L'économie du Logiciel Libre:

organisation coopérative et incitation à l'innovation

par

Dominique FORAY[+] et Jean−Benoît ZIMMERMANN[++]

Résumé:

Un logiciel libre est un logiciel dont le code source est rendu ouvertement disponible et ne peut faire l'objet d'une appropriation privative. Le mode d'organisation coopérative du "bazar", qui en fonde la production, a trouvé dans Internet un support particulièrement puissant et a pu jusqu'ici fonctionner efficacement dans la mesure où utilisateurs et développeurs des produits s'identifiaient en une seule et même catégorie. Aujourd'hui, le logiciel libre prend le virage du marché grand public et cette forme d'organisation collective peut être source de rentes privées. La viabilité de ce modèle est nécessairement dépendante de sa capacité à entretenir une dynamique de l'innovation efficiente. Elle dépend par conséquent d'un problème d'incitations individuelles, auxquelles l'éthique collective ne saurait être considérée comme un substitut efficace.

The Economics of Open Source Software:

Cooperation and incentives to innovate

Abstract:

An open source software is defined by the availability of its source−code that cannot be privately appropriated. Its production is based on a special cooperative organisation mode, called the "bazaar", which functions particularly well in the context of Internet. Its functional viability till now was related to the fact that the users and developers of these software products considered themselves in the same category of actors. Today, open source software is entering the market and the existing form of organisation can now be the source of profits. The model's viability is related to its ability to sustain a high level of innovation. Hence, it depends on individual incentives to participate, and public spiritedness can no longer be relied upon as an effective substitute.

JEL: D23, H41, L86, O31, O34

[+] Directeur de Recherche CNRS, IMRI, Université de Paris−Dauphine et OCDE/CERI − dominique.foray@oecd.org

[++] Directeur de Recherche CNRS, GREQAM−EHESS, Marseille − jbenoit@ehess.cnrs−mrs.fr

Introduction*

Les problèmes de propriété intellectuelle qui se posent à l'économie de l'innovation relèvent de trois séries de questions emboîtées. Premièrement, une connaissance nouvelle peut−elle ou doit−elle être appropriée de façon privée? Deuxièmement, si oui, sous quel régime de droit de propriété intellectuelle doit−elle être enregistrée? Et troisièmement, s'il s'agit du régime des brevets, quelle pratique de création et d'usage du brevet s'avérera−t−elle favorable à l'établissement d'un bon équilibre entre la protection de l'inventeur et l'usage social de son invention? Il est remarquable de constater que le logiciel, comme d'ailleurs les créations génétiques, sont des objets pour lesquels il n'y a de réponse assurée à aucune de ces trois questions. La première question rend compte de l'opposition entre logiciel propriétaire et logiciel libre et les termes de cette question sont principalement économiques comme nous le verrons par la suite : ce n'est pas le "peut−il être" qui pose problème comme dans le cas des créations génétiques (question notamment éthique) mais c'est plutôt le "doit−il être" qui fait question. La deuxième question rend compte de l'incertitude sur le régime de propriété intellectuelle, qu'il convient d'activer dès lors que l'on a répondu oui à la question sur l'appropriation privée. La protection par brevet est−elle préférable à la protection par droit d'auteur ou bien, compte tenu du caractère insatisfaisant de ces deux régimes pour les logiciels, ne vaudrait−il pas mieux créer un droit *sui generis*, ainsi qu'on l'a fait notamment pour les variétés végétales ? Enfin, la troisième question fait référence aux situations de blocage qui peuvent découler d'une mauvaise utilisation du brevet : soit parce que la surface du brevet est très étendue et bloque les projets d'innovation ultérieurs sur le domaine ; soit parce qu'un nombre très important de brevets couvre les composants élémentaires d'une innovation, ce qui disperse la base de connaissances et ralentit le développement d'un produit final par un agent particulier.

Cet article porte principalement sur les deux premières questions et les traitera dans un ordre inversé. C'est parce qu'aucun des régimes de droit de propriété intellectuelle n'offre une solution totalement satisfaisante − et cela compte tenu de ce que l'usage de la propriété intellectuelle nous situe de toute façon dans un cadre de solutions de second rang (section 1) − que l'on "remontera" vers la question du logiciel libre. Après en avoir identifié les bonnes propriétés du point de vue de l'économie de l'innovation (section 2), nous évoquerons les fondements institutionnels du logiciel libre (section 3) puis nous examinerons le rôle des incitations individuelles dans ce modèle (sections 4 et 5).

* Les auteurs tiennent à remercier toutes les personnes qui ont débattu avec eux des questions abordées ici et tout particulièrement Nicolas Curien et Thierry Pénard pour leur commentaires constructifs sur une première version de cet article.

1. Les problèmes de la propriété intellectuelle dans le domaine du logiciel informatique

La question de la protection de la propriété intellectuelle pour le logiciel s'est posée à partir des années 70, avec son accès au statut de marchandise, dès lors que la législation antitrust américaine avait mis un coup d'arrêt aux pratiques de facturation liée introduites par IBM. Longtemps classé au rang des activités de services ou considéré comme produit joint de la construction d'ordinateurs, le logiciel est donc considéré aujourd'hui comme une activité maîtresse et autonome au sein de l'industrie informatique. Il reste cependant une activité industrielle très particulière, essentiellement vouée au développement de technologies de process de calcul ou de commande et à leur mise en forme opérationnelle, en conformité avec des standards opératoires : formatage des données, interfaces d'entrée−sortie, protocoles de communication, interfaces homme−machine. Marchandise un peu particulière pour l'analyse économique, il peut s'apparenter à de la connaissance, bien qu'en même temps sa mise en ?uvre par un utilisateur donné ne requière pas de la part de celui−ci une appropriation effective. Mais, compilé, c'est−à−dire traduit en langage−machine, sous la forme d'un "code−objet", il revêt un caractère particulier d'autonomie ; il n'est plus seulement information sur la conduite d'un processus, d'une action, il est lui−même action, il est processus de coordination de stations logiques (les modules d'une architecture informatique), en vue de réaliser une tâche de calcul, de traitement et de manipulation de données, de commande, c'est−à−dire une tâche de nature informationnelle (Zimmermann, 1995).

La protection de la propriété intellectuelle répond à une double préoccupation. Dans un sens, elle constitue une incitation à l'innovation, en octroyant à l'inventeur un pouvoir de monopole temporaire pour l'exploitation industrielle et commerciale de son invention. Dans l'autre, elle vise, du moins en ce qui concerne le système des brevets, une meilleure diffusion de la connaissance, de par l'obligation qui est faite à l'inventeur, pour obtenir la protection, de révéler les principes et la nature du produit ou du procédé sur lequel porte la requête. Ces deux ordres opposés de préoccupations constituent le fondement de ce que l'on désigne parfois comme le "dilemme schumpétérien", auquel une solution est généralement recherchée en termes d'étendue[1] et de durée optimale de la protection en vue de "constituer un facteur d'incitation suffisant ex−ante sans trop entraver l'imitation et la diffusion ex−post" (Heraud, 1995). Une telle solution ne peut être considérée comme satisfaisante que dans un contexte défini en fonction des caractéristiques de l'industrie et de la dynamique de l'innovation, tout particulièrement en ce qui concerne le rythme de l'innovation et ses propriétés cumulatives.

[1] Selon Scotchmer (1991), le système nord−américain ayant tendance à octroyer une forte étendue de la protection au premier inventeur, risque de réduire les incitations pour les innovations suivantes, secondaires ou induites.

S’agissant du logiciel, le problème principal réside dans l’incertitude sur la classe de droit de propriété intellectuelle (droit d’auteur ou brevet) qui serait la plus adaptée aux caractéristiques de l’objet. Selon Lucas (1987), "le programme qui est au coeur du logiciel se caractérise d’abord par son contenu en tant que procédé permettant de tirer parti des ressources de la machine en vue d’un résultat déterminé. A ce titre, sa protection pose un problème de brevetabilité. Mais, en même temps, il se présente en lui−même, apparemment au moins, comme une ?uvre de l’esprit susceptible de donner prise au droit d’auteur (...). De là vient qu’(il) puisse *a priori* prétendre à la fois au bénéfice du brevet d’invention et à celui du droit d’auteur. De là vient aussi qu’il ne puisse trouver commodément sa place ni dans l’une ni dans l’autre de ces deux branches de la propriété industrielle, ce qui fait toute la différence en la matière".

Néanmoins, plutôt que de concevoir un cadre juridique spécifique, l’orientation générale, aux Etats−Unis, puis en Europe et au Japon, a consacré la prédominance du droit d’auteur. Cette unanimité s’accommode toutefois de variantes, relatives tant aux contextes juridiques nationaux de la propriété intellectuelle, qu’au rôle et à la place de la jurisprudence (le droit anglo−saxon, beaucoup moins codifié que le droit latin, lui confère à cet égard une fonction de premier ordre), voire aux attitudes culturelles vis−à−vis de la propriété intellectuelle.

En Europe, et notamment en France, le système juridique de la propriété intellectuelle a jusqu’ici exclu assez clairement le logiciel du champ d’applicabilité des brevets, hormis dans le cadre d’un procédé industriel brevetable, la protection restant alors limitée au cadre du procédé de référence. Aux Etats−Unis, en revanche, la jurisprudence a ouvert la voie, dans le courant des années quatre−vingt−dix, à une dérive vers un recours accru au système des brevets, suscitant une opposition radicale entre partisans et opposants de la brevetabilité du logiciel. Depuis lors, le débat s’est ouvert en Europe, en vue de la nécessaire détermination d’une approche commune. Une étude a été réalisée à la demande de la Commission Européenne sur l’impact économique de la brevetabilité des programmes informatiques (Hart, Holmes et Reid, 2000). Une consultation ouverte a été lancée, via l’Internet, parallèlement à la Conférence Intergouvernementale de Munich de Novembre 2000.

En soi, l’observation des effets de ces pratiques sur le comportement des acteurs de l’industrie du logiciel aux Etats−Unis semble indiquer que la brevetabilité du logiciel conduirait à une forme de cloisonnement des savoirs et des procédures, qui interdirait ou du moins freinerait toute pratique combinatoire. Parmi la multitude de brevets accordés, on trouve un grand nombre de procédures, voire d’algorithmes. Beaucoup d’entre eux apparaissent très éloignés des critères de nouveauté et d’originalité qui conditionnent, en principe, la délivrance d’un brevet. Ils s’apparentent souvent à des procédures conventionnelles et largement répandues, à

l'utilisation desquelles les détenteurs de brevets pourraient donc désormais s'opposer. Un des aspects principaux du problème est que le législateur ne peut statuer ici de manière acceptable en l'absence d'un état de l'art patenté, alors même que la législation américaine est supposée appliquer le principe du *first to invent*, par opposition à la plupart des autres régions du monde qui pratiquent celui du *first to file* (Heraud, 1995).

Dans des travaux récents, Bessen et Maskin (2000) mettent en avant l'argument que, dans une industrie comme le logiciel, où l'innovation est à la fois *séquentielle*[2] et *complémentaire*[3], l'introduction d'un système fort de propriété intellectuelle fondé sur les brevets aurait un effet inhibiteur sur la dynamique de l'innovation. En revanche, un régime de faible protection, parce qu'il ouvre des possibilités d'imitation, tend paradoxalement à encourager les efforts de recherche et d'innovation. Ces résultats ne tiennent certes pas compte de la procédure particulière des brevets qui permet à l'auteur d'une innovation de perfectionnement d'obtenir une licence d'exploitation d'un brevet pionnier antérieur. L'apport du modèle théorique semble cependant bien corroboré par l'observation de l'industrie américaine du logiciel, qui a connu une stagnation, voire une régression des efforts de R&D, au moment même où la jurisprudence accordait de plus en plus largement la délivrance de brevets pour des programmes d'ordinateurs. Et cette tendance régressive apparaissait particulièrement marquée pour les firmes qui brevetaient le plus.

Mais, de son côté, la législation du droit d'auteur, appliquée au logiciel, a cette particularité qu'elle procure la protection de l'?uvre sans obliger à en dévoiler le contenu. La double nature de l'expression d'un programme, sous la forme d'un code−source explicite et d'un code−objet, après compilation, en langage machine, permet de disposer des potentialités du produit logiciel sans nécessiter l'accès à son expression explicite. Par conséquent, l'immense majorité des éditeurs commercialisent les programmes sous la seule version de leur code−objet et gardent secret le code−source. "La création étant fonctionnelle, l'utilisateur peut obtenir le résultat attendu : les fonctions ou, à l'américaine, les fonctionnalités, sans avoir à se préoccuper de ce que les juristes ont élu comme objet de droit : la forme" (M.Vivant, 1993). La question immédiatement corollaire est celle de l'ingénierie inverse, ou décompilation, qui vise à remonter au code−source à partir du code−objet. Malgré une certaine diversité des législations et des jurisprudences, la limite de tolérance en la matière, telle que l'a exprimée la Directive Européenne, est celle de la révélation des interfaces en vue de permettre l'interopérabilité des programmes, à condition toutefois que l'éditeur du logiciel n'ait pas accepté de révéler par lui−même ces spécifications d'interfaces. Au delà de cette limite, la

[2] dans le sens du caractère cumulatif du progrès technique, chaque invention successive prenant appui sur les précédentes.

[3] dans le sens où des recherches menées en parallèle et avec des approches différentes sur un même sujet accroissent la probabilité de réussite à horizon temporel donné.

violation du droit d'auteur est considérée comme avérée. Ainsi le droit d'auteur, appliqué aux logiciels ne répond−il qu'à l'une des deux préoccupations à l'origine de la protection de la propriété industrielle, la protection de l'inventeur et non la diffusion des idées. "Pour la première fois depuis Sybaris, cinq siècles avant Jésus−Christ, qui dans sa loi sur l'appropriation des recettes de cuisine en imposait la divulgation au public, voici donc réconciliés propriété et secret!" (M.Vivant, 1993).

En réalité, la véritable opposition n'est pas celle entre brevets et droit d'auteur, mais entre deux modèles de développement du logiciel, deux approches fondées sur des principes radicalement opposés et donc prédisposées à diverger. Ces deux modèles sont ceux fondés sur l'approche marchande et de l'appropriabilité, d'une part, et ceux fondés sur la culture de la "connaissance pure" et de la coopération, d'autre part. Cette opposition est précisément celle qui divisait Bill Gates et ses collègues académiques dans les années soixante−dix au MIT. Le premier ne concevait pas que d'autres puissent utiliser librement le fruit de son travail de conception. Ses collègues, baignés de culture scientifique et communautaire, considéraient en revanche qu'un code est de la connaissance pure et constitue à ce titre un bien public qui ne peut être approprié de manière privative (Smets et Faucon, 1999). Si Bill Gates a connu la carrière industrielle que l'on sait, les seconds, qui s'inscrivent dans la traduction et la logique de la *science ouverte* (Dasgupta et David, 1994), ont été à l'origine du *logiciel libre*. Mais ils se situent aussi dans la tradition de tout un milieu de développeurs qui, même intégré dans un contexte d'entreprises, a fondé ses pratiques sur la diffusion des idées et le caractère cumulatif du progrès technique, aussi bien en matière de concepts et d'outils, que d'algorithmes de résolution de problèmes ou de méthodes et de styles de programmation.

2. Les bonnes propriétés du logiciel libre

Un logiciel libre est un logiciel dont le code−source, c'est−à−dire la série d'instructions qui forme le programme avant la compilation, est rendu ouvertement disponible et ne peut faire l'objet d'une appropriation privative. Le développement des logiciels libres est fondé sur le volontariat et le bénévolat des participants, dans un mode d'organisation coopératif qui s'appuie largement sur les commodités organisationnelles issues d'Internet[4]. Le produit phare en est aujourd'hui le système d'exploitation Linux, inspiré d'Unix et portable sur des architectures de micro−ordinateurs. Mais beaucoup d'autres produits sont également disponibles auprès d'entreprises spécialisées ou même, le plus souvent, téléchargeables gratuitement sur Internet : serveurs de site web, utilitaires bureautiques, scientifiques, de traitement d'image ?

[4] Voir à ce propos D.Desbois, N.Jullien, T.Pénard, A.Poulain−Maubant, J.Vétois et JB.Zimmermann (Eds.) (1999).

Ce que nous souhaitons montrer dans cette section est que ce qui caractérise le modèle du logiciel libre n'est pas principalement la gratuité, qui ne constitue d'ailleurs pas une règle, mais surtout la liberté de modifier et d'améliorer les versions existantes, ainsi que l'impossibilité de s'approprier les améliorations apportées. L'économie du logiciel libre est donc d'abord un modèle d'innovation, d'accumulation de la connaissance et de recombinaison des savoirs et non pas une stratégie de marketing de type "offre gratuite de produits".

Le fait de donner à l'utilisateur l'accès au code source permet d'engendrer des effets très importants d'apprentissage par l'usage, c'est à dire d'exploiter au mieux une très grande intelligence distribuée : les millions d'usagers qui révèlent des problèmes et les milliers de programmeurs qui trouvent comment les éliminer. D'après les termes de la Free Software Foundation, chacun peut utiliser le code et le modifier, à la condition de communiquer la modification à l'organisation, pour qu'elle soit vérifiée et évaluée. On retrouve ici " les bonnes propriétés " de la distribution de la connaissance et des systèmes de savoir ouvert : seule la circulation rapide et élargie des savoirs permet de bénéficier du potentiel unique d'un très grand nombre d'individus compétents. D'un point de vue concret, une distribution rapide de la connaissance facilite la coordination entre les agents, elle diminue les risques de duplication entre projets de recherche et elle permet de concentrer l'apprentissage sur les « meilleures » inventions. Elle constitue en outre une assurance de qualité, puisqu'une fois la connaissance produite, elle est testée et donc vérifiée par de nombreux agents. Enfin, en propageant la connaissance au sein d'une population diversifiée de chercheurs et d'entrepreneurs, elle accroît la probabilité de découvertes et d'inventions ultérieures, en même temps qu'elle diminue le risque que cette connaissance soit détenue par des agents incapables d'en exploiter les potentialités (David et Foray, 1995).

Ces bonnes propriétés du savoir ouvert sont amplifiées dans le cas des logiciels libres:

- premièrement, un logiciel constitue un objet scientifique ou technologique complexe, qui est donc caractérisé par des processus d'apprentissage quasi−illimités : un système composé de milliers de développeurs travaillant sur le même logiciel durant une longue période restera longtemps, presque indéfiniment, dans une phase de rendements croissants − ce qui ne serait pas le cas d'un système de milliers d'ingénieurs travaillant à l'amélioration d'une brouette.
- Deuxièmement, un logiciel est exprimé sous la forme d'un ensemble d'instructions codifiées qui circulent parfaitement sur le réseau électronique. Ainsi, la circulation des améliorations apportées est rapide, parfaite, et son coût marginal est quasi−nul. Ceci accroît l'efficacité globale du processus de recherche collective. Ceci accroît aussi l'incitation des agents à envoyer de l'information. Comme l'ont bien montré Lakhani et

Von Hippel (2000) dans leur étude empirique du système Apache, si tant d'agents acceptent de coopérer, c'est notamment parce que le temps passé à envoyer de l'information ne dépasse pas cinq minutes.

- Troisièmement, les logiciels appartiennent à une certaine classe de technologie ayant la propriété particulière de réduire, voir même d'annuler la distance entre producteurs et consommateurs (Quah, 1999). Les millions d'usagers qui révèlent les problèmes sont, pour une part, les développeurs qui proposeront les solutions. Le va−et−vient entre identification des problèmes et formulation des solutions est donc quasi−instantané. Là encore, Lakhani et Von Hippel (2000) ont un résultat intéressant : les gens s'entraident parce que la solution recherchée par l'un est « sur étagère ». Autrement−dit, elle est connue par un autre qui n'aura donc pas à consentir un effort très grand pour retrouver et envoyer celle−ci.

Ces différents aspects ont une traduction évidente et indiscutable en termes de performance des produits. Le taux d'innovation, la qualité et la fiabilité sont nettement supérieurs à ce que l'on trouve dans le monde des logiciels propriétaires. Si les évidences empiriques sur ce point restent parcellaires, le simple fait que de nombreuses entreprises et administrations se rallient au monde du logiciel libre est un bon indicateur du potentiel des performances attendues.

Pour ces raisons, le système Linux ne doit pas simplement être analysé en tant que mode privilégié d'expression des convictions éthiques et altruistes des individus, ou même d'un sentiment communautaire, mais il doit être surtout vu comme un mécanisme générateur d'efficacité économique.

3. Les fondements institutionnels d'un modèle coopératif

Le mode de développement coopératif du logiciel libre, fréquemment désigné comme le modèle du "bazar" (Raymond, 1998), prend appui sur le potentiel de diffusion et de communication offert par l'Internet et la mutualisation de ressources qu'il autorise. Il est fondé sur le principe de la disponibilité des codes−sources, afin de permettre à n'importe quel développeur de réaliser toutes modifications ou améliorations qui pourraient lui sembler utiles. Mais ces modifications n'ont d'intérêt, à un niveau collectif, que si leur auteur les rend à son tour publiques, afin qu'elles puissent être éventuellement intégrées à la construction d'ensemble[5]. Le fonctionnement effectif d'un mode d'innovation continu, fondé sur une logique de don/contre−don (Dang Nguyen et Pénard, 2001), nécessitait de se doter des outils

[5] Pour ce qui concerne Linux, le code source initial représentait environ 50 000 lignes de programme ; il dépasse aujourd'hui le million. Un petit groupe informel d'experts sélectionne et valide les propositions d'amendements ou d'ajouts. Mais ce type de coordination informelle, sur un modèle associatif, risque d'atteindre rapidement ses limites, avec la croissance du nombre des participants, d'autant qu'il n'y a ni financement ni rémunération des travaux de programmation et de coordination. (Ganagé et Bonnet, 1998).

juridiques minimaux indispensables pour protéger la communauté des développeurs contre les comportements opportunistes, et notamment contre l'appropriation privée de tout ou partie du code[6]. Pour éviter cet écueil, qui est celui du logiciel du domaine public, il était nécessaire de rentrer dans le champ du droit d'auteur, avec cependant un positionnement très différent, voire opposé à celui du copyright fondé sur la propriété privée de l'expression. De ces préoccupations sont nés les principes du "copyLeft" institués par la Free Software Foundation. Ces principes sont ceux de la libre utilisation du code et de sa libre modification, sous réserve d'en communiquer la teneur à l'organisation en vue de sa "vérification" et sa "labélisation". Ainsi, dans le cas de Linux, "la décision ultime d'intégrer ou non les changements est du ressort de l'un des "dictateurs bienveillants" qui "dirigent" le secteur concerné par le projet Linux" (Alper, 1999).

En réalité, le besoin de principes de cet ordre s'est imposé dès lors que se mettait en place un premier projet collectif de développement, GNU[7] en 1983. La nécessité de pourvoir à un cadre juridique susceptible de garantir le respect des principes du développement coopératif a donné naissance à la licence GPL ou GNU−*General Public License*[8]. Le principe fondamental en est que, les codes concernés étant libres, chaque programme qui intègre des lignes de code GPL doit aussi être disponible sous licence GPL. Ainsi, les auteurs n'abandonnent pas leurs droits, mais la seule rente de monopole que ces droits autoriseraient dans un régime de copyright. Le logiciel reste de la sorte la propriété de ses créateurs. Ceux−ci autorisent quiconque à en faire usage (modifications, améliorations, compléments ?), sous la seule condition que toute nouvelle version puisse, elle aussi, circuler librement.

Ceci n'est pas exclusif d'une éventuelle commercialisation de ces programmes et ne limite donc pas le cadre du logiciel libre à la sphère non−marchande. Mieux encore, certaines entreprises ont fondé leurs stratégies de développement sur les principes de développement coopératif et de libre accès aux codes−sources, élargissant ainsi le cadre conceptuel du logiciel libre à un champ que l'on désigne fréquemment comme celui des OSS (*Open Source Software*). Cet élargissement peut, soit porter sur des activités de développement, d'édition et de distribution de logiciels libres, à l'instar de la société américaine Redhat Software ; soit conduire à une ouverture des sources de certains produits de la part de sociétés de logiciel traditionnelles. Ainsi, un logiciel peut fort bien devenir "libre" après avoir été protégé et payant, comme l'a fait Netscape en 1998 pour son produit *Communicator*, dont le code−source est librement diffusé. Cette stratégie avait pour objectif de chercher à contrer

[6] Se prémunir contre l'appropriation privée du code n'exclut pas pour autant une éventuelle exploitation marchande de ce code.

[7] *GNU is Not Unix*, projet développé en vue de créer une plate−forme "libre" de type Unix. Ce projet s'inscrivait, de manière prémonitoire, en opposition aux réflexes propriétaires qui allaient, dès la fin des années quatre−vingt, reprendre le dessus dans le monde Unix et en faire éclater le potentiel d'ouverture.

[8] Pour une analyse fine des fondements juridiques de la GPL, voir Clément−Fontaine (1999).

Microsoft sur son propre terrain, en mobilisant un important effectif de programmeurs sur un produit−clef, afin de l'imposer comme standard de fait. La logique économique pour Netscape était de réaliser ensuite ses gains sur les produits complémentaires, lesquels peuvent sans difficulté rester de l'ordre des architectures propriétaires.

Cette implication des activités marchandes dans le domaine du libre a dicté la nécessité de concevoir un élargissement du cadre juridique du copyLeft, au delà de la trop stricte licence GNU−GPL. De là est né un ensemble de licences plus hybrides, visant à concilier développement coopératif et intérêts privés au sein des OSS. Les modèles abondent et traduisent une variété de situations et de stratégies particulières, qui ne peuvent s'intégrer dans aucun régime universel. Ils correspondent à des positionnements divers qui associent, en des proportions et des modes variés, la logique du copyright à celle du copyleft (Smets et Faucon, 1999).

Ainsi, dans son opération d'ouverture de *Communicator*, Netscape a mis au point deux licences complémentaires : NPL (*Netscape Public License*), qui permet d'intégrer les nouveaux développements à ses serveurs sans nécessiter de faire entrer ces derniers dans le champ du copyleft, et MPL (*Mozilla Public License*)[9], qui couvre le développement de nouveaux modules. De la sorte, toute modification à partir du code source de *Communicator* doit être publiée, mais Netscape se réserve le droit d'intégrer dans *Communicator* des modules propriétaires et se garde de contraindre son utilisation dans sa propre gamme de produits. De la même façon, Sun a créé la licence SCSL (*Sun Community Source License)* pour "libérer" son protocole de communication *Jini*. De cette manière, il se réserve l'exclusivité de la certification en compatibilité de tout produit issu de *Jini* et destiné à une exploitation commerciale. D'autres, comme Novell ou IBM, ont suivi des voies similaires et de multiples variantes combinent les droits d'utilisation ou d'intégration des OSS avec les logiciels propriétaires et leur usage marchand et non−marchand. L'idée générale est au fond de rendre compatible un modèle d'accès libre à la technologie, favorisant la progression du savoir et l'adoption du standard, avec un modèle d'exclusivité sur certains éléments, qui garantit un certain niveau de revenus.

4. L'incitation dans un système d'innovation collective

Le modèle du logiciel libre aborde aujourd'hui le virage de son entrée dans le monde marchand et de son élargissement au marché grand public. Ce virage est sans doute particulièrement délicat, car il découle du succès, voire de l'engouement, que les logiciels libre suscitent depuis peu auprès d'une catégorie élargie d'utilisateurs. Jusqu'à une période

[9] Mozilla est le nom que Netscape a donné au projet de développement coopératif né de l'ouverture de Communicator.

récente, le développement du libre était souvent considéré comme une affaire d'informaticiens, dans la mesure où il ne débordait guère de cette communauté virtuelle d'utilisateurs−développeurs sur laquelle se fonde le modèle du "bazar". Or voilà que le succès du libre finit par atteindre la catégorie des simples utilisateurs, en commençant par la part la plus dynamique d'entre eux, souvent très jeune, férue d'Internet et qui contribue à diffuser, à la fois chez les particuliers et dans les entreprises, des concepts innovants, des pratiques en évolution constante, des produits et des standards fondés sur des règles nouvelles. A travers cette catégorie d'utilisateurs éclairés, c'est le marché grand public (particuliers et entreprises), un marché de masse, qui commence à s'ouvrir au monde du libre. L'émergence de cette nouvelle catégorie d'utilisateurs, ainsi que l'intérêt du monde marchand pour le modèle du libre, traduisent une situation nouvelle où des agents sont en mesure de réaliser des bénéfices privés par la commercialisation ou l'utilisation des résultats des efforts collectifs de la communauté des développeurs dont la participation n'est, à la base, fondée sur aucune rémunération. Ceci aggrave bien entendu, pour ces développeurs, les problèmes d'incitations.

L'économie de la science et de la technologie a analysé de nombreux systèmes de "savoirs libres", faisant référence, soit à des situations locales et historiques −les Canuts lyonnais (Foray et Hilaire Perez, 2000), la métallurgie du Lancashire (Allen, 1983)−, soit à des situations générales −la science ouverte (Dasgupta et David, 1994). Or, dans tous ces cas, on s'aperçoit qu'au delà des bonnes propriétés de la circulation rapide et élargie des savoirs, qui confèrent au système son efficacité économique, s'impose toujours une dimension d'incitation individuelle. Celle−ci requiert le fonctionnement de mécanismes visant à offrir un crédit aux inventeurs, sans que ce crédit ne se traduise en droit d'exclusivité. En d'autres termes, un système de savoir ouvert requiert des mécanismes d'incitation, compatibles avec le principe de partage et de circulation des inventions.

C'est le système de la règle de priorité dans le domaine de la science ouverte, qui donne à celui qui invente et qui publie son invention un droit de propriété moral. Ce droit ne se concrétise pas par une situation d'exclusivité, mais il est à la base de la constitution d'un capital de réputation, élément décisif pour obtenir des subventions ou gagner un prix. La compatibilité entre le caractère de bien public de la connaissance et le mécanisme d'incitation est ainsi assurée par ce mécanisme remarquable, qui permet de créer un actif privé, une forme de propriété intellectuelle, qui résulte de l'acte même de renoncer à la possession exclusive de la nouvelle connaissance. En ce sens, la règle de priorité offre des incitations non marchandes (elle crée des contextes de course ou de tournoi) à la production de biens publics.

Dans le cas de la soierie lyonnaise du XVIII° siècle, on accordait une prime au Canut qui inventait un nouveau procédé, en échange de la libre circulation de l'invention. En outre, le

montant de la prime était calculé en fonction de la participation de l'inventeur lui−même à la diffusion et à l'adoption par les autres de son invention. Là encore, il y a compatibilité entre incitations individuelles et mécanisme collectif. Cependant, ce système n'évite pas la question difficile de l'estimation *ex ante* de la valeur sociale de l'invention.

Dans le monde du logiciel libre, l'effort collectif d'amélioration d'une base de logiciels engendre pour chacun une externalité d'usage. Cette externalité est générale au sein de la communauté des utilisateurs−développeurs, en ce sens qu'elle équivaut à la production d'un bien public non−rival et non−exclusif. Si elle constitue, à une échelle collective, une incitation à coopérer, elle engendre aussi à l'échelle individuelle, et lorsqu'elle est considérée isolément, une incitation à la défection et au comportement opportuniste (le bénéfice de l'externalité n'est pas contingent de la coopération). Indispensables à sa viabilité, les incitations individuelles sont donc présentes et au c?ur de la dynamique du modèle. Au delà de l'éthique académique (sentiment de contribuer à une ?uvre collective), elles sont principalement de deux ordres : l'apprentissage et la réputation. Une condition supplémentaire tient au faible coût marginal du travail exigé pour contribuer à l'entreprise coopérative.

Le premier ordre d'incitations est relatif aux effets d'apprentissage. La programmation est un art non stabilisé et dont les méthodes restent encore fondées sur des procédures et méthodes relativement artisanales et peu codifiées, y compris au sein des organisations productives des grands éditeurs[10]. Le fait, pour un développeur, de suivre et de participer aux efforts de la communauté du libre, a pour effet en retour de contribuer à améliorer son propre niveau de compétence informatique, et ceci pour au moins deux raisons. La première relève d'un apprentissage par la pratique, issu d'une participation à un effort collectif de développement. La seconde résulte de la confrontation avec un code qui, lui même, résulte d'une multiplicité de contributions relevant d'une diversité de savoirs, de méthodes et de styles de programmation. Cet apprentissage par interaction constitue une source d'amélioration individuelle de la compétence du programmeur, bien au−delà de ce que celui−ci pourrait attendre de sa participation à une équipe plus traditionnelle de développement, fût−ce même au sein de grandes entreprises, dont l'organisation reste à ce jour cloisonnée et fractionnée en petites équipes. L'incitation à coopérer est claire pour le développeur individuel, dans la mesure où il s'agit pour lui de rester à la hauteur de l'évolution des savoirs, et ce d'autant plus qu'il se situe dans le contexte d'un jeu répété.

[10] Zimmermann (1998)

Le deuxième ordre d'incitations joue un rôle pivot dans notre propos. Il est relatif aux effets individuels de réputation[11] et fonctionne de manière très similaire à celui en vigueur dans le milieu académique. Il résulte d'un processus de reconnaissance par les pairs, dans la mesure où les modifications et améliorations acceptées et labellisées dans un logiciel libre comportent la signature explicite de leur auteur, qui apparaît comme telle dans les copies en circulation du programme. Ici, cependant, ce capital de reconnaissance a la vertu essentielle de pouvoir être converti sur un mode pécuniaire, à travers une embauche dans une entreprise où un accès privilégié à des sources de financement, à la différence du monde académique où cet objectif à terme reste une modalité secondaire.

On pourrait rassembler enfin dans un ensemble de conditions favorables, les facteurs qui contribuent à la minimisation du coût marginal de l'effort nécessaire pour contribuer à l'entreprise coopérative. Lakhani et von Hippel (2000) montrent l'importance de ces conditions, dans le cas d'un système en ligne d'assistance entre usagers. Le système fonctionne, non seulement parce qu'il y a des bénéfices individuels (apprentissage et réputation) mais aussi parce qu'il n'y a pratiquement pas de coût : il y a dans le système au moins un usager pour lequel l'effort de trouver la solution demandée par un autre usager est minime (la probabilité que la solution soit disponible quelque part dans le système est très grande) et le coût de transmission de cette solution est quasi−nul. L'effet Internet (au sens d'une minimisation du coût marginal de stockage et de transmission de l'information) est ici très important.

Dans le contexte qui prévalait jusqu'à une période récente, les individus qui tiraient bénéfice de l'utilisation des logiciels libres étaient, pour l'essentiel, ceux−là mêmes qui participaient à leur développement. La participation à l'effort d'amélioration de la base de logiciels supposait, pour chacun, que les bénéfices tirés de l'usage de cette base et les gains espérés en termes d'apprentissage et de réputation se cumulent en un avantage suffisamment élevé, pour compenser le coût minime de la participation. La dynamique interne du bazar montre que cette condition pouvait être considérée comme généralement satisfaite sans que la présence de comportement opportunistes ne mette en péril la viabilité du système[12]. Nous montrerons à la section 5, sur la base d'un modèle de coalition très simplifié, que l'équilibre d'une telle configuration correspond à des conditions internes et externes assez facilement réalisables.

[11] Lerner et Tirole (2000) regroupent, au sein d'une même catégorie dite "signal incentives", deux types d'effets "distincts quoique difficiles à distinguer", d'une part les effets de réputation et d'autre part les effets de satisfaction ("ego gratification") directement liés à la reconnaissance par les pairs.

[12] − Dans un article récent sur les comportements coopératifs dans la science ouverte, P.David (1997) développe un modèle d'interactions stochastiques entre individus rationnels, engagés dans un processus continu d'observation expérimentale, échange d'informations et révision de choix. Il montre que le fonctionnement coopératif du réseau peut tolérer les comportements opportunistes. Autrement dit, les performances cognitives du réseau (la capacité à produire collectivement un énoncé scientifique, tout en préservant la diversité des opinions) sont assurées tant que le nombre de comportements opportunistes est maintenu en dessous d'un seuil critique.

Nous nous interrogerons également sur les effets de taille et montrerons qu'une simple hypothèse de stricte concavité dans la production du bien public engendre un effet limite sur la taille qui justifie la nécessité d'organiser la communauté du libre autour de projets dans lesquels ne tendent à s'investir qu'une fraction du total des développeurs.

Mais la principale question qui nous préoccupe ici est celle de savoir ce qu'il advient de cet équilibre, dans un contexte où un nombre croissant d'utilisateurs simples bénéficient des efforts de développement de la communauté du libre, sans être redevables d'aucune contrepartie, notamment sur le plan pécuniaire. Cet élargissement de la population concernée résulte d'un ensemble de trois facteurs complémentaires. Il s'agit, en premier lieu, de l'arrivée à un certain niveau de maturité des produits du logiciel libre ; les produits sont dorénavant très performants et d'un niveau de fiabilité très élevé, du fait de la capacité de test inégalée que représente le modèle du bazar comparativement à toute structure d'entreprise. En deuxième lieu, cet élargissement est porté par l'extraordinaire moyen de diffusion que représente Internet et qui surajoute cette capacité aux propriétés d'interconnexion de la communauté des développeurs. En dernier lieu, la constitution d'entreprises commerciales, vouées à la distribution de logiciels libres édités avec des designs plus élaborés, des interfaces−utilisateur, des manuels d'utilisation et des utilitaires d'aide, constitue un facteur décisif pour une catégorie d'utilisateurs peu technicienne et peu rompue à la navigation sur Internet. Ce troisième type d'acteurs dans le monde du logiciel libre trouve à concilier logique commerciale et principes de non−appropriation, en fondant sa rentabilité sur la vente de services liés : "hot line", mises à jour, "débogage", ingénierie de systèmes d'information, formation ... Il y aurait lieu, en outre, d'ajouter à ces trois facteurs explicatifs un quatrième argument, qui tient à la logique de l'affrontement entre deux modèles de production radicalement opposés. Dans le combat que mènent bon nombre de "militants" du libre contre l'appropriation privée des codes et contre la structure oligopolistique de l'industrie du logiciel, l'adoption des logiciels libres par un nombre sans cesse croissant d'utilisateurs constitue un indicateur pertinent, en termes de part de marché, du terrain gagné.

On comprend facilement que la possibilité, pour un nombre croissant d'agents, de tirer des revenus conséquents de la commercialisation et de l'utilisation de logiciels libres, sans avoir contribué à leur développement, remet en cause de manière radicale la tolérance du système aux comportements opportunistes. Tout d'abord l'élargissement d'audience, auquel fait face aujourd'hui le libre, constitue un clair facteur de reconnaissance, donc de succès, et il serait limitatif de n'y voir que la multiplication de comportements opportunistes. Au niveau de chaque programmeur, ensuite, les succès et la reconnaissance du libre confortent sans aucun doute les effets de réputation dont bénéficient les développeurs, mais ils engendrent aussi une insatisfaction relativement à l'absence de rémunération des efforts consentis. Cette

frustration, dont l'intensité s'accroît avec l'élargissement du "marché" du libre, finit par peser, dans l'arbitrage des individus, d'autant plus fort que leurs attentes en matière d'apprentissage et de réputation sont peu élevées. Nous montrerons dans la section 5 que les individus le plus facilement en manque d'incitations sont précisément ceux qui offrent le plus fort niveau de compétences. Leur défection risquerait de précipiter le système du bazar dans une dynamique de démissions et d'effondrement hors de tout équilibre. C'est le succès du logiciel libre qui en aurait alors engendré la chute.

Maintenir l'équilibre, alors même que les adopteurs et bénéficiaires du logiciel libre sont de plus en plus nombreux, suppose par conséquent l'instauration de mécanismes nouveaux susceptibles de compenser les effets de frustration à l'origine des défections. Cette compensation d'incitation peut, de la manière la plus directe, prendre la forme d'une incitation pécuniaire, par le biais d'une rémunération. Deux modes alternatifs et complémentaires s'imposent. Le premier, qui est déjà largement effectif, est celui d'une embauche au sein des entreprises qui se sont ralliées à des degrés divers au modèle du logiciel libre ; ces embauches se traduisent le plus souvent par une allocation du temps de travail de l'individu entre, d'une part des tâches dévolues aux objectifs propres de l'entreprise et, d'autre part, une implication dans les activités coopératives du bazar. Le second mode de compensation relève du secteur public et doit être considéré comme l'une des manières d'intégrer le soutien au logiciel libre dans les objectifs de politique technologique. Il pourrait s'agir d'un principe de rémunération bénéficiant à des développeurs confirmés, impliqués pour partie de leur temps dans des projets publics de recherche.

5. Un modèle de coalition

De manière très stylisée, soit I la population des utilisateurs du libre, de cardinal N, lesquels peuvent faire le choix individuel de participer ou non aux efforts collectifs de développement. On considère que ces efforts collectifs engendrent, pour chacun des N individus de I, une externalité d'usage, résultant des améliorations apportées à la base des logiciels. Nous représentons cette externalité sous la forme d'une fonction $\phi(n)$ du nombre d'individus coopérant. $\phi(n)$ est bien entendu croissante et on admettra facilement que, du fait des difficultés de coordination qui croissent avec le nombre d'individus impliqués, $\phi(n)$ est strictement concave.

De manière très simplifiée, on suppose, dans un premier temps, que le coût individuel de participation, par unité de temps, est uniforme et désigné par CP, de même que pour les effets espérés en termes d'apprentissage et de réputation, désignés par KR. On peut alors écrire le

gain net, à chaque période de temps, pour un développeur individuel i, selon qu'il coopère ou non:

$G_c = \phi(n) + KR - CP$ s'il coopère
$G_d = \phi(n)$ s'il ne coopère pas

Un sous−ensemble J de I, de cardinal n*, constitue une coalition stable, sous la double condition de
stabilité interne:

$$\phi(n^*) - \phi(n^*-1) + KR - CP > 0$$

stabilité externe:

$$\phi(n^*+1) - \phi(n^*) + KR - CP < 0$$

soit donc que

$$\Delta\phi(n^*) > CP - KR > \Delta\phi(n^*+1)$$

Un équilibre existe donc, du fait de la concavité de ϕ , sous la seule condition d'une consistance suffisante des effets d'apprentissage et réputation KR vis−à−vis du coût CP de la coopération. Dans de telles conditions, n* constitue une taille limite d'efficience et, si N est suffisamment grand, il peut devenir nécessaire de structurer le développement du libre autour de projets qui rassemblent des communautés de développeurs−utilisateurs partiellement spécialisées, quoique non−imperméables les unes aux autres.

Considérons à présent l'élargissement contemporain du logiciel libre en introduisant, dans l'expression des gains individuels, une externalité négative, décrite comme une fonction FR(N) croissante et concave du nombre N des utilisateurs. On suppose en outre que les effets d'apprentissage et de réputation KR(i) espérés par un individu i, en retour de sa participation à l'effort collectif, sont l'objet d'une certaine distribution sur la population des développeurs.

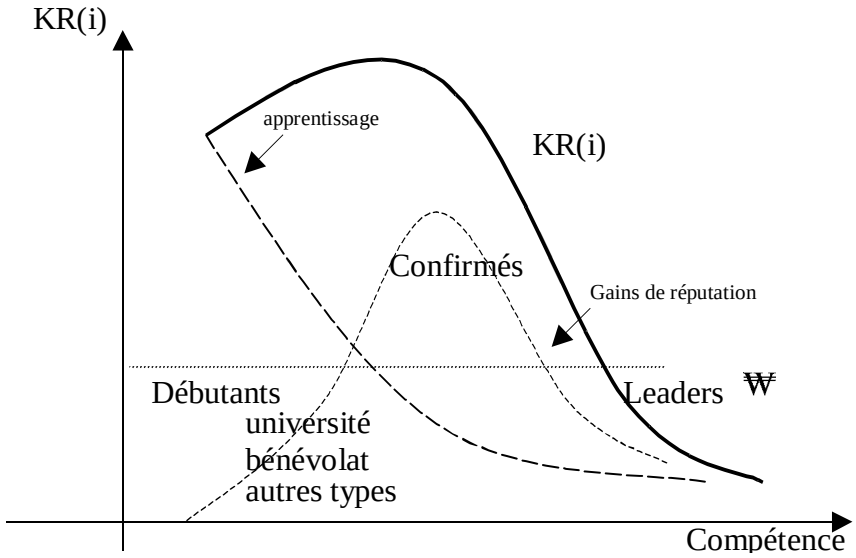
Pour un développeur, l'expression générale du gain relatif, entre coopération et non−coopération, peut alors s'écrire

$$\Delta g(i) = \Delta\phi(n) + KR(i) - FR(N) - CP$$

Cette fois, la condition de consistance des effets incitatifs est plus difficile à réaliser du fait de la présence de cet effet de frustration FR(N). Si l'on considère une sous−population J de taille n*, coalition stable réalisable en l'absence de l'externalité négative, l'introduction de cette dernière va entraîner dans le camp de la défection une fraction $\eta(N)$ de la population, correspondant aux individus dotés des KR(i) les plus faibles et pour lesquels le gain relatif

$\Delta g(i)$ devient négatif. Ces défections ont elles mêmes une répercussion à la baisse sur l'externalité positive $\Delta\phi\ (n-\eta(N))$ et le système peut alors évoluer vers un nouveau point fixe $n' > 0$ qui constituerait un nouvel équilibre, de moindre efficacité ou même s'effondrer hors de toute coalition réalisable, si N est suffisamment grand.

Le profil de la distribution des KR(i) joue bien entendu un rôle déterminant dans la dynamique de recomposition ou d'effondrement du système. Les premiers individus incités à faire défection sont ceux qui sont le moins en attente d'effets d'apprentissage et de réputation. Or ces effets varient de manière importante, avec le niveau de compétence atteint par l'individu. Ainsi on peut attendre qu'un individu proche de ses débuts compensera de faibles gains en réputation par d'importants effets d'apprentissage, avant d'entamer une progressive construction de sa réputation qui s'appuieront largement sur les compétences acquises. En revanche, un individu qui bénéficie d'une certaine reconnaissance acquise, cumulera des gains de réputation de plus en plus faibles.



Effets d'apprentissage et de réputation selon le niveau de compétence

Par conséquent, les individus les plus enclins à faire défection, lorsque l'audience des logiciels libres s'élargit, sont précisément les individus les plus compétents. Si nous cherchions à affiner les termes du modèle en tenant compte des niveaux de compétence dans la contribution à la production du bien public, ce sont clairement les individus dont l'implication est la plus efficiente, qui seraient les premiers à prendre leurs distances avec le modèle du bazar. En outre, on peut aussi considérer que l'effet de frustration individuelle est d'autant plus important que l'individu est d'un niveau de compétence élevé et se trouve en position d'attendre des gains plus importants, ce qui ne peut que renforcer le phénomène. De telles considérations pourraient cependant être nuancées par la prise en compte d'un niveau

d’engagement militant des individus, en particulier pour les leaders du mouvement, dont la défection supposerait l’abandon de cet objectif militant et aurait évidemment des conséquences fortes en termes de rupture de l’équilibre.

Conclusion

Ainsi, le logiciel libre constitue−t−il un modèle intéressant, à la fois parce qu’il paraît bien adapté aux caractéristiques de l’innovation dans un domaine industriel comme le logiciel, et parce qu’il est, sans aucun doute, un modèle de mutualisation des connaissances, garant de la qualité des solutions adoptées et du caractère concurrentiel des marchés.

Longtemps confiné dans un milieu restreint, le logiciel libre est entré aujourd’hui, du fait d’Internet, dans une phase de démultiplication des interactions entre développeurs et de pénétration du monde marchand. Son poids, en termes de part de marché, reste encore extrêmement réduit, mais sa popularité va grandissante et son potentiel est immense. Reste toutefois que, dans son opposition avec la logique propriétaire, le décollage du libre est contingent d’un certain nombre de facteurs qui sont avant tout institutionnels et politiques. Deux aspects nous en semblent particulièrement à retenir ici.

Le premier est relatif aux régimes de protection de la propriété intellectuelle qui seront appliqués, dans le futur, au domaine du logiciel. Le choix d’une systématisation du recours à la brevetabilité constituerait une menace redoutable pour le devenir du libre.

Le deuxième facteur est relatif à la nécessité de corriger les défauts d’incitation du modèle, en introduisant un principe de rémunération des développeurs, du moins des plus compétents d’entre eux. Cette pratique est dores et déjà avalisée par les pratiques des entreprises qui se sont ralliées au monde du libre. Reste à déterminer, pour les pouvoirs publics, si elle doit à son tour compter parmi les mesures de politique technologique qui pourraient viser à renforcer la viabilité du modèle du libre. On conçoit facilement qu’une telle orientation de politique technologique ne pourrait se concevoir à la seule échelle nationale et devrait, pour le moins, viser une mise en place à l’échelle de l’Union Européenne.

Bibliographie

Allen R. (1983), "Collective invention", *Journal of Economic Behavior and Organization*, 4, p.1−24

Alper J. (1999), "L’envol des logiciels libres", *La Recherche*, 319, Avril, p.27−29

Bessen J. et Maskin K (2000), "Sequential Innovation, Patents and Imitation", MIT, Dept of Economics, *Working Paper* N°00−01, January.

Clément−Fontaine M. (1999), "La licence Publique Générale GNU", Mémoire de DEA "Droit des Créations Immatérielles", Université de Montpellier I, http://crao.net/gpl/ .

Dasgupta P. et David P.A. (1994), "Towards a new economics of science", *Research Policy* 23(5), p.487−521

David, P. (1997), « Communication norms and the collective cognitive performance of 'invisible colleges' », in G.B.Navaretti (ed.), *Creation and transfer of knowledge : institutions and incentives*, Physica Verlag Series

David, P. et Foray D. (1995), "Accessing and expanding the science and technology knowledge base", *STI Review*, n°16, OCDE, Paris, p.13−68

Dang−Nguyen G. et Pénard T. (2001) "Don et coopération sur Internet. Un essai d'explication économique", *Revue Economique*, ce numéro

D.Desbois, N.Jullien, T.Pénard, A.Poulain−Maubant, J.Vétois et JB.Zimmermann (Eds.) (1999) *Logiciels Libres: de l'utopie au marché*, L'Harmattan/Terminal .

Foray D. et Hilaire Perez L. (2000), "The economics of open technology: collective organization and individual claims in the "fabrique lyonnaise" during the old regime", *Conference in honor of Paul David*, Turin, Mai 2000

Ganagé M. et Bonnet R. (1998), "Linux: l'explosion du phénomène des logiciels libres", *L'Ordinateur Individuel*, N°97, Juillet−Août.

Hart R., Holmes P. et Reid J. (2000), "The Economic Impact of Patentability of Computer Programs", *Report to the European Commission*, Study Contract ETD/99/B5−3000/E/106.

Heraud J.A. (1995), "Brevet et contexte institutionnel de la création technologique", in Baslé M., Dufourt D., Héraud J.A. et Perrin J., *Changement institutionnel et changement technique*, CNRS Editions, Paris

Lakhani K. et von Hippel E. (2000), « How open source software works : « Free » user−to−user assistance », MIT Sloan School of Management, *Working Paper* #4117

Lerner J. et Tirole J. (2000), "The simple Economics of Open Source", Harvard Business School et Institut d'Economie Industrielle, *Mimeo*

Lucas A. (1987), "Le droit de l'informatique", Thémis, PUF.

Quah (1999), "The weightless economy in economic development", London School of Economics, Economic Department, Londres

Raymond E.S. (1998), *La Cathédrale et le Bazar*, traduit par Blondeel S., http://www.lifl.fr/~blondeel/traduc/Cathedral−bazaar/Main_file.html .

Scotchmer S. (1991), "Standing on the shoulders of giants: cumulative research and the patents law", *Journal of Economic Perspectives*, vol.5, n°1, hiver, pp.29−41.

Smets−Solanes J.P. et Faucon B. (1999), "Logiciels libre − Liberté, Egalité, Business", Edispher, Paris.

Vivant M. (1993), "Une épreuve de vérité pour les droits de propriété intellectuelle: le développement de l'informatique", in *L'avenir de la propriété intellectuelle*, Librairies Techniques, collection le Droit des Affaires, Paris

Zimmermann J.B. (1995), "L'industrie du logiciel: de la protection à la normalisation", in Baslé M. et al. (1995)

Zimmermann J.B. (1998), "L'industrie du logiciel: ébauche d'une approche prospective" *Terminal* N°75, Hiver 97 − Printemps 98.