* change de nom. *OpenSSH* étant développé par un grand nombre de programmeurs ne maintenant aucune structure organisée formelle, *SSH Communications Security* pourrait avoir de la difficulté à obtenir ce qu'elle demande. Ce semble d'ailleurs être le cas puisque après plus d'un an, l'entreprise n'a toujours pas donné suite à sa lettre de mise en demeure.

[11] Aussi, le deuxième objectif de cette étude vise à déterminer de quelle façon la responsabilité civile peut être appliquée aux développeurs de logiciels libres. Pour y arriver, les questions relatives à leur *contraignabilité* sont abordées. Est-ce que les développeurs peuvent être identifiés adéquatement? Comment doit-on réagir face à leur dispersion à l'échelle internationale? La principale difficulté réside dans l'analyse des critères permettant d'établir un lien causal entre les agissements d'un développeur et les dommages subis. Parmi les nombreuses personnes impliquées, lesquelles ont légalement causé les pertes?

[12] Compte tenu du contexte global dans lequel s'inscrit le développement des logiciels libres, une analyse internationale s'avère nécessaire pour atteindre ces objectifs. Aussi, cette étude aborde tout autant le point de vue du droit civil que celui de la common law. Toutefois, le droit américain occupant une position prédominante en matières technologiques, celui-ci fait l'objet d'une attention particulière.

[13] Par ailleurs, avant d'aborder l'analyse juridique des questions de responsabilité civile qui se rattachent aux logiciels libres, il est nécessaire de comprendre ce que sont réellement ces derniers. Pour cette raison, il convient de débuter par la présentation de quelques notions fondamentales à leur sujet.

PARTIE I - Les logiciels libres dans un contexte légal

Chapitre 1 - Un aperçu des logiciels libres

[14] Malgré la vague de popularité que connaît l'univers du logiciel libre depuis quelques années, de nombreuses personnes n'ont toujours pas conscience des différences fondamentales qui distinguent les logiciels libres des autres types de logiciels disponibles sur le marché. Ceci s'explique probablement par l'existence d'une vaste gamme de statuts juridiques distincts en matière de logiciel. De plus, pour compliquer la chose, certains développeurs de logiciels désireux de profiter de l'étiquette favorable dont jouissent les logiciels libres « jouent sur des ambiguïtés pour rendre difficile la distinction entre libre et non libre »[19]. Dans ces circonstances, il est nécessaire d'établir sans équivoque ce qu'est un logiciel libre.

Section 1 - L'historique

[15] L'histoire des logiciels libres remonte aux origines de l'informatique moderne. Au cours des années 60, puisqu'il n'y avait aucune compatibilité entre les ordinateurs des différents fabricants, « [...] sofware was not bought and sold ; it was given away with the expensive hardware, which was useless without it »[20]. À cette époque, le code source était généralement fourni avec les logiciels afin que les acheteurs puissent les adapter à leurs besoins particuliers. Dans ce contexte, les programmeurs percevaient les logiciels comme des biens collectifs auxquels chacun pouvait apporter ses propres

---

[19] Benjamin DRIEU, « Licences, du copyright au copyleft », (2000) *APRIL*, source : <http://www.april.org/articles/divers/licences.html>.

[20] Donald K. ROSENBERG, « Copyleft and the Religious Wars of the 21st Century », (1998) *Stromian Technologies*, source : <http://www.stromian.com/copyleft.htm>.

améliorations. C'est ce qui permet à certains d'affirmer : « Originally, all software was free »[21].

[16] Cette période, considérée comme l'âge d'or du logiciel libre, s'étendit jusqu'au début des années 1970. Par la suite, plusieurs facteurs contribuèrent à l'apparition des logiciels propriétaires. Tout d'abord, l'augmentation de la demande pour le développement de nouveaux logiciels entraîna la naissance de l'industrie du logiciel telle que nous la connaissons aujourd'hui. Par la suite, *AT&T* s'appropria le système d'exploitation *UNIX*, dont le code source était auparavant accessible[22]. Finalement, le gouvernement américain décida de retirer son support pour les standards informatiques ouverts[23]. À ce moment, les logiciels libres étaient déjà en voie de disparition.

[17] Comme l'explique Richard Stallman :

> « In 1984, it was impossible to use a modern computer without installing a proprietary operating system, which you would have to obtain under a restrictive license. No one was allowed to share software freely with fellow computer users, and hardly anyone could change software to fit his or her own needs. »[24]

[18] C'est cet état de fait qui amena Richard Stallman à constater les effets négatifs de la propriété des logiciels sur leur développement. Il se rend compte qu'en cachant les codes sources d'un logiciel, il devient impossible

---

[21] Robert J. CHASSELL, « Software Freedom : Rights, Duty, Metaphor, and Making a Living », (2002) *Rons.net.cn*, source : <http://www.rons.net.cn/english/FSM/bob>.

[22] Eric S. RAYMOND, « A Brief History of Hackerdom », dans Chris DIBONA, Sam OCKMAN et Mark STONE (dir.), *Open Sources : Voices from the Open Source Revolution*, O'Reilly, 1999, source : <http://www.oreilly.com/catalog/opensources/book/raymond.html>.

[23] Nathan NEWMAN, « The Origins and Future of Open Source Software », (1999) *Net Action*, source : <http://www.netaction.org/opensrc/future/oss-whole.html>.

[24] Richard M. STALLMAN, « 15 Years of Free Software », (1999) *LinuxToday*, source : <http://linuxtoday.com/news_story.php3?ltsn=1999-03-17-003-10-NW-LF>.

pour l'utilisateur de l'adapter à ses propres besoins, et par conséquent d'améliorer le produit. Il fonde donc, en 1983, le projet *GNU* (Gnu's Not Unix) sous l'égide de la *Free Software Foundation* (*FSF*) afin de développer un système d'exploitation complètement libre.

[19] Au début des années 90, le projet *GNU* avait rassemblé tous les éléments essentiels d'un système d'exploitation sauf le kernel, noyau du système. Au même moment, un étudiant finlandais nommé Linus Torvalds, entreprenait le développement de Linux en utilisant les outils distribués par la *FSF*[25]. En peu de temps, *GNU/Linux* devint une plate-forme raisonnablement stable et des distributions commerciales se mirent à apparaître (*Debian*, *Red Hat*, *Suse*, etc.). Un système d'exploitation libre était finalement accessible au grand public.

[20] Cependant, *GNU/Linux* n'était pas le seul système d'exploitation libre destiné à voir le jour. Au cours de la même période, la distribution *Net/2* de *BSD Unix* prenait forme. Ne contenant pas de code informatique soumis au droit d'auteur d'*AT&T*, *Net/2* donna également naissance à de nombreuses distributions (*NetBSD*, *FreeBSD*, *OpenBSD*)[26].

[21] De façon parallèle, le développement d'Internet contribua grandement au renouveau des logiciels libres. Tout d'abord parce le réseau Internet repose lui-même sur des protocoles et des logiciels non-propriétaires dont les codes informatiques sont disponibles, tel que TCP/IP, *BIND* ou *Sendmail*. Ensuite parce qu'Internet facilite les communications entre les développeurs, étape essentielle pour l'élaboration et la distribution des logiciels libres. Il n'en fallait pas plus pour amorcer le retour en force des logiciels libres.

---

[25] Nicholai BEZROUKOV, « A Slightly Skeptical View on Linus Torvalds », (2000) *Softpanorama*, source: <http://www.softpanorama.org/People/Torvalds/index.shtml>.

[26] BSDTODAY, « A Brief History of BSD », (2000) *BSDToday*, source : <http://www.bsdtoday.com/tmp-ba8274c5/bsd-history.html>.

Section 2 - La définition

[22] Comme son nom l'indique la notion de logiciel libre implique l'existence d'un certain degré de liberté. Celle-ci fait référence aux droits qui sont accordés à l'utilisateur du logiciel. Toutefois, ce qui paraît simple aux francophones ne l'est pas nécessairement pour les anglophones. En anglais, le terme *free* signifie à la fois liberté et gratuité. Or, tous les logiciels libres ne sont pas nécessairement gratuits. Par exemple, il existe de nombreuses distributions commerciales de *GNU/Linux*. Une certaine ambiguïté résulte donc de ce choix terminologique[27].

[23] Cela mis à part, un logiciel sera considéré libre si ses utilisateurs bénéficient de quatre libertés principales :

- La liberté d'exécuter le programme informatique pour n'importe quel usage ;
- La liberté d'étudier le fonctionnement du programme informatique et de l'adapter à leurs propres besoins ;
- La liberté de redistribuer des copies du logiciel ;
- La liberté d'améliorer le logiciel et de distribuer les copies modifiées[28].

[24] L'existence de ces libertés implique premièrement qu'il ne doit pas être nécessaire de demander ou de payer pour obtenir l'autorisation de les exercer. Incidemment, le coût du logiciel ne doit pas représenter une compensation pour les droits d'auteur, contrairement à la pratique bien

---

[27] FREE SOFTWARE FOUNDATION, Pourquoi « Free Software » est-il meilleur que 'Open Source' », (2001) *FSF*, source : <http://www.fsf.org/philosophy/free-software-for-freedom.fr.html>.

[28] FREE SOFTWARE FOUNDATION, « Qu'est-ce qu'un Logiciel Libre? », (2001) *FSF*, source : <http://www.fsf.org/philosophy/free-sw.fr.html>.

établie en matière de logiciels propriétaires. Seuls les frais de reproduction et de distribution pourront donc être réclamés dans le montant exigé en contre-partie du logiciel.

[25] Deuxièmement, il est essentiel que le code source soit disponible pour que l'utilisateur soit en mesure d'en étudier le fonctionnement et de le modifier. Celui-ci doit être joint au logiciel ou disponible sur demande. L'accès au code source constitue la caractéristique fondamentale des logiciels libres.

[26] Troisièmement, pour que ces libertés soit significatives, elles doivent être irrévocables. Aussi, si le donneur de licence se réserve la possibilité de révoquer unilatéralement les droits concédés, le logiciel ne peut être considéré comme libre.

[27] L'utilisateur d'un logiciel libre est donc investi de pouvoirs importants par rapport à l'utilisateur d'un logiciel propriétaire. Corrélativement, la tradition veut que l'exercice de ces pouvoirs s'accompagnent du devoir de faire bénéficier la communauté de tout travail utile ayant été effectué. Ainsi, les utilisateurs sont incités à s'impliquer activement dans le développement du logiciel. Ceci peut concerner, par exemple, une modification du code source, la rédaction de documentation, la mise en place de ressources de développement, etc. Ces droits et obligations sont importants car ils influent nécessairement sur l'établissement de la responsabilité civile.

Section 3 - Les logiciels libres et les logiciels *open source*

[28] En 1998, une rupture s'est opérée au sein de la communauté du libre[29]. Certains programmeurs, principalement sous l'influence d'Éric Raymond et de Bruce Perens, se sont distancés de l'idéologie de la *FSF*, qu'ils jugent mal adaptée à la réalité économique d'aujourd'hui. Selon eux, l'expression logiciel libre n'est pas susceptible d'inciter l'industrie du logiciel à adopter leur modèle de développement parce qu'elle implique la prédominance de notions éthiques et morales[30]. Ils adoptèrent donc une nouvelle stratégie fondée sur la notion de logiciels *open source*.

[29] Tout comme pour la définition des logiciels libres, celle des logiciels *open source* protège les droits d'utilisation, de redistribution, la disponibilité du code source et le droit de le modifier[31]. En fait, les logiciels libres et les logiciels *open source* sont théoriquement identiques. Seule la terminologie à été modifiée afin de mettre l'emphase sur la disponibilité du code source plutôt que sur la liberté de l'utilisateur. Aussi, la distinction est surtout philosophique, le modèle ouvert mettant l'accent sur le pragmatisme alors que le modèle libre repose sur l'éthique[32].

[30] Toutefois, d'un point de vue pratique, certaines différences sont apparues entre les deux mouvements. Celles-ci concernent principalement les entreprises qui offrent des logiciels en dévoilant leur code source tout en restreignant les autres droits essentiels de l'utilisateur. Ces pratiques ont

---

[29] OPENSOURCE.ORG, « History of the OSI », (1999) *Opensource*, source : <http://www.opensource.org/docs/history.html>.

[30] Eben MOGLEN, « Free Software Matters: Free Software or Open Source? », (2000) *Moglen*, source : <http://moglen.law.columbia.edu/publications/lu-07.html>.

[31] OPENSOURCE.ORG, « The Open Source Definition », (2001) *Opensource*, source : <http://www.opensource.org/docs/definition.html>.

[32] Brett WATSON, « Philosophies of Free Software and Intellectual Property », (1999) *RAM*, source : <http://www.ram.org/ramblings/philosophy/fmp/free-software-philosophy.html>.

parfois été acceptées par les promoteurs des logiciels *open source* alors qu'elles sont définitivement rejetées par les partisans des logiciels libres. Par exemple, la *Apple Public Source License* (*APSL*) retenue par *Apple* pour son système d'exploitation *Mac OS X*, oblige les licenciés à publier toute version modifiée du programme informatique qu'ils ont déployée[33]. Cette limitation de la liberté de modifier le logiciel satisfait les critères de la définition des logiciels *open source*, mais non ceux de la définition des logiciels libres[34].

[31] Sur le plan juridique, il existe très peu de différences entre les logiciels libres et les logiciels *open source*. Pour s'en convaincre, il suffit de constater que les deux mouvements reposent sur les mêmes fondements et la même méthode de développement. Aussi, bien que le champ de ce mémoire soit limité aux logiciels libres, les conclusions qui en sont tirées devraient généralement trouver application aux logiciels *open source*, sous réserve du contenu de leurs licences respectives.

Section 4 - Les autres types de logiciels

[32] Mis à part les logiciels libres, il existe plusieurs autres types de logiciels possédant des statuts juridiques distincts. Ainsi, la vaste majorité des logiciels disponibles aujourd'hui sont propriétaires, c'est-à-dire que leur licence restreint considérablement leur utilisation, leur redistribution ou la possibilité de les modifier[35]. Pour assurer l'efficacité de ces restrictions, ceux-ci sont distribués sous forme d'exécutables binaires incompréhensibles pour l'être humain. Par ailleurs, les logiciels propriétaires ont recours à différentes méthodes de distribution sur Internet. Par exemple, la redistribution des

---

[33] APPLE COMPUTER, « Apple Public Source License », (2001) *Apple*, art. 2.2 (c), source : <http://www.opensource.apple.com/apsl>.

[34] FREE SOFTWARE FOUNDATION, « Les problèmes de la licence d'Apple », (2001) *FSF*, source : <http://www.fsf.org/philosophy/apsl.fr.html>.

[35] FREE SOFTWARE FOUNDATION, « Catégories de logiciels libres et non libres », (2001) *FSF*, source : <http://www.fsf.org/philosophy/categories.fr.html>.

gratuiciels, ou *freewares*, est généralement permise car ils sont distribués gratuitement dès l'origine[36]. Toutefois, leur utilisation reste limitée et leur modification interdite. De la même façon, les partagiciels, ou *sharewares*, peuvent être redistribués et utilisés gratuitement pendant une certaine période de temps. Cependant, à l'expiration de ce délai, le licencié doit payer le donneur de licence pour continuer à utiliser le logiciel[37].

[33] D'autres logiciels sont dit néo-libres ou semi-libres. Dans certains cas, il s'agit de logiciels dont la licence accorde des droits d'utilisation, de redistribution et de modification mais qui en limite l'exercice aux usages sans but lucratif[38]. Cette façon de procéder permet d'éviter que des tiers réalisent des bénéfices en utilisant ou distribuant un logiciel libre sans que le titulaire des droits d'auteur puisse en bénéficier. Parfois, il s'agit de logiciels dont la licence autorise l'intégration de code informatique libre et non-libre[39]. Grâce à cette technique, il devient possible de dévoiler le code source de certains éléments du programme informatique sans pour autant être obligé de dévoiler la totalité du code source d'un ensemble plus vaste. Ces solutions étant dérivées des logiciels libres, le cadre juridique de ces derniers leur est applicable, dans la mesure où il est compatible avec les termes spécifiques de chaque licence.

[34] Finalement, certains logiciels appartiennent au domaine public, ce qui signifie que leurs développeurs abandonnent toutes les protections qui leur

[36] PCS, « What's Freeware? », (2002) *Why-not*, source : <http://www.why-not.com/software/ware.htm>.

[37] PCS, « What's Shareware? », (2002) *Why-not*, source : <http://www.why-not.com/software/ware.htm>.

[38] FREE SOFTWARE FOUNDATION, *loc. cit.*, note 35.

[39] Eric DI FILIPPO, « Les logiciels libres », (1999) *Juriscom*, p. 57, source : <http://www.juriscom.net/uni/mem/10/log02.rtf>.

sont accordées par le droit d'auteur[40]. Un logiciel de ce genre est nécessairement libre, puisque aucune restriction ne peut y être attachée.

* * *

[35] C'est donc l'amplitude des droits accordés aux utilisateurs qui distinguent les logiciels libres, *open source*, du domaine public, et même semi-libres, des logiciels propriétaires traditionnels. C'est l'existence de ces droits qui a permis l'apparition de la méthode de développement unique qui bouleverse actuellement l'industrie du logiciel. Celle-ci ayant fait ses preuves, bon nombre d'entreprises tentent aujourd'hui d'en tirer profit. Cette façon particulière de concevoir les logiciels a nécessairement une influence sur l'application de la responsabilité civile.

Chapitre 2 - Un modèle de développement distribué

[36] Traditionnellement, les logiciels résultent des efforts d'un groupe restreint de programmeurs travaillant ensemble à l'intérieur de la hiérarchie d'une organisation. Au contraire, « The structure of work and communication in the hacker community is decentralized and distributed »[41]. En effet, l'évolution de la majorité des projets libres repose sur la contribution volontaire des utilisateurs du logiciel. Ceux-ci, tout en étant dispersés sur la surface du globe, s'y impliquent en fonction de leur expertise et de leurs besoins respectifs. Pour cette raison, le fonctionnement de ce modèle se rapproche beaucoup de celui de la recherche scientifique[42].

---

[40] Jean-Paul SMETS-SOLANES et Benoît FAUCON, *Logiciels Libres : Liberté, Egalite, Business, Freepatents*, Edispher, 1999, source : <http://www.freepatents.org/liberty/droit.html>.

[41] Eric S. RAYMOND, cité dans William C. TAYLOR, « Inspired by Work », (1999) 29 *Fastcompany* 200, source : <http://www.fastcompany.com/online/29/inspired.html>.

[42] Nicolai BEZROUKOV, « Open Source Software Development as a Special Type of Academic Research (Critique of Vulgar Raymondism) », (1999) 10 *First Monday*, source :

Section 1 - La méthodologie employée

[37] L'article *The Cathedral and the Bazaar*[43] d'Eric Raymond contient certainement la description la plus connue de cette méthode de développement. La communauté du libre y est assimilée à un bazar où les marchands interagissent en public sans être soumis à une structure organisationnelle. À l'opposé, le développement propriétaire traditionnel y est comparé à une cathédrale où les tâches sont effectuées dans un milieu clos et hiérarchisé.

[38] Selon Eric Raymond, certains principes doivent être respectés pour qu'un projet fonctionne à l'intérieur de ce bazar. Tout d'abord, pour qu'une communauté se crée autour d'un logiciel, il essentiel que celui-ci offre un minimum de potentiel dès le départ. « The implementation can be crude or incomplete, but it must convince others of its potential »[44]. Les promoteurs du projet doivent également faire preuve de flexibilité en reconnaissant les bonnes idées provenant des autres et en changeant d'approche lorsque les besoins de la communauté le requièrent. Un autre principe directeur consiste à publier le code source du logiciel dès le début de son développement et de continuer à le faire paraître fréquemment par la suite. Cette technique réduit les dédoublements d'efforts, récompense les contributeurs et les stimulent à poursuivre la mise au point du logiciel. Finalement, le recours à la communauté pour examiner le code source est l'élément le plus important de la stratégie : « Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone »[45].

<http://www.firstmonday.org/issues/issue4_10/bezroukov/index.html>.

[43] Eric S. RAYMOND, « The Cathedral and the Bazaar », (1998) 3 *First Monday*, source : <http://www.firstmonday.org/issues/issue3_3/raymond/index.html>.

[44] Kim JOHNSON, « Open-Source Software Development », (1999) *Johnson*, source : <http://www.cpsc.ucalgary.ca/~johnsonk/SENG/SENG691/open.htm>.

[45] E. S. RAYMOND, *loc. cit.*, note 43.

[39] Le modèle du bazar ne colle cependant pas exactement à la réalité. Tous les projets de logiciels libres possèdent un minimum d'organisation.[46] Le plus souvent, celle-ci est assurée par le promoteur du projet qui sert de guide à l'évolution du programme informatique, met en place des procédures et insuffle de la motivation aux développeurs. De plus, une structure est nécessaire afin de résoudre les conflits qui peuvent survenir lorsque des solutions contradictoires sont proposées. Dans tous les cas, les principes d'efficacité mis de l'avant par les partisans des logiciels libres sont universels et peuvent même être appliqués au sein d'une entreprise traditionnelle[47]. D'ailleurs, une étude récente démontre qu'un petit nombre de promoteurs contribuent souvent à l'élaboration de la plus grande partie du code des logiciels libres, les nombreux développeurs occasionnels ne fournissant qu'une aide complémentaire. Par conséquent : « Free Software development is less a bazaar of several developers involved in several projects [and] more a collation of projects developped single-mindeddly by a large number of authors »[48].

[40] Il n'en demeure pas moins que la composition d'un logiciel libre est souvent complexe, car les contributions des différents développeurs s'y entremêlent. Certaines d'entre elles ont pour but d'ajouter de nouvelles fonctionnalités alors que d'autres visent à régler des bogues préexistants. Même pour un projet de petite taille, tel que le lecteur audio *XMMS*, les interactions entre les contributions sont nombreuses[49]. Ceci est d'autant plus vrai lorsqu'un projet entier est intégré à l'intérieur d'un autre. La coexistence

---

[46] Charles CONNELL, « Open Source Projects Manage Themselves? Dream On », (2000) *Lotus*, source : <http://www.lotus.com/developers/devbase.nsf/articles/doc2000091200>.

[47] Jonathan EUNICE, « Beyond the Cathedral, Beyond the Bazaar », (1998) *Illuminata*, source : <http://www.illuminata.com/public/content/cathedral/intro.htm>.

[48] Rishab GOSH et Vipul Ved PRAKASH, « The Orbiten Free Software Survey », (2000) 5 *First Monday*, source : <http://www.firstmonday.dk/issues/issue5_7/ghosh/index.html>.

[49] XMMS, « Changelog », (2002) *XMMS*, source : <http://www.xmms.org/ChangeLog>.

simultanée de plusieurs versions du logiciel, officielle ou non, accentue encore cet état de fait.

Section 2 - Les développeurs impliqués

[41] L'élaboration de la majorité des logiciels libres repose grandement sur le travail volontaire d'individus. Ceux-ci agissent parfois de façon isolée, parfois par le biais d'institutions telles que la *FSF*. Certains ont un impact direct sur l'évolution du logiciel en modifiant son programme informatique ou en rédigeant sa documentation. D'autres, bien qu'ils ne peuvent être considérés comme des développeurs, participent au projet de manière indirecte en rapportant les problèmes éprouvés ou en maintenant un site Web sur le sujet.

[42] De nombreuse motivations incitent les programmeurs à s'impliquer activement dans le développement d'un logiciel libre. Le plaisir de résoudre des problèmes informatiques intéressants est définitivement l'une d'elle. Comme le souligne Linus Torvald, « most of the good programmers do programming not because they expect to get paid or get adulation by the public, but because it is fun to program »[50]. Leurs efforts rapportent également des récompenses immédiates plus concrètes. Tout d'abord, leurs contributions permettent d'apporter des solutions aux difficultés qu'ils ont eux-mêmes éprouvées avec le logiciel[51]. Ensuite, cela leur fournit une expérience nouvelle qui contribue à améliorer leurs compétences et leurs talents[52]. Enfin, l'une des motivations essentielles des développeurs de logiciels libres a trait au prestige et la réputation qu'ils obtiennent à l'intérieur de leur

---

[50] Linus TORVALDS cité dans FIRST MONDAY, « FM Interview with Linus Torvalds: What Motivates free Software Developers? », (1998) 3 *First Monday*, source : <http://www.firstmonday.org/issues/issue3_3/torvalds/index.html>.

[51] Josh LERNER et Jean TIROLE, « The Simple Economics of Open Source », (2000) *National Bureau of Economic Research*, source : <http://www.nber.org/papers/w7600>.

[52] INTERNATIONAL INSTITUTE OF INFONOMICS, « Free/Libre and Open Source Software: Survey and Study », (2002) *FLOSS*, source : <http://floss1.infonomics.nl/stats.php?id=31>.

communauté[53]. La valeur de cette réputation se matérialise dans la reconnaissance accordée par leurs pairs ainsi que dans les opportunités de carrières et autres bénéfices économiques qui en découlent. Par exemple, Linus Torvald s'est vu offrir des actions lors des offres publiques d'achat de certaines compagnies dont le modèle économique repose sur *Gnu/Linux*[54]. Aussi, bien qu'Eric Raymond compare l'économie du logiciel libre a une « 'gift culture' in which members compete for status by giving things away »[55], les développeurs ne semblent pas agir de façon totalement désintéressée.

[43] D'ailleurs, ce n'est pas un hasard si l'entreprise privée prend une place grandissante dans l'évolution des logiciels libres. Cette contribution prend parfois la forme d'améliorations à un logiciel libre préexistant. Par exemple, la plupart des entreprises offrant une distribution de *GNU/Linux* investissent dans le développement des logiciels de fenêtrage *Gnome* ou *KDE*. Parfois encore, les entreprises initient des projets libres en remplaçant la licence propriétaire d'un logiciel par une licence libre. C'est ce qu'a fait *Netscape* en 1998 avec son navigateur Web[56]. En agissant de la sorte, les entreprises peuvent répondre plus rapidement aux besoins de leurs propres clients, résoudre un problème spécifique pour un coût moindre ou tout simplement améliorer la performance de l'un de leurs logiciels. Cette implication de l'entreprise démontre que des motivations économiques soutiennent le développement des logiciels libres. Seulement, la valeur recherchée « is not measured in terms of its price, but rathers in [the free software] use value as a tool »[57].

---

[53] Nicholas THOMPSON, « Reboot! », (2000) 3 *The Washington Monthly*, source : <http://www.washingtonmonthly.com/features/2000/0003.thompson.html>.

[54] J. LERNER et J. TIROLE, loc. cit., note 51, note 9.

[55] Eric S. RAYMOND, « The Magic Cauldron », (1999) *Raymond*, source: <http://www.tuxedo.org/~esr/writings/magic-cauldron/magic-cauldron.html>.

[56] NETSCAPE, « Netscape Announces Plans to Make Next-Generation Communicator Source Code Avalaible Free on the Net », (1998) *Netscape*, source : <http:/wp.netscape.com/newsref/pr/newsrelease558.html>.

Section 3 - Deux exemples : *Linux* et *Apache*

[44] Le développement du kernel de *Linux* est sans aucun doute le projet libre le plus connu aujourd'hui. *Linux* est le noyau d'un système d'exploitation de type *UNIX* complet et compatible avec de multiples plate-formes qui compte aujourd'hui environ 2,5 millions de lignes de code[58]. Linus Torvalds en fut l'instigateur en 1991, alors qu'il décida d'apporter des améliorations au système d'exploitation *Minix*. Utilisant Internet à son plein potentiel, Torvald réussit à réunir des centaines de développeurs autour de *Linux*. Avec la croissance du projet, il délégua la supervision de certaines sections du programme informatique à d'autres programmeurs. Le développement de *Linux* est donc devenu très modulaire. Malgré tout, « eventually all changes still go through Torvalds and he has the final [word] on what goes into the kernel and what stays out »[59].

[45] *Apache* est un autre logiciel libre célèbre puisqu'il est le serveur Web le plus utilisé depuis de nombreuses années. Ce projet fut fondé en 1995, par un groupe de webmestres réagissant à l'arrêt du développement du serveur Web du *National Center for Supercomputing Applications* (*NCSA*). Aujourd'hui, l'évolution du logiciel *Apache* est entre les mains d'un groupe composé d'une vingtaines de personnes qui ont la responsabilité de voter les modifications proposées par l'importante communauté de développeurs. Pour accéder au groupe, ces derniers doivent contribuer activement au projet pendant une certaine période de temps : « The *Apache* Group is a

---

[57] Patrick K. BOBKO, « Open-source Software and the Demise of Copyright », (2001) 27 *Rutgers Computer & Tech. L. J.* 51, 84.

[58] David A. WHEELER, « More Than a Gigabuck: Estimating GNU/Linux's Size », (2002) *Dwheeler*, source : <http://www.dwheeler.com/sloc/redhat71-v1/redhat71sloc.html>.

[59] Davor CUBRANIC, « Open-Source Software Development », (1999) *Cubranic*, source : <http://sern.ucalgary.ca/~maurer/ICSE99WS/Submissions/Cubranic/Cubranic.html>.

meritocracy – the more work you have done, the more you are allowed to do »[60].

* * *

[46] Ainsi, alors que les logiciels propriétaires ne proviennent généralement que d'une seule source, les logiciels libres sont formés par le conglomérat des efforts de différents développeurs, qu'ils soient individuels ou institutionnels. Cette décentralisation du développement est la caractéristique qui pose le plus de défis au droit de la responsabilité civile. Or, la viabilité de cette méthode de développement particulière repose en grande partie sur des notions juridiques, soit celles du droit d'auteur et des licences d'utilisation.

Chapitre 3 - Les assises légales des logiciels libres

[47] Qu'ils soient libres ou propriétaires, distribués gratuitement ou contre une rémunération, la vaste majorité des logiciels sont accompagnés d'une licence d'utilisation. Le recours à ces licences découle de l'assujettissement des logiciels au régime de la propriété intellectuelle, et particulièrement à celui des droits d'auteur. Pourtant, la nature des licences d'utilisation des logiciels varie énormément : les licences propriétaires tendent à limiter les droits concédés alors que les licences libres visent à les étendre. Au surplus, toutes les licences libres ne sont pas équivalentes. Toujours est-il que les droits et obligations qui découlent de ces conventions ont un impact important sur la responsabilité civile des développeurs de logiciels.

---

[60] THE APACHE GROUP, « F.A.Q. », (2002) *Apache*, source : <http://www.apache.org/foundation/faq.html>.

Section 1 - Les logiciels et les droits d'auteur

[48] Au niveau international, les droits d'auteur sont régi par la *Convention de Berne pour la protection des oeuvres littéraires et artistiques*[61] (*Convention de Berne*). Ce document établit un « régime minimal de protection des oeuvres littéraires et artistiques »[62] qui assure la conservation des intérêts de l'auteur. Celui-ci se voit accorder des droits sur la forme de sa création, mais non sur l'idée qui en est à l'origine. Il obtient ainsi le droit de contrôler, entre autres choses, la reproduction, la transformation et la distribution de son oeuvre. Ces droits ont l'avantage d'être transférables pour la plupart. Bien que le traitement des infractions aux droits d'auteur demeure de compétence nationale, la *Convention de Berne* permet d'étendre les droits de l'auteur au territoire de l'ensemble des 139 États qui en sont membres. Enfin, elle établit que l'existence des droits d'auteur ne nécessite aucune formalité, c'est-à-dire qu'ils existent dès la création de l'oeuvre.

[49] Aujourd'hui, il ne fait plus aucun doute que les logiciels bénéficient du régime des droits d'auteur. D'ailleurs, la plupart des législations nationales sur le sujet contiennent des clauses visant à inclure spécifiquement les programmes d'ordinateur[63]. Le droit international se dirige également dans cette direction dans le cadre de l'élaboration du *Traité de l'OMPI sur le droit d'auteur (Traité de l'OMPI)*[64]. En conséquence, le code source et la version binaire d'un logiciel jouissent de la même protection que les oeuvres littéraires. Cependant, tous les composants d'un logiciel ne sont pas protégés avec la même force. Ainsi, les tribunaux américains ont précisé à

---

[61] *Convention de Berne pour la protection des oeuvres littéraires et artistiques*, Juris International, source : <http://www.jurisint.org/pub/01/fr/147.htm>.

[62] Pierre TRUDEL et autres, *Droit du cyberespace*, Montréal, Les Éditions Thémis, 1997, p. 16-3.

[63] Au Canada, *Loi sur le droit d'auteur*, L.R.C. (1985), c. C-42, art. 2 ; en France, *Code de la propriété intellectuelle*, art. L. 112-2 ; aux États-Unis, *Copyright Act*, 17 U.S.C., s. 101.

[64] *Traité de l'OMPI sur le droit d'auteur*, Juris International, art. 4, source : <http://www.jurisint.org/pub/01/fr/149.htm>.

plusieurs reprises que les interfaces ne bénéficient parfois d'aucune protection[65].

[50] Dans ce contexte, la licence est un contrat par lequel le titulaire des droits d'un logiciel autorise un tiers à poser des gestes qui autrement les enfreindraient. En ce qui concerne le licencié, la pratique commerciale traditionnelle lui accorde seulement un droit d'utilisation, auquel se greffe de nombreuses restrictions[66]. Il lui est donc interdit de modifier, de copier ou de redistribuer la copie du logiciel qui lui est remise. Les licences d'utilisation comportent généralement de nombreuses autres limitations spécifiques. Elles ont trait, par exemple, aux garanties accordées, aux questions de responsabilité, au choix de la juridiction, etc.

[51] Évidemment, seul le véritable titulaire des droits sur le logiciel est autorisé à agir en tant que donneur de licence. Bien qu'il puisse s'agir de l'auteur du logiciel lui-même, ce n'est pas toujours le cas. Ainsi, lorsque le logiciel résulte du travail d'un employé agissant dans le cadre de son emploi, la loi attribue parfois la titularité des droits d'auteur à l'employeur[67]. Les contrats de travail contiennent souvent une clause ayant le même effet. De plus, certaines institutions, telle que la *FSF*, incitent les développeurs de logiciels libres à leur céder la titularité des droits afin d'en assurer la protection.

---

[65] *Lotus Development Corp.* c. *Borland International, Inc.*, 49 F.3d 807 (1st Cir. 1995), 819 ; *Computer Associates International* c. *Altai Inc.*, 982 F.2d 693, 703-712.

[66] William H. NEUKOM et Robert W. GOMULKIEWICZ, « Licensing Rights to Computer Software », (1993) 354 *PLI/Pat* 775.

[67] Au Canada, Loi sur le droit d'auteur, précitée, note 63, art. 13 (3) ; aux États-Unis, *Copyright Act*, précité, note 63, s. 201.

Section 2 - Les licences libres *copyleftées*

[52] Le terme *copyleft* fut mis de l'avant par la *FSF* et s'oppose au *copyright*, équivalent anglophone du droit d'auteur. L'objectif du *copyleft* est d'utiliser les restrictions imposées par les lois sur le droit d'auteur afin de garantir la liberté des logiciels[68].

[53] Les licences *copyleftées* accordent tout d'abord un niveau élevé de liberté au licencié en lui permettant d'utiliser, de modifier et de redistribuer le logiciel. Toutefois, leur particularité consiste à garantir les mêmes libertés aux licenciés de tous les logiciels qui sont dérivés du logiciel original[69]. Ceci se réalise par l'inclusion d'une clause empêchant le licencié qui désire redistribuer une version modifiée du logiciel de le faire sous une autre licence. Par conséquent, l'inclusion de code informatique soumis à une licence *copyleftée* à l'intérieur d'un logiciel propriétaire est interdite.

[54] Évidemment, cette façon de procéder favorise la réutilisation du logiciel par des tiers. Or, une fois que le code a été modifié et redistribué à de multiple reprises, le licencié se trouve à contracter avec de nombreux donneurs de licence distincts[70]. En effet, les développeurs possédant tous des droits autonomes sur leur apport respectif au logiciel, un engagement contractuel est obtenu de chacun d'eux lors de l'acceptation de la licence. De la même façon, le licencié est redevable envers chacun des titulaires de droits dans le logiciel. Il est vrai que certaines juridictions reconnaissent l'existence d'oeuvres collectives[71]. Dans ce cas, il est possible de prétendre que l'initiateur du projet est le seul et unique donneur de licence, sous

[68] Shawn W. POTTER, « Opening Up to Open Source », (2000) 6 *Rich. J. L. & Tech.* 24, par. 58.

[69] FREE SOFTWARE FOUNDATION, « Qu'est-ce que le copyleft? », (2001) *FSF*, source : <http://www.fsf.org/copyleft/copyleft.fr.html>.

[70] Daniel B. RAVICHER, « Facilitating Collaborative Software Development: The Enforceability of Mass-Market Public Software Licenses », (2000) 5 *Va. J. L. & Tech* 11, par. 84.

[71] En France, *Code de la propriété intellectuelle*, précité, note 63, art. L113-2 al 3.

prétexte qu'il est celui qui divulgue l'oeuvre[72]. Cependant, cette interprétation paraît douteuse dans la mesure où il est possible d'investir chacun des auteurs d'un droit distinct, que l'initiateur du projet peut très bien cesser de divulguer le logiciel sans que cela ne stoppe son développement, et que celui-ci n'a aucune influence déterminante sur les autres contributeurs.

[55] La licence *copyleftée* la plus utilisée est la *GNU General Public License* (*GPL*)[73]. Celle-ci régit aujourd'hui la plus grande partie des logiciels libres, dont le système d'exploitation *GNU/Linux*. Elle accorde au licencié les différents droits promus par la *FSF* à condition que celui-ci accepte :

- De ne pas établir de droit de propriété dans le logiciel ;
- De fournir le code source à tout ceux qui se procurent le code binaire ;
- De placer une indication à l'intérieur du logiciel stipulant que la *GPL* s'y applique ;
- De reconnaître que le logiciel est fourni sans garantie ;
- De licencier toute version dérivée sous les même conditions[74].

[56] Toutefois, la *GPL* n'est pas l'outil le plus approprié dans toutes les situations. C'est le cas, par exemple, lorsque l'on tente de l'appliquer à une librairie destinée à être partagée entre différents logiciels. En effet, il est possible d'interpréter la *GPL* de façon à laisser croire que tout logiciel utilisant cette librairie doit nécessairement être distribué sous la même licence. Or, si

---

[72] Jean-Baptiste SOUFRON, « La licence publique générale : un système original de protection juridique pour les créations issues des sytèmes de développement coopératifs », (2002) *Droit & Nouvelles Technologies*, source : <http://www.droit-technologie.org/2_1.asp?dossier_id=79>.

[73] FREE SOFTWARE FOUNDATION, « GNU General Public License version 2 », voir annexe A.

[74] Ira V. HEFFAN, « Copyleft : Licensing Collaborative Works in the Digital Age », (1997) 49 *Stan. L. R.* 1487, 1508.

cela s'avère exact, très peu de développeurs de logiciels propriétaires seraient enclins à utiliser des librairies *copyleftées*[75]. Pour résoudre ce problème, la *FSF* conçue une licence plus souple, la *GNU Lesser General Public License* (*LGPL*)[76].

Section 3 - Les licences libres *non-copyleftées*

[57] Certaines licences libres n'ont pas recours au mécanisme du *copyleft*. Leur statut juridique se rapproche plutôt de celui des logiciels du domaine public. Pour cette raison, certains auteurs y font référence en tant que « licences de type domaine public »[77]. Ceci s'explique par l'étendue des droits que ces licences concèdent. En effet, elles autorisent généralement l'utilisation, la modification et la redistribution du code source, sans imposer d'autres restrictions. Cependant, les logiciels qui leur sont soumis ne font pas partie du domaine public puisque celles-ci précisent que le titulaire des droits d'auteur conserve ses droits. D'ailleurs, elles contiennent souvent l'obligation de divulguer l'existence de ces droits à l'intérieur des versions modifiées du logiciel.

[58] Les licences libres *non-copyleftées*, contrairement à celles qui le sont, n'interdisent pas d'inclure le code source qui leur est soumis à l'intérieur d'un logiciel propriétaire. Tout licencié peut donc modifier le logiciel et le redistribuer sous la licence de son choix. Encore une fois, de nombreux développeurs sont donc susceptibles d'avoir contribué au logiciel et, par le fait même, d'être partie au contrat de licence.

---

[75] Dennis M. KENNEDY, « A Primer on Open Source Licensing Legal Issues : Copyright, Copyleft and Copyfuture », (2001) 20 *St. Louis U. Pub. L. Rev.* 345, 362.

[76] FREE SOFTWARE FOUNDATION, « GNU Lesser General Public License version 2.1 », (1999) *FSF*, source : <http://www.fsf.org/copyleft/lesser.html>.

[77] J.-P. SMETS-SOLANES et B. FAUCON, *loc. cit.*, note 40.

[59] Parmi toutes les licences libres *non-copyleftées*, la plus connue est sans aucun doute la licence *Berkeley Software Distribution* (*BSD*)[78]. Celle-ci est utilisée, entre autres, par les différentes variantes du système d'exploitation *BSD Unix*. Cette licence « is very non-restrictive in its term, basically allowing anyone to do anything with code covered »[79], pour autant que le licencié accepte :

- De mentionner l'existence de droits d'auteur dans le code source ;
- De mentionner l'existence de droits d'auteur dans la documentation ;
- De reconnaître que le logiciel est fourni sans garantie.

[60] Jusqu'en 1999, la licence *BSD* comprenait également une clause publicitaire qui obligeait le licencié à mentionner les auteurs du logiciel lors de sa redistribution. Le résultat est qu'après de nombreuses années de développement, la mention obligatoire de certains logiciels contenait des dizaines, voir des centaines, de noms[80]. Pour cette raison, la licence fut modifiée afin de retirer la clause publicitaire.

Section 4 - La validité des licences libres

[61] Les contrats de licence des logiciels libres possèdent une nature et un contenu qui leur est propre. Certains assimilent même les licences *copyleftées* à des contrats viraux, puisque celles-ci suivent d'elles-mêmes le contenu numérique[81]. En raison de ces particularités, les détracteurs des logiciels libres prétendent que ces licences ne peuvent être considérés

---

[78] OPENSOURCE.ORG, « The BSD License », voir annexe B.

[79] MOZILLA.ORG, « Netscape Public license FAQ », (1999) *Mozilla*, source : <http://www.mozilla.org/NPL/FAQ.html>.

[80] FREE SOFTWARE FOUNDATION, « Le problème de la licence BSD », (2000) *FSF*, source : <http://www.fsf.org/philosophy/bsd.fr.html>.

comme des documents juridiques contraignables. Toutefois, aucune instance judiciaire n'a eu à trancher définitivement cette question jusqu'à maintenant. De plus, chaque licence étant soumis aux multiples législations nationales, la solution est très certainement différente d'une juridiction à l'autre.

[62] Aux États-Unis, certains auteurs questionnent la viabilité de la philosophie de la *FSF* parce que celle-ci va à l'encontre des principes promus par les droits d'auteur. Selon cette conception, la validité d'une licence utilisant la loi sur les droits d'auteur américaine pour éliminer la propriété sur les oeuvres dérivées pourrait être remise en question[82]. Toutefois, cet argument est insatisfaisant dans la mesure où le donneur d'une licence libre ne fait qu'exercer un droit qui lui est conféré par la loi et ne contrevient à aucune disposition légale. Dans le même sens, Robert Merge prétend que les licences *copyleftées* sont de simples restrictions informelles grevant l'information et qu'elles ne seraient pas exécutables lorsque le licencié n'a pas contracté directement avec les nombreux donneurs de licence précédents[83]. À cela, Eben Moglen, conseiller juridique de la *FSF*, répond :

> « There's no absence of privity [which isn't required anyway]. [...] In the case of the *GPL*, no one is bound to anything in particular unless she redistributes the software, modified or unmodified. Because copying and redistribution, or the making of derivatives, are never authorized in the absence of a license, undertaking to redistribute is clear acceptance of our terms for redistribution. »[84]

---

[81] Margaret Jane RADIN, « Humans, Computers, and Binding Commitment », (2000) 75 *Ind. L. J.* 1125, 1132.

[82] Mark LEMLEY, Peter MENELL, Robert MERGE et Pamela SAMUELSON, *Software and Internet Law*, New York, Aspen Publisher, 2001, p. 532ff.

[83] Robert P. MERGE, « The End of Friction? Property Rights and Contract in the "Newtonian" World of On-Line Commerce », (1997) 12 *Berkeley Tech. L. J.* 115.

[84] Eben MOGLEN cité dans Denis E. POWELL, « Judgment Day for the GPL? », (2000) *LinuxPlanet*, source : <http://www.linuxplanet.com/linuxplanet/reports/2000/1>.

[63] En effet, les licences telle que la *GPL* semblent conforme aux exigences juridiques dans la mesure où le consentement du licencié peut être constaté[85]. C'est le fait de conclure à l'acceptation des termes de la licence dès lors qu'il y a redistribution du logiciel qui laisse planer des doutes quand à l'opposabilité du document. En effet, si l'utilisateur ignore les implications de son geste, il n'est pas certain qu'une convention existe entre les parties. Pour cette raison, chaque situation représente un cas d'espèce soumis au pouvoir d'appréciation des tribunaux. Par exemple, un tribunal américain a déjà refusé de reconnaître l'application d'une licence de type *browse-wrap*, c'est-à-dire affichée sur un site Web, parce que l'utilisateur n'avait pas à accepter ou visionner le document avant de télécharger le logiciel en question[86]. De façon générale, pour que les conditions de la licence soient opposables au licencié, l'attention de ce dernier devra avoir été suffisamment portée sur ses clauses[87].

[64] Le plus souvent, les licences libres sont jointes au logiciel qu'elles grèvent. Cette façon de faire permet de les rapprocher du régime des licences *shrinkwrap* et *clickwrap*, en vertu desquelles le consentement du licencié est obtenu par des méthodes similaires. Bien qu'il subsiste toujours un débat quant à la validité de ces licences, le droit américain semble les accepter graduellement. La décision *ProCD, Inc* c. *Zeidenberg*[88], qui a confirmée la validité d'une licence *shrinkwrap,* et les amendements législatifs proposés dans le cadre du *Uniform Computer Information Transactions Act*[89]

---

[85] Mélanie CLÉMENT-FONTAINE, « La licence publique générale GNU », (1999) *Juriscom*, par. 32, source : <http://www.juriscom.net/uni/mem/08/presentation.htm>.

[86] *Specht* c. *Netscape Communications Corp.*, 150 F. Supp. 2d 585 (SDNY, 2001).

[87] David MCGOWAN, « Legal Implication of Open-Source Software », (2001) *U. Ill. L. Rev.* 241, 290.

[88] *ProCD, Inc* c. *Zeidenberg*, 86 F.3d 1447 (7e Cir. 1996).

[89] *Uniform Computer Information Transactions Act*, 2001, source : <http://www.law.upenn.edu/bll/ulc/ucita/ucita01.htm>.

(*UCITA*) le démontre. La majorité des auteurs concluent que les licences libres jouiraient du même statut[90].

[65] La même conclusion peut être déduite d'un jugement en injonction provisoire ayant été rendu dans l'affaire *MySQL* c. *Progress Software*[91]. Ce litige concerne le logiciel propriétaire *Gemini* de *Progress Software* dont le code informatique est lié dynamiquement au logiciel de base de données libre de *MySQL*. Comme le code source de *Gemini* n'était pas disponible, *MySQL* réclamait que sa distribution soit interrompue. Bien que cette demande n'ait pas été accordée, *MySQL* n'ayant pu démontrer un préjudice irréparable, la juge a considéré la *GPL* comme une licence contraignable dans les circonstances. Toutefois, un règlement hors cours étant intervenu dans cette affaire depuis, aucun procès ne viendra confirmer cette décision préliminaire[92].

[66] La balance des probabilités semble donc peser en faveur de la validité des licences libres. De plus, des normes sociales puissantes incitent les programmeurs à utiliser et à respecter les droits mis en place par celles-ci. À titre d'exemple, de nombreux conflits étant survenus jusqu'à aujourd'hui au sujet de l'application de la *GPL* se sont réglés au stade de la négociation[93]. Le plus souvent, les développeurs de logiciels préfèrent retirer le code informatique en infraction ou soumettre l'ensemble de leur logiciel à la *GPL* plutôt que de se mettre à dos une part importante de leur communauté[94]. Le droit peut donc difficilement se permettre de déclarer ces licences non-

---

[90] Patrick K. BOBKO, « Linux and General Public Licenses : Can Copyright Keep « Open Source » Software free? », 28 *AIPLA Q. J.* 81 ; I. V. HEFFAN, *loc. cit.*, note 74 ; D. MCGOWAN, *loc. cit.*, note 87 ; D. B. RAVICHER, *loc. cit.*, note 70.

[91] *Progress Software Corp.* c. *MySQL AB*, 195 F.Supp.2d 328.

[92] MYSQL, « MySQL AB and Nusphere Corporation Announce Settlement », (2002) *MySQL*, source : <http://www.mysql.com/press/release_2002_14.html>.

[93] Emmett Charles PLANT, « GPL Violation – NVIDIA », (2000) *Slashdot*, source : <http://www.slashdot.org/features/00/05/01/0047219.shtml> ; SENGAN, « Initio Violating the GPL? », (1998) *Slashdot*, source : <http://slashdot.org/articles/98/10/15/1520254_F.shtml>.

exécutoires alors qu'une proportion grandissante de l'industrie du logiciel repose sur celles-ci. En conséquence, il est raisonnable de tenir la validité de ces licences pour acquise afin d'évaluer leur portée.

* * *

[67] Ironiquement, c'est le régime des droits d'auteur qui permet aux partisans des logiciels libres d'assurer la liberté dont ils jouissent. Il est vrai que certaines licences libres offrent de meilleures garanties que d'autres, mais elles visent toutes la mise en commun des connaissances. Pour cette raison, les règles de la responsabilité civile doivent être en mesure de faire face à des produits conçus par une multitude de personnes distinctes. Pour y arriver, il est d'abord nécessaire de déterminer quelles sont les situations susceptibles d'entraîner la responsabilité civile des développeurs d'un logiciel libre.

---

[94] Mark H WEBBINK, « Open Source Software – Bridging the Chasm », (2002) 691 *PLI/Pat* 663, 683.

PARTIE II - Détermination de la responsabilité civile

Chapitre 1 - Les éléments contractuels

[68] La personne la plus susceptible de rechercher un dédommagement de la part des développeurs d'un logiciel libre est l'utilisateur de ce logiciel. Or, dès que l'utilisateur accepte les termes de la licence, une relation contractuelle s'établit entre lui et chacun des titulaires de droits dans le logiciel. Cette entente constitue la principale base des recours du licencié. Pour en bénéficier, il doit démontrer que l'un ou l'autre des donneurs de licence n'a pas exécuté une obligation dont il a la charge en vertu du contrat. Les licences libres étant spécialement conçues pour limiter la responsabilité civile de ces derniers, ce fardeau peut s'avérer difficile à surmonter.

Section 1 - La qualification des licences libres

[69] L'une des caractéristiques des contrats de licence libre est qu'ils sont imposés par les donneurs de licence. Le plus souvent, ceux-ci reprennent l'intégralité d'un contrat-type, tel que la *GPL*, et l'imposent à tous ceux qui désirent utiliser le logiciel. Le licencié n'a donc aucune possibilité d'en négocier les termes. Pour cette raison, les licences libres peuvent être considérées comme des contrats d'adhésion[95]. Même si cette qualification n'influence pas directement les droits et obligations des parties, elle peut avoir un impact sur la validité des clauses d'exclusion de responsabilité qui y apparaissent[96].

---

[95] M. CLÉMENT-FONTAINE, *loc. cit.*, note 85, par. 26.

[96] Marcus MAHER, « Open Source Software : The Success of an Alternative Intellectual Property Incentive Paradigm », (2000) 10 *Fordham Intell. Prop. Media & Ent. L. J.* 619, note 342.

[70] Le rattachement des licences libres au régime de l'un des contrats nommés prend plus d'importance. Malheureusement, même en ce qui a trait aux logiciels propriétaires traditionnels, cette problématique ne fait pas l'objet d'une solution unanime. Par exemple, aux États-Unis, la doctrine et les tribunaux qualifient les licences propriétaires de vente afin de les soumettre aux règles de l'article 2 du *Uniform Commercial Code*[97] (*UCC*). Il est vrai que cette catégorisation est plus ou moins juste puisque « Article 2, although titled 'Sales', applies broadly to 'transactions in goods' »[98]. En droit civil, c'est principalement le régime du louage qui est appliqué aux licences propriétaires[99]. Cependant, il est également possible d'y voir une vente en se fondant sur la relation traditionnelle qui existe entre le distributeur et le consommateur[100].

[71] Pourtant, ces deux solutions sont insatisfaisantes. Les licences d'utilisation de logiciels ne constituent manifestement pas des ventes puisqu'elles ne transfèrent aucun droit réel, élément essentiel à ce type de contrat. Il est tout aussi difficile de prétendre qu'il s'agit de locations puisque le licencié est généralement tenu à une redevance forfaitaire alors que le louage est un contrat à exécution successive.

[72] En principe, il apparaît donc plus approprié de considérer les licences propriétaires comme des contrats innomés. Ceci s'explique tout autant par leur objet spécifique (produit immatériel) que par le contenu des droits qu'elles transmettent (droit d'usage limité et non-exclusif)[101]. Pourtant, les

---

[97] *Uniform Commercial Code*, 1992, art. 2, source : <http://www.law.cornell.edu/ucc/2/overview.html>.

[98] Douglas E. PHILLIPS, « When Software Fails: Emerging Standards of Vendor Liability Under the Uniform Commercial Code », (1994) 50 *Bus. Law.* 151, 157.

[99] Michel VIVANTet autres, *Lamy droit de l'informatique et des réseaux*, Paris, Lamy, 2001, no. 840, p. 522 .

[100] Ejan MACKAAY, « Le marché du progiciel – licence ou vente? », (1994) 6 *Cahiers de propriété intellectuelle* 401, 412.

[101] Frédérique TOUBOL, *Le logiciel : Analyse juridique*, Paris, Feduci – L.G.D.J., 1986, p. 128.

tribunaux préfèrent leur appliquer le droit de la vente ou du louage afin d'y introduire les garanties implicites de ces régimes. L'équité implique que le donneur de licence exploitant un logiciel libre de manière commerciale devrait invariablement se voir appliquer le même traitement que celui des donneurs de licences propriétaires puisque le licencié lui a également versé une contre-partie.

[73] Cependant, la solution n'est pas nécessairement la même lorsque le logiciel est distribué gratuitement. Dans ce cas, le licencié n'est tenu à aucune redevance, ce qui exclut l'application du droit de la vente ou du louage. En droit civil, il est possible de considérer ces ententes en tant que prêt à usage. L'obligation de restitution qui est à la base du contrat de prêt perd alors de son importance puisque le logiciel est un bien immatériel pouvant être reproduit facilement[102]. En droit américain, il est possible que le *UCC* puisse tout de même être appliqué aux logiciels libres gratuits à cause de la grande portée de son article 2. Au surplus, un régime similaire de garanties légales pourrait leur être appliqué par le biais du *UCITA*, celui-ci s'appliquant à l'ensemble des contrats relatifs à l'information électronique[103]. Toutefois, à cause de l'opposition de nombreux groupes de pression, les États américains sont réticents à adopter cette législation. Actuellement, seuls le Maryland et la Virginie l'ont incorporé dans leurs systèmes juridiques[104]. Finalement, dans les juridictions où aucun régime légal similaire n'existe, il est fort probable que les contrats de licence faits à titre gratuit soient considérés comme des contrats innomés. Dans une telle situation, toute forme de garantie implicite est nécessairement exclue.

---

[102] M. CLÉMENT-FONTAINE, *loc. cit.*, note 85, par. 24.

[103] *Uniform Computer Information Transactions Act*, précité, note 89, art. 102(a)(10) et (11).

[104] UCITA ONLINE, « What's Happening to UCITA in the States », (2001) *UcitaOnline*, source : <http://www.ucitaonline.com/whathap.html>.

Section 2 - Les obligations contractuelles des développeurs

[74] L'objectif des licences d'utilisation des logiciels étant de délimiter les droits des licenciés, elles ne contiennent généralement aucune mention explicite quant aux obligations du donneur de licence. Les licences libres ne font pas exception à cette règle. Pourtant, les donneurs de licences sont tout de même tenus de remplir un certain nombre d'obligations envers le licencié. Ainsi, « l'obligation essentielle d'un prestataire est la délivrance de l'objet du contrat »[105]. Dans le contexte des licences libres, ceci signifie que le donneur de licence est tenu de mettre le logiciel à la disposition du licencié. Le fait de le mettre en ligne est donc suffisant. En ce qui concerne l'étendue de cette obligation, elle ne peut être déduite qu'en interprétant l'esprit des contrats de licences libres.

[75] La principale problématique découlant de l'obligation de délivrance consiste à déterminer si le licencié a le droit d'exiger la disponibilité du code source du logiciel. En fait, lorsque le licencié accepte les termes d'une licence libre, il s'attend à disposer de ce code. D'ailleurs, les licences libres *copyleftées* obligent le licencié à le fournir s'il redistribue lui-même le logiciel[106]. Toutefois, cette obligation le rend responsable envers son propre donneur de licence et non envers ses licenciés. Ceux-ci devraient tout de même être en mesure d'engager la responsabilité des développeurs pour l'absence du code source car celui-ci est nécessaire à l'exercice des droits qui leur sont conférés par la licence. À l'inverse, le code binaire perd de l'importance puisqu'il ne permet pas de modifier le logiciel. Au surplus, ce dernier n'est pas essentiel puisqu'il peut être obtenu par compilation. Il est donc possible d'envisager que l'objet des contrats de licences libres soit précisément la fourniture du logiciel sous la forme de code source. En ce

---

[105] M. VIVANTet autres, *op. cit.*, note 98, no. 1282, p. 734.

[106] FREE SOFTWARE FOUNDATION, *loc. cit.*, note 73, preamble.

sens, l'obligation de délivrance demeurerait inexécutée tant que celui-ci n'est pas disponible.

[76] L'obligation de délivrance peut également être interprétée de façon à inclure la documentation du logiciel à titre d'accessoire. C'est, entre autres, la solution qu'ont retenu les tribunaux français face aux licences propriétaires[107]. Il est loin d'être certain que ce principe puisse être appliqué aux licences libres puisqu'il s'oppose à l'esprit de ces conventions. En effet, la documentation et le programme informatique y sont perçus comme des éléments autonomes. Alors que le licencié d'un logiciel propriétaire s'attend à ce que la documentation soit jointe à celui-ci, le licencié d'un logiciel libre s'attend plutôt à trouver cette documentation au sein de la communauté qui gravite autour du logiciel. L'étendue de l'obligation de délivrance étant soumise à la volonté des parties, la documentation devrait donc en être exclue en matière de logiciels libres. De plus, bien que certains auteurs définissent la notion de code source de façon à y inclure les commentaires nécessaires à sa compréhension[108], les licences libres font uniquement référence au programme informatique. Il ne serait donc pas possible de tenir un donneur de licence responsable pour le manque de commentaires à l'intérieur du code source.

[77] Une autre obligation pouvant être attribuée aux donneurs de licences libres concerne l'information devant être fournie aux licenciés. En droit civil, l'équité contractuelle fait parfois reposer une obligation minimale de renseignement sur les épaules du contractant qui sait quelque chose dont la connaissance est utile à l'autre[109]. Dans le contexte des logiciels, les

---

[107] Philippe LE TOURNEAU, *Théorie et pratique des contrats informatiques*, Paris, Éditions Dalloz, 2000, p. 97.

[108] Hervé CROZE et Franck SAUNIER, *Logiciels : retour aux sources*, Paris, JCP ed. Générale, 1996, doctrine 3909, s. 7, p. 94.

[109] Claude LUCAS DE LEYSSAC, *L'obligation de renseignements dans les contrats*, coll. « l'information en droit privé », Paris, LGDJ, 1978, p. 305.

renseignements ainsi fournis devraient attirer l'attention sur les risques.[110] Par exemple, le matériel pré-requis au bon fonctionnement du logiciel et ses principaux bogues connus devraient être indiqués au licencié. Dans le même ordre d'idée, tant que la première version officielle d'un logiciel n'est pas au point, il paraît important de mentionner clairement qu'il s'agit d'une version préliminaire. En cas d'omission, le licencié lésé pourrait réussir à impliquer la responsabilité du donneur de licence, particulièrement s'il s'agit d'un logiciel libre pour lequel une contre-partie a été versée. Cette obligation demeure toutefois inexistante en common law[111].

[78] Enfin, lorsque les développeurs font la promotion des caractéristiques et des fonctions de leur logiciel, il est possible que les tribunaux considèrent que leurs affirmations constituent des garanties. De façon générale, une garantie « is a statement of fact [...] respecting the quality or character of the goods »[112]. Aussi, le donneur de licence qui désire éviter d'engager sa responsabilité civile doit agir prudemment quant à la façon dont il présente le logiciel libre au licencié, car ses déclarations pourraient être interprétées comme des engagements. Par exemple, ce pourrait être le cas s'il prétend que le logiciel est compatible avec un standard reconnu[113]. Pour bénéficier de ces garanties, le licencié devra généralement prouver que les déclarations en question sont des éléments de la convention. Compte tenu des circonstances, ce fardeau de preuve peut être difficile à surmonter. Toutefois, certaines législations allègent la charge du licencié en élargissant l'étendue de l'entente entre les parties. Ainsi, le *UCITA* américain précise que la publicité relative à l'information peut faire partie du contrat[114].

---

[110] M. VIVANTet autres, *op. cit.*, note 99, no. 924, p. 556.

[111] Allan E. FARNSWORTH, « The Concept of "Good Faith" in American Law », (1993) *Consiglio Nazionale delle Ricerche*, source : <http://www.cnr.it/CRDCS/frames10.htm>.

[112] Cem KANER, « The Law of Software Quality », (2000) *Software management Conference San Jose*, source: <http://www.kaner.com/pdfs/slides/asmlaw.pdf>.

[113] Cem KANER, « Liability for Product Incompatibility », (1998), 4 *Software QA* 33, source: <http://www.kaner.com/liability.html>.

[114] *Uniform Computer Information Transactions Act*, précité, note 89, art. 402.

Section 3 - Les garanties implicites

[79] Les obligations à la charge des donneurs de licence ne sont pas les seuls éléments de responsabilité contractuelle pouvant être invoqués par le licencié ayant subi un dommage. Dans bien des cas, ce dernier sera également protégé par une ou plusieurs garanties implicites. Celles-ci sont insérées implicitement à l'intérieur des conventions par le droit afin de rétablir l'équilibre contractuel entre les parties. Elles sont propres à chaque régime juridique et varient en fonction de la qualification du contrat. Parmi la multitude de garanties implicites existantes, quelques-unes sont susceptibles d'être introduites dans les contrats de licences libres.

[80] Les garanties contre l'éviction sont certainement les plus universelles. Elles existent tout autant en droit civil qu'en common law, où elles prennent le nom de *warranty of good title*. Elles trouvent application dans le cadre des contrats de vente et de louage. De façon générale, elles assurent au licencié que son utilisation du logiciel ne sera perturbé ni en fait, ni en droit, par son partenaire ou des tiers. Elle empêche donc le donneur de licence de revenir sur l'objet de sa concession tant que le licencié n'a pas excédé les droits qu'il tenait de lui[115]. L'application la plus probable de cette garantie a trait aux troubles de droit provenant de tiers. Par exemple, il est possible que l'utilisation du logiciel enfreigne un brevet resté inconnu des développeurs. Dans ce cas, le licencié pourrait être tenu responsable et se voir interdire l'utilisation du logiciel. Dans la mesure où cette interdiction lui cause un dommage, il pourrait avoir intérêt à poursuivre le donneur de licence en invoquant la garantie contre l'éviction[116].

---

[115] M. CLÉMENT-FONTAINE, *loc. cit.*, note 85, par. 24.

[116] Stephen M. MCJOHN, « The Paradoxes of Free Software », (2000) 9 *Geo. Mason L. Rev.* 25, 35.

[81] La garantie de qualité loyale et marchande de la common law pourrait aussi avoir un impact important sur les contrats de licences libres. D'ailleurs, celle-ci est reprise par les *UCC* et *UCITA* américains[117]. Cette garantie précise que « [the] goods will be fit for the purpose which they are ordinarily intended »[118]. Le logiciel doit donc être en mesure d'effectuer ce qui en est attendu par un licencié raisonnable. Ainsi, un serveur Web devrait être en mesure de diffuser des documents sur Internet. Si ce n'est pas le cas, le donneur de licence pourrait être tenu des dommages résultant de cette incapacité. Cette garantie implique donc une charge potentiellement importante pour les développeurs. Ceci est particulièrement vrai tant qu'un logiciel libre demeure dans sa phase initiale de conception car son fonctionnement n'est pas nécessairement stable.

[82] Le droit civil possède aussi une garantie implicite qui lui est propre. Il s'agit de la garantie contre les vices cachés. Elle s'applique essentiellement aux contrats de vente et de louage. Elle peut aussi s'introduire dans les contrats de prêt, mais uniquement lorsque le prêteur connaît le vice. Le vice visé par cette garantie est celui qui rend le logiciel impropre à son usage ou qui diminue tellement cet usage que le licencié n'aurait pas accepté les termes de la licence, ou aurait payé un prix inférieur. Il s'agit donc d'un défaut majeur du logiciel. De plus, il doit être caché, c'est-à-dire qu'il n'est pas apparent et qu'il n'est pas connu du licencié au moment où celui-ci accepte les termes de la licence. Enfin, le vice doit exister lors de la conclusion du contrat[119]. Toutefois, dans la mesure où le contrat de licence est qualifié de louage et que l'on considère son exécution comme successive, certains prétendent que la garantie doit également être étendue aux défauts

---

[117] *Uniform Commercial Code*, précité, note 97, art. 2-314 et 2A-212 ; *Uniform Computer Information Transactions Act*, précité, note 89, art. 403(a)(1).

[118] Matthew J. SMITH, « An Overview of the Uniform Computer Information Transactions Act: Warranties, Self-Help, and Contract Formation Why Ucita Should Be Renamed 'The Licensors' Protection Act' », (2001) 25 *S. Ill. U. L. J.* 389, 402.

[119] F. TOUBOL, *op. cit.*, note 101, p. 134.

qui apparaissent par la suite[120]. Cette interprétation semble exagérée, car elle étend cette garantie bien au delà de sa portée initiale. Au surplus, cette garantie semble difficile à appliquer aux logiciels libres vu les termes généreux de leurs licences qui laissent difficilement entrevoir pourquoi le licencié les refuseraient malgré l'existence d'un vice.

[83] Finalement, l'application de garanties de conformité est envisageable. Aux États-Unis, elles se nomment *warranty of fitness* et se retrouvent autant dans le *UCC* que dans le *UCITA*[121]. En droit civil elles sont plutôt intégrées à l'intérieur de l'obligation de délivrance[122]. Dans tous les cas, elles obligent le donneur de licence à fournir un logiciel conforme à l'objectif particulier du licencié à partir du moment où il prend connaissance de cet objectif. Compte tenu de cette dernière limitation, le recours à ces garanties devrait être extrêmement restreint en ce qui a trait aux logiciels libres.

[84] Dans la mesure où ces protections juridiques sont intégrées à l'intérieur des contrats de licence, le droit oblige les développeurs à concevoir des logiciels possédant des qualités minimales. C'est donc dans le but de se soustraire à ces garanties que des clauses d'exclusion de responsabilité sont inclues dans l'ensemble des licences libres.

---

[120] M. CLÉMENT-FONTAINE, *loc. cit.*, note 85, par. 54.

[121] Uniform Commercial Code, précité, note 96, art. 2-315 ; *Uniform Computer Information Transactions Act*, précité, note 89, art. 405(a)(1).

[122] F. TOUBOL, *op. cit.*, note 101, p. 131.

Section 4 - Les clauses d'exclusion de responsabilité

[85] Les garanties implicites constituent le principal risque de responsabilité contractuelle pour les développeurs d'un logiciel libre. Pour cette raison, ceux-ci tentent d'éviter leur application en ayant recours à des clauses d'exclusion totale de responsabilité. Il s'agit du principal point commun de toutes les licences libres[123]. Ainsi, la licence *BSD* contient la clause suivante :

> « This software is provided by the author "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the author be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage. »[124]

[86] Pourtant, le droit ne permet pas toujours aux parties contractantes d'exclure entièrement leur responsabilité civile. Par ailleurs, lorsque les clauses d'exclusions sont autorisées, les tribunaux se réservent généralement le privilège d'en limiter la portée en tenant compte des circonstances. L'effet escompté par les développeurs de logiciels libres n'est donc pas assuré.

[87] En droit civil, la validité des exclusions conventionnelles varie en fonction de la qualification du contrat. Le régime de la vente est le seul à interdire les clauses de non-garantie dans certaines circonstances,

---

[123] Bruce PERENS, « The Open Source Definition », dans DIBONA, OCKMAN et STONE (dir.), *op. cit.*, note 22.

[124] OPENSOURCE.ORG, *loc. cit.*, note 78.

particulièrement lorsque le vendeur est un professionnel[125]. Au contraire, elles sont généralement permises en matière de louage. En ce qui concerne les garanties contre l'éviction, elles empêchent le licencié de réclamer des dommages et intérêts mais son droit à la résiliation du contrat ou à la diminution du prix n'est pas affecté[126]. De plus, en matière de vices cachés, rien n'interdit aux parties de reléguer la totalité des risques sur les épaules du licencié. Ce dernier principe vaut également en matière de prêt, dans le mesure où la garantie contre les vices cachés peut y être appliquée. Toutefois, le donneur de licence n'est pas en mesure d'exclure sa responsabilité civile lorsqu'il a commis une faute lourde ou intentionnelle[127]. La bonne foi des développeurs est donc un élément déterminant lors de l'évaluation de la validité de ces clauses.

[88] En common law, le principe de la liberté contractuelle permet aux parties de s'entendre entre elles quant au partage des risques. En choisissant d'exclure totalement la responsabilité du donneur de licence, elles ne font qu'exercer ce droit. Dans ce contexte, les clauses des licences libres sont à prime abord valides. Par contre, certaines lois viennent quelque fois restreindre la possibilité de limiter sa responsabilité civile par le biais de conventions. C'est notamment le cas du UCC américain, qui exige qu'une telle clause soit conçue de façon à ce que la personne contre laquelle elle s'applique soit en mesure de la remarquer[128]. À cela, la jurisprudence a ajouté que la mention doit être non-ambiguë[129]. Quant au UCITA, il reprend les principes du UCC tout en apportant quelques éléments d'évaluation. Par

---

[125] E. MACKAAY, *loc. cit.*, note 100, 415.

[126] Marcel PLANIOL et Georges RIPERT, *Traité élémentaire de droit civil*, t. 2, 10e éd., L.G.D.J., Paris, 1926, no 525.

[127] Jean-louis BAUDOUIN et Pierre-Gabriel JOBIN, *Les obligations*, 5e Éd., Éditions Yvon Blais, Cowansville, 1998, no 872 p. 707.

[128] M. MAHER, *loc. cit.*, note 96, note 339.

[129] Michael D. SCOTT, *Scott on Computer Law*, Vol. 2, 2d ed., Englewoods Cliff, New Jersey, 1998, no. 162, p. 7-40.1 et no. 168, p.7-40.2.

exemple, le fait que la clause soit rédigée en lettres majuscules contrastant avec le reste du texte pourrait servir à justifier sa validité[130]. Puisque les licences libres respectent généralement ces critères, l'ensemble des garanties implicites établies par ces lois leur seraient inapplicables. Une exception permet cependant d'impliquer la responsabilité du donneur de licence lorsque celui-ci n'agit pas avec « good faith, diligence, reasonableness, and care »[131].

[89] Selon les juridictions, les clauses de non-garantie peuvent aussi être inefficaces pour certains types de dommages. C'est notamment le cas en droit québécois, où il n'est pas possible d' « exclure ou limiter sa responsabilité pour le préjudice corporel ou moral causé à autrui »[132]. L'exclusion demeure tout de même utile dans ces circonstances puisqu'elle équivaut à la dénonciation d'un danger, ce qui peut mener à la réduction de la responsabilité du donneur de licence.

[90] En somme, les clauses d'exclusion de responsabilité paraissent autorisée dans la plupart des cas. Malgré tout, il demeure possible que l'une d'entre elles soit déclarée inopérante dans le cadre d'une situation particulière. En effet, les tribunaux ont tendance à les interpréter de façon restrictive et se réservent généralement le privilège d'en atténuer les effets[133]. Plusieurs éléments sont susceptibles d'être pris en considération lors de la prise d'une telle décision. Ceux-ci comprennent :

- Le pouvoir de marchandage des parties ;
- Les options disponibles ;

---

[130] M. J. SMITH, *loc. cit.*, note 118, 406.

[131] Uniform Commercial Code, précité, note 96, art. 1-102(3) ; *Uniform Computer Information Transactions Act*, précité, note 89, art. 113(a)(1).

[132] *Code civil du Québec*, L.Q. 1991, c. 64, art. 1474.

[133] M. VIVANTet autres, *op. cit.*, note 99, no. 1129, p. 662.

- Les avantages accordés en contre-partie de la clause ;
- La possibilité d'obtenir une entente sans clause similaire avec une autre personne ;
- La connaissance de l'existence de la clause[134].

[91] Lorsqu'un logiciel libre est impliqué, quelques éléments jouent en faveur de l'intervention judiciaire. Premièrement, il est évident que le pouvoir de marchandage y est monopolisé par les donneurs de licence. À ce propos, la simple qualification de contrat d'adhésion peut parfois suffire à faire tomber l'exclusion de responsabilité[135]. Deuxièmement, il est important de considérer que toutes les licences de logiciels, qu'elles soient libres ou propriétaires, contiennent des clauses similaires. En ce sens, le licencié n'a souvent d'autre choix que d'accepter de se soumettre à ce régime. Troisièmement, lorsque le logiciel est exploité commercialement, l'absence de protection contractuelle risque d'être injustifiée en raison du montant payé. Quatrièmement, les licences libres qui tentent d'exclure d'emblée toute forme de responsabilité civile par le biais d'une clause unique peuvent être défavorisées. En effet, cette technique pourrait être considérée comme abusive puisqu'il est possible d'utiliser des clauses spécifiques pour chaque risque envisagé[136].

[92] À l'inverse, plusieurs arguments peuvent être apportés à l'appui des clauses d'exclusion de responsabilité des licences libres. Tout d'abord, il existe une alternative à l'acceptation des risques puisque le licencié possède toujours l'option de s'assurer ou de contracter avec un fournisseur de service garantissant la solution informatique qu'il met en place. Ensuite, il est

---

[134] David SLEE, « Liability for Information Provision », (1992) 3 *The Law Librarian* 155, 157.

[135] Au Québec, *Code civil du Québec*, précité, note 131, art. 1437 ; aux États-Unis, *Vault Corp.* c. *Quaid Software Ltd.*, 847 F.2d 255 (5e Cir. 1988).

[136] D. SLEE, *loc. cit.*, note 134, 158.

essentiel de prendre en compte la quantité de droits concédés par les développeurs de logiciels libres en échange de cette exonération. Le licencié se voyant accorder l'autorisation de copier, de modifier et de redistribuer le logiciel, l'équilibre contractuel semble respecté. De plus, la clause d'exclusion étant toujours mise en évidence et rédigée en termes clairs, le licencié peut difficilement prétendre qu'il n'en avait pas connaissance au moment où il a accepté les termes de la licence. Finalement, il faut considérer le fait que les donneurs de licence font, le plus souvent, don de leur temps et de leurs connaissances. Dans l'hypothèse où leur responsabilité contractuelle est exclue, un très grand nombre de personnes ont les moyens de s'impliquer dans le développement des logiciels libres et leur coût de production peut demeurer bas[137].

[93] Dans l'ensemble, il semble que la balance des probabilités penche en faveur de la validité des clauses de non-garantie des logiciels libres gratuits. En ce qui concerne les logiciels libres distribués contre une rémunération, la solution sera probablement différente pour chaque cas d'espèce, compte tenu des circonstances particulières et du montant payé par le licencié.

Section 5 - Les lois de protection des consommateurs

[94] En plus du droit commun des contrats, un très grand nombre d'États ont édictés des lois particulières visant à protéger les consommateurs. Ces législations peuvent avoir des effets importants sur les contrats qui y sont soumis. Dans certains cas elles impliquent un certain formalisme qui, s'il n'est pas respecté, entraîne la nullité de l'entente. Par exemple, la *Loi de protection du consommateur* de l'Ontario exige l'existence d'un écrit pour tous les contrats d'une valeur supérieure à 50 dollars[138]. Dans d'autres cas, elles

---

[137] Robert W. GOMULKIEWICZ, « How Copyleft Uses License Rights to Succeed in the Open Source Software Revolution and the Implications for Article 2b », (1999) 36 *Hous. L. Rev.* 179, 192.

[138] *Loi sur la protection du consommateur*, L.R.O., 1990, c. C-31, art. 19.

accordent des garanties au consommateur. Le plus souvent cette protection équivaut à ce qui existe sous le régime de la garantie de qualité loyale et marchande[139]. Plusieurs d'entres elles abordent également la problématique des clauses abusives[140]. Ces dispositions étant généralement d'ordre public, elles ne peuvent être exclues par le commerçant. Aussi, dans l'hypothèse où les lois de protection des consommateurs s'appliquent aux licences libres, la responsabilité civile des développeurs se trouve grandement accrue.

[95] À priori, il semble que ces lois devraient à tout le moins protéger les licenciés ayant payé pour obtenir un logiciel libre. Cela ne pose aucune difficulté dans la mesure où ces conventions sont qualifiées de vente ou de louage puisqu'il s'agit du champ d'application traditionnel de ces législations. Par contre, si elles sont considérées en tant que contrats innomés, il n'est pas certain que le consommateur puisse en bénéficier. Il est fort probable que les tribunaux rectifient cette situation en considérant tout de même ces contrats de licence comme des contrats de consommation. Ceci s'explique par l'interprétation large que doivent recevoir les lois de protection des consommateurs qui ont « vocation à englober tous les contrats que le consommateur conclut dans le marché de masse »[141].

[96] Il est peu plausible que les logiciels libres gratuits reçoivent le même traitement. Dans l'hypothèse où leurs licences sont perçues comme des prêts, celles-ci devraient être automatiquement exclues du régime de la protection des consommateurs qui cible particulièrement les contrats de vente et de louage. D'autre part, l'objectif de ces lois est de protéger les

---

[139] Au Québec, *Loi sur la protection du consommateur*, L.R.Q., 1977, c. P-40.1, art. 37 et 38 ; aux Etats-Unis, *Magnusson-Moss Warranty Act*, 15 U.S.C., 2302(c).

[140] Pierre-Emmanuel MOYSE et Vincent GAUTRAIS, « Droit des auteurs et droit de la consommation dans le cyberespace : la relation auteur/utilisateur », (1996) *Léger Robic Richard*, source : <http://www.robic.ca/publications/070.shtml>.

[141] Vincent GAUTRAIS et Ejan MACKAAY, « Les contrats informatiques », dans Denys-Claude LAMONTAGNE, *Contrats spéciaux*, Cowansville, Éditions Yvon Blais, 2001, p. 279.

intérêts économiques d'une partie faible, le consommateur, face à ceux d'une partie forte, le commerçant. Il serait paradoxal qu'elles trouvent application dans une situation où le consommateur se voit accorder de nombreux droits par un commerçant renonçant à toute compensation.

[97] L'application du droit de la consommation aux logiciels pose également le problème de la qualification des parties. Dans le contexte des logiciels propriétaires, cela consiste exclusivement à déterminer si le licencié correspond à la définition qui est donnée du consommateur. De façon générale, c'est le cas lorsque le contrat de licence n'a pas de rapport direct avec l'activité professionnelle exercée par l'individu licencié[142]. Toutefois, dans le contexte des logiciels libres, la qualification du donneur de licence en tant que commerçant doit aussi être évaluée. En effet, une bonne partie des développeurs de logiciels libres ne sont manifestement pas des commerçants puisqu'en ayant recours aux licences libres ils n'accomplissent pas des actes de commerce dans le cadre de leur profession. Ces derniers, bien que fréquemment professionnels de l'informatique, agissent le plus souvent sans but lucratif et en dehors du cadre de leur activité principale. A priori , les lois de protection des consommateurs ne devraient donc pas leur être appliquées. Par contre, un certain nombre d'entreprises dont le modèle économique est fondé sur la distribution de logiciels libres peuvent certainement être considérées comme des commerçants, ce qui a pour conséquence de les assujettir à ces lois.

* * *

[98] Les développeurs sont donc en mesure d'exclure leur responsabilité contractuelle relativement efficacement lorsqu'ils distribuent leur logiciel libre gratuitement. Dans cette situation, seules quelques exceptions semblent en

---

[142] M. VIVANTet autres, *op. cit.*, note 99, no. 1286, p. 736.

mesure de faire tomber les protections mises en place par les licences libres. À l'inverse, les développeurs qui distribuent un logiciel libre sur une base commerciale semblent moins bien protégés. Entres autres, il est possible que les exclusions de responsabilité incluses dans les licences de leurs logiciels soient jugées abusives et que le droit de la consommation soit appliqué à leur égard. Pourtant, une telle situation ne s'est encore jamais présentée. De plus, dans un cas comme dans l'autre, la responsabilité civile des développeurs peut aussi être impliquée à l'extérieur du cadre de la relation contractuelle avec le licencié.

Chapitre 2 - Les éléments extra-contractuels

[99] Le licencié n'est pas la seule personne susceptible d'être lésée par un logiciel. Dans bien des cas, le dommage sera subi par un tiers qui ignore les particularités de la relation contractuelle existante entre le donneur de licence et le licencié. C'est le cas, par exemple, lorsqu'un logiciel effectue une erreur de calcul dans un traitement de radiothérapie et cause la mort d'un patient[143] Dans une telle situation, les termes de la licence du logiciel, qu'elle soit libre ou propriétaire, prennent une importance secondaire. Les obligations et les agissements des développeurs doivent alors être évalués en fonction d'une norme objective.

Section 1 - Le droit commun

[100] L'objectif du droit commun de la responsabilité civile extra-contractuelle est d'assurer le respect de l'équilibre économique entre les différentes personnes qui composent la société. Pour y arriver, celles-ci peuvent être tenues de réparer le préjudice corporel, matériel et moral causé

---

[143] *Jones* c. *Minnesota Mining & Manufacturing Co.*, 669 P. 2d 744 (N.M.App. 1983).

à autrui par leurs actes ou leurs omissions, dans la mesure où les gestes posés sont jugés répréhensibles[144]. Ainsi, trois éléments essentiels composent la responsabilité civile extra-contractuelle : une faute (en droit civil) ou une négligence (en common law), un dommage et un lien de causalité entre les deux. Un dédommagement est requis dès lors que les trois éléments sont réunis. En ce qui concerne la détermination de la responsabilité civile des développeurs de logiciels libres, l'évaluation de la faute ou de la négligence est particulièrement pertinente.

[101] La faute ou la négligence des développeurs peut se révéler de différentes façons. Le dysfonctionnement du logiciel est certainement la plus évidente. Cependant, comme le démontre la jurisprudence américaine en la matière, « It can be expected that in future software-implicated injuries, the program will usually 'perform as programmed' »[145]. Le plus souvent, le dommage est donc causé par une mauvaise conception du logiciel, et non par une simple erreur de programmation. Cette mauvaise conception peut, par exemple, prendre la forme d'une faille de sécurité[146]. Elle peut s'étendre à tous les éléments du logiciel, qu'il s'agisse du programme informatique, du contenu informationnel ou de la documentation[147]. Elle peut aussi bien surgir à l'étape de la programmation que lors de la mise en place d'un environnement de travail inadéquat[148]

---

[144] Jean-Louis BAUDOUIN et Patrice DESLAURIERS, *La responsabilité civile*, 5e Éd., Éditions Yvon Blais, Cowansville, 1998, no. 6, p. 4 ; Allen M. LINDEN, *Canadian Tort Law*, 6th Ed., Butterworth, Markham, 1997, p. 4.

[145] Thomas G. WOLPERT, « Product Liability and Software Implicated in Personal Injury », (1993) 60 *Def. Couns. J.* 519, 532.

[146] Alan Charles RAUL, Frank R. VOLPE et Gabriel S. MEYER, « Liability for Computer Glitches and Online Security Lapses », (2001) 31 *BNA Electronic Commerce Law Report* 849, source : <http://www.sidley.com/cyberlaw/features/liability.asp>.

[147] Cem KANER, « Liability for Defective Content », (1996) 3 *Software QA* 56, source : <http://www.kaner.com/badcont.htm>.

[148] Mary L. BEYER, « Managing the Risk of Virus Liability », (1993) 12 *Computer Law.* 22.

[102] Tout comme en matière contractuelle, la faute délictuelle du droit civil implique l'inexécution d'un devoir par les développeurs. Selon les circonstances, il peut s'agir du manquement à un devoir ou de la violation d'une norme de conduite[149]. La situation n'est pas tellement différente en common law où la commission d'un acte négligent est déterminée « by identifying the appropriate standard of care and applying it to the facts of the case »[150]. Dans les deux cas, ces gestes constituent une faute ou une négligence dans la mesure où ils s'éloignent du comportement d'une personne raisonnable ayant les mêmes connaissances et placée dans les mêmes circonstances. Il s'agit donc, au minimum, d'un comportement défaillant par rapport à une norme reconnue à un moment donné. Lorsque la défaillance est grave au point où elle démontre une insouciance grossière ou le mépris total des intérêts d'autrui, elle peut être assimilée à un geste posé avec l'intention de causer un dommage. Enfin, cette évaluation s'effectue toujours sur une base individuelle[151]. Ceci implique que pour chaque logiciel libre ayant causé un dommage, il y a nécessairement des développeurs responsables et d'autres à qui rien ne peut être reproché.

[103] Par ailleurs, les professionnels sont généralement tenus de respecter des critères de prudence plus élevés. En effet, leur comportement est soumis aux règles qui régissent leur activité professionnelle. Ainsi, aux États-Unis, le *Restatement (Second) of Torts* précise que ceux qui « undertake to render services in the practice of a profession or trade are required to exercise the skills and knowledge normally possessed by members of that profession »[152]. Cependant, il est difficile d'établir de tels critères pour les programmeurs. Ceci s'explique par la diversité de leur formation, le manque de normes communes tel que des standards, des règles d'éthique ou des

---

[149] J.-L. BAUDOUIN et P. DESLAURIERS, *op. cit.*, note 144, no. 125, p. 96.

[150] P. H. OSBORNE, *op. cit.*, note 144, p. 24.

[151] Léon MAZEAUD, « Les fautes collectives », (1956) 16 *R. du B.* 405, 407.

[152] *Restatement (Second) of Torts*, 1965, art. 299A.

mesures disciplinaires et l'absence de réglementation en la matière[153]. Le fait qu'il n'existe toujours aucun ordre professionnel compétent en la matière accentue cette problématique. Malgré tout, les efforts d'organisations comme l'*Association for Computing Machinery* (*ACM*), l'*Institute for Certification of Computer Professionals* (*ICCP*) et l'*International Standards Organization* (*ISO*), montrent qu'une normalisation du domaine s'amorce. Ainsi une proposition récemment soumise à la *Internet Engineering Task Force* (*IETF*) vise à standardiser la dénonciation des vulnérabilités aux fabricants de logiciels ainsi que la réaction de ces derniers une fois qu'ils ont été informés du danger[154]. Le devoir de prudence et de diligence des programmeurs à l'intérieur de leur champ de compétence devrait donc être rehaussé en proportion des initiatives de ce genre[155].

[104] Quelque soit le standard retenu pour évaluer le comportement des programmeurs, il ne saurait être question d'obliger les développeurs de logiciels libres à fournir un produit exempt d'imperfection. Ceci se comprend dans la mesure où la vérification complète d'un logiciel est impossible[156]. Le caractère raisonnable des décisions des développeurs doit donc être apprécié au cas par cas. Les éléments d'évaluation pertinents comprennent :

- La connaissance antérieure du problème ayant causé le dommage ;
- L'expertise des développeurs ;
- Le respect des standards de l'industrie ;

---

[153] Richard A. GLASER et Leslee M. LEWIS, « Redefining the Professional: The Policies and Unregulated Development of Consultant Malpractice Liability », (1995) 72 *U. Det. Mercy L. Rev.* 563, 577.

[154] Steve CHRISTEY et Chris WYSOPAL, « Responsible Vulnerability Disclosure Process », 2002, *Internet Engineering Task Force*, source : <http://www.ietf.org/internet-drafts/draft-christey-wysopal-vuln-disclosure-00.txt>.

[155] Donald R. BALLMAN, « Software Tort: Evaluating Software Harm by Duty of Function and Form », (1997) 3 *Conn. Ins. L. J.* 417, 459.

[156] Cem KANER, « The Impossibility of Complete Testing », (1997) 4 *Software QA* 28, source : <http://www.kaner.com/imposs.htm>.

- L'analyse de la sécurité et des risques ;
- La prise en compte des erreurs potentielles d'utilisation ;
- La prise en charge de l'information provenant des utilisateurs ;
- La méthodologie utilisée pour tester le logiciel[157].

[105] En tenant compte de ces éléments, il semble que l'adoption d'une licence libre et d'un modèle de développement distribué joue en faveur des développeurs. Tout d'abord, le nombre de développeurs impliqués dans le projet pouvant être multiplié, les problèmes connus sont susceptibles d'être réglés beaucoup plus rapidement, ce qui réduit les risques. Ensuite, les standards informatiques y ont plus de chances d'être respectés à cause de la diversité des intérêts qui motivent les différents contributeurs. De plus, il est difficile d'imaginer une technique prenant plus efficacement en charge l'information provenant des utilisateurs puisque ces derniers sont en mesure d'effectuer eux-même les modifications nécessaires. Enfin, en mettant leur code source à la disposition des utilisateurs, les développeurs facilitent la découverte des faiblesses de conception ou de programmation du programme informatique. Le logiciel est ainsi testé beaucoup plus efficacement, dans la mesure où il est suffisamment distribué et utilisé.

[106] Il est vrai qu'une méthode de développement distribuée ne permet pas d'adopter de protocole strict quand au contrôle de la qualité du logiciel. L'expertise des développeurs demeurant extrêmement variable, l'analyse de la sécurité et des risques peut être inexistante. Par contre, les développeurs de logiciels libres étant soumis à une révision par leurs pairs, le risques qu'un logiciel de piètre qualité soit diffusé à grande échelle se trouve réduit.

---

[157] C. KANER, *loc. cit.*, note 112.

[107] Cette évaluation doit cependant être adaptée à chaque situation afin de tenir compte de l'usage particulier auquel le logiciel est destiné.[158] Dans certains domaines d'activité, les probabilités qu'un dommage survienne sont plus élevées. Dans d'autres, c'est la sévérité du dommage potentiel qui est plus grande. À titre d'exemple, il suffit de penser aux logiciels utilisés pour la gestion des réacteurs nucléaires. Les développeurs qui oeuvrent dans de tels milieux doivent nécessairement faire preuve de plus de prudence et de diligence[159]. Dans ce contexte, il est possible qu'un autre modèle de développement soit parfois mieux adapté. Or, le fait de ne pas avoir recours à une technologie existante peut engendrer une responsabilité si les règles fondamentales de prudence recommandent cette utilisation[160]. La diffusion de certains logiciels sous une licence libre pourrait donc être perçue de cette façon. Ce serait probablement le cas lorsqu'une analyse méthodique du code informatique est indispensable ou lorsqu'un nombre infime de programmeurs possèdent les connaissances nécessaires pour participer au projet.

[108] Bien que le régime du droit commun de la responsabilité extra-contractuelle s'adresse avant tout aux tiers ayant subi un dommage, certaines juridictions permettent également au licencié de s'en prévaloir. C'est particulièrement le cas des États soumis à la common law[161]. Toutefois, en agréant aux termes de la licence, le licencié d'un logiciel libre accepte parfois d'assumer la totalité des risques. C'est le cas, par exemple, lorsque la *GPL* est impliquée. Ainsi, l'article 11 de cette licence précise : « The entire

---

[158] Paul A. MATHEW, « Architects, Engineers, Computer Product and the Law: A Matter of Anticipation », (1982) 3 *Computer L. J.* 337, 351.

[159] Cem KANER, « Software Negligence and Testing Coverage », (1995) 2 *Software QA* 18, source : <http://www.kaner.com/coverage.htm>.

[160] Michael RUSTAD et Lori E. EISENSCHMIDT, « The Commercial Law of Internet Security », (1995) 10 *High Tech L. J.* 213.

[161] Jody ARMOUR et Watts S. HUMPHREY, « Software Product Liability », (1993) *Carnegie Mellon Software Engineering Institute*, source : <http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.013.html>.

risk as to the quality and performance of the program is with you [the licensee] »[162]. Dans une telle situation, le licencié ne devrait pas être en mesure d'invoquer ce régime juridique tant et aussi longtemps que la clause d'allocation des risques trouve application à son égard. Pour juger de cette validité, les critères relatifs aux clauses d'exclusion de responsabilité demeurent pertinents[163].

[109] Dans le même ordre d'idée, la prise en charge contractuelle des risques par le licencié permet aux donneurs de licence d'espérer lui faire assumer les pertes qu'ils ont causés. En effet, même si le contrat de licence n'a pas d'effet à l'égard des tiers, il leur est opposable[164]. Pour cette raison, le licencié pourrait être introduit dans les procédures entreprises à l'encontre des développeurs par le biais d'un appel en garantie. Cependant, cette protection n'est pas infaillible, même si le partage des risques est considéré valide, puisque l'ajout de ce nouveau débiteur ne fait pas perdre à la victime ses droits à l'égard des développeurs. La *contraignabilité* et la solvabilité du licencié n'étant pas assurées, la responsabilité civile des développeurs ne peut être écartée dans tous les cas.

Section 2 - Le dommage causé intentionnellement

[110] Les gestes des développeurs peuvent également avoir été posés avec l'intention de nuire à autrui. Par exemple, les concepteurs d'un logiciel d'administration à distance ont parfois l'objectif de permettre l'accès non-autorisé à des systèmes informatiques[165]. Ce type de comportement permet

---

[162] FREE SOFTWARE FOUNDATION, *loc. cit.*, note 73.

[163] Beth Hahn GERWIN, « Computer Related Litigation Using Tort Concepts », (1985) 9 *Am. J. Trial Advoc.* 97, 98.

[164] J.-L. BAUDOUIN et P.-G. JOBIN, *op. cit.*, note 127, no. 457, p. 377.

[165] SOURCEFORGE, « Project: Back Orifice 2000 », (1999) *SourceForge*, source: <http://sourceforge.net/projects/bo2k>.

à la victime d'invoquer à son avantage le caractère intentionnel des gestes posés. Pour y arriver, elle n'a pas nécessairement à prouver l'intention de causer un dommage. Il lui suffit de démontrer que les développeurs savaient, ou auraient dû savoir, que le dommage qu'elle a subi résulterait presque certainement des gestes posés[166]. Lorsque cette preuve est établie, même le licencié est en mesure d'impliquer la responsabilité des développeurs puisque les clauses de non-garantie, ou d'allocation des risques, sont inefficaces dans une telle situation.

[111] En droit civil, le dommage causé intentionnellement relève du même régime juridique général que la faute délictuelle. Ainsi, la détermination de la responsabilité civile doit passer une fois de plus par l'évaluation de la faute de chacun des développeurs. Toutefois, dans les juridictions de common law, ce sont les différents *intentional torts* qui trouvent application. Or, quelques-uns sont pertinents en matière de logiciels libres. Le *trespass to chattel* en est un exemple. Celui-ci prend effet lorsque « the defendant directly and intentionally interferes with a chattel in the possession of the plaintiff »[167]. La victime qui désire s'en prévaloir doit donc démontrer que le logiciel libre a détruit ou endommagé l'un des biens dont elle avait la possession. La responsabilité pour *conversion* en est un autre. Il s'agit d'un « intentional exercise of domination or control over another's chattel that substantially interferes with the other's right to control »[168]. Sous ce régime, la victime a donc la charge de prouver que le logiciel libre l'empêche de contrôler pleinement l'un de ses biens. Une telle situation survient, par exemple, lorsqu'un logiciel libre s'accapare les ressources d'un serveur distant sans autorisation. Enfin, les *intentional interferences with economic interests torts* permettent à la victime d'impliquer la responsabilité des développeurs pour la

---

[166] Vicky H. ROBBINS, « Vendor Liability for Computer Viruses and Undisclosed Disabling Devices in Software », (1993) 7 *Computer Law.* 20, 26.

[167] P. H. OSBORNE, *op. cit.*, note 144, p. 263.

[168] *Restatement (Second) of Torts*, précité, note 152, art. 222A.

conception d'un logiciel libre lui faisant perdre des droits contractuels auprès d'un tiers[169].

[112] La preuve de la volonté de causer un dommage peut aussi être facilitée par l'existence de législations pénales sur les crimes informatiques. Ainsi, au Canada, l'élaboration d'un logiciel permettant d'accéder sans autorisation à des systèmes informatiques est considérée comme un crime[170]. La situation de la victime lui permet parfois de recourir à une telle loi afin d'établir les critères de la faute ou de la négligence dans le cadre d'une action civile[171]. Celles-ci sont susceptibles d'établir les éléments de l'obligation à laquelle sont tenus les développeurs ainsi que la façon dont son inexécution peut être constatée. Dans certaines circonstances, cette technique pourrait même être utile en présence d'une faute non-intentionnelle ou de négligence. En effet, la preuve des éléments composant une négligence criminelle sera généralement suffisante pour engendrer la responsabilité civile des développeurs.

[113] L'utilité de l'établissement d'une distinction entre le régime de droit commun et celui des dommages causés intentionnellement est surtout pratique. En effet, en plus d'accorder un recours extra-contractuel au licencié, cette deuxième catégorie permet parfois d'envisager l'octroi de dommages-intérêts punitifs ou exemplaires. Ceci est particulièrement vrai en common law où cette règle est intégrée dans le droit commun[172]. Dans les juridictions civilistes, les dommages punitifs sont exceptionnels mais leur attribution demeure possible. Par exemple, en droit québécois, ils pourraient être

---

[169] Robin A. BROOKS, « Deterring the Spread of Viruses Online: Can Tort Law Tighten The 'Net'? », (1998) 17 *Rev. Litig.* 343, 367.

[170] *Code criminel*, L.R.C. (1985), c. C-46, art. 342.2.

[171] Susan C. LYMAN, « Civil Remedies for the Victims of Computer Viruses », (1992) 11 *Comp. L. J.* 607, 622.

[172] Gerald FRIDMAN, « Punitive Damages in Tort », (1970) 48 *R. du B. can.* 373.

accordés dans l'hypothèse où un logiciel libre enfreindrait intentionnellement les droits et libertés fondamentaux de la victime[173].

Section 3 - La responsabilité sans faute du fabricant

[114] Bien que la responsabilité extra-contractuelle repose essentiellement sur la commission d'actes inadéquats, le droit exempte parfois le demandeur d'en faire la preuve. Dans la majorité des juridictions, la victime d'un produit défectueux bénéficie de cet avantage vis-à-vis du fabricant l'ayant mis en circulation[174]. Il lui suffit alors de démontrer que le défaut du produit le rend dangereux ou non sécuritaire pour exiger un dédommagement. Puisque ce régime de la responsabilité s'applique malgré toute stipulation contraire, ce recours s'offre au cocontractant de la même façon qu'aux tiers. Incidemment, les développeurs ne peuvent compter sur les clauses d'allocation des risques des licences libres pour reléguer cette responsabilité au licencié. Cette lourde charge est imposée au fabricant parce qu'il est « in the best position to identify and eliminate defects »[175].

[115] Aujourd'hui, il est généralement reconnu que les logiciels sont des produits au sens des règles de la responsabilité sans faute du fabricant. Ceci s'explique en grande partie par l'objectif de protection des consommateurs qui les sous-tend et qui en exige une interprétation large[176]. De plus, la vaste majorité des logiciels ne peuvent être qualifiés de service dans la mesure où l'on définit ceux-ci comme « something which is rarely duplicated, allowing

---

[173] *Charte des droits et libertés de la personne*, L.R.Q., 1977, c. C-12, art. 49.

[174] Au Québec, *Code civil du Québec*, précité, note 132, art. 1468 ; en Europe, *Directive relative au rapprochement des dispositions législatives, réglementaires et administratives des États membres en matière de responsabilité du fait des produits défectueux*, Dir. Cons. CE 85/374, 25 juill. 1985, JOCE 7 août 1985, no. L 210, p. 29 ; aux États-Unis, *Restatement (Third) of Torts : Product Liability*, 1997, art. 1.

[175] B. H. GERWIN, *loc. cit.*, note 163, 113.

[176] M. VIVANTet autres, *op. cit.*, note 99, no. 627, p. 422.

little chance of quality control or defect testing »[177]. D'ailleurs, la distribution et la commercialisation des logiciels ressemble beaucoup plus à celle des produits matériels traditionnels qu'à celle des services[178]. Dans tous les cas, cette question est définitivement réglée en Europe où la Commission européenne a statué que la *Directive relative au rapprochement des dispositions législatives, réglementaires et administratives des États membres en matière de responsabilité du fait des produits défectueux*[179] (*Directive relative à la responsabilité des produits défectueux*) doit être appliquée aux logiciels[180]. Malgré tout, le régime de la responsabilité sans faute du fabricant a été très peu utilisé à l'endroit de l'industrie du logiciel jusqu'à maintenant. Pour cette raison, la *National Academy of Sciences* (*NAS*) a récemment émis un rapport enjoignant le gouvernement américain d'adopter une législation de ce type spécifique aux logiciels[181].

[116] L'application de la responsabilité sans faute du fabricant aux développeurs de logiciels libres implique une difficulté supplémentaire. En effet, une grande partie des programmeurs de ce milieu ne correspondent pas à l'image du fabricant traditionnel. Or, l'objectif de ce régime particulier est de protéger la victime contre le producteur qui agit à titre professionnel[182]. Pour cette raison, les programmeurs indépendants ne devraient pas s'inquiéter :

---

[177] L. Nancy BIRNBAUM, « Strict Products Liability and Computer Software », (1988) 8 *Comp. L. J.* 135, 146.

[178] Roland DESILETS, « Note, Software Vendor's Exposure to Products Liability for Computer Viruses », (1989) 9 *Comp. L. J.* 509, 524.

[179] *Directive relative au rapprochement des dispositions législatives, réglementaires et administratives des États membres en matière de responsabilité du fait des produits défectueux*, précitée, note 171, no. L210, p. 29.

[180] *Rép. min.*, 15 nov. 1988, JOCE 8 mai 1989, no. C 144, p. 42.

[181] NATIONAL ACADEMY OF SCIENCES, « Cybersecurity Today and Tomorrow: Pay Now or Pay Later », (2002) *NAS*, source : <http://books.nap.edu/html/cybersecurity>.

[182] Alexandre MENAIS, « Commentaires sur la loi du 19 mai 1998 relative à la responsabilité des produits défectueux », (1998) *Juriscom*, source : <http://www.juriscom.net/pro/1/resp19980601.htm>.

> « Ces dispositions pourraient s'appliquer au donneur de licence, que la *GPL* [ou toute autre licence libre] soit à titre gratuit ou onéreux, dès lors qu'il s'agit d'un fournisseur professionnel. Ainsi, le développeur indépendant dont la profession n'est pas de fournir des logiciels, ne devrait pas être inquiété par cette lourde responsabilité. »[183]

[117] Par ailleurs, dans le cas des fournisseurs professionnels de logiciels libres, la qualification du défaut est nécessaire. Le plus souvent, son origine importe peu puisque la victime jouit d'une protection générale, qu'il s'agisse d'un vice de conception, de réalisation ou de l'absence d'avis de sécurité[184]. Le critère déterminant se situe plutôt au niveau de la gravité du vice. Celui-ci doit être dangereux pour la personne ou les biens d'une façon telle que le grand public ne peut légitimement s'y attendre. Aussi, il est possible de distribuer un logiciel libre intrinsèquement dangereux pour autant que des avertissements suffisants permettent aux utilisateurs de prendre conscience du risque. À l'inverse, un logiciel libre relativement sécuritaire peut être défectueux si les utilisateurs ne sont pas en mesure d'évaluer correctement le danger qu'il représente. Cette qualification implique aussi que le dommage doit posséder un certain degré de matérialité. La victime ne peut donc invoquer un défaut de sécurité lorsqu'elle subit des pertes exclusivement économiques[185].

[118] Les développeurs conservent tout de même la possibilité de s'exonérer en prouvant que le défaut ne pouvait être connu, compte tenu des connaissances scientifiques et de l'expertise technologique disponible au moment où le logiciel a été conçu et distribué. Pour y arriver, ils devront généralement démontrer que le vice n'a pas été trouvé, et n'aurait pu l'être,

---

[183] M. CLÉMENT-FONTAINE, *loc. cit.*, note 85, par. 55.

[184] Patrick T. MIYAKI, « Computer Software Defects: Should Computer Software Manufacturers Be Held Strictly Liable for Computer Software Defects? », (1992) 8 *Santa Clara Computer & High Tech. L.J.* 121, 128.

[185] John M. CONLEY, « Tort Theories of Recovery Against Vendors of Defective Software », (1987) 13 *Rutgers Comp. & Tech. L. J.* 1, 28.

malgré le recours à des procédures adéquates pour tester le programme informatique[186]. Ainsi, des développeurs professionnels se fiant uniquement à l'apport de la communauté pour tester leur logiciel libre ne pourraient bénéficier de ce moyen de défense dans l'hypothèse où un groupe de contrôle de la qualité traditionnel aurait été en mesure de dépister le défaut. Cette situation est particulièrement susceptible de s'appliquer aux logiciels libres attirant un faible nombre de contributeurs.

Section 4 - Les responsabilités statutaires

[119] En plus du régime de droit commun, de nombreuses lois peuvent contenir des dispositions permettant d'impliquer la responsabilité civile des développeurs de logiciels. La plupart de ces lois visent à protéger des intérêts spécifiques dans des champs d'activité où un groupe de personnes se trouve en situation d'infériorité. Les développeurs doivent donc éviter d'enfreindre les normes qui prévalent dans les domaines auxquels se destinent leurs logiciels libres spécialisés. Par exemple, dans le domaine bancaire, différentes législations assurant la protection des données personnelles doivent être respectées[187]

[120] Cependant, les logiciels étant avant tout des créations de l'esprit, ce sont les recours fondés sur les lois de protection de la propriété intellectuelle qui ont le plus de chance d'être invoqués à l'encontre des développeurs de logiciels libres. Les législations sur les droits d'auteur, les brevets et les marques de commerce permettent toutes au titulaire d'un droit ayant été violé d'exiger des dommages-intérêts de la part de ceux qui ont commis l'infraction.

---

[186] L. N. BIRNBAUM, *loc. cit.*, note 177, 154.

[187] Aux Etats-Unis, *Gramm-Leach-Bliley Act* (2001) 15 U.S.C., s. 6801 ; en Europe, *Directive relative à la protection des personnes physiques à l'égard du traitement des données à caractère personnel et à la libre circulation de ces données*, Dir. Cons. CE 95/46, 24 oct. 1995, JOCE 23 novembre 1995, no. L291, p. 31.

En ce qui concerne les droits d'auteur, les développeurs sont responsables du préjudice causé dès lors qu'ils effectuent « tout acte dévolu exclusivement aux titulaires »[188]. La réutilisation de code source protégé est particulièrement susceptible d'entraîner cette conséquence. Dans plusieurs juridictions, ils peuvent également être tenus responsables pour la violation commise par un utilisateur, dans la mesure où leur logiciel y contribue, l'incite ou la cause. Le *Digital Millennium Copyright Act* (*DMCA*) américain va encore plus loin en interdisant de contourner toute mesure technologique visant à contrôler l'accès à une oeuvre[189]. Néanmoins, une utilisation équitable des oeuvres protégées est généralement permise[190]. En ce qui concerne les brevets, les risques demeurent réduits puisque « la plupart des législations en matière de brevet prévoient une exclusion formelle de la *brevetabilité* des programmes d'ordinateurs »[191]. Certains pays, dont les États-Unis et le Japon, acceptent tout de même de breveter les logiciels. La même règle pourrait bientôt s'appliquer à l'Europe où un débat fait rage à ce sujet[192]. Les développeurs résidant dans ces pays engagent donc leur responsabilité lorsqu'ils s'inspirent d'un mécanisme informatique protégé afin de concevoir un logiciel libre. Enfin, lorsque la désignation d'un logiciel porte à confusion avec une marque de commerce d'un logiciel concurrent, ses développeurs sont responsables des pertes qui en découlent.

[121] Finalement, des dispositions affectant la responsabilité civile des développeurs de logiciels libres peuvent aussi se retrouver dans les lois

---

[188] P. TRUDEL et autres, *op. cit.*, note 62, p. 16-70.

[189] *Digital Millennium Copyright Act*, (1998) 112 Stat. 2860, sec. 1201.

[190] Mark E. HARRINGTON, « On-line Copyright Infringement Liability for Internet Service Providers : Context, Cases & Recently Enacted Legislation », (1999) *B. C. Intell. Prop. & Tech. F.* 060499.

[191] Pierre BREESE, « Brevetabilité des logiciels », (2001) *Breese Majerowicz*, source : <http://www.breese.fr/guide/htm/Logiciel/main1.htm>.

[192] COMMISSION EUROPÉENNE, « Brevetabilité des inventions mises en oeuvre par ordinateur », (2002) *Europa.eu.int*, source : <http://europa.eu.int/comm/internal_market/fr/indprop/comp/index.htm>.

d'ordre criminel. En effet, les victimes d'actes criminels sont parfois en mesure d'obtenir une compensation pour leur préjudice sans avoir à entamer de poursuite civile[193]. Les recours de ce type peuvent, entre autres, avoir une utilité face aux développeurs de virus informatiques[194].

* * *

[122] La responsabilité extra-contractuelle des développeurs de logiciels libres peut donc être impliquée de différentes façons. Comme toute autre personne, ils sont tenu de réparer le préjudice causé par leur faute ou leur négligence. Il est vrai que, dans bien des cas, les clauses d'allocation des risques des licences libres leur permettent de reléguer l'obligation de dédommagement au licencié. Cependant, ces clauses ne sont d'aucune utilité lorsque le licencié est insolvable ou que les gestes ont été posés avec l'intention de causer un dommage. Cela est également vrai lorsque les développeurs agissent à titre professionnel et que leur logiciel est grevé d'un défaut de sécurité. Dans ce cas, ils peuvent même être tenus responsables malgré une absence de faute ou de négligence de leur part. Enfin, le non-respect de dispositions législatives peut également les obliger envers un tiers. Les éléments extra-contractuels représentent donc un risque potentiel pour la responsabilité civile des développeurs de logiciels libres. Un dédommagement pouvant être exigé d'eux, il est essentiel d'évaluer comment la responsabilité civile peut être appliquée à leur situation particulière.

---

[193] Au Canada, *Loi sur l'indemnisation des victimes d'actes criminels*, L.R.Q., 1977, c. I-6, art. 8.

[194] S. C. LYMAN, *loc. cit.*, note 171, 623.

PARTIE III - Application de la responsabilité civile

Chapitre 1 - Les problématiques liées à la *contraignabilité* des développeurs

[123] Puisque la responsabilité civile des développeurs d'un logiciel libre peut être établie, la question de son application doit nécessairement être abordée. Or, il s'avère que la méthode de développement distribuée propre aux logiciels libres entraîne un certain nombre de difficultés à cet égard. D'abord, l'absence de structure organisationnelle formelle qui caractérise l'élaboration des logiciels libres rend l'exercice des droits des personnes lésées plus difficile. Ainsi, lorsque la *traçabilité* du code n'a pas été assurée, l'identification complète des auteurs du logiciel peut être impossible à réaliser. De plus, même lorsque tous les responsables peuvent être identifiés, leur répartition géographique est susceptible de poser de sérieux problèmes à celui qui cherche à établir leur responsabilité civile.

Section 1 - Les difficultés d'identification

[124] Bien que la vaste majorité des développeurs de logiciels libres considèrent qu'il est important d'identifier adéquatement la paternité du code source[195], il est toujours possible de recenser de nombreux cas de code libre non crédité[196]. Ceci s'explique en grande partie par la taille souvent impressionnante du bassin de contributeurs des projets libres et par la rapidité avec laquelle les modifications du code se succèdent. Dans de telles conditions, l'historique de l'ensemble des changements effectués n'est pas

[195] INTERNATIONAL INSTITUTE OF INFONOMICS, *loc. cit.*, note 52.

[196] Rishab Aiyer GHOSH, « Clustering and Dependencies in Free/Open Source Software Development : Methodology and Preliminary Analysis », (2002) *Institut d'économie industrielle*, p. 11, source : <http://www.idei.asso.fr/Commun/Conferences/Internet/OSS2002/Papiers/Ghosh.PDF>.

toujours conservé et l'auteur d'une partie non négligeable du logiciel peut demeurer introuvable. Même lorsqu'un mécanisme assure la *traçabilité* du code, les méthodes de travail des développeurs peuvent engendrer des incertitudes quant à l'auteur réel d'une partie de celui-ci. Par exemple, les développeurs du serveur Web *Apache* n'indiquent pas leurs noms individuellement lorsqu'ils effectuent des modifications au logiciel. D'un point de vue juridique, cet état de fait est tout aussi néfaste pour les personnes ayant subi un dommage que pour les développeurs. En effet, les premières risque de se retrouver dans l'incapacité d'identifier le responsable de leur préjudice et les seconds de perdre les bénéfices liés à l'existence d'un co-responsable.

[125] Pourtant, en théorie, ces pratiques n'ont pas de raison d'être, du moins en ce qui concerne les logiciels soumis à la *GPL*. En effet, l'article 2.a) de cette licence précise que lors de la redistribution d'un logiciel « You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change »[197]. Une application généralisée de cet article permettrait sans aucun doute d'assurer la *traçabilité* du code. Cependant, beaucoup de développeurs considèrent cet article mal adapté à leur méthode de développement et ne s'y soumettent pas. Selon eux, le fait d'insérer ces informations dans un fichier à chaque fois qu'une rustine lui est appliquée l'alourdit inutilement et implique beaucoup de travail supplémentaire.

[126] La vaste majorité des auteurs peuvent tout de même être identifiés par le biais des fichiers journaux de modification ou des systèmes de contrôle de versions des projets. Ces outils sont utilisés afin de suivre l'évolution du développement des logiciels. Les fichiers journaux de modifications sont de simples fichiers textes où les changements apportés aux différentes versions

---

[197] FREE SOFTWARE FOUNDATION, *loc. cit.*, note 73, art. 2.a).

du logiciel sont notés. De leur côté, les systèmes de contrôle de versions, tel que *Concurrent Versions System*[198] (*CVS*), sont des logiciels qui gèrent l'accès des développeurs au code source et enregistrent toutes les modifications y étant apportées. Ils facilitent le développement distribué du logiciel et, ce faisant, accumulent des informations permettant l'identification des auteurs du code.

[127] Malgré tout, ces outils ne garantissent pas la *traçabilité* totale du code source. Ainsi, lorsque le code d'un projet est intégré à l'intérieur d'un autre, il est possible que l'historique du projet original disparaisse. De plus, lorsque des développeurs ayant accès au code officiel d'un logiciel appliquent des rustines fournies par des développeurs étrangers au projet, l'identité de ces derniers risque de ne pas être mentionnée[199]. Finalement, si les auteurs sont en mesure de conserver l'anonymat, la possibilité de retracer l'origine du code ne sera parfois d'aucune utilité.

[128] Pour toutes ces raisons, d'autres mécanismes d'identification devraient idéalement être mis de l'avant par les développeurs de logiciels libres. Par exemple, une vérification d'identité peut être effectuée avant l'octroi d'un accès au code officiel du logiciel. Les développeurs de la distribution *Gnu/Linux Debian* agissent déjà de cette façon[200]. Certains proposent même de chiffrer et de certifier la version officielle des logiciels libres afin d'en assurer l'authenticité[201]. À tout le moins, toutes les rustines appliquées

---

[198] CONCURRENT VERSIONS SYSTEM, « Concurrent Versions System : The Open Standard for Version Control », (2002) *CVSHome*, source : <http://www.cvshome.org>.

[199] RAPHAEL, « The GPL, the Contributors, the ChangeLog and CVS », (2002) *Advogado*, source : <http://www.advogado.org/article/183.html>.

[200] Adam DI CARLO, Christian SCHWARZ et Ian JACKSON, « Debian Developer's Reference Chapter 2 - Applying to Become a Maintainer », (2002) *Debian*, source : <http://www.debian.org/doc/packaging-manuals/developers-reference/ch-new-maintainer.en.html>.

[201] Bruce PERENS, « The Trojan Horse », (1998) *Lwn.net*, source : <http://old.lwn.net/1998/1119/Trojan.html>.

devraient être conservées et les développeurs devraient s'assurer que leurs fichiers journaux de modification et leurs systèmes de contrôle de versions soient exhaustifs.

Section 2 - La répartition géographique

[129] Le modèle de développement distribué des logiciels libres étant basé sur Internet, les différents développeurs d'un même projet se trouvent souvent répartis aux quatre coins du globe, bien que la majorité d'entre eux soient généralement regroupés en Amérique du Nord et en Europe[202]. Dans l'hypothèse d'un litige les impliquant, l'inévitable caractère international du débat entraîne un certain nombre de difficultés au niveau juridique. Il s'agit d'ailleurs de problèmes récurrents en ce qui concerne les interactions dans les environnements électroniques[203]. Dans un premier temps, la juridiction compétente pour entendre les parties doit être déterminée. Ensuite, se pose la question de l'exécution internationale du jugement rendu.

[130] Relativement à la juridiction compétente,

> « International law, as it has developed from being part of the conflict of laws to its present standing, has not fusnished the national legislature with a comprehensive set of norms defining, with reasonable precision, the permissible limits to State jurisdiction. »[204]

[131] Aussi, les normes applicables en la matière ont été élaborées de façon distincte d'une juridiction à l'autre, avec toutes les disparités que cela

---

[202] TECHNICAL UNIVERSITY OF BERLIN, « Who Is Doing It? Knowing More About Developers », (2001) *WIDI*, source : <http://widi.berlios.de> ; INTERNATIONAL INSTITUTE OF INFONOMICS, *loc. cit.*, note 52.

[203] P. TRUDEL et autres, *op. cit.*, note 62, p. 4-29.

[204] Tapio PUURUNEN, « The Legislative Jurisdiction of State over Transactions in International Electronic Commerce », (2000) 18 *J. Marshall J. Computer & Info. L.* 689, 699.

implique. Il est toutefois reconnu qu'une personne ne peut être poursuivie dans un forum sans qu'il y ait un facteur de rattachement quelconque entre celui-ci et l'affaire dont il est question[205]. Or, puisque différents facteurs de rattachement peuvent relier un litige à des forums distincts et qu'il n'existe pas de règle de préférence universellement reconnue entre eux, les juridictions nationales ne sont pas exclusives[206]. Il en découle que certains litiges sont susceptibles d'être portés devant plus d'une juridiction, le demandeur devant alors choisir celle qu'il privilégie. Il est fort probable que la personne ayant été lésée par les développeurs d'un logiciel libre se retrouve dans une telle situation.

[132] La première catégorie de facteurs de rattachement potentiellement utiles a trait aux défendeurs. Les tribunaux soumis à une tradition civiliste se considèrent généralement justifiés d'intervenir lorsque le domicile du défendeur se trouve sur leur territoire[207]. La situation est plus complexe lorsque la common law doit être appliquée, puisqu'une compétence sur l'objet du litige doit s'ajouter à celle sur la personne[208]. Par ailleurs, le lien entre le défendeur et le forum n'est pas toujours limité au lieu de son domicile. Par exemple, les juges américains se considèrent compétents dès lors que le défendeur a des contacts minima suffisant avec les États-Unis[209]. Toutefois, la détermination de la juridiction sur les défendeurs étant un processus individuel, il est possible qu'un tribunal soit compétent face à un défendeur, mais non vis-à-vis des autres. Cette situation tend à protéger les

---

[205] Simon JOHNSON, « Internet Activity and Jurisdiction over Foreign Defendants », (1999) *Gary Dunn Computer & Technology Law*, source : <http://www.dunn.com/papers/paper_3.shtml>.

[206] Brigitte STERN, « Quelques observations sur les règles internationales relatives a l'application extraterritoriale du droit » (1986) 32 *A.F.D.I.* 34, 43.

[207] Friedrich JUENGER, « Judicial Jurisdiction in the United States and in the European Communities : A Comparison », (1984) 82 *Mich. L. Rev.* 1195, 1203.

[208] Peter S. SELVIN, « Personal Jurisdiction Over Non-U.S. Defendants - Recent Developments », (2002) 670 *PLI/Lit* 11, 13.

[209] *World-Wide Volkswagen Corp.* c. *Woodson*, (1980) 444 U.S. 286.

développeurs de logiciels libres en forçant le demandeur à mener des procédures parallèles dans plusieurs juridictions distinctes. Cette solution étant fort onéreuse, le demandeur a donc avantage à invoquer l'existence d'autres facteurs de rattachement.

[133] Lorsque la relation entre les parties est régie par une licence, le contexte contractuel peut aussi fournir des facteurs de rattachement. Tout d'abord, « contract law, since it is enforced by agreement, trancends the problem of national borders »[210]. Ainsi, les tribunaux se soumettent aisément aux dispositions contractuelles permettant d'identifier le forum compétent. Quelques licences libres comportent des clauses de ce type. Ainsi, la clause 11 de la *Mozilla Public licence* (*MPL*) précise : « This License shall be governed by California law provisions (except to the extent applicable law, if any, provides otherwise), excluding its conflict-of-law provisions »[211]. En l'absence de précisions, les tribunaux se considèrent généralement compétents si les obligations découlant d'un contrat doivent être exécutées sur leur territoire[212]. Or, lorsque la licence d'un logiciel libre est impliquée, il peut s'avérer difficile de déterminer le lieu d'exécution des obligations. Cela requiert « fictitiously localizing the place of performance somewhere, perhaps at the residence of the person who is to receive the information »[213]. Cette solution est compréhensible dans la mesure où l'obligation principale du donneur de licence consiste à délivrer le logiciel libre au licencié[214]. Une telle

---

[210] Robert DUNNE, « Deterring Unauthorized Access to Computer : Controling Behavior in Cyberspace Through a Contract Law Paradigm », (1994) 35 *Jurimetrics J.* 1, 10.

[211] MOZILLA.ORG, « Mozilla Public License version 1.1 », (2002) *Mozilla*, art. 11, source : <http://www.mozilla.org/MPL/MPL-1.1.html>.

[212] F. JUENGER, *loc. cit.*, note 207.

[213] Jeffrey TALPIS, *If I am from Grand-Mère, Why Am I Being Sued in Texas? : Responding to Inappropriate Foreign Jurisdiction in Quebec - United States Crossborder Litigations*, Montréal, Éditions Thémis, 2001, p. 28.

[214] Thibault VERBIEST, « Droit international privé et commerce électronique : état des lieux », (2001) *Juriscom*, par. 6, source : <http://www.juriscom.net/pro/2/ce20010213.htm>.

façon de procéder favorise nettement ce dernier en lui permettant de prendre action dans sa propre juridiction.

[134] De façon similaire, en matière extra-contractuelle, les tribunaux reconnaissent parfois leur compétence si les gestes posés ou le dommage se sont produits sur leur territoire[215]. Bien que la localisation du dommage ne pose pas de difficultés, les gestes dommageables des développeurs d'un logiciel libre sont plus difficiles à situer. En effet, les décisions relatives à l'élaboration du logiciel sont généralement prises de façon distribuées à travers de nombreuses juridictions. Aussi, tout comme dans certaines affaires relatives au commerce électronique, l'emplacement du serveur rendant le logiciel accessible pourrait servir à localiser les gestes posés[216]. Cette solution n'est toutefois pas sans reproches puisqu'en l'absence d'autres facteurs de rattachement un serveur localisé à l'étranger permettrait aux développeurs d'échapper aux conséquences légales de leurs actes. Elle demeure tout de même pertinente lorsqu'une connexion substantielle peut être établie entre le développement du logiciel libre et l'endroit d'où il est diffusé[217].

[135] Enfin, les tribunaux américains interprètent maintenant leur juridiction de façon à s'octroyer compétence à partir du moment où les agissements en ligne des défendeurs ont causés des effets suffisamment importants sur leur territoire[218]. Bien entendu, le recours aux effets comme facteur de rattachement nécessite la prise en compte de différents critères :

- nature du contact électronique ;

---

[215] Kevin M. CLERMONT, « Jurisdictional Salvation and the Hague Treaty », (1999) 85 *Cornell L. Rev.* 89, 91.

[216] Au Canada, *Investors Group Inc.* c. *Hudson*, [1999] R.J.Q. 599 ; en France, TGI Paris, 3e ch., 12 févr. 1999, Chaumet, Juris-data no. 041421.

[217] J. TALPIS, *op. cit.*, note 213, p. 29.

[218] *Zippo Mfg. Co.* c. *Zippo Dot Com Inc.*, (1997) 952 F. Supp. 1119.

- nature du litige ;
- niveau de passivité ou d'interractivité du médium ;
- existence d'autres contacts avec le forum[219].

[136] Avec la multiplication des conflits d'envergure internationale engendrés par l'utilisation d'Internet, cette forme d'interprétation extensive de la juridiction pourrait se généraliser. Encore une fois, la personne qui cherche à établir la responsabilité civile des développeurs d'un logiciel libre y gagne la possibilité d'entamer des procédures dans sa propre juridiction.

[137] Malgré tout, l'avantage de choisir parmi les juridictions compétentes dont bénéficie le demandeur n'est pas illimité. En effet, les tribunaux des juridictions de common law, par le biais de la théorie du *forum non conveniens*, procèdent à une analyse de la *raisonnabilité* du choix du forum. Dans l'hypothèse où un autre forum est mieux placé pour entendre le litige, le juge peut refuser de procéder, même si sa compétence est établie[220]. Bien que les tribunaux civilistes donnent traditionnellement préséance au premier forum choisi, la théorie du *forum non conveniens* tend à être graduellement acceptée par ces derniers[221]. Aussi, le demandeur devrait toujours tenter de choisir la juridiction la plus adéquate, par exemple celle où réside la majorité des développeurs du logiciel libre.

[138] Une forte concentration de développeurs de logiciels libres étant domiciliés aux États-Unis, il est probable que cette juridiction soit souvent la plus appropriée. Il est vrai que les questions de compétence peuvent soulever des difficultés à l'intérieur même des États-Unis, vu la structure

---

[219] Jose ROJAS, « Avoiding Entanglement in a Jurisdictional Web », (2000) 590 *PLI/Pat* 11, 17.

[220] Ronald A. BRAND, « Comparative Forum Non Conveniens and the Hague Convention on Jurisdiction and Judgments », (2002) 37 *Tex. Int'l L. J.* 467, 468.

[221] Au Québec, *Code civil du Québec*, précité, note 131, art. 3135 ; en Allemagne, OLG Nurnberg IPRspr. (1961) *AWD* 18 ; au Japon, *Mukoda et al.* c. *Boeing Co.*, (1986) 604 Hanrei taimuzu 138.

fédérale de ce pays. Néanmoins, l'interprétation extensive appliquée par les tribunaux américains tend à assurer leur compétence. Les avantages dont bénéficie le demandeur en droit américain doivent également être pris en compte : pressions financières sur le défendeur, règles de divulgation de la preuve intrusives, méthodes avantageuses de compensation des avocats, possibilité d'obtenir des dommages punitifs, droit de recours à un jury, etc[222]. Pour toutes ces raisons, il est permis de présumer que la plupart des litiges impliquant la responsabilité civile des développeurs de logiciels libres se dérouleront aux États-Unis.

[139] Par ailleurs, l'obtention d'un jugement ne résout pas toutes les difficultés puisque le demandeur doit encore être en mesure de le faire exécuter contre un nombre suffisant de développeurs. Or, l'exécution des jugements à l'étranger est, elle aussi, soumise aux règles nationales de chaque État[223]. En conséquence, une forme ou une autre d'homologation doit être obtenue à l'intérieur de chaque juridiction où réside un développeur du logiciel concerné. Le plus souvent, cette procédure est accompagnée d'une évaluation plus ou moins élaborée du jugement ayant été rendu. Les critères d'évaluation les plus communs comprennent la vérification de la juridiction du tribunal original et le respect des règles minimales de justice et de procédure. Le respect des intérêts nationaux et de l'ordre public local est un autre critère pratiquement universel. Ainsi, un tribunal américain a récemment refusé de reconnaître une ordonnance française enjoignant à *Yahoo! Inc.* de cesser la vente d'articles nazis sur son site Web parce que celle-ci était contraire aux principes de la Constitution des États-Unis[224]. Les règles de certaines

---

[222] Ugo MATTEI et Jeffrey LENA, « U.S. Jurisdiction Over Conflicts Arising Outside of the United States : Some Hegemonic Implications », (2001) 24 *Hastings Int'l & Comp. L. Rev.* 381, 393.

[223] CENTER FOR INTERNATIONAL LEGAL STUDIES, *International Execution Against Judgment Debtors*, Oceana Publications Inc., Dobbs Ferry, 1998.

[224] *Yahoo! Inc.* c. *La ligue contre le racisme et l'antisemitisme*, (2001) 169 F. Supp. 2D 1181, 1192.

juridictions sont encore plus sévères et exigent que le raisonnement et les conclusions du jugement soient entièrement conformes au droit national[225]. Toutefois, puisque qu'un jugement impliquant la responsabilité civile des développeurs d'un logiciel libre est nécessairement de nature privée et que ses conclusions se limiteront le plus souvent à l'octroi d'une somme d'argent, sa reconnaissance à l'étranger devrait poser peu de problèmes. Par contre, certains de ses éléments, tels que les dommages punitifs obtenus en vertu du droit américain, pourraient s'avérer difficiles à faire appliquer à l'échelle internationale. Il n'en demeure pas moins que certains développeurs domiciliés là où les critères de reconnaissance des jugements sont plus sévères pourraient échapper à son exécution.

[140] Il existe également quelques documents internationaux permettant de faciliter la détermination de la juridiction compétente et l'exécution des jugements à l'étranger. Le plus important est sans conteste la *Convention de Bruxelles concernant la compétence judiciaire et l'exécution des décisions en matière civile et commerciale*[226] (*Convention de Bruxelles*). Celle-ci s'applique aux États membres de l'Union européenne et fixe des règles juridictionnelles précises. Elle accorde, par exemple, une juridiction contre tous les co-défendeurs dès lors que la compétence du tribunal peut être établie contre l'un d'entre eux[227]. Cette disposition est évidemment fort utile à celui qui recherche la responsabilité civile de développeurs répartis dans toute l'Europe. De plus, la *Convention de Bruxelles* assure au demandeur que le jugement obtenu dans l'un des pays signataires sera reconnu par les autres. Un traité international similaire est d'ailleurs en cours d'élaboration dans le cadre de la *Conférence de La Haye de droit international privé* et prend actuellement la forme de l'*Avant-projet de convention sur la*

[225] Ronald A. BRAND, *Enforcing Foreign Judgments in the United States and United States Judgments Abroad*, Chigago, American Bar Association, 1992.

[226] *Convention du 27 septembre 1968 concernant la compétence judiciaire et l'exécution des décisions en matière civile et commerciale*, (1990) O.J. (C 189) 2.

[227] *Id.*, art. 3.

compétence et les jugements étrangers en matière civile et commerciale[228] (*Convention de La Haye*). L'un des objectifs de ce document est de faciliter le règlement des conflits résultant des interactions sur Internet[229]. Il pourrait donc constituer un outil de taille pour les personnes ayant été lésées par les développeurs d'un logiciel libre. Cependant, il n'est pas certain que le processus de négociation aboutisse rapidement, vu l'écart existant entre les positions américaines et européennes à ce sujet. Quelques autres traités bilatéraux en matière d'exécution des jugements, tel que celui entre le Canada et le Royaume-Uni[230], permettent d'apporter des solutions partielles supplémentaires à cette problématique.

* * *

[141] La *contraignabilité* des développeurs de logiciels libres n'est donc pas un problème aussi absolu qu'il ne le paraît. Il est vrai que certains responsables se trouveront parfois hors de portée de la justice, soit parce qu'ils ne peuvent être retracés, soit parce qu'ils résident dans un État interprétant sa juridiction de façon restrictive ou acceptant difficilement la reconnaissance des jugements étrangers. Toutefois, les personnes ayant subi un dommage ne sont pas sans recours pour autant. Premièrement, les outils de développement des logiciels libres permettent d'identifier la vaste majorité des auteurs d'un logiciel. Deuxièmement, différents mécanismes visant à améliorer la gestion des litiges à l'échelle internationale existent déjà, qu'il s'agisse de l'interprétation extensive de la juridiction ou de traités entre

---

[228] *Avant-projet de convention sur la compétence et les jugements étrangers en matière civile et commerciale*, 1999, Source : <http://www.hcch.net/f/conventions/draft36f.html>.

[229] Mary SHANNON MARTIN, « Keep it Online : The Hague Convention and the Need for Online Alternative Dispute Resolution in International Business-To-Consumer E-Commerce », (2002) 20 *B.U. Int'l L. J.* 125, 127.

[230] *Convention entre le Canada et le Royaume-Uni de Grande-Bretagne et d'Irlande du Nord pour assurer la reconnaissance et l'exécution réciproques des jugements en matière civile et commerciale,* voir annexe de la *Loi sur la Convention Canada-Royaume-Uni relative aux jugements en matière civile et commerciale*, L.R.C. (1985), c. C-30.

les États. Par ailleurs, d'autres difficultés découlent de l'application de la responsabilité civile aux développeurs de logiciels libres, particulièrement en ce qui a trait à l'établissement du lien de causalité.

Chapitre 2 - La problématique de la causalité

[142] Une deuxième source de difficultés découlant de l'application de la responsabilité civile aux développeurs de logiciels libres se rattache à l'établissement du lien causal qui doit nécessairement exister entre les agissements des développeurs et le dommage. Il s'agit de définir les critères permettant de déterminer, parmi l'ensemble des développeurs ayant omis d'exécuter des obligations contractuelles ou extra-contractuelles, ceux dont les gestes ont légalement causé le préjudice. Ce problème s'avère complexe à résoudre, d'abord à cause de la nature même des logiciels libres, ensuite à cause de l'incapacité du droit à expliquer efficacement le fonctionnement de la causalité.

Section 1 - L'énigme de la causalité

[143] L'établissement d'un lien de causalité est l'un des éléments essentiels de l'attribution de la responsabilité civile. Ainsi, peu importe le système juridique, lorsque le droit exige la commission d'un acte répréhensible et l'existence d'un dommage comme conditions d'une indemnisation, un lien doit pouvoir être établi entre les deux. Il est vrai que les analyses juridiques de la causalité sont souvent superficielles, cette dernière pouvant généralement être déduite implicitement de la situation factuelle[231]. Cependant, lorsque de multiples évènements s'enchaînent et qu'il en résulte un préjudice, une telle déduction est plus difficile à réaliser. Il devient alors nécessaire de

[231] J.-L. BAUDOUIN et P. DESLAURIERS, *op. cit.*, note 144, no. 512, p. 343.

déterminer « À quelle condition un antécédent d'un dommage doit en être considéré comme une cause génératrice »[232]. Il s'agit donc de faire un tri parmi les évènements afin de ne retenir que ceux qui constituent une cause efficiente. Plusieurs causes de ce type pouvant coexister, des critères relatifs au partage de la responsabilité entre elles sont également nécessaires.

[144] Ce processus revêt une importance particulière ce qui concerne l'attribution de la responsabilité civile aux développeurs de logiciels libres. Ceci s'explique principalement par le caractère distribué de l'élaboration du logiciel. En effet, le préjudice sera généralement précédé d'une multitude d'interactions entre les apports respectifs des nombreux développeurs indépendants. Dans une telle situation, le dommage est inévitablement la « résultante d'une série de causes, qui se suivent et se conditionnent plus ou moins »[233]. Dans la mesure où la responsabilité peut être attachée aux agissements de plus d'un développeur, il devient nécessaire d'identifier ceux qui ont légalement causé les pertes pour ensuite partager cette responsabilité entre eux.

[145] Le problème de la causalité se pose d'abord en matière extra-contractuelle. Ainsi, plusieurs développeurs peuvent avoir posé des gestes répréhensibles susceptibles d'engager leur responsabilité civile. Pourtant, tous ces agissements n'ont pas nécessairement causé le préjudice. Par exemple, dans l'hypothèse où l'ajout d'une fonctionnalité est jugée fautive, il n'est pas certain que le développeur l'ayant implémentée dans un module du logiciel doit être tenu d'indemniser la victime dont les pertes résultent de l'appel à la même fonctionnalité ayant été intégrée par la suite dans un module différent par un second développeur. Le même raisonnement

---

[232] François TERRÉ, Philippe SIMLER et Yves LEQUETTE, *Droit civil - Les obligations*, Éditions Dalloz, Paris, 1999, no. 819, p. 756.

[233] *Id.*, p. 754.

s'applique au non-respect d'une disposition légale spécifique, sous réserve des précisions contenues dans le texte législatif.

[146] L'établissement du lien de causalité en matière contractuelle est tout aussi important. Il est vrai que ce sujet suscite traditionnellement moins de discussion. Cette rareté de la doctrine s'explique peut-être par le fait que, en matière de rupture de contrat, les problèmes de causalité sont assez simples par rapport à ceux concernant l'étendue de l'obligation de dédommagement[234]. Par contre, la situation diffère lorsqu'un logiciel libre est impliqué. En effet, le licencié ayant contracté avec plusieurs donneurs de licence, chacun d'eux étant uniquement responsable de son apport personnel, la cause du préjudice peut provenir de multiples sources. Il en résulte que, parmi tous les développeurs ayant omis d'exécuter une obligation contractuelle, seuls ceux dont les gestes ont causé le dommage doivent être tenus responsables.

[147] La nécessité de rechercher les causes efficientes implique également que les développeurs ne sont pas les seuls responsables potentiels. En effet, d'autres personnes peuvent avoir posées des gestes susceptibles d'engager leur responsabilité civile. C'est le cas de la victime qui peut avoir été négligente dans son utilisation du logiciel ou y avoir introduit des données inacceptables.[235] Elle peut aussi s'être engagée dans une activité risquée en toute connaissance de cause[236]. Dans la même optique, un utilisateur peut contribuer à causer des pertes à un tiers en faisant fonctionner le logiciel de façon inadéquate. Aussi, l'ensemble de ces gestes doit être prise en compte lors de la détermination des causes..

---

[234] Herbert HART et Tony HONORÉ, *Causation in the Law*, 2e Éd., Clarendon Press, Oxford, 1985, p. 277.

[235] Viki MCBROOM, « Civil Liability for Software Flaws », (1989) 09 *International Comp. L. Ad.* 15, 17.

[236] J.-L. BAUDOUIN et P. DESLAURIERS, *op. cit.*, note 144, no. 533, p. 362.

[148] Malheureusement, malgré l'importance des problèmes posés par l'utilisation de la notion de causalité dans le domaine juridique, aucune solution globale ne s'offre pour résoudre l'ensemble des difficultés qui en découlent. Plusieurs auteurs de doctrine mettent même en doute la possibilité de définir adéquatement le lien de causalité. Pour cette raison, « Le problème de la causalité est considéré comme une 'énigme de notre droit' »[237]. Il n'en demeure pas moins que des tentatives d'explication théorique ont été avancées et qu'une doctrine abondante aborde cette problématique[238]. Aucune de ces théories n'étant parfaite, les tribunaux hésitent à adopter l'une plutôt que l'autre et préfèrent choisir celle qui leur paraît le mieux adaptée à chaque cas particulier[239]. En conséquence, toutes les théories existantes sont susceptibles d'être utilisées lors de l'évaluation de la causalité des gestes posés par les développeurs de logiciels libres.

Section 2 - Les théories de la causalité

[149] La première théorie de la causalité est dite de l'équivalence des conditions. Celle-ci consiste à rechercher les évènements qui ont constitué une condition indispensable , *sine qua non*, du dommage : « would the harm have occured but for the defendant's conduct? »[240]. Selon cette conception, la responsabilité devrait s'attacher à tous les actes reprochables sans lesquels le préjudice ne serait pas survenu. Le principal avantage de cette théorie est qu'elle facilite, pour le défendeur, la démonstration de l'inexistence du lien causal dans le cas précis où les pertes auraient été subies même

---

[237] Alain BÉNABENT, *Droit civil - Les obligations*, 6e Éd., Montchrestien, Paris, 1997, no. 556, p. 333.

[238] Tony HONORÉ, « Causation and Remoteness of Damages », dans *International Encyclopedia of Comparative Law*, Tübingen, Mohr, 1984, vol. XI, chap. 7, no. 45.

[239] Arnaud MEYRAN, « La notion de causalité dans le droit de la responsabilité civile », (2002) *Net-Iris*, source : <http://www.net-iris.com/publication/author/document.php3?document=64>.

[240] Jane STAPLETON, « Law, Causation, and Common Sense », (1988) 8 *Oxford J. Legal Studies* 111, 114.

sans son intervention[241]. Pour cette raison, les juridictions de common law ont largement recours à la théorie de l'équivalence des conditions[242]. Elle est toutefois rejetée par la vaste majorité des auteurs civilistes parce qu'elle « étend beaucoup trop loin les conséquences de la responsabilité »[243]. En effet, elle tend à retenir une multitude de faits générateurs, même lorsque ceux-ci n'ont qu'une relation lointaine avec le dommage.

[150] Appliquée au contexte des logiciels libres, la théorie de l'équivalence des conditions implique que tous les développeurs dont l'apport a contribué à la réalisation des pertes sont responsables, même si leur implication dans le projet est minime. Ainsi, le développeur ayant simplement contribué à régler un bogue peut se trouver dans la même position que l'auteur de la plus grande partie du logiciel. Il s'agit d'un avantage certain pour le demandeur, puisqu'il est alors en mesure d'impliquer la responsabilité d'un nombre important de développeurs. Par contre, cette situation s'avère inéquitable pour les développeurs ayant peu participé au projet mais dont la contribution, bien qu'éloignée du dommage, constitue un élément essentiel de la chaîne d'évènements l'ayant causé. Enfin, le recours à cette théorie complexifie indûment l'identification des responsables, puisqu'il devient nécessaire de remonter très loin dans le processus de conception du logiciel.

[151] Une deuxième théorie, celle de la proximité de la cause, contraste nettement avec le caractère non sélectif de la théorie de l'équivalence des conditions. Celle-ci retient comme unique cause de l'évènement celle qui est située le plus près dans le temps du dommage, c'est-à-dire la *causa proxima*. Cette approche permet d'effectuer un tri parmi les divers antécédents en

---

[241] Tony HONORÉ, « Causation in the Law », (2001) *Standford Encyclopedia of Philosophy*, source : <http://plato.stanford.edu/entries/causation-law>.

[242] P. H. OSBORNE, *op. cit.*, note 144, p. 51.

[243] Jean CARBONNIER, *Droit civil - Les Obligations*, Presses universitaires de France, Paris, 1996, no. 216, p. 368.

attachant la responsabilité à celui qui est chronologiquement le dernier[244]. Néanmoins, le caractère mécanique et simpliste de cette théorie peut entraîner des absurdités, particulièrement lorsqu'une cause plus éloignée du préjudice a joué un rôle important dans sa réalisation[245].

[152] Ce type de situation constitue sans doute la norme en ce qui a trait aux dommages découlant de l'utilisation d'un logiciel libre. Par exemple, les gestes de la victime étant nécessairement posés après le développement du logiciel, la théorie de la proximité de la cause prive cette dernière de tout recours dès lors qu'elle a elle-même partiellement contribué à ses pertes. De plus, lorsque le dommage découle d'une fonctionnalité particulière du logiciel, il est possible de considérer que les développeurs ayant mis celle-ci au point contribuent beaucoup plus aux pertes que celui qui, par la suite, adapte le logiciel à une nouvelle plate-forme informatique. Pour ces raisons, cette théorie ne permet pas de résoudre efficacement les problèmes de causalité posés par l'application de la responsabilité civile aux développeurs de logiciels libres.

[153] Pour sa part, la théorie de la causalité adéquate découle des travaux des auteurs modernes afin d'établir des critères plus rationnels devant présider à la détermination des causes. Pour certains, la causalité adéquate signifie que la responsabilité doit être attachée aux faits générateurs qui accroissent sensiblement la possibilité de réalisation du dommage, suivant le cours habituel des choses. Ainsi :

> « la cause adéquate est la cause qui, *normalement*, entraîne toujours un dommage de l'espèce considérée, par opposition aux

---

[244] Philippe CONTE et Patrick MAISTRE DU CHAMBON, *La responsabilité délictuelle*, Presses universitaires de Grenoble, Grenoble, 2000, no. 150, p. 134.

[245] Patrice JOURDAIN, *Les principes de la responsabilité civile*, 2e Éd., Dalloz, Paris, 2000, p. 68.

> causes qui n'entraînent un tel dommage que par suite de circonstances extraordinaires »[246].

[154] En d'autres mots, il s'agit d'évaluer les implications potentielles de la cause en fonction du sens commun[247]. Pour d'autres, le critère réside plutôt dans la possibilité objective du résultat. Par exemple, il est possible de considérer causal l'évènement qui constitue un élément nécessaire d'un ensemble de conditions conjointement suffisantes pour la réalisation du dommage[248]. Cette méthode, basée sur un *necessary element of a sufficient set*, est celle qui a retenue le plus d'attention au cours des dernières années.

[155] Malgré tout, quelque soit la méthode utilisée pour établir la cause adéquate, cette théorie demeure considérablement subjective. En effet, certains faits générateurs « rendent la survenance du préjudice simplement possible, d'autres plausible, d'autres enfin probable »[249]. Aussi, le tri nécessaire à la sélection des causes requiert, dans chaque cas, un jugement de valeur. La théorie de la causalité adéquate joue tout de même un rôle prédominant dans les juridictions civilistes[250].

[156] L'application de la causalité adéquate en matière de logiciels libres semble donner de meilleurs résultats que ceux découlant des deux autres théories. Tout d'abord, contrairement à ce qui se produit avec la théorie de l'équivalence des conditions, il s'avère possible de rejeter la responsabilité du développeur dont les agissements sont essentiels au dommage mais qui ne l'aurait pas causé sans la contribution prépondérante d'un autre. Ainsi, peut-on prétendre que le développeur ayant appliqué négligemment une rustine ne

---

[246] J. CARBONNIER, *op. cit.*, note 243, no. 216, p. 368.

[247] H. HART et T. HONORÉ, *op. cit.*, note 234, p. 133.

[248] Richard W. WRIGHT, « Causation in Tort Law », (1985) 73 *Calif. L. Rev.* 1735.

[249] J.-L. BAUDOUIN et P. DESLAURIERS, *op. cit.*, note 144, no. 518, p. 346.

[250] A. BÉNABENT, *op. cit.*, note 236, no. 558, p. 334.

devrait pas être responsable du code erroné qu'elle contient, contrairement à celui qui a été négligent en la programmant. Dans un second temps, aucune responsabilité n'est prématurément rejetée sur la base de critères arbitraires, tel qu'il résulte de l'application de la théorie de la proximité de la cause. Il apparaît donc que cette théorie devrait être privilégiée même si elle ne permet pas de généraliser suffisamment le mécanisme d'établissement du lien de causalité pour fournir une réponse précise dans chaque situation.

[157] Aucune de ces solutions n'étant parfaite, certains auteurs proposent de remplacer la causalité par une notion de rechange[251]. Bien qu'aucune juridiction ne se soit encore complètement détachée de la recherche du lien causal, ces idées influencent parfois les décisions des tribunaux. C'est le cas de la théorie de la relativité aquilienne qui propose de substituer l'analyse du but de la règle de droit à celle de la causalité[252]. Selon les partisans de cette théorie, la responsabilité peut être attribuée lorsque le texte législatif enfreint a justement pour but d'éviter les pertes similaires à celles réclamées. De cette façon, un tribunal a déjà présumé l'existence d'un lien de causalité entre la maladie d'un employé et le milieu de travail inadéquat mis en place par son employeur, malgré l'impossibilité scientifique de prouver l'origine de l'infection[253]. De façon similaire, la théorie du risque prétend distribuer la responsabilité en fonction du risque potentiel découlant de chaque fait générateur[254]. Selon ce point de vue, la répartition de la responsabilité sur la base de probabilités permettrait de mieux gérer les situations où de multiples causes se confondent. D'autres auteurs recommandent encore de remplacer

---

[251] Geneviève VINEY et Patrice JOURDAIN, *Les conditions de la responsabilité,* 2e Ed., L.G.D.J., Paris, 1998, no. 336, p. 156.

[252] Denis PHILIPPE, « La théorie de la relativité aquilienne », dans *Mélanges Dalcq*, Larcier, Bruxelles, 1994, p. 467.

[253] Au Royaume-Uni, *McGhee* c. *National Coal Board*, [1973] 1 W.L.R. 1.

[254] John FLEMING, « Probabilistic Causation in Tort Law », (1989) 68 *R. du B.* 661.

la notion de causalité par celle de l'équité, c'est-à-dire par la prise en compte de la situation économique des parties[255].

[158] La large discrétion judiciaire dont jouissent les tribunaux au niveau de l'évaluation de causalité permet à ces notions étrangères d'avoir un impact sur l'attribution de la responsabilité civile. Cette marge de manoeuvre est essentielle dans la mesure où les théories classiques ne permettent pas d'identifier objectivement la cause légale d'un dommage. L'analyse des faits n'étant pas suffisante, les juges n'ont alors d'autres choix que de prendre en compte diverses politiques judiciaires, tel l'obtention d'un effet dissuasif, l'assurance d'un dédommagement pour le demandeur ou le respect de la moralité[256]. Pour la majorité des auteurs, l'application de ces politiques judiciaires est déterminante :

> « the threshold question of *sine qua non* [...] is the only factual question in the area called 'legal causation'. [...] after that question is dealt with, the remaining issues of 'legal causation', 'proximate cause', or 'remoteness of damage' (even if expressed in terms of causation), are inextricably tied up with questions of policy : when *should* the defendant be liable. »[257]

Section 3 - Les considérations économiques

[159] Lorsque la responsabilité civile des développeurs d'un logiciel libre est impliquée, l'un des facteurs stratégiques pertinent a trait à l'obtention d'un résultat économiquement adéquat. En effet, les décisions concernant l'attribution de la responsabilité ont un impact important sur l'utilisation des ressources et le partage de la richesse dans une société[258]. Le modèle économique sur lequel repose les logiciels libres étant unique, l'application de

[255] T. HONORÉ, *loc. cit.*, note 237, vol. XI, chap. 7, no. 101.

[256] John COOKE et David W. OUGHTON, *The Common law of Obligations*, Butterworths, London, 1989, p. 238.

[257] J. STAPLETON, *loc. cit.*, note 240, 114.

la responsabilité civile doit nécessairement y être adapté. Ceci est d'autant plus vrai que le droit de la responsabilité civile est en mesure de mettre en péril l'existence même des logiciels libres. Un tel résultat n'est évidemment pas souhaitable puisque les pertes subies au niveau de l'industrie du logiciel et de l'innovation dépasseraient alors les gains réalisés par les demandeurs ayant reçu un dédommagement. Il est donc important que les tribunaux considèrent les implications économiques de leurs décisions au stade de l'établissement de la causalité.

[160] Tout d'abord, il ne fait aucun doute que l'un des principaux effet de la structure légale des licences libres est de minimiser le coût de production des logiciels. Cette caractéristique assure d'ailleurs la viabilité des projets libres[259]. Dans la mesure où l'imposition de la responsabilité aux développeurs entraîne l'effet inverse, beaucoup y voient un élément susceptible de stopper les contributions :

> « The *GNU* Project wants to encourage programmers to donate their programming time, but legal liability could detract from this goal, since personal liability for defects might discourage programmers from participating. »[260]

[161] L'anticipation de ce résultat découle de la présomption que l'application de la responsabilité civile entraînerait, pour les développeurs de logiciels libres, des inconvénients supérieurs aux avantages découlant de leur participation. Or, cette présomption n'est pas justifiée si la responsabilité peut être assumée par des développeurs dont les bénéfices sont suffisants pour supporter ce risque supplémentaire. En conséquence, les tribunaux devraient s'attarder à identifier les développeurs répondant à ce critère et

---

[258] Harold DEMSETZ, « When Does the Rule of Liability Matter? », (1972) 1 *Journal of Legal Studies* 13.

[259] R. W. GOMULKIEWICZ, *loc. cit.*, note 137, 192

[260] I. V. HEFFAN, *loc. cit.*, note 74, 1509.

privilégier l'établissement d'un lien causal entre leurs gestes et les pertes subies.

[162] Pour y arriver, la totalité des bénéfices directs et indirects motivant les développeurs de logiciels libres doivent être pris en compte. Ceux-ci étant diversifiés et souvent sans valeur monétaire intrinsèque[261], cette tâche peut paraître difficile à accomplir. Cependant une catégorisation en trois groupes distincts s'avère réalisable :

- Les bénéfices monétaires (M) comprennent, entre autres, la rémunération reçue en contre-partie du travail fourni et l'économie de temps réalisée grâce au logiciel amélioré ;
- Les bénéfices hédonistes (H) englobent ce qui a trait au plaisir de programmer et d'appartenir à un groupe ;
- les bénéfices socio-psychologiques (SP) concernent principalement les avantages liés à l'apprentissage et à la réputation mais incluent également toutes les implications culturelles découlant de l'apport fourni ;

[163] En fonction de ces trois groupes, chaque développeur tirera un bénéfice différent de sa participation au développement d'un logiciel libre :

$$B = M + H + SP^{262}$$

[164] Selon cette formule simplifiée, un développeur devrait théoriquement continuer de contribuer à un projet, tant et aussi longtemps que B demeure positif à son égard. De plus, même lorsqu'aucun bénéfice monétaire n'est tiré

---

[261] J. LERNER et J. TIROLE, *loc. cit.*, note 51.

[262] Yochai BENKLER, « Coase's Penguin, or, Linux and the Nature of the Firm », (2002) 112 *Yale L. J.*, à paraître.

d'un logiciel, le risque découlant de l'imposition potentielle de la responsabilité civile n'incite pas automatiquement les développeurs à cesser d'y contribuer. En effet, les développeurs pour qui H et SP sont élevés accepteront malgré tout de courir le risque. Or, la vaste majorité des contributeurs occasionnels d'un projet ne correspondent pas à cette description. Leur apport étant trop restreint pour influencer leur réputation, le bénéfice découlant de leur implication se trouve ainsi limité. Au contraire, les promoteurs du logiciel, c'est-à-dire ceux qui influencent grandement son orientation et son élaboration, voient leurs noms associés au projet et leurs bénéfices se trouvent décuplés. Cette position leur permet donc d'accepter un niveau de risque plus élevé. Dans ces circonstances, il semble adéquat de ne pas retenir inutilement la responsabilité civile des développeurs occasionnels lorsque leurs agissements constituent une cause éloignée du dommage. Les tribunaux devraient plutôt tenter d'identifier un nombre restreint de causes rapprochées, particulièrement lorsque celles-ci découlent des agissements des promoteurs du logiciel.

[165] Au surplus, une analyse économique traditionnelle de l'établissement de la causalité parmi les développeurs d'un logiciel libre mène à la même conclusion. Vu sous cet angle, le principal objectif des règles de responsabilité civile est d'atteindre un équilibre optimal entre le coût des dommages et celui des mesures permettant de les éviter[263]. Il en découle que les gestes du défendeur ne devraient être considérés comme la cause du préjudice que si cela a pour effet de promouvoir une allocation efficace des ressources au niveau de la prévention. En d'autres mots, « the injurer 'causes' the injury when he is the cheaper cost avoider ; not otherwise »[264]. Or, lorsqu'un logiciel libre est concerné, les promoteurs du projet sont bien souvent les développeurs les mieux placés pour éviter le dommage,

---

[263] Richard A. POSNER, « A Theory of Negligence », (1972) 1 *Journal of Legal Studies* 29, 33.

[264] William M. LANDES et Richard A. POSNER, « Causation in Tort Law : An Economic Approach » (1983) 12 *Journal of Legal Studies* 109, 110.

particulièrement si les gestes reprochés concernent le logiciel dans son ensemble. Ceci s'explique en grande partie par la vision globale de l'évolution du logiciel qu'ils partagent, par l'information qui est en leur possession et par l'influence qu'ils exercent sur les contributeurs occasionnels. À l'inverse, pour ces derniers, le coût de la prévention sera minime uniquement dans les cas où une fonctionnalité précise du logiciel sur laquelle ils ont travaillé est impliquée. Dans toutes les autres situations, l'important nombre d'heures de travail qu'ils devraient fournir pour éviter la survenance du préjudice pousse à retenir uniquement la causalité des gestes posés par les promoteurs du logiciel.

Section 4 - Le partage de la responsabilité

[166] Peu importe la théorie retenue et les politiques judiciaires considérées, l'établissement de la causalité parmi les développeurs d'un logiciel libre entraînera presque inévitablement la sélection de plusieurs causes efficientes. Une fois cette première étape franchie, encore faut-il partager la responsabilité entre les développeurs à l'origine de ces gestes. En fonction des rapports qui existent entre les diverses causes efficientes, ce partage peut prendre plusieurs formes.

[167] Lorsque les causes efficientes ayant concourues au dommage sont de nature contractuelle, le droit soumet le partage de la responsabilité à la volonté des parties[265]. C'est donc dans le texte des licences libres que la solution doit d'abord être recherchée. Dans la mesure où celles-ci ne placent aucune obligation sur les épaules des donneurs de licence, le partage de la responsabilité n'y est évidemment jamais abordé. En droit civil, les obligations contractuelles étant présumées conjointes[266], la responsabilité de

---

[265] Bernard ZAHND, *Pluralité de responsables et solidarité*, Held SA, Lausanne, 1980, p. 23.

[266] J.-L. BAUDOUIN et P.-G. JOBIN, *op. cit.*, note 127, no. 605, p. 472.

chaque donneur de licence devrait donc être limitée au montant de sa part dans le dommage. La même solution devrait également prévaloir en common law, quoi que dans ce cas « The presumption is that a contract made by two or more persons is joint »[267]. Ceci implique que, a priori, le paiement de la totalité de la dette est exigible de chacun des responsables, sous réserve du recours que celui qui a payé peut engager contre les autres. Toutefois, cette présomption ne devrait pas trouver application aux donneurs de licence d'un logiciel libre puisque chacun d'eux s'est engagé indépendamment pour son propre apport au logiciel. L'objet de leurs obligations respectives est donc différent. Au surplus, l'essence même des contrats de licence libres semble s'opposer à l'idée de toute solidarité entre les développeurs. Ceci découle des clauses d'exclusion de responsabilité qui sont omniprésentes. En conséquence, l'utilisateur recherchant la responsabilité civile des donneurs de licence d'un logiciel libre se doit d'impliquer tous les responsables dans ses procédures judiciaires s'il désire être indemnisé intégralement. Cette situation n'est évidemment pas à son avantage puisqu'il se trouve ainsi à assumer le risque que représente les développeurs insolvables et non-contraignables.

[168] La présomption de *non-solidarité* s'étendant aux obligations légales, le même résultat devrait prévaloir en matière extra-contractuelle tant et aussi longtemps qu'un texte législatif n'établit pas la solidarité de façon expresse[268]. Toutefois, en vertu du droit commun de la responsabilité civile, la solidarité est la norme en droit civil comme en common law. Dans les juridictions civilistes, elle découle tout autant de la jurisprudence que de la législation. Par exemple, les tribunaux français considèrent qu'une obligation *in solidum* existe entre les co-responsables délictuels[269], ce qui équivaut à la solidarité avec la différence qu'une mise en demeure ou une interruption de la

---

[267] Glanville L. WILLIAMS, *Joint Obligations*, Butterworth, London, 1949, p. 35.

[268] J.-L. BAUDOUIN et P.-G. JOBIN, *op. cit.*, note 126, no. 610, p. 475.

[269] *Cass. Civ.*, 4 déc. 1939, DC 1941.124.

prescription pour l'un des débiteurs ne vaut pas pour les autres[270]. Au contraire, au Québec c'est l'article 1526 du code civil qui établit la solidarité en matière extra-contractuelle. Pour les juridictions de common law, les mêmes conséquences découlent de législations et de décisions judiciaires ayant modifiées le principe de la *joint and several liability*. Traditionnellement, chaque développeur trouvé responsable aurait pu être contraint de dédommager le demandeur intégralement, sans possibilité de recouvrement envers les autres co-responsables[271]. Aujourd'hui, la contribution de ces derniers est presque universellement assurée. Cette solidarité représente un avantage majeur pour la victime puisque celle-ci n'a qu'à prouver la responsabilité d'un seul développeur pour obtenir pleine compensation. Le risque d'insolvabilité et de *non-contraignabilité* est ainsi transféré sur les épaules de ce dernier. Une conséquence supplémentaire est que les entreprises et les institutions dont la capacité de payer est assurée deviennent des cibles de choix par rapport aux développeurs individuels.

[169] Par ailleurs, que l'obligation de dédommagement soit conjointe ou solidaire, la forme de la contribution attachée à chaque cause efficiente peut varier. Théoriquement, chaque cause efficiente ayant contribué à l'intégralité du dommage, celui-ci devrait toujours être divisé en parts égales[272]. Cependant, la vaste majorité des législateurs préfèrent relier le partage de la responsabilité à la gravité respective des gestes posés[273]. D'autres encore, admettent un partage égal mais uniquement pour les cas où « il s'avère trop difficile de déterminer la part de responsabilité attribuable [...] à l'égard de la

---

[270] Pierre RAYNAUD, « La nature de l'obligation des coauteurs d'un dommage. Obligation 'in solidum' ou solidarité?, dans *Mélanges Vincent*, 1981, p. 317.

[271] Glanville L. WILLIAMS, *Joint torts and Contributory Negligence*, Stevens & Sons, London, 1951.

[272] Henri et Léon MAZEAUD et André TUNC, *Traité théorique et pratique de la responsabilité civile délictuelle et contractuelle*, 6e éd., Éditions Montchrestien , Paris, 1965, t. 2, no. 1505, p. 621.

[273] Au Québec, *Code civil du Québec*, précité, note 131, art. 1478 ; aux États-Unis, *Restatement (Third) of Torts : Apportionnement of Liability*, 1999, art. 26.

faute ou de la négligence »[274]. Ces deux dernières solutions paraissent particulièrement adéquates en ce qui concerne les logiciels libres, car elles évitent aux développeurs occasionnels ayant peu contribué aux pertes d'avoir à assumer une responsabilité disproportionnée.

[170] Enfin, l'une des causes efficientes du dommage peut parfois être attribuable au demandeur lui-même. Dans une telle situation, le droit civil permet d'exonérer les développeurs en proportion de la responsabilité qu'on lui attribue, « sans pouvoir aller jusqu'à l'exonération totale »[275]. Au contraire, la common law traditionnelle, par le biais de la *contributory negligence*, tend à retenir l'entière responsabilité du demandeur dès lors que ses agissements ont contribué en partie à ses pertes[276]. Toutefois, la vaste majorité des législations sur le partage de la responsabilité civile empêchent aujourd'hui les développeurs de se fonder sur la *contributory negligence* pour obtenir une défense complète[277]. En somme, le demandeur peut être tenu d'assumer une part de ses pertes, sans que cela ne dégage complètement les développeurs dont les gestes constituent des causes efficientes.

Section 5 - Les régimes exceptionnels de la causalité

[171] Le fardeau de la preuve de la causalité appartenant au demandeur[278], c'est la personne lésée par le logiciel libre qui doit démontrer le lien causal entre les agissements des développeurs et ses pertes. Compte tenu de la situation factuelle, cette preuve peut être difficile, voire impossible, à établir. Pour des raisons similaires, les tribunaux ont élaboré des régimes exceptionnels facilitant l'établissement du lien de causalité dans certaines

[274] En Ontario, *Loi sur le partage de la responsabilité*, L.R.O., 1990, c. N-1, art. 4.

[275] A. BÉNABENT, *op. cit.*, note 237, no. 564-1, p. 339.

[276] William PROSSER, « Comparative Negligence », (1953) 51 *Mich. Law Rev.* 465.

[277] A. M. LINDEN, *op. cit.*, note 144, p. 461.

[278] J.-L. BAUDOUIN et P. DESLAURIERS, *op. cit.*, note 144, no. 542, p. 358.

situations. Quelques-uns d'entre eux sont susceptibles d'être invoqués à l'encontre des développeurs d'un logiciel libre.

[172] L'application du régime de la faute commune est limitée au droit commun de la responsabilité extra-contractuelle. Comme son nom l'indique, il permet de considérer que des gestes répréhensibles identiques ont été posés par l'ensemble des responsables. Ceux-ci peuvent être trouvés « soit dans la participation à une activité spécialement dangereuse, soit dans l'organisation défectueuse de l'activité »[279]. Pour s'en prévaloir, la victime d'un logiciel libre doit être en mesure de démontrer que la simple implication dans l'élaboration du logiciel est fautive. C'est le cas, entre autres, du logiciel dont l'objectif même est de causer un dommage, tel qu'un virus informatique. Une fois cette preuve établie, le fardeau de preuve de la victime est considérablement réduit puisque le nombre de liens causals se trouve réduit à un seul. Quant à la répartition du préjudice entre les développeurs impliqués, la division ne peut être qu'identique puisqu'une seule faute a été commise[280]. Par conséquent, le régime de la faute commune tend à faire supporter une portion disproportionnée de la responsabilité aux développeurs ayant peu contribué au projet. Cependant, cette approche apparaît particulièrement efficiente là où les gestes des développeurs ont été posés dans l'intention de causer un dommage.

[173] Le régime de la responsabilité alternative est, quant à lui, fondé sur l'existence de plusieurs faits distincts générateurs de responsabilité. Il trouve application quand il est impossible d'identifier la personne ayant causée le préjudice, bien qu'il soit certain que celle-ci est comprise à l'intérieur d'un groupe déterminé de personnes ayant posé des gestes auxquels la responsabilité peut être attachée[281]. Par exemple, dans l'hypothèse où le dommage peut être relié à une ligne de code en particulier et que l'auteur de

[279] Danièle MAYER, « La 'garde' en commun », (1975) 74 *R.T.D.C.* 197, 207.

[280] J.-L. BAUDOUIN et P. DESLAURIERS, *op. cit.*, note 144, no. 561, p. 368.

cette ligne n'est pas connu, la responsabilité alternative peut être utilisée si tous les développeurs ayant négligemment travaillés sur cette partie du logiciel le sont. Évidemment, le caractère distribué du développement des logiciels libres empêchera parfois la détermination d'un groupe précis. Cependant, dans la mesure où cela est possible, les tribunaux qui reconnaissent le régime de la responsabilité alternative considèrent qu'un « burden shift back to the defendants on the issue of factual causation [is] necessitated as a matter of fairness »[282]. Par conséquent, la responsabilité de l'ensemble des développeurs incapables de se disculper peut être retenue, même si un seul d'entre eux a réellement causé le préjudice. En pareil cas, la responsabilité doit nécessairement être partagée entre eux en parts égales. Les développeurs occasionnels sont donc également pénalisés sous ce régime.

[174] Finalement, la responsabilité en fonction de la part de marché vise elle aussi à garantir une indemnisation dans les cas où la cause du dommage s'avère impossible à identifier de façon précise. Cette théorie découle de la constatation des tribunaux américains que « [in] our complex industrialized society, advances in science and technology have created fungible goods that may harm consumers and that are difficult to trace to a specific producer »[283]. En conséquence, ceux-ci acceptent parfois d'imposer la responsabilité à de nombreux fabricants fautifs en proportion de leur part respective dans le marché d'un produit spécifique[284]. Il est vrai que ce régime peut difficilement être appliqué aux logiciels libres sous sa formulation actuelle, aucune part de marché distincte ne pouvant être attribuée aux

---

[281] Hassen ABERKANE, « Du dommage causé par une personne indéterminée dans un groupe déterminé de personnes », (1958) 56 *R.T.D.C.* 516, 517.

[282] Michelle ADAMS, « Causation and Responsibility in Tort and Affirmative Action », (2001) 79 *Texas L. Rev.* 643, 674.

[283] David M. SCHULTZ, « Market Share Liability in DES Cases : The Unwarranted Erosion of Causation in Fact », (1991) 40 *Depaul L. Rev.* 771, 782.

[284] *Sindell* c. *Abbott Laboratories*, 26 Cal. 3D 588, 610.

différents développeurs. Cependant, une méthode similaire de répartition de la responsabilité basée sur l'apport respectif de chacun des développeurs au projet est envisageable. Une telle façon de procéder aurait l'avantage de permettre un partage de la responsabilité nettement plus équitable que celui découlant des régimes de la faute commune et de la responsabilité alternative.

* * *

[175] La problématique de la causalité représente certainement la plus grande source de difficultés pour l'application de la responsabilité civile aux développeurs de logiciels libres. Le droit fournit tout de même plusieurs pistes de solution facilitant le dédommagement des personnes ayant subi un préjudice, même si la théorie est loin d'expliquer efficacement le mécanisme de la causalité. Ainsi, la marge de manoeuvre dont disposent les tribunaux leur permet de tenir compte de diverses politiques judiciaires et au besoin de recourir à l'un des régimes exceptionnels de la causalité. La solidarité des responsables agit dans le même sens. Malgré tout, les utilisateurs fondant leur recours sur les licences libres se trouvent encore une fois désavantagés par rapport aux tiers. En effet, les obligations des donneurs de licence sont conjointes et les régimes exceptionnels de la causalité sont mal adaptés à ce contexte contractuel particulier.

Conclusion

[176] L'incertitude juridique à laquelle les développeurs de logiciels libres font face en matière de responsabilité civile est caractéristique de l'avènement de toute nouveauté technologique. Elle est accentuée par l'organisation décentralisée qui caractérise le développement des logiciels libres et les nouveaux outils juridiques que constituent les licences libres. Elle est compréhensible car « Le développement de technologies engendre des transformations qui remettent en cause les catégories par lesquelles on avait l'habitude de définir leur cadre juridique »[285]. Néanmoins, l'analyse présentée montre que le droit, tel qu'il existe actuellement, est en mesure de résoudre la majorité des problèmes relatifs à la détermination et l'application de la responsabilité civile en matière de logiciels libres. Il n'en demeure pas moins que les réponses apportées par les différents systèmes juridiques ne sont pas toujours identiques et que leur mise en oeuvre n'obtiendra définitivement pas l'unanimité.

[177] La responsabilité civile des développeurs de logiciels libres peut découler de différentes sources. Tout d'abord, en ce qui concerne leur relation avec le licencié, ceux-ci sont protégés efficacement par les licences d'utilisation auxquelles ils ont recours. Cette protection n'est toutefois pas infaillible puisque certaines circonstances contribuent à rendre les clauses d'exclusion de responsabilité des licences libres invalides. C'est le cas, notamment, lorsque les gestes des développeurs constituent une faute ou une négligence grave, qu'ils sont posés avec l'intention de causer un dommage ou lorsque le droit interdit les exclusions contractuelles d'un type de garantie ou de dommage particulier. Pour les développeurs distribuant un logiciel libre de façon commerciale, les risques sont beaucoup plus élevés puisque les circonstances militent en faveur de l'atténuation des effets des

[285] P. TRUDEL et autres, *op. cit.*, note 62, p. intr-3.

clauses d'exclusion de responsabilité et que les protections du droit de la consommation peuvent leur être opposées.

[178] Les dommages susceptibles d'être causés à des tiers peuvent aussi entraîner la responsabilité des développeurs de logiciels libres. Dans ce cas, les protections mises en place par les licences libres s'avèrent moins efficaces. En effet, même lorsque la licence contient une clause d'allocation des risques, le dédommagement du préjudice ne peut être systématiquement relégué au licencié. Ainsi, les développeurs peuvent être tenus de dédommager un tiers pour leur faute ou leur négligence si le licencié est insolvable ou non-contraignable. Il en va de même lorsque les gestes des développeurs ont été posés avec l'intention de causer un dommage ou que le logiciel est grevé d'un défaut de sécurité, dans la mesure où ils agissent à titre de professionnels. Enfin, une multitude de textes législatifs ont le potentiel d'engendrer la responsabilité civile des développeurs sans que ceux-ci n'aient la possibilité de la reléguer au licencié.

[179] Par ailleurs, l'application de cette responsabilité est réalisable. Ainsi, plusieurs éléments permettent d'assurer la *contraignabilité* des développeurs. Par exemple, la *traçabilité* du code d'un logiciel libre peut être assurée par le respect des politiques de la *FSF* qui requièrent l'identification exhaustive des auteurs ainsi que par l'utilisation de fichiers journaux de modification ou de systèmes de contrôle de versions. De la même façon, certains mécanismes juridiques permettent de répondre à la répartition géographique des développeurs. C'est notamment le cas des théories américaines du contact minimum et du recours aux effets comme facteur de rattachement. Celles-ci incitent maintenant les tribunaux à interpréter leur juridiction de façon extensive. Parallèlement, les accords internationaux récents ont tendance à faciliter l'exécution des jugements à l'étranger. Bien que ces éléments ne permettent pas de garantir la *contraignabilité* de tous les développeurs de

logiciels libres, la vaste majorité d'entre eux ne sont pas en position d'échapper à l'application de la responsabilité civile.

[180] Le droit fournit également des critères permettant d'identifier, parmi l'ensemble des développeurs d'un logiciel libre, ceux dont les gestes constituent la cause légale du préjudice. Ainsi, la théorie de la causalité adéquate permet au juge de retenir uniquement la responsabilité de ceux qui ont contribué à faire augmenter les probabilités de la survenance du préjudice. La prise en compte des facteurs économiques suggère également de limiter l'application de la responsabilité aux promoteurs d'un logiciel, plutôt que de l'étendre aux contributeurs occasionnels. Enfin, le partage de la responsabilité et l'application des régimes exceptionnels de la causalité contribuent tout deux à favoriser le dédommagement des tiers ayant subi un dommage.

[181] En somme, les règles de la responsabilité civile représentent un risque potentiel pour les développeurs de logiciels libres, même s'ils sont relativement bien protégés par les contextes juridiques et factuels. Pourtant, aucune décision judiciaire n'a encore abordé cette question de près ou de loin. Ceci n'est pas étonnant dans la mesure où jusqu'à maintenant, même les logiciels propriétaires destinés au grand public ont été à l'abri des conséquences de la responsabilité civile. Pourtant, ceux-ci occupent le marché depuis de nombreuses années. Ceci s'explique probablement par l'absence de précédents et le manque de moyens de preuve qui découle de l'absence d'accès au code des logiciels. Pour plusieurs, « Today there are no real consequences for having bad security, or having low-quality software of any kind. In fact, the marketplace rewards low quality »[286]. En conséquence,

---

[286] Bruce SCHNEIER, « Crypto-Gram Newsletter », (2002) *Counterpane Internet Security*, source : <http://www.counterpane.com/crypto-gram-0204.html>.

de nombreuses voix s'élèvent aujourd'hui pour réclamer une application plus efficace des règles de la responsabilité civile aux fabricants de logiciels[287].

[182] Si cette perception des choses se concrétise par des amendements législatifs ou l'établissement d'une jurisprudence sur cette question, les chances de voir la responsabilité civile des développeurs d'un logiciel libre impliquée dans un litige augmenteront. Ceci est d'autant plus vrai qu'en matière de logiciels libres, les difficultés reliées à l'obtention de moyens de preuve sont éliminées par la disponibilité du code source. L'utilisation de plus en plus courante des logiciels libres pour effectuer des tâches critiques contribue également à ce résultat.

[183] Dans cette optique, les développeurs de logiciels libres ont avantage à transférer les risques découlant de cette responsabilité à des tiers. Ceci peut d'abord être réalisé par le recours à des distributeurs garantissant les logiciels après les avoir révisés. Ceux-ci y trouveraient leur profit en commercialisant leur apport sous la forme d'un service à valeur ajoutée[288]. Ces distributeurs pourraient même mettre en place des systèmes de certification des logiciels libres sur la base de critères de qualité ou de compatibilité[289]. Il est également possible d'y arriver en souscrivant une assurance sur le logiciel. À ce propos, les développeurs de logiciels libres sont avantagés puisque certaines compagnies d'assurance réclament maintenant des primes moins élevées à leurs assurés utilisant un système d'exploitation libre, plutôt qu'un système d'exploitation propriétaire[290]. Il est probable que le marché réponde à l'accroissement de la responsabilité des

---

[287] Elinor MILLS ABREU, « Microsoft at Heart of Calls for Software Liability », (2002) *CNET News*, source : <http://news.cnet.com/investor/news/newsitem/0-9900-1028-20045512-0.html> ; Denis FISHER, « Software Liability Gaining Attention », (2002) *EWeek*, source : <http://www.eweek.com/print_article/0,3668,a=21030,00.asp> ; Ira SAGER et Jay GREENE, « The Best Way to Make Software Secure : Liability », (2002) *BusinessWeek Online*, source : <http://www.businessweek.com:/print/magazine/content/02_11/b3774071.htm?mainwindow>.

[288] I. V. HEFFAN, *loc. cit.*, note 74, 1509.

[289] S. W. POTTER, *loc. cit.*, note 68, par. 90.

développeurs de logiciels libres en mettant ces mesures en place de lui-même.

[184] Cependant, tous les développeurs ne sont pas en position d'avoir recours à ces solutions de contournement. Pour cette raison, il est essentiel que les tribunaux tiennent compte des particularités du modèle de développement des logiciels libres afin de réduire les impacts de la responsabilité civile sur leur évolution. L'équité exige que les programmeurs indépendants qui fournissent une aide gratuite à leur communauté ne soient pas tenus de verser des sommes exorbitantes en guise de dédommagement. D'ailleurs, le droit accorde déjà des exonérations de responsabilité dans plusieurs circonstances similaires. Ainsi, dans certaines juridictions, celui qui donne un bien est exonéré de la responsabilité civile pouvant découler de cet acte[291]. Les volontaires oeuvrant pour un organisme à but non-lucratif bénéficient parfois de la même protection[292]. Peut-être est-il temps d'étendre ces règles aux développeurs de logiciels libres?

---

[290] Erich LUENING, « Windows Users Pay for Hacker Insurance », (2001) *CNET News*, source : <http://news.com.com/2102-1001-258392.html>.

[291] Au Québec, *Code civil du Québec*, précité, note 132, art. 1471.

[292] Aux États-Unis, *Volunteer Protection Act*, 42 U.S.C. s. 14503(a).

Table de la législation

Textes internationaux

Avant-projet de convention sur la compétence et les jugements étrangers en matière civile et commerciale, 1999, Source : <http://www.hcch.net/f/conventions/draft36f.html>

Convention de Berne pour la protection des oeuvres littéraires et artistiques, Juris International, source : <http://www.jurisint.org/pub/01/fr/147.htm>

Convention du 27 septembre 1968 concernant la compétence judiciaire et l'exécution des décisions en matière civile et commerciale, (1990) O.J. (C 189) 2

Convention entre le Canada et le Royaume-Uni de Grande-Bretagne et d'Irlande du Nord pour assurer la reconnaissance et l'exécution réciproques des jugements en matière civile et commerciale, voir annexe de la *Loi sur la Convention Canada-Royaume-Uni relative aux jugements en matière civile et commerciale*, L.R.C. (1985), c. C-30

Traité de l'OMPI sur le droit d'auteur, Juris International, source : <http://www.jurisint.org/pub/01/fr/149.htm>

Textes canadiens

Charte des droits et libertés de la personne, L.R.Q., 1977, c. C-12

Code civil du Québec, L.Q. 1991, c. 64

Code criminel, L.R.C. (1985), c. C-46

Loi sur la protection du consommateur, L.R.O., 1990, c. C-31

Loi sur la protection du consommateur, L.R.Q., 1977, c. P-40.1

Loi sur le droit d'auteur, L.R.C. (1985), c. C-42

Loi sur le partage de la responsabilité, L.R.O., 1990, c. N-1

Loi sur l'indemnisation des victimes d'actes criminels, L.R.Q., 1977, c. I-6

Textes américains

Copyright Act, 17 U.S.C.

Digital Millennium Copyright Act, (1998) 112 Stat. 2860

Gramm-Leach-Bliley Act (2001) 15 U.S.C. 6801

Magnusson-Moss Warranty Act, 15 U.S.C., 2302(c)

Restatement (Second) of Torts, 1965

Restatement (Third) of Torts : Apportionnement of Liability, 1999

Restatement (Third) of Torts : Product Liability, 1997

Uniform Commercial Code, 1992, source : <http://www.law.cornell.edu/ucc/2/overview.html>

Uniform Computer Information Transactions Act, 2001, source : <http://www.law.upenn.edu/bll/ulc/ucita/ucita01.htm>

Volunteer Protection Act, 42 U.S.C.

Textes européens

Directive relative à la protection des personnes physiques à l'égard du traitement des données à caractère personnel et à la libre circulation de ces données, Dir. Cons. CE 95/46, 24 oct. 1995, JOCE 23 novembre 1995

Directive relative au rapprochement des dispositions législatives, réglementaires et administratives des États membres en matière de responsabilité du fait des produits défectueux, Dir. Cons. CE 85/374, 25 juill. 1985, JOCE 7 août 1985

Rép. min., 15 nov. 1988, JOCE 8 mai 1989

Textes français

Code de la propriété intellectuelle

Table de la jurisprudence

Textes canadiens

Textes américains

Textes allemands

Textes anglais

McGhee c. National Coal Board, [1973] 1 W.L.R. 1

Textes français

Cass. Civ., 4 déc. 1939, DC 1941.124

TGI Paris, 3e ch., 12 févr. 1999, Chaumet, Juris-data no. 041421

Textes japonnais

Mukoda et al. c. Boeing Co., (1986) 604 Hanrei taimuzu 138

Bibliographie

ABERKANE, H., « Du dommage causé par une personne indéterminée dans un groupe déterminé de personnes », (1958) 56 R.T.D.C. 516

ADAMS, M., « Causation and Responsibility in Tort and Affirmative Action », (2001) 79 Texas L. Rev. 643

AGENCE POUR LES TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION DANS L'ADMINISTRATION, « L'approche en matière de logiciels et de logiciels libres », (2001) ATICA, source : <http://www.atica.pm.gouv.fr/bouquet-libre/ll_fr.pdf>

APPLE COMPUTER, « Apple Public Source License », (2001) Apple, source : <http://www.opensource.apple.com/apsl>

ARMOUR, J., W. S. HUMPHREY, « Software Product Liability », (1993) Carnegie Mellon Software Engineering Institute, source : <http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.013.html>

ASSOCIATED PRESS, « The Penguin Continues Its March », (2002) Wired News, source : <http://www.wired.com/news/linux/0,1411,52863,00.html>

BALLMAN, D. R., « Software Tort: Evaluating Software Harm by Duty of Function and Form », (1997) 3 Conn. Ins. L. J. 417

BAUDOUIN, J.-L., P. DESLAURIERS, La responsabilité civile, 5e Éd., Cowansville , Éditions Yvon Blais, 1998

BAUDOUIN, J.-L., P.-G. JOBIN, Les obligations, 5e Éd., Éditions Yvon Blais, Cowansville, 1998

BÉNABENT, A., Droit civil - Les obligations, 6e Éd., Montchrestien, Paris, 1997

BENKLER, Y., « Coase's Penguin, or, Linux and the Nature of the Firm », (2002) 112 Yale L. J., à paraître

BEZROUKOV, N., « A Slightly Skeptical View on Linus Torvalds », (2000) Softpanorama, source: <http://www.softpanorama.org/People/Torvalds/index.shtml>

BEZROUKOV, N., « Open Source Software Development as a Special Type of Academic Research (Critique of Vulgar Raymondism) », (1999) 10 First Monday, source : <http://www.firstmonday.org/issues/issue4_10/bezroukov/index.html>

BEYER, M. L., « Managing the Risk of Virus Liability », (1993) 12 Computer Law. 22

BIRNBAUM, L. N., « Strict Products Liability and Computer Software », (1988) 8 Comp. L. J. 135

BLOOMBERG NEWS, « Microsoft Executive Says Linux Threatens Innovation », (2001) News.com, source : <http://news.cnet.com/investor/news/newsitem/0-9900-1028-4825719-RHAT.html?tag=ltnc>

BOBKO, P. K., « Linux and General Public Licenses : Can Copyright Keep « Open Source » Software free? », 28 AIPLA Q. J. 81

BOBKO, P. K., « Open-source Software and the Demise of Copyright », (2001) 27 Rutgers Computer & Tech. L. J. 51

BRAND, R. A., « Comparative Forum Non Conveniens and the Hague Convention on Jurisdiction and Judgments », (2002) 37 Tex. Int'l L. J. 467

BRAND, R. A., Enforcing Foreign Judgments in the United States and United States Judgments Abroad, Chigago, American Bar Association, 1992

BREESE, P., « Brevetabilité des logiciels », (2001) Breese Majerowicz, source : <http://www.breese.fr/guide/htm/Logiciel/main1.htm>

BROOKS, R. A., « Deterring the Spread of Viruses Online: Can Tort Law Tighten The 'Net'? », (1998) 17 Rev. Litig. 343

BSDTODAY, « A Brief History of BSD », (2000) BSDToday, source : <http://www.bsdtoday.com/tmp-ba8274c5/bsd-history.html>

CARBONNIER, J., Droit civil - Les Obligations, Presses universitaires de France, Paris, 1996

CENTER FOR INTERNATIONAL LEGAL STUDIES, International Execution Against Judgment Debtors, Oceana Publications Inc., Dobbs Ferry, 1998

CHASSELL, R. J., « Software Freedom : Rights, Duty, Metaphor, and Making a Living », (2002) Rons.net.cn, source : <http://www.rons.net.cn/english/FSM/bob>

CHRISTEY, S., C. WYSOPAL, « Responsible Vulnerability Disclosure Process », 2002, Internet Engineering Task Force, source : <http://www.ietf.org/internet-drafts/draft-christey-wysopal-vuln-disclosure-00.txt>

CLÉMENT-FONTAINE, M., « La licence publique générale GNU », (1999) Juriscom, source : <http://www.juriscom.net/uni/mem/08/presentation.htm>

CLERMONT, C. M., « Jurisdictional Salvation and the Hague Treaty », (1999) 85 Cornell L. Rev. 89

CONCURRENT VERSIONS SYSTEM, « Concurrent Versions System : The Open Standard for Version Control », (2002) CVSHome, source : <http://www.cvshome.org>

CONLEY, J. M., « Tort Theories of Recovery Against Vendors of Defective Software », (1987) 13 Rutgers Comp. & Tech. L. J. 1

CONNELL, C., « Open Source Projects Manage Themselves? Dream On », (2000) Lotus, source : <http://www.lotus.com/developers/devbase.nsf/articles/doc2000091200>

CONTE, P., P. MAISTRE DU CHAMBON, La responsabilité délictuelle, Presses universitaires de Grenoble, Grenoble, 2000

COMMISSION EUROPÉENNE, « Brevetabilité des inventions mises en oeuvre par ordinateur », (2002) Europa.eu.int, source : <http://europa.eu.int/comm/internal_market/fr/indprop/comp/index.htm>

COOKE, J., OUGHTON, D. W., The Common law of Obligations, Butterworths, London, 1989

CROZE, H., F. SAUNIER, Logiciels : retour aux sources, Paris, JCP ed. Générale, 1996

CUBRANIC, D., « Open-Source Software Development », (1999) Cubranic, source : <http://sern.ucalgary.ca/~maurer/ICSE99WS/Submissions/Cubranic/Cubranic.html>

DEMSETZ, H., « When Does the Rule of Liability Matter? », (1972) 1 Journal of Legal Studies 13

DESILETS, R., « Note, Software Vendor's Exposure to Products Liability for Computer Viruses », (1989) 9 Comp. L. J. 509

DIBONA, C., S. OCKMAN et M. STONE (dir.), Open Sources : Voices from the Open Source Revolution, O'Reilly, 1999, source : <http://www.oreilly.com/catalog/opensources/book/raymond.html>

DI CARLO, A., C. SCHWARZ et I. JACKSON, « Debian Developer's Reference Chapter 2 - Applying to Become a Maintainer », (2002) Debian, source : <http://www.debian.org/doc/packaging-manuals/developers-reference/ch-new-maintainer.en.html>

DI FILIPPO, E., « Les logiciels libres », (1999) Juriscom, source : <http://www.juriscom.net/uni/mem/10/log02.rtf>

DRIEU, B., « Licences, du copyright au copyleft », (2000) APRIL, source : <http://www.april.org/articles/divers/licences.html>

DUNNE, R., « Deterring Unauthorized Access to Computer : Controling Behavior in Cyberspace Through a Contract Law Paradigm », (1994) 35 Jurimetrics J. 1

EUNICE, J., « Beyond the Cathedral, Beyond the Bazaar », (1998) Illuminata, source : <http://www.illuminata.com/public/content/cathedral/intro.htm>

FARNSWORTH, A. E., « The Concept of "Good Faith" in American Law », (1993) Consiglio Nazionale delle Ricerche, source : <http://www.cnr.it/CRDCS/frames10.htm>

FLEMING, J., « Probabilistic Causation in Tort Law », (1989) 68 R. du B. 661

FIRST MONDAY, « FM Interview with Linus Torvalds: What Motivates free Software Developers? », (1998) 3 First Monday, source : <http://www.firstmonday.org/issues/issue3_3/torvalds/index.html>

FISHER, D., « Software Liability Gaining Attention », (2002) EWeek, source : <http://www.eweek.com/print_article/0,3668,a=21030,00.asp>

FREE SOFTWARE FOUNDATION, « Catégories de logiciels libres et non libres », (2001) FSF, source : <http://www.fsf.org/philosophy/categories.fr.html>

FREE SOFTWARE FOUNDATION, « GNU Lesser General Public License version 2.1 », (1999) FSF, source : <http://www.fsf.org/copyleft/lesser.html>

FREE SOFTWARE FOUNDATION, « Le problème de la licence BSD », (2000) FSF, source : <http://www.fsf.org/philosophy/bsd.fr.html>

FREE SOFTWARE FOUNDATION, « Les problèmes de la licence d'Apple », (2001) FSF, source : <http://www.fsf.org/philosophy/apsl.fr.html>

FREE SOFTWARE FOUNDATION, « Pourquoi « Free Software » est-il meilleur que 'Open Source' », (2001) FSF, source : <http://www.fsf.org/philosophy/free-software-for-freedom.fr.html>

FREE SOFTWARE FOUNDATION, « Qu'est-ce que le copyleft? », (2001) FSF, source : <http://www.fsf.org/copyleft/copyleft.fr.html>

FREE SOFTWARE FOUNDATION, « Qu'est-ce qu'un Logiciel Libre? », (2001) FSF, source : <http://www.fsf.org/philosophy/free-sw.fr.html>

FRICKE, P., « Linux Strategies and Solutions : Linux Server Suppliers Contend for Leadership », (2002) D.H. Brown Associates, source : <http://www-1.ibm.com/linux/news/pb020405.pdf>

FRIDMAN, G., « Punitive Damages in Tort », (1970) 48 R. du B. Can. 373

GAUTRAIS, V., E. MACKAAY, « Les contrats informatiques », dans Denys-Claude LAMONTAGNE, Contrats spéciaux, Cowansville, Éditions Yvon Blais, 2002, à paraître

GERWIN, B. H., « Computer Related Litigation Using Tort Concepts », (1985) 9 Am. J. Trial Advoc. 97

GHOSH, R. A., « Clustering and Dependencies in Free/Open Source Software Development : Methodology and Preliminary Analysis », (2002) Institut d'économie industrielle, source : <http://www.idei.asso.fr/Commun/Conferences/Internet/OSS2002/Papiers/Ghosh.PDF>

GLASER, R. A., L. M. LEWIS, « Redefining the Professional: The Policies and Unregulated Development of Consultant Malpractice Liability », (1995) 72 U. Det. Mercy L. Rev. 563

GOMULKIEWICZ, R. W., « How Copyleft Uses License Rights to Succeed in the Open Source Software Revolution and the Implications for Article 2b », (1999) 36 Hous. L. Rev. 179

GOSH, R., V. V. PRAKASH, « The Orbiten Free Software Survey », (2000) 5 First Monday, source : <http://www.firstmonday.dk/issues/issue5_7/ghosh/index.html>

HARMS, S., « Linux Anesthesia Modular Devices Interface », (2000) LAMDI, source : <http://gasnet.med.yale.edu/lamdi>

HARRINGTON, M. E., « On-line Copyright Infringement Liability for Internet Service Providers : Context, Cases & Recently Enacted Legislation », (1999) B. C. Intell. Prop. & Tech. F. 060499

HART, H., T. HONORÉ, Causation in the Law, 2e Éd., Clarendon Press, Oxford, 1985

HEFFAN, I. V., « Copyleft : Licensing Collaborative Works in the Digital Age », (1997) 49 Stan. L. R. 1487

HEWLETT-PACKARD, « Linux and HP », (2002) HP, source : <http://www.hp.com/united-states/linux/>

HONORÉ, T., « Causation and Remoteness of Damages », dans International Encyclopedia of Comparative Law, Tübingen, Mohr, 1984

HONORÉ, T., « Causation in the Law », (2001) Standford Encyclopedia of Philosophy, source : <http://plato.stanford.edu/entries/causation-law>

IBM, « IBM Public License Version 1.0 », (2001) IBM source : <http://oss.software.ibm.com/developerworks/opensource/license10.html>

INTERNATIONAL INSTITUTE OF INFONOMICS, « Free/Libre and Open Source Software: Survey and Study », (2002) FLOSS, source : <http://floss1.infonomics.nl/stats.php?id=31>

JOHNSON, K., « Open-Source Software Development », (1999) Johnson, source : <http://www.cpsc.ucalgary.ca/~johnsonk/SENG/SENG691/open.htm>

JOHNSON, S., « Internet Activity and Jurisdiction over Foreign Defendants », (1999) Gary Dunn Computer & Technology Law, source : <http://www.dunn.com/papers/paper_3.shtml>

JOURDAIN, P., Les principes de la responsabilité civile, 2e Éd., Dalloz, Paris, 2000

JUENGER, F., « Judicial Jurisdiction in the United States and in the European Communities : A Comparison », (1984) 82 Mich. L. Rev. 1195

KANER, C., « Liability for Defective Content », (1996) 3 Software QA 56, source : <http://www.kaner.com/badcont.htm>

KANER, C., « Liability for Product Incompatibility », (1998), 4 Software QA 33, source: <http://www.kaner.com/liability.html>

KANER, C., « Software Negligence and Testing Coverage », (1995) 2 Software QA 18, source : <http://www.kaner.com/coverage.htm>

KANER, C., « The Impossibility of Complete Testing », (1997) 4 Software QA 28, source : <http://www.kaner.com/imposs.htm>

KANER, C., « The Law of Software Quality », (2000) Software management Conference San Jose, source: <http://www.kaner.com/pdfs/slides/asmlaw.pdf>

KENNEDY, D. M., « A Primer on Open Source Licensing Legal Issues : Copyright, Copyleft and Copyfuture », (2001) 20 St. Louis U. Pub. L. Rev. 345

LANDES, W. M., R. A. POSNER, « Causation in Tort Law : An Economic Approach » (1983) 12 Journal of Legal Studies 109

LEIBOVITCH, E., « Open Source's Quiet Revenge », (2001) ZDNet News, source : <http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2688215,00.html>

LEMLEY, M., P. MENELL, R. MERGE, P. SAMUELSON, Software and Internet Law, New York, Aspen Publisher, 2001

LEONARD, A., « Life or Death Software », (1999) Salon, source : <http://www.salon.com/tech/feature/1999/08/05/anesthesia/index.html>

LERNER, J., J. TIROLE, « The Simple Economics of Open Source », (2000) National Bureau of Economic Research, source : <http://www.nber.org/papers/w7600>

LE TOURNEAU, P., Théorie et pratique des contrats informatiques, Paris, Éditions Dalloz, 2000

LUCAS DE LEYSSAC, C., L'obligation de renseignements dans les contrats, coll. « l'information en droit privé », Paris, LGDJ, 1978

LINDEN, A. L., Canadian Tort Law, 6th Ed., Butterworth, Markham, 1997

LIU, E., « Governments Embrace Open Source », (2002) OsOpinion, source : <http://www.osopinion.com/perl/story/18157.html>

LUENING, E., « Windows Users Pay for Hacker Insurance », (2001) CNET News, source : <http://news.com.com/2102-1001-258392.html>

LYMAN, J., « IBM To Pump $1 Billion Into Linux », (2000) E-Commerce Times, source : <http://www.ecommercetimes.com/perl/story/6027.html>

LYMAN, S. C., « Civil Remedies for the Victims of Computer Viruses », (1992) 11 Comp. L. J. 607

MACKAAY, E., « Le marché du progiciel – licence ou vente? », (1994) 6 Cahiers de propriété intellectuelle 401

MAHER, M., « Open Source Software : The Success of an Alternative Intellectual Property Incentive Paradigm », (2000) 10 Fordham Intell. Prop. Media & Ent. L. J. 619

MATHEW, P. A., « Architects, Engineers, Computer Product and the Law: A Matter of Anticipation », (1982) 3 Computer L. J. 337

MATTEI, U., J. LENA, « U.S. Jurisdiction Over Conflicts Arising Outside of the United States : Some Hegemonic Implications », (2001) 24 Hastings Int'l & Comp. L. Rev. 381

MAYER, D., « La 'garde' en commun », (1975) 74 R.T.D.C. 197

MAZEAUD, L., « Les fautes collectives », (1956) 16 R. du B. 405

MAZEAUD, H. Et L., A. TUNC, Traité théorique et pratique de la responsabilité civile délictuelle et contractuelle, 6e éd., Éditions Montchrestien , Paris, 1965

MCBROOM, V., « Civil Liability for Software Flaws », (1989) 09 International Comp. L. Ad. 15

MCGOWAN, D., « Legal Implication of Open-Source Software », (2001) U. Ill. L. Rev. 241

MCJOHN, S. M., « The Paradoxes of Free Software », (2000) 9 Geo. Mason L. Rev. 25

MENAIS, A., « Commentaires sur la loi du 19 mai 1998 relative à la responsabilité des produits défectueux », (1998) Juriscom, source : <http://www.juriscom.net/pro/1/resp19980601.htm>

MERGE, R. P., « The End of Friction? Property Rights and Contract in the « Newtonian » World of On-Line Commerce », (1997) 12 Berkeley Tech. L. J. 115

MEYRAN, A., « La notion de causalité dans le droit de la responsabilité civile », (2002) Net-Iris, source : <http://www.net-iris.com/publication/author/document.php3?document=64>

MILLS ABREU, E., « Microsoft at Heart of Calls for Software Liability », (2002) CNET News, source : <http://news.cnet.com/investor/news/newsitem/0-9900-1028-20045512-0.html>

MIYAKI, P. T., « Computer Software Defects: Should Computer Software Manufacturers Be Held Strictly Liable for Computer Software Defects? », (1992) 8 Santa Clara Computer & High Tech. L.J. 121

MOGLEN, E., « Free Software Matters: Free Software or Open Source? », (2000) Moglen, source : <http://moglen.law.columbia.edu/publications/lu-07.html>

MOYSE, P.-E., V. GAUTRAIS, « Droit des auteurs et droit de la consommation dans le cyberespace : la relation auteur/utilisateur », (1996) Léger Robic Richard, source : <http://www.robic.ca/publications/070.shtml>

MOZILLA.ORG, « Mozilla Public License version 1.1 », (2002) Mozilla, source : <http://www.mozilla.org/MPL/MPL-1.1.html>

MOZILLA.ORG, « Netscape Public license FAQ », (1999) Mozilla, source : <http://www.mozilla.org/NPL/FAQ.html>

MYSQL, « MySQL AB and Nusphere Corporation Announce Settlement », (2002) MySQL, source : <http://www.mysql.com/press/release_2002_14.html>

NATIONAL ACADEMY OF SCIENCES, « Cybersecurity Today and Tomorrow: Pay Now or Pay Later », (2002) NAS, source : <http://books.nap.edu/html/cybersecurity>

NETSCAPE, « Netscape Announces Plans to Make Next-Generation Communicator Source Code Avalaible Free on the Net », (1998) Netscape, source : <http:/wp.netscape.com/newsref/pr/newsrelease558.html>

NEUKOM, W. H., R. W. GOMULKIEWICZ, « Licensing Rights to Computer Software », (1993) 354 PLI/Pat 775

NEWMAN, N., « The Origins and Future of Open Source Software », (1999) Net Action, source : <http://www.netaction.org/opensrc/future/oss-whole.html>

OPENOFFICE, « About Us: OpenOffice.org », (2002) OpenOffice, source : <http://www.openoffice.org/about.html>

OPENSOURCE.ORG, « History of the OSI », (1999) Opensource, source : <http://www.opensource.org/docs/history.html>

OPENSOURCE.ORG, « The Open Source Definition », (2001) Opensource, source : <http://www.opensource.org/docs/definition.html>

OSBORNE, P. H., The Law of Tort, Irwin Law, Toronto, 2000

PCS, « What's Freeware? », (2002) Why-not, source : <http://www.why-not.com/software/ware.htm>

PCS, « What's Shareware? », (2002) Why-not, source : <http://www.why-not.com/software/ware.htm>

PERENS, B., « The Trojan Horse », (1998) Lwn.net, source : <http://old.lwn.net/1998/1119/Trojan.html>

PHILIPPE, D., « La théorie de la relativité aquilienne », dans Mélanges Dalcq, Larcier, Bruxelles, 1994

PHILLIPS, D. E., « When Software Fails: Emerging Standards of Vendor Liability Under the Uniform Commercial Code », (1994) 50 Bus. Law. 151

PLANIOL, M., G. RIPERT, Traité élémentaire de droit civil, t. 2, 10e éd., L.G.D.J., Paris, 1926

PLANT, E. C., « GPL Violation – NVIDIA », (2000) Slashdot, source : <http://www.slashdot.org/features/00/05/01/0047219.shtml>

POSNER, R. A., « A Theory of Negligence », (1972) 1 Journal of Legal Studies 29

POTTER, S. W., « Opening Up to Open Source », (2000) 6 Rich. J. L. & Tech. 24

POWELL, D. E., « Judgment Day for the GPL? », (2000) LinuxPlanet, source : <http://www.linuxplanet.com/linuxplanet/reports/2000/1>

PROSSER, W., « Comparative Negligence », (1953) 51 Mich. Law Rev. 465

PUURUNEN, T., « The Legislative Jurisdiction of State over Transactions in International Electronic Commerce », (2000) 18 J. Marshall J. Computer & Info. L. 689

RADDING, A., « IT Managers Become More 'Open'-Minded », (2001) InformationWeek, source : <http://www.informationweek.com/833/opensource.htm>

RADIN, M. J., « Humans, Computers, and Binding Commitment », (2000) 75 Ind. L. J. 1125

RAPHAEL, « The GPL, the Contributors, the ChangeLog and CVS », (2002) Advogado, source : <http://www.advogado.org/article/183.html>

RAUL, A. C., F. R. VOLPE et G. S. MEYER, « Liability for Computer Glitches and Online Security Lapses », (2001) 31 BNA Electronic Commerce Law Report 849, source : <http://www.sidley.com/cyberlaw/features/liability.asp>

RAVICHER, D. B., « Facilitating Collaborative Software Development: The Enforceability of Mass-Market Public Software Licenses », (2000) 5 Va. J. L. & Tech 11

RAYMOND, E. R., « The Cathedral and the Bazaar », (1998) 3 First Monday, source : <http://www.firstmonday.org/issues/issue3_3/raymond/index.html>

RAYMOND, E. S., « The Magic Cauldron », (1999) Raymond, source: <http://www.tuxedo.org/~esr/writings/magic-cauldron/magic-cauldron.html>

RAYNAUD, P., « La nature de l'obligation des coauteurs d'un dommage. Obligation 'in solidum' ou solidarité?, dans Mélanges Vincent, 1981

ROBBINS, V. H., « Vendor Liability for Computer Viruses and Undisclosed Disabling Devices in Software », (1993) 7 Computer Law. 20

ROJAS, J., « Avoiding Entanglement in a Jurisdictional Web », (2000) 590 PLI/Pat 11

ROSENBERG, D. K., « Copyleft and the Religious Wars of the 21st Century », (1998) Stromian Technologies, source : <http://www.stromian.com/copyleft.htm>

RUSTAD, M., L. E. EISENSCHMIDT, « The Commercial Law of Internet Security », (1995) 10 High Tech L. J. 213

SAGER, I., J. GREENE, « The Best Way to Make Software Secure : Liability », (2002) BusinessWeek Online, source : <http://www.businessweek.com:/print/magazine/content/02_11/b3774071.htm?mainwindow>

SCHNEIER, B., « Crypto-Gram Newsletter », (2002) Counterpane Internet Security, source : <http://www.counterpane.com/crypto-gram-0204.html>

SCHULTZ, D. M., « Market Share Liability in DES Cases : The Unwarranted Erosion of Causation in Fact », (1991) 40 Depaul L. Rev. 771

SCOTT, M. D., Scott on Computer Law, Vol. 2, 2d ed., Englewoods Cliff, New Jersey, 1998

SELVIN, P. S., « Personal Jurisdiction Over Non-U.S. Defendants - Recent Developments », (2002) 670 PLI/Lit 11

SENGAN, « Initio Violating the GPL? », (1998) Slashdot, source : <http://slashdot.org/articles/98/10/15/1520254_F.shtml>

SHANKLAND, S., « IBM Drills Linux into Oil Industry », (2002) CNET News, source : <http://news.zdnet.co.uk/story/0,,t289-s2110816,00.html>

SHANNON MARTIN, M., « Keep it Online : The Hague Convention and the Need for Online Alternative Dispute Resolution in International Business-To-Consumer E-Commerce », (2002) 20 B.U. Int'l L. J. 125

SLEE, D., « Liability for Information Provision », (1992) 3 The Law Librarian 155

SMETS-SOLANES, J.-P., B. FAUCON, Logiciels Libres : Liberté, Egalite, Business, Freepatents, Edispher, 1999, source : <http://www.freepatents.org/liberty/droit.html>

SMITH, M. J., « An Overview of the Uniform Computer Information Transactions Act: Warranties, Self-Help, and Contract Formation Why Ucita Should Be Renamed 'The Licensors' Protection Act' », (2001) 25 S. Ill. U. L. J. 389

SOUFRON, J.-B., « La licence publique générale : un système original de protection juridique pour les créations issues des sytèmes de développement coopératifs », (2002) Droit & Nouvelles Technologies, source : <http://www.droit-technologie.org/2_1.asp?dossier_id=79>

SOURCEFORGE, « Project: Back Orifice 2000 », (1999) SourceForge, source: <http://sourceforge.net/projects/bo2k>

STALLMAN, R. M., « 15 Years of Free Software », (1999) LinuxToday, source : <http://linuxtoday.com/news_story.php3?ltsn=1999-03-17-003-10-NW-LF>

STAPLETON, J., « Law, Causation, and Common Sense », (1988) 8 Oxford J. Legal Studies 111

STERN, B., « Quelques observations sur les règles internationales relatives a l'application extraterritoriale du droit » (1986) 32 A.F.D.I. 34

TALPIS, J., If I am from Grand-Mère, Why Am I Being Sued in Texas? : Responding to Inappropriate Foreign Jurisdiction in Quebec - United States Crossborder Litigations, Montréal, Éditions Thémis, 2001

TAYLOR, C., « Inspired by Work », (1999) 29 Fastcompany 200, source : <http://www.fastcompany.com/online/29/inspired.html>

TECHNICAL UNIVERSITY OF BERLIN, « Who Is Doing It? Knowing More About Developers », (2001) WIDI, source : <http://widi.berlios.de>

TERRÉ, F., P. SIMLER, Y. LEQUETTE, Droit civil - Les obligations, Éditions Dalloz, Paris, 1999, no. 819

THE APACHE GROUP, « F.A.Q. », (2002) Apache, source : <http://www.apache.org/foundation/faq.html>

THOMPSON, N., « Reboot! », (2000) 3 The Washington Monthly, source : <http://www.washingtonmonthly.com/features/2000/0003.thompson.html >

TOUBOL, F., Le logiciel : Analyse juridique, Paris, Feduci – L.G.D.J., 1986

TRUDEL, P., et autres, Droit du cyberespace, Montréal, Les Éditions Thémis, 1997

UCITA ONLINE, « What's Happening to UCITA in the States », (2001) UcitaOnline, source : <http://www.ucitaonline.com/whathap.html>

VALLOPPILLIL, V., « Open Source Software A (New?) Development Methodology », (1998) Open Source Initiative, source : <http://www.opensource.org/halloween/halloween1.html>

VERBIEST, T., « Droit international privé et commerce électronique : état des lieux », (2001) Juriscom, source : <http://www.juriscom.net/pro/2/ce20010213.htm>

VINEY, G., P. JOURDAIN, Les conditions de la responsabilité, 2e Ed., L.G.D.J., Paris, 1998

VIVANT, M., et autres, Lamy droit de l'informatique et des réseaux, Paris, Lamy, 2001

WATSON, B., « Philosophies of Free Software and Intellectual Property », (1999) RAM, source : <http://www.ram.org/ramblings/philosophy/fmp/free-software-philosophy.html>

WEBBINK, M. H., « Open Source Software – Bridging the Chasm », (2002) 691 PLI/Pat 663

WHEELER, D. A., « More Than a Gigabuck: Estimating GNU/Linux's Size », (2002) Dwheeler, source : <http://www.dwheeler.com/sloc/redhat71-v1/redhat71sloc.html>

WILCOX, J., « IBM to Spend $1 Billion on Linux in 2001 », (2000) Cnet News, source : <http://news.cnet.com/news/0-1003-200-4111945.html>

WILLIAMS, G. L., Joint Obligations, Butterworth, London, 1949

WILLIAMS, G. L., Joint torts and Contributory Negligence, Stevens & Sons, London, 1951

WOLPERT, T. G., « Product Liability and Software Implicated in Personal Injury », (1993) 60 Def. Couns. J. 519

WRIGHT, R. W., « Causation in Tort Law », (1985) 73 Calif. L. Rev. 1735

XMMS, « Changelog », (2002) XMMS, source : <http://www.xmms.org/ChangeLog>

ZAHND, B., Pluralité de responsables et solidarité, Held SA, Lausanne, 1980

Annexe A – GNU General Public license

Cette portion du document a été retirée pour respecter des règles de confidentialité ou de droits d'auteurs.
Pour consulter l'intégralité du document, veuillez vous référer au document original.

Cette portion du document a été retirée pour respecter des règles de confidentialité ou de droits d'auteurs.
Pour consulter l'intégralité du document, veuillez vous référer au document original.

Annexe B – BSD Licence

Cette portion du document a été retirée pour respecter des règles de confidentialité ou de droits d'auteurs.
Pour consulter l'intégralité du document, veuillez vous référer au document original.