CULTURE DU WEB

Points de vue sur l'open source

Analyses et décryptages

Patrice **Bertrand**
Directeur Général

Smile
OPEN SOURCE SOLUTIONS

www.smile.fr • +33 (0)1 41 40 11 00 • contact@smile.fr
www.smile-oss.com • blog.smile.fr • twitter: @GroupeSmile

PREAMBULE

SMILE

Smile est une **société d'ingénieurs experts** dans la mise en œuvre de **solutions open source** et l'intégration de systèmes appuyés sur l'open source. Smile est membre de l'**APRIL**, l'association pour la promotion et la défense du logiciel libre, du **PLOSS** – le réseau des entreprises du Logiciel Libre en Ile-de-France et du **CNLL – le conseil national du logiciel libre**.

Smile compte près de 700 collaborateurs dans le monde, dont plus de 500 en France (mai 2013), ce qui en fait *le premier intégrateur français et européen de solutions open source.*

Depuis 2000, environ, **Smile mène une action active de veille technologique** qui lui permet de découvrir les produits les plus prometteurs de l'open source, de les qualifier et de les évaluer, de manière à proposer à ses clients les produits les plus aboutis, les plus robustes et les plus pérennes. Cette démarche a donné lieu à **toute une gamme de *livres blancs*** couvrant différents domaines d'application. La gestion de contenus (2004), les portails (2005), la business intelligence (2006), la virtualisation (2007), la gestion électronique de documents (2008), les PGIs/ERPs (2008), les VPN open source (2009), les Firewall et Contrôle de flux (2009), les Middleware orientés messages (2009), l'ecommerce et les Réseaux Sociaux d'Entreprise (2010) et plus récemment, le Guide de l'open source et NoSQL (2011). Chacun de **ces ouvrages présente une sélection des meilleures solutions open source** dans le domaine considéré, leurs qualités respectives, ainsi que des retours d'expérience opérationnels.

Au fur et à mesure que des solutions open source solides gagnent de nouveaux domaines, Smile sera présent pour proposer à ses clients d'en bénéficier sans risque. Smile apparaît dans le paysage informatique français comme **le prestataire intégrateur de choix** pour **accompagner** les plus grandes entreprises dans l'adoption des meilleures solutions open source.

Ces dernières années, Smile a également étendu la gamme des services proposés. Depuis 2005, un département consulting accompagne nos clients, tant dans les phases d'avant-projet, en recherche de solutions, qu'en accompagnement de projet. Depuis 2000, Smile dispose d'un studio graphique, devenu en 2007 Smile Digital – agence interactive, proposant outre la création graphique, une expertise e-marketing, éditoriale, et interfaces riches. Smile dispose aussi d'une agence spécialisée dans la TMA (support et l'exploitation des applications) et d'un centre de formation complet, Smile Training. **Enfin, Smile est implanté à Paris, Lille, Lyon, Grenoble, Nantes, Bordeaux, Marseille et Montpellier. Et présent également en Espagne, en Suisse, au Benelux, en Ukraine, au Maroc et en Côte d'Ivoire**.

QUELQUES RÉFÉRENCES DE SMILE

Sites Internet

EMI Music, Salon de l'Agriculture, Mazars, Areva, Société Générale, Gîtes de France, Patrice Pichet, Groupama, Eco-Emballage, CFnews, CEA, Prisma Pub, Véolia, NRJ, JCDecaux, 01 Informatique, Spie, PSA, Boiron, Larousse, Dassault-Systèmes, Action Contre la Faim, BNP Paribas, Air Pays de Loire, Forum des Images, IFP, BHV, ZeMedical, Gallimard, Cheval Mag, Afssaps, Bénéteau, Carrefour, AG2R La Mondiale, Groupe Bayard, Association de la Prévention Routière, Secours Catholique, Canson, Veolia, Bouygues Telecom, CNIL…

Portails, Intranets et Systèmes d'Information

HEC, Bouygues Telecom, Prisma, Veolia, Arjowiggins, INA, Primagaz, Croix Rouge, Eurosport, Invivo, Faceo, Château de Versailles, Eurosport, Ipsos, VSC Technologies, Sanef, Explorimmo, Bureau Veritas, Région Centre, Dassault Systèmes, Fondation d'Auteuil, INRA, Gaz Electricité de Grenoble, Ville de Niort, Ministère de la Culture, PagesJaunes Annonces…

E-Commerce

Krys, La Halle, The North Face, Kipling, Vans, Pepe Jeans, Hackett, Minelli, Un Jour Ailleurs, Decitre, ANWB, Solaris, Gibert Joseph, De Dietrich, Adenclassifieds, Macif, Furet du Nord, Gîtes de France, Camif Collectivité, GPdis, Projectif, ETS, Bain & Spa, Yves Rocher, Bouygues Immobilier, Nestlé, Stanhome, AVF Périmédical, CCI, Pompiers de France, Snowleader, Darjeeling…

ERP et Décisionnel

Veolia Transport, Solucom, Casden Banques Populaires, La Poste, Christian Louboutin, PubAudit, Effia Trasnport, France 24, Inra, Publicis, Nomadvantage, Nouvelles Frontières, Anevia, Jus de Fruits de Mooréa, Espace Loggia, Bureau Veritas, Skyrock, Lafarge, Cadremploi, Groupe Vinci, IEDOM (Banque de France), Carrefour, Corsair, Le Bon Coin, Jardiland, Trésorerie Générale du Maroc, Ville de Genève, ESCP, Sofia, Faiveley Transport, INRA, Deloitte, Yves Rocher, ETS, DGAC, Generalitat de Catalunya, Gilbert Joseph, Perouse Médical…

Gestion documentaire

Generali, HEC, JCDecaux, Serimax, Pierre Audoin Consultant, Alstom Power services, NetasQ, CS informatique, SNCF - Direction du matériel, Mazars, EDF R&D, EDF Nucléaire, Conseil Régional du Centre, Leroy Merlin, Primagaz, Renault F1, INRIA, Ministère belge de la Communauté Française, APAVE, CNIL, Services du Premier Ministre...

Infrastructure et Hébergement

Agence Nationale pour les Chèques Vacances, Pierre Audoin Consultants, Rexel, Motor Presse, OSEO, Sport24, Eco-Emballage, Institut Mutualiste Montsouris, ETS, Ionis, Osmoz,

SIDEL, Atel Hotels, Cadremploi, SETRAG, Institut Français du Pétrole, Mutualité Française, Bouygues Telecom, Carrefour, HEC, Jardiland, Orange, TNS Sofres, Manpower, Ministère de l’économie, Eram, Kantar Worldpanel, Fiducial…

Consultez nos références, en ligne, à l’adresse : http://www.smile.fr/clients.

Ce livre blanc

Depuis plus de 10 années maintenant, Smile est aux avant-postes du logiciel libre et open source en France. Etant le premier des intégrateurs open source, nous avons bien évidemment un poste d'observation unique sur les usages de l'open source, leur pénétration dans les entreprises, les métiers, les entreprises et les modèles économiques qui forment cet écosystème si dynamique.

Mais au delà de son activité économique, Smile a également été partie prenante dans l'animation de cet écosystème, dans la promotion du logiciel libre et open source, dans la défense des valeurs qui l'ont fondé.

Smile est adhérent de l'APRIL, l'association pour la promotion et la défense du logiciel libre. Smile est également adhérent des très nombreux clusters d'entreprises acteurs du logiciel libre en France. En 2010, ces clusters, associations d'entreprises et groupes thématiques au sein de pôles de compétitivité se sont réunis pour former le CNLL, Conseil National du Logiciel Libre. Les présidents d'associations, qui forment le bureau du CNLL, m'ont élu comme porte parole du CNLL, et m'ont renouvelé leur confiance jusqu'à ce jour. C'est un honneur, et une responsabilité, que de porter la voix des entreprises de cette filière.

En tant qu'acteur du logiciel libre et open source, en tant qu'observateur de cette industrie, en tant que dirigeant de l'une des entreprises qui la composent, je me suis attaché à décrire et analyser ce mouvement et cet écosystème sous tous ses aspects : technologiques, économiques, sociétaux, humanistes, et même marketing. Ce sont ces articles et tribunes, parus sur les trois dernières années, qui sont réunis ici pour présenter la vision d'ensemble d'un acteur engagé, Smile.

J'espère que vous en trouverez la lecture enrichissante.

N'hésitez pas à nous transmettre vos avis et évaluations sur ce livre blanc.
Une seule adresse : contact@smile.fr

open source

WWW.SMILE.FR

SOMMAIRE

Ce que l'open source a changé

L'open source est une idée qui a pris naissance dans le monde du logiciel, mais a inspiré et bousculé bien d'autres domaines.

Logiciel libre et open source

Revenons aux origines. Le logiciel libre est imaginé dans les années 80 par Richard Stallman. Il affirme que les programmes informatiques doivent pouvoir être librement utilisés, et surtout étudiés et modifiés. Utopique pour certains, il amorce pourtant une véritable révolution, qui 20 ans plus tard a bousculé toute l'économie du logiciel, et bien au delà. Fin des années 90, certains préfèrent l'appellation alternative de logiciel "open source" pour désigner à peu près la même chose, mais en mettant en avant non pas tant la liberté, que les qualités spécifiques de ces programmes réalisés de manière collective, peu centralisée, dont le code source (le programme tel qu'il est écrit par un informaticien) est disponible et peut être modifié, utilisé pour créer de nouveaux programmes, des œuvres dérivées.

A certains égards, l'open source est un mouvement humaniste. Il considère que le logiciel est, à la manière de la connaissance scientifique, une forme de patrimoine de l'humanité, un bien commun que nous enrichissons collectivement, pour le bien être de tous.

L'open source, disons ici plutôt le logiciel libre, porte aussi un message particulièrement d'actualité: le logiciel nous contrôle, il est vital pour nous de contrôler le logiciel. Des pans de plus en plus grands de notre vie sont sous la maîtrise de logiciels. Un logiciel détermine si votre voiture va freiner, un autre si votre pacemaker va faire battre votre cœur, et un autre peut-être déterminera pour qui vous avez voulu voter aux présidentielles. Le logiciel fait désormais plus que nous "rendre service", il nous contrôle. Ce n'est pas un mal en soi, à condition seulement que le contrôlions aussi, que nous sachions ce qu'il fait exactement, et ayons le droit de le modifier si besoin. Cette exigence première du logiciel libre est plus que jamais essentielle.

Ces 20 dernières années, le logiciel libre et open source, réuni sous l'appellation FLOSS, a apporté d'incroyables bouleversements.

Dans l'informatique, une révolution aux multiples facettes

D'abord dans la manière de créer des programmes. Dans les années 90, peu après la naissance du web, c'est une révélation : les programmes les plus critiques de la toile, les programmes les plus utilisés, les programmes les plus complexes, sont des programmes open source. Même Bill Gates en prend soudain conscience, et adresse en 1998 un mémo à ses troupes, où il s'alarme de cette transformation, de ces logiciels aussi bons et parfois meilleurs, de cette nouvelle forme de concurrence.

L'open source a apporté une rupture dans l'économie du logiciel en abaissant les coûts d'une manière incroyable. Tout ce qui constitue le socle d'une plateforme informatique, d'une plateforme web, est devenu tout simplement gratuit : système d'exploitation, bases de données, logiciels serveurs, outils de développement, outils d'administration. Bien sûr, le coût total de possession n'est jamais nul : il faut du matériel, du support et de l'expertise humaine pour déployer et faire marcher tout cela. Mais pour une startup, la barrière à l'entrée a été abaissée de manière phénoménale, stimulant et accélérant la création d'entreprises innovantes. Et pour les entreprises utilisatrices, cette nouvelle donne s'est traduite en gains de compétitivité.

Comme toutes les révolutions technologiques depuis la machine à vapeur, l'open source a amené une forme de destruction créatrice, comme l'avait décrit l'économiste Joseph Schumpeter. En produisant des alternatives quasi-gratuites à des logiciels anciennement coûteux, l'open source a fait disparaître des acteurs devenus non compétitifs, et réduit les marges de quelques autres. Mais le contexte nouveau d'un socle logiciel devenu un bien commun a permis l'émergence de milliers d'acteurs, de startups innovantes, dont certaines sont déjà grandes. Et a permis, plus largement, l'émergence du web, de ses acteurs géants, et des milliers d'acteurs plus petits mais innovants et grandissants.

Le développement logiciel a été profondément modifié lui aussi. L'approche moderne du développement consiste à assembler des composants, grands et petits, pour l'essentiel open source. Une part déterminante du développement consiste donc à sélectionner les bons composants et les intégrer, en ne développant réellement que les parties spécifiques, qui concentrent la valeur ajoutée de l'application. C'est une transformation du développement logiciel qui a apporté d'importants gains de productivité.

L'open source domine sur les serveurs et dans le cloud

L'open source a eu des succès mitigés sur le poste de travail, sur le PC ordinaire. Et pourtant, moins visible et moins connue du public, la victoire de l'open source a été écrasante du côté serveurs et Cloud. Si Windows domine sur les postes de travail, le système d'exploitation Linux a une domination plus grande encore sur les millions de serveurs des grandes plateformes du web, de Google, Facebook, Amazon, ou eBay, mais des plus petits acteurs de la même manière. Une étude récente estimait à 90 % la part de marché de Linux sur le Cloud de Amazon.

Dans beaucoup de domaines, l'open source est en pointe, faisant naître les outils de demain. Citons par exemple l'émergence du "Big Data", la manipulation des données à une échelle nouvelle, où les outils de bases de données anciens atteignent leurs limites, et où des technologies nouvelles sont nécessaires. Ces nouvelles bases, dites "NoSql", sont pratiquement toutes des logiciels open source.

Open Innovation

L'open source a apporté aussi une nouvelle approche de la R&D. Une belle illustration est donnée par le projet open source Genivi, qui a l'initiative de BMW et PSA a réuni des grands

constructeurs automobiles et équipementiers dans une démarche typique de R&D mutualisée, construisant ensemble une plateforme logicielle destinée à leurs véhicules. Pour réussir ce projet stratégique, ces grands industriels ont adopté le modèle open source tant en termes de socle, de développement, de diffusion, que de gouvernance. Et l'on pourrait citer évidemment le noyau du système Linux lui-même, auquel contribuent des dizaines d'entreprise, en faisant sans doutes le plus bel exemple de R&D mutualisée, à l'échelle mondiale. Les démarches appelées parfois « open innovation » ont montré les bénéfices d'une innovation plus ouverte sur le monde, moins cachée, fonctionnant en réseau.

Open Art

Certains ont présenté l'open source comme antagoniste à la propriété intellectuelle. C'est tout le contraire, puisque l'open source se définit par ses licences d'utilisation, qui s'appuient elles-mêmes sur le droit d'auteur. L'auteur, titulaire des droits, donne à l'utilisateur des droits étendus, et quelques devoirs. Ce principe par lequel l'auteur d'une œuvre reste parfaitement identifié, conserve ses droits, mais autorise différentes utilisations et la redistribution de son œuvre a été étendue à de nombreux domaines, bien au delà du logiciel.

L'open source se décline dans l'art également. Les licences Creative Commons ont permis de diffuser des œuvres de toutes natures en donnant des droits étendus, en particulier une libre rediffusion, avec ou sans le droit de modifier l'œuvre originale.

Ainsi, la fondation Blender, qui développe l'un des meilleurs programmes d'animation 3D du monde, un programme open source, réalise des "open movies", des films d'animation dont tous les fichiers source, qui permettent de générer le film, sont rendus disponibles et peuvent être modifiés. Comme un roman dont on pourrait réécrire la fin.

Open hardware

L'open source a gagné le matériel également, sous l'appellation de « open hardware ». Il s'agit ici de partager les plans de circuits et d'équipements entiers. Un bel exemple d'open hardware, le projet Arduino est un microcontrôleur programmable totalement open source, matériel et logiciel, qui peut être adapté pour toutes formes de traitement du signal, ou de contrôle de process. Il peut être programmé pour réagir aux signaux de capteurs externes, les traiter, et commander des actions. Depuis 2005 il s'enrichit d'année en année, et plus de 300 000 unités ont été fabriquées. La diffusion de l'open hardware est encore modeste, mais souvenons-nous que c'était le cas aussi de l'open source logiciel à ses débuts : un « truc de geek ». Mais ces trucs de geeks font tourner les plateformes du web aujourd'hui.

Le mot clé derrière ces projets, ces démarches, est celui de réappropriation de la technologie. La technologie n'est pas le domaine réservé d'une élite minuscule, du fond de la Silicon Valley. Nous pouvons la maîtriser, et particulièrement si nous réunissons nos forces. C'est le principe des FabLabs … Nous ne sommes pas que des consommateurs idiots qui s'endettent pour acheter le dernier smartphone, dont on n'aura pas le droit même de changer la batterie. Avec quelques amis, avec un peu d'aide, avec des plans et des logiciels open source, nous pouvons construire des choses extraordinaires, dans notre garage. Pas tout à fait le dernier smartphone, mais pas très loin.

Les imprimantes 3D ouvrent de nouvelles frontières pour ces démarches. Après avoir pris le contrôle des logiciels, il sera possible de reprendre le contrôle sur le matériel. On rêve déjà de pouvoir télécharger, sous licence libre, les plans d'une pièce de rechange pour sa cafetière, d'imprimer chez soi sa pièce en 3D. Et un peu plus tard, d'imprimer la cafetière open source elle-même ! Utopique ? Mais justement, c'est la plus grande révolution de l'open source, de montrer que l'utopie gagne, parfois.

Open médecine ?

Les systèmes open source ne sont pas que pour les bricoleurs du dimanche. Ils gagnent par exemple la recherche en médecine. Merveilleux exemple de matériel et de logiciel open source associé à une démarche de recherche : des chercheurs ont développé Raven, un robot chirurgien open source, mis à disposition des équipes de recherche du monde entier afin de faire progresser les logiciels et technologies de chirurgie assistée. D'autres chercheurs travaillent à une machine combinant scanneur et radiothérapie, dont les plans, le code source, et les instructions de fabrication seront open source. Il est intéressant de remarquer que certains de ces projets de médecine open source ont reçu le soutien de la FDA, qui est un peu l'équivalent de l'AFSSAPS, avec l'espoir en particulier que le logiciel open source améliore la qualité, jugée insuffisante, des équipements propriétaires.

Fédérer les énergies citoyennes

L'open source a montré aussi que l'on pouvait fédérer et organiser les efforts d'un grand nombre de personnes sur un projet commun. Il était précurseur de ce qu'on a appelé plus tard le "crowdsourcing", ces projets qui impliquent un grand nombre de contributeurs bénévoles, dont la réussite emblématique est celle de Wikipedia, mais qui a aussi donné OpenStreetMap.. Avec un double crédo : d'une part la connaissance est un bien commun qui doit être accessible à tous sans barrière économique, d'autre part les citoyens peuvent gérer eux-mêmes ce patrimoine, dans le cadre d'une organisation décentralisée, et d'une gouvernance ouverte.

Parmi les déclinaisons de l'open source, on peut citer aussi le mouvement de l'open data, la mise à disposition des données publiques, mais aussi des données de certaines entreprises. Une démarche citoyenne et démocratique d'une part, mais aussi le socle de nombreuses initiatives et modèles économiques nouveaux appuyés sur ces données.

Des combats citoyens

L'open source a fédéré des combats citoyens fondamentaux. Les militants de l'open source ont une force particulière : ils réfléchissent aux tendances sociétales, mais sont aussi au cœur des technologies nouvelles et parfois de leurs rouages économiques. Ils ont compris par exemple l'importance de standards réellement ouverts, dont la spécification soit librement accessible, dont la gouvernance soit ouverte, dont l'utilisation soit gratuite. Ils se battent pour la neutralité du Net, ce principe fondateur de non-discrimination des flux sur le réseau mondial, qui a permis l'émergence de toute une industrie du web et qui est menacée aujourd'hui. Ils tentent d'expliquer aux politiques pourquoi les brevets ne sont pas applicables au monde du logiciel, où la seule protection du copyright est amplement

suffisante. Dans le monde du logiciel, les brevets sont contre-productifs, ils découragent l'innovation, ils sont l'arme d'un oligopole de géants et d'entités mafieuses appelées "patent trolls". Pour les premiers il s'agit d'effrayer les petits concurrents plus innovants. Pour les seconds, d'extorquer une rente sur l'innovation des autres.

Une industrie florissante

L'open source n'est pas à l'écart de l'économie, au contraire. Les développeurs qui construisent les programmes open source ne sont pas toujours des bénévoles : la plupart sont payés par des entreprises qui voient un intérêt bien analysé dans leurs participations à ces travaux : elles bénéficient de logiciels performants dont elles n'ont eu à financer qu'une fraction de la R&D, elles ont une parfaite maîtrise de ces technologies qui deviennent des standards, elles ont un rôle dans la gouvernance de ces projets.

En France, l'économie du logiciel libre représente plus de 300 PME et ETI, éditeurs de logiciels ou sociétés de services, dédiées au logiciel libre. Elles sont souvent réunies en associations régionales, elles-mêmes fédérées au sein du CNLL, le Conseil National du Logiciel Libre. Elles représentent ensemble plus de 3000 salariés, et connaissent une croissance annuelle de près de 30 %. Si on comptabilise également les emplois liés au logiciel libre dans les sociétés de services généralistes, l'industrie (notamment aéronautique) et les télécommunications, le chiffre d'affaires global lié à l'open source est estimé à 2.5 Mrds d'Euros, soit 6% du marché des logiciels et des services informatiques, et plus de 30000 emplois, en croissance annuelle de ~30%. [Source : Pierre Audoin Consultants].

Conclusion

On le voit, les déclinaisons de l'open source sont nombreuses, les impacts de l'open source vont bien au-delà du logiciel, des nouvelles technologies, ils s'étendent à d'autres industries, à l'ensemble de la société, à nos conceptions de la citoyenneté, de la démocratie. Toutes ces facettes de l'open source, à l'articulation de la technologie et du sociétal, sont représentées à l'Open World Forum. C'est ce qui fait la personnalité unique de cet événement, c'est ce qui attire à Paris des intervenants de renom, et une audience internationale.

Financer l'édition de logiciel open source

On a écrit beaucoup déjà sur les différents modèles économiques de l'open source, mais celui des éditeurs de logiciel open source est encore mal compris.

Le modèle des fondations

On sait qu'une grande majorité du logiciel open source est développé non par des bénévoles, mais par des informaticiens qui sont rétribués, au travers de divers modèles économiques.

Le modèle économique des fondations, qui est un modèle de mutualisation de R&D par les entreprises, fonctionne particulièrement bien. Il fait fonctionner le développement du noyau Linux, des nombreux produits de la fondation Apache, ou plus récemment du Cloud "Openstack". Des entreprises trouvent économiquement pertinent de faire travailler quelques-uns de leurs développeurs sur les programmes de la fondation. Le modèle des fondations est florissant, mais dans de nombreux domaines il n'a pas amené de produit majeur. Ce sont précisément les domaines qui ont été conquis par des éditeurs open source.

Modèle économique d'éditeur open source

Le modèle économique des éditeurs open source est très différent de celui des fondations. L'éditeur est une société normale, c'est-à-dire à but lucratif, qui fait le choix de distribuer tout ou partie de ses produits sous une licence open source. L'éditeur investit dans le développement d'un produit logiciel, et espère trouver un retour sur son investissement. Ces dix dernières années ont vu un développement spectaculaire de ce modèle. Il semblait même que toutes les startups du logiciel avaient choisi l'open source.

Un logiciel open source – s'il est de qualité bien sûr – se répand comme une traînée de poudre, au niveau mondial. Les success-stories n'ont pas manqué, ces dix dernières années, et incluent quelques belles sociétés françaises. Et cependant, gagner de l'argent, pour un éditeur open source, n'est pas toujours aisé.

Comme on le sait, la distribution d'un produit sous une licence open source permet une libre utilisation, sans limitation aucune, et autorise la modification et la redistribution du logiciel. Il n'est donc pas possible d'exiger des utilisateurs du produit le paiement d'un quelconque droit d'utilisation.

Des modèles fondés sur le support

On a coutume de dire que, en conséquence, les modèles économiques d'éditeurs open source sont fondés sur le support. L'éditeur propose un contrat de support, par souscription en général annuelle. En l'échange du paiement de la souscription, l'éditeur offre un ensemble de garanties et de services, en général l'accès immédiat aux correctifs du logiciel, mais aussi une prestation de support, c'est-à-dire la promesse d'une intervention garantie, sur tous

types de problèmes en relation avec le logiciel. Aux États-Unis, les éditeurs associent leur offre de support à une forme d'assurance relative à la propriété intellectuelle, qui peut être très séduisante pour les services juridiques de leurs clients.

Les difficultés d'un modèle de support

Certains éditeurs ont un produit destiné uniquement aux grandes entreprises, et sur un domaine critique. En général, les grandes entreprises ne conçoivent pas qu'un logiciel critique ne soit pas l'objet d'un contrat de support solide.

Mais un modèle fondé exclusivement sur le support présente des difficultés.

a) Il est difficile de rendre le prix proportionnel à l'usage du logiciel, et donc aux bénéfices qu'il apporte à son utilisateur,

b) Le prix du support doit financer à la fois la prestation de support, qui doit être d'une qualité irréprochable, mais aussi le développement du produit,

c) Plus le produit est de qualité, moins le client rencontre de difficulté, plus il lui semble possible de renoncer au support,

d) Des prestataires tiers, autres que l'éditeur et qui n'ont pas investi dans le développement du produit, peuvent proposer des alternatives de support, moins légitimes et moins expertes, mais éventuellement moins chères.

Jouer les grands nombres

Certains éditeurs peuvent compter sur une sorte de loi des grands nombres : un utilisateur sur 100 de leurs produits deviendra client, mais ce n'est pas un problème puisqu'ils ont 100 000 déploiements. Pour certains c'est 1 sur 1 000, pour d'autres 1 sur 10 000. MySql, historiquement le premier des éditeurs open source à succès, faisait vivre une société de 400 personnes en proposant un support optionnel de sa base de données. Mais c'est un modèle qui ne vaut que pour certains types de produits, combinant un marché global immense, et une position de leadership forte. J'entendais récemment Fabien Potencier, PDG de SensioLabs, expliquer que c'était sa vision du modèle économique pour son produit Symfony, un framework PHP au succès mondial : la loi des grands nombres. Pour un éditeur qui est dans cette logique, chaque déploiement nouveau est bon à prendre, qu'il rapporte ou non, il participe de la diffusion du produit, il contribuera à le faire connaître et apprécier.

Mais c'est un modèle qui ne convient pas à tous les éditeurs, soit que leur marché soit intrinsèquement trop petit pour faire jouer les grands nombres, soit que le marché soit fragmenté, et donc leur part trop petite.

Éditeurs intégrateurs

Les éditeurs open source dont le produit n'a pas un déploiement mondial peuvent avoir de grandes difficultés à devenir rentable. Nombreux sont ceux qui sont également intégrateurs de leur produit. C'est la manière la plus simple de trouver des revenus, au moins pour démarrer, et on pourrait penser que personne ne saura mieux déployer un produit que celui

qui l'a créé. Mais c'est un modèle qui n'est pas extensible, et qui tend à brider l'expansion de l'entreprise. Pour se déployer, en particulier à l'international, l'éditeur doit s'appuyer sur un réseau d'intégrateurs partenaires, et il ne parviendra jamais à bâtir ce réseau s'il est aussi le premier concurrent de ses partenaires. Les éditeurs-intégrateurs open source ont donc un modèle qui tient la route, mais qui les confine à un petit marché. D'une certaine manière, le produit est là pour aider à vendre l'intégration, la vraie source de revenus ; il n'est qu'un atout concurrentiel au service d'un coeur de métier d'intégrateur.

"Payer ?? Je croyais que c'était open source ?"

Certains éditeurs open source ont connu un succès planétaire. Mais nous connaissons aussi des petits éditeurs qui souffrent. Leur produit est utilisé, mais personne ne semble se soucier de financer leur effort de développement. Certains utilisateurs sont capables même de se plaindre d'un bug qui tarde à être corrigé, d'une fonctionnalité manquante, d'une réponse tardive à leurs questions. Mais payer ?? "Je croyais que c'était open source ? "

Et les mêmes qui assurent avec force que, en matière de logiciel, "libre ne veut pas dire gratuit", fustigent les éditeurs qui ont une politique commerciale un peu, disons, volontariste. Il faut pourtant bien comprendre une chose : tant que payer est optionnel, on paye très peu. Les utilisateurs de logiciels open source, et particulièrement les entreprises, ne paieront pas au nom d'un devoir moral. C'est dommage, mais c'est ainsi. Et la nécessité de rendre durable le modèle sous-jacent, dans leur intérêt même, ne semble pas être une motivation suffisante. On peut, on doit, faire leur éducation. Mais en attendant, il faut vivre avec cette forme de tragédie des biens communs : chacun préfère que les autres s'occupent de rendre durable le modèle.

Des modèles à deux licences

Pour construire un modèle viable, certains éditeurs ont fait le choix d'une distribution sous deux licences distinctes, l'une open source, l'autre non.

Il en existe différentes variantes. Dans certains cas, il n'y a qu'une version du logiciel, mais elle est distribuée sous deux licences, et le support n'est proposé que sur la licence dite, "entreprise", dont la licence n'est pas open source. Ce peut être le moyen, pour un même produit, de rendre le prix de support proportionnel à l'usage.

Pour d'autres éditeurs, la version dite "entreprise" n'est pas le même produit. Elle offrira des fonctionnalités supérieures, destinées aux utilisations plus professionnelles ou plus ambitieuses. D'autres enfin définissent leur version "entreprise" par une qualité supérieure, un niveau de stabilité, de test, de robustesse, convenant aux utilisations professionnelles.

Un modèle à deux licences et deux versions, parfois appelé "freemium", ou encore "open core", oblige à trouver le juste point d'équilibre entre les possibilités des deux versions : la version open source doit être assez complète et assez robuste pour être diffusée, donner envie, construire une image positive du produit. L'éditeur doit donc accepter l'idée que sa version open source, quand bien même il l'aurait affublée d'un nom inquiétant, doit être utilisable en production de manière satisfaisante. Si elle ne l'est pas, si c'est une sorte de

version d'évaluation, alors elle ne participera pas à la diffusion et à la renommée du produit. Mais, jugeant que le taux de conversion à la version "entreprise" est trop faible, certains éditeurs sont tentés de dénigrer leur propre version open source, la décrivant eux-mêmes comme inutilisable ! Nous pensons que c'est un très mauvais calcul. Mais il n'est pas aisé de trouver un modèle totalement satisfaisant.

J'entendais dernièrement une société éditrice de logiciels open source qui dénonçait les horribles modèles freemiums et, dans la même phrase, disait sa colère de voir une administration utiliser son produit sans rien payer. Il y a un peu d'inconséquence à cela.

Lorsqu'un éditeur voit chaque déploiement non payant de son produit comme une anomalie douloureuse, insupportable, quasiment du vol, c'est sans doute qu'un modèle open source ne lui convenait pas.

Modèle troublant ?

On peut trouver à redire à ces modèles à double licences/double produits, certaines critiques sont fondées. Ainsi, il est incontestable qu'ils estompent la frontière entre open source et propriétaire : le produit revendique une forme de label "open source", qu'il ne mérite que pour une de ses versions. Ils mettent en avant les multiples bénéfices de l'open source, mais n'en offrent réellement qu'un sous-ensemble.

Certains critiques vont jusqu'à décrire les modèles à double licence comme une arnaque, une démarche de petit dealer : la première dose est gratuite, mais aussitôt que vous aurez aimé le produit, le piège se referme, vous voilà accro et condamné à payer ! C'est une vision tout à fait ridicule. D'une part, les décideurs informatiques ne sont pas des enfants, ils comprennent parfaitement, ce qui est open source et ce qui ne l'est pas, et savent apprécier les bénéfices en regard du coût, c'est leur métier. D'autre part, la version open source fonctionne, elle a peut-être des fonctionnalités plus réduites, mais elle n'a pas de date limite, elle peut être utilisée en l'état, gratuitement, autant qu'on voudra. Et son code source librement étudié, modifié, redistribué.

Pour beaucoup d'éditeurs open source, y compris en mode "open core", le ratio entre les installations "entreprise", sous contrat avec l'éditeur, et les installations libres est de 1 à 10, souvent de 1 à 100. Si 90 ou 99 % des utilisateurs bénéficient d'un produit open source de qualité, financé par des utilisateurs plus exigeants qui eux-mêmes sont satisfaits de leur choix, on se demande où serait le mal ?

Pas de concurrence entre éditeurs et fondations

On peut apprécier ou non les modèles à double licences, les choisir ou non, mais il est clair qu'ils ne font de tort à personne, et sûrement pas au patrimoine logiciel des fondations.

Les éditeurs open source ne sont pratiquement jamais en concurrence avec les produits open source des fondations et communautés. D'abord parce qu'ils adressent des segments de marché très différents. L'open source des fondations, qu'on pourrait appeler "non marchand", est particulièrement riche dans l'infrastructure, les outils de développement et frameworks, et certains domaines connexes. Ce n'est pas limitatif, bien sûr, et l'on doit citer

aussi des Libre Office, VLC, Blender et bien d'autres. Mais du moins, il laisse des domaines non couverts, et ce sont précisément ces domaines qui sont occupés par des éditeurs open source. Les éditeurs open source ne sont presque jamais en concurrence avec les produits de fondations, simplement parce que s'ils l'étaient, ils perdraient, et disparaitraient.

On peut donc refuser de déployer ces logiciels, ou bien on peut choisir d'en utiliser la version véritablement open source, fusse-t-elle appelée "community". Mais on ne peut pas à la fois déployer cette version et pester contre l'éditeur parce qu'il propose par ailleurs une version "entreprise", non open source. Ce serait incohérent, puisque c'est bien cette version entreprise qui finance le développement de l'autre version. Si l'on refuse le modèle, il faut le refuser complètement. Mais la réciproque est vraie : l'éditeur pour sa part, ne peut pas à la fois construire un marketing fondé sur le caractère open source et fustiger les utilisateurs qui ont l'audace d'utiliser cette version ! Au final, chacun doit simplement assumer ses positions.

Retour sur la circulaire Ayrault

Le gouvernement a publié en septembre 2012 une circulaire qui a fait grand bruit et qui a réjoui les défenseurs du logiciel libre et open source. Elle décrit de manière détaillée les domaines où l'open source apporte des bénéfices spécifiques, et où leur déploiement est recommandé, dans les systèmes d'information de l'État. Un aspect de cette circulaire n'a pas été très remarqué, il concerne précisément les produits à double licences, qui sont jugés moins désirables. Le principal risque cité est qu'ils "risquent de rebasculer dans un mode propriétaire". Nous ne voyons pas un grand risque en cela : les exemples sont très rares de logiciels open source redevenant propriétaires, et la menace du "fork", la reprise du code source par un tiers, reste fortement dissuasive. Et l'État fait souvent le choix de contrats de support globalisés, sans s'assurer qu'ils incluent une juste rémunération des éditeurs.

L'État peut préférer des produits open source de la sphère non marchande, mais s'il utilise des produits d'éditeurs, et particulièrement ceux de petits éditeurs français, il devrait selon nous s'assurer qu'une juste rétribution est permise à ces entreprises afin qu'elles puissent croître et se lancer sur le marché mondial. On lira sur ce sujet, la tribune de Miguel Valdes, qui dirige l'éditeur Bonita, parue ici même il y a peu.

Éditeurs open source et communautés

Les logiciels open source d'éditeurs ont d'autres caractéristiques que celle de leur modèle économique. Dans une majorité de cas, la partie centrale du produit est développée par une équipe unique, celle de l'éditeur. Certains acceptent les contributions de tiers, mais dans l'ensemble les contributeurs bénévoles sont un peu plus réticents à s'investir pour un produit au bénéfice d'une entreprise. Et à l'évidence, la gouvernance du produit est entre les mains de l'éditeur. La notion de communauté, pour ces logiciels, est donc différente, mais elle a du sens malgré tout. En particulier, nombreux sont les éditeurs open source qui ont adopté un modèle d'architecture noyau-extensions, un modèle où de nombreuses fonctionnalités du produit sont apportées par des développements qui viennent s'accrocher au produit pour le compléter.

C'est un modèle qui permet d'enrichir un produit sans avoir à toucher directement son code source noyau : il permet donc un bon compromis entre le besoin de robustesse du produit, et la richesse d'un foisonnement de type communautaire. Certains produits open source d'éditeurs ont ainsi des centaines, certains des milliers, d'extensions développées par des tiers. Et chaque déploiement du produit s'accompagne alors de quantités d'extensions, dont certaines sont devenues incontournables. Ici, on peut parler d'une communauté, même si elle est un peu différente : ces développeurs travaillent plutôt en parallèle, que de manière coordonnée. Mais le résultat est là : le produit s'enrichit par les efforts non seulement de l'éditeur, mais des centaines de développeurs d'extensions.

EN CONCLUSION

On sait que le développement de logiciel open source ne peut pas et ne doit pas être à l'écart des mécanismes économiques et de la sphère des entreprises. Le patrimoine logiciel porté par des fondations, qui a des fondements humanistes, a néanmoins une logique, des motivations économiques. Il a donné lieu aux logiciels qui forment le socle de l'informatique moderne. Mais il ne couvre pas tous les domaines, et a laissé du champ aux éditeurs open source. Quelles que soient les variantes de modèles économiques retenues, ceux-ci apportent une contribution essentielle au patrimoine commun de l'open source. Loin d'être antagonistes, l'open source du secteur non marchand des fondations et communautés, et l'open source marchand des éditeurs s'alimentent l'un l'autre, ils sont au contraire interdépendants.

OPEN SOURCE NATIONAL ?

PAS DE SOUVERAINETÉ NATIONALE POUR LA DÉMARCHE HUMANISTE DE L'OPEN SOURCE

Faut-il un "Cloud français", un "moteur de recherche français", du "big data français" ? Une chose est certaine, il faut que la France, ses chercheurs et ses entreprises, soit au plus haut niveau dans tous les domaines de pointe de l'informatique. Et ces domaines de pointes sont souvent faits de logiciels open source. Mais l'open source peut-il, doit-il, être spécifiquement français ? L'open source est-il du ressort de la connaissance partagée, sans frontières ? Est-il universel et humaniste ? Ou bien un modèle économique d'entreprise, un outil marketing ? Un terrain de guerre économique, un enjeu de compétitivité industrielle ?

Pour y répondre, il convient, comme souvent en matière d'open source, de distinguer l'univers non marchand, celui des fondations et de communautés, et l'univers marchand, celui des éditeurs de logiciel open source. Considérons aujourd'hui le premier.

L'open source des fondations, l'open source de l'univers non marchand, relève d'une démarche humaniste, par laquelle nous bâtissons ensemble un patrimoine de connaissances, disponible pour l'humanité entière, sous la forme de code source. Pour autant, la motivation des contributeurs, qui sont pour une large part des entreprises, n'est pas philanthrope. Ils visent un bénéfice économique assez direct, avec la mutualisation à grande échelle de son investissement de recherche et développement. L'humanisme n'est pas le moteur principal, c'est un effet de bord.

Les grands projets de cet univers, par exemple ceux de la fondation Apache, sont à une échelle globale. Ils sont utilisés dans le monde entier, sont souvent devenus des standards, et leurs contributeurs sont de tous les pays. Certes, la fondation Apache est installée aux États-Unis, et l'essentiel des échanges s'y fait en anglais, mais la gouvernance des projets ne donne aucune prééminence particulière aux Américains. Elle est fondée sur la méritocratie très simple : ceux qui contribuent le plus et le mieux ont le pouvoir. Et il existe de nombreux projets dont le leadership n'est pas américain.

Lorsque des projets relèvent d'une telle démarche et d'une telle gouvernance, et sont déjà reconnus, lancer un projet open source concurrent, soit-il français, n'est pas toujours une bonne idée. C'est un phénomène bien connu, mais regrettable : tout le monde aimerait avoir "son projet", plutôt que de contribuer aux projets "des autres". À la fois parce qu'il y voit une plus grande gloire, parce que l'informaticien en général est tenté de refaire les choses à sa façon, et parce que comprendre ce qu'on fait les autres est difficile. Mais, bien que très humaine, cette tentation a pour effet de disperser les efforts, et amène donc une moindre dynamique de progrès. Pour autant, la concurrence peut aussi être bénéfique, si véritablement le nouveau projet a des ruptures positives à proposer : on parle parfois de darwinisme dans l'open source.

Revenons à la question posée : faut-il oeuvrer pour que ces programmes relevant de l'open source non marchand soient davantage français ?

Nous pensons que non que dans cette démarche humaniste, la nationalité du logiciel importe peu. Il faut que les informaticiens et entreprises françaises soient de plus grands contributeurs de ces logiciels. Les entreprises françaises qui auront fait cet investissement auront acquis une expertise précieuse, elles pourront mieux déployer ces logiciels au service de leurs clients, elles auront construit un avantage compétitif. Et elles auront gagné, au passage, une voix dans la gouvernance.

Et cependant, il serait bon qu'une fondation open source semblable à la fondation Apache puisse voire le jour en Europe. Non pour lui faire concurrence, mais pour la compléter : les domaines où l'open source se déploie sont innombrables, et tous ne sont pas encore couverts. On peut imaginer que les entreprises françaises et européennes participeraient plus facilement aux projets d'une fondation plus proche d'elles. La fondation sélectionnerait quelques projets dont le caractère d'utilité publique serait manifeste, et des mesures fiscales encourageant la participation à ces projets donneraient un élan extraordinaire à l'open source en France, mais aussi à l'innovation et à la compétitivité.

Mais sur le fond, de même que les nations savent travailler ensemble au CERN, et financer ensemble ces recherches, de la même manière le patrimoine open source doit être alimenté conjointement. Les questions de souveraineté technologique et d'indépendance nationale ne sont pas hors de propos, mais pour y travailler, la meilleure démarche est que chaque pays tienne son rang dans l'effort commun, en aidant ses chercheurs et ses entreprises à cultiver et maîtriser ces logiciels open source qui deviennent les standards de l'informatique.

L'open source n'est pas qu'affaire de biens communs, il relève aussi d'un univers marchand, et nous verrons dans un prochain article qu'il s'analyse différemment.

AIDER L'OPEN SOURCE FRANÇAIS

Faut-il un "Cloud français", un "moteur de recherche français", du "big data français" ? Dans pratiquement tous les domaines de pointes, le logiciel open source joue un rôle central. Mais l'open source peut-il, doit-il, être spécifiquement français ?

Nous avons vu précédemment que dans la démarche humaniste de l'open source non marchand, celui des fondations et des communautés, la nationalité du projet importait peu. Mais l'open source n'est pas qu'affaire de biens communs, il existe aussi des entreprises qui ont construit leur modèle économique sur la création et la diffusion de logiciels open source. Ce sont en premier lieu les éditeurs open source, ils sont au cœur d'une filière industrielle où figurent aussi nombre de prestataires de services open source, d'intégration, de formation, de support, d'hébergement. Comme en témoigne une récente étude menée par le CNLL, cette filière comporte beaucoup d'entreprises de petite taille, qui sont en croissance malgré la crise.

Doivent-ils être français ? Ici, sans hésiter, on répondra Oui, évidemment : il faut qu'il y ait davantage d'éditeurs open source français, et surtout qu'il y en ait davantage qui réussissent à l'échelle globale où se joue le marché du logiciel. Les éditeurs et intégrateurs open source sont des entreprises souvent jeunes et dynamiques et certains sont déjà sur un marché mondial. Ils sont les catalyseurs d'un large écosystème qui représente 30 000 emplois en France, et porte une croissance annuelle de près de 30 %.

Il faut qu'ils réussissent, qu'ils exportent, qu'ils conquièrent le monde, c'est une évidence. Ils sont la relève des industries anciennes déclinantes, et il nous semble que le gouvernement devrait s'en préoccuper plus encore.

Justement, c'est la question plus particulière que l'on veut poser ici : faut-il les aider les entreprises françaises dans leur R&D open source ?

Les éditeurs open source sont parmi les premiers visés par les dispositifs d'aide à la R&D dans le logiciel. Au sein du pôle de compétitivité Systematic-Paris-Region, le Groupe Thématique Logiciel Libre (GTLL) a fait éclore 33 projets en 5 ans, représentant un effort de R&D open source de plus de 140 millions d'Euros.

Si l'État aide la R&D, y compris open source, des entreprises françaises, ce n'est pas pour faire progresser la connaissance et la technologie en général, mais pour les aider spécifiquement à être mieux armées que leurs concurrents, en particulier étrangers. Nous ne sommes plus ici dans la sphère humaniste. Cette finalité nationale est légitime, puisque ces aides de l'État proviennent des impôts et taxes prélevés sur les entreprises et les contribuables français : d'une certaine manière, il s'agit de rendre aux entreprises françaises une partie de ce qui leur a été pris, sous la condition d'une utilisation bien définie, jugée bénéfique pour l'économie.

En ces temps de crise, tant de l'économie que des finances publiques, les pouvoirs publics seront naturellement soucieux des retombées de leurs aides, tant de manières directes par les emplois de R&D induits, que de manières indirectes et avec effet de levier fort, par la compétitivité améliorée des entreprises qui en bénéficient. En clair : les aides, et les retombées des aides ne doivent pas quitter la France, et même, pour certaines, la région ; si elles bénéficiaient à d'autres, elles perdraient leur caractère de différentiels compétitifs.

L'open source a quelques caractéristiques fondamentales qui se prêtent mal à ce cloisonnement, du moins qui lui confèrent une moindre étanchéité. Comme on le sait, un logiciel open source peut être librement utilisé, étudié, modifié et redistribué, sans condition. Ainsi, un logiciel open source novateur, issu d'un projet ayant bénéficié d'aides publiques françaises ou européennes, pourra servir aux concurrents étrangers des entreprises françaises. Et les entreprises qui ont participé à ce projet, et qui ont elles-mêmes financé l'investissement ne pourront revendiquer un monopole dans l'exploitation de ce logiciel.

Il est donc essentiel de faire comprendre aux pouvoirs publics que, même en l'absence de monopole d'exploitation, la R&D fondée sur l'open source est un vecteur puissant de compétitivité. Et de parvenir à faire évoluer les dispositifs d'évaluation, mais aussi de construction des projets, pour en tenir compte.

La démonstration est assez facile : il existe tellement d'entreprises éditrices de logiciel open source qui ont connu un succès rapide et mondial. On peut citer quelques champions français de l'open source, parmi lesquels Talend, Nuxeo, Sensio (Symfony), Obeo, Nexedi, etc.

Open source ou non, l'éditeur d'un produit conserve une légitimité incontournable, quasiment exclusive, pour commercialiser une variété de services autour de son produit, et construire un modèle économique solide. Un éditeur open source permet à une majorité d'utilisateurs de son produit de l'utiliser sans être source de revenus pour l'entreprise et si cela lui posait problème, il faudrait qu'il change de modèle. Le deal fondateur du modèle est en général celui-ci : le caractère open source permettra une diffusion mondiale ultrarapide – si le produit est bon, bien sûr – et le fait qu'une partie seulement des utilisateurs soit source de revenus pour l'éditeur est une concession nécessaire pour ce marketing planétaire gratuit. Et si tant d'éditeurs nouveaux font le choix de ce modèle, c'est que le deal est favorable, bien sûr.

Il est vrai qu'un concurrent pourrait proposer une gamme de services autour du même produit, sans avoir investi pour le créer. À commencer par des services de support. Il pourrait aller plus loin même, et créer un "fork" du produit, c'est-à-dire une branche de développement nouvelle. Tout cela est autorisé, irrévocablement, par une licence open source. Mais dans la pratique, ça n'arrive que très peu. Le coût serait énorme, et la légitimité de l'éditeur sur le support du produit que ses équipes ont bâti est difficile à concurrencer. De plus, on peut imaginer que les clients eux-mêmes hésiteraient à envoyer leur budget de support à ces concurrents moins experts, et à mettre en danger la pérennité du produit.

Dans la pratique, les "forks" sont très rares, et sont plutôt initiés, non par des entreprises concurrentes, mais par des communautés craignant qu'un produit open source ne se referme, que l'éditeur ne le transforme en produit propriétaire.

Il faut évidemment des éditeurs open source français, il en faut plus et de plus grands, et tant mieux s'ils concurrencent des solutions d'autres pays, et tant mieux s'ils gagnent. Eh Oui, il faut les aider à se lancer et à se développer. Le modèle open source, s'il permet en général une meilleure diffusion de la connaissance, n'empêche pas la construction d'un avantage compétitif fort, de barrières à l'entrée en forme d'expertise unique. Si les éditeurs de logiciels font le choix de l'open source, c'est qu'ils ont bien compris que ce sera un atout pour eux, sur leur marché national et plus encore pour conquérir le monde.

Nouveau Moteur pour l'Open Source

J'aimerais évoquer une tendance importante des 10 dernières années, qui n'a peut-être pas eu l'attention qu'elle mérite.

Pendant quelques décennies, les avancées des technologies de l'information ont été pour une bonne part conduites par de grandes sociétés informatiques, disons telles que IBM, Oracle ou Microsoft, et des moins grandes de même nature. Quel que soit le domaine technologique, le modèle économique était le même : investir en R&D, créer des produits, puis vendre le fruit de ces travaux aux entreprises et aux particuliers. Et bien sûr, pour sécuriser cet investissement, elles avaient besoin de copyright et de brevets.

Bien que défenseur du logiciel libre et open source, je n'ai pas de problème avec le principe d'un tel modèle économique création-vente ou investissement-retour. Du moins, sauf pour la partie brevets - mais ce n'est pas le sujet ici.

Mais la grande transformation de ces dix dernières années est le passage de ce modèle dominant des vendeurs de technologie, à un modèle tiré par les consommateurs de technologie. Je parle des immenses consommateurs de technologie que sont Google, Facebook, Twitter et consors. On sait que ces géants ont construit toute leur infrastructure sur des socles open source. Ce n'est pas juste qu'ils y trouvaient des produits performants, c'était aussi une nécessité vitale : quand on déploie des serveurs par centaines de milliers, l'open source change radicalement l'équation économique. Et ils visent aussi une parfaite maîtrise de leur informatique, n'imaginent pas avoir à appeler au secours une autre société en cas de problème.

Bref, ces grands consommateurs ont un besoin vital de technologies et produits open source. Parce qu'ils ont des besoins hors-normes, des besoins rarement observés précédemment, ces grands acteurs sont obligés de faire avancer l'état de l'art, et ont attiré à eux quelques-uns des développeurs les plus brillants de leur génération.

Mais, et c'est là le point clé, leur modèle économique n'est pas un modèle de vendeur de technologie. La technologie est un coût nécessaire pour eux, non un revenu potentiel direct. Ils développent des outils nouveaux, mais n'ont pas vocation à les monétiser. Le seul bénéfice de mieux faire tourner leurs milliers de serveurs a déjà rentabilisé l'investissement. Du moins pour une partie de ces développements, car certains évidemment resteront perçus comme un avantage concurrentiel spécifique.

Alors, ayant appuyé leurs travaux sur des socles open source, ayant apprécié les bénéfices immenses qu'ils pouvaient tirer de ce patrimoine librement disponible, ils estiment souvent pertinent de restituer une partie de leur R&D comme contributions aux logiciels open source qu'ils utilisent, ou comme logiciels nouveaux. Non pas pour "faire le bien", mais pour apporter de la matière et nourrir l'effet boule de neige. Le domaine que l'on appelle le "Big Data" est une belle illustration de ce phénomène. Parce qu'ils gèrent des volumes de données jamais vus jusqu'alors, ces grands consommateurs de technologie ont du faire progresser des

technologies telles que Hadoop, Cassandra, SolR, etc. De sorte que l’open source règne sur les technologies de traitement des données à très grande échelle.

C’est la grande révolution qui s’est opérée discrètement : l’informatique est tirée par ses plus grands consommateurs de technologie, davantage que par les vendeurs de technologie. Et si les vendeurs de technologie voient parfois l’open source comme un risque, les consommateurs de technologie ne le voient que comme une opportunité. Ils n’ont rien à perdre, beaucoup à gagner, à rendre disponible une partie de leur R&D sous licence open source : ils consolident leur propre socle, en font un standard de fait, peuvent bénéficier ainsi de la R&D mutualisée des grandes fondations du logiciel libre et communauté, venant compléter la leur.

Bien sûr, les vendeurs de technologie tirent encore une part de l’innovation, mais la part croissante des grands consommateurs, à la fois appuyée sur le patrimoine open source et nourrissant ce patrimoine, est un phénomène puissant de ces dernières années.

BREVETS LOGICIELS : LA GRANDE IMPOSTURE DE LA PROPRIÉTÉ INTELLECTUELLE

La protection de la propriété intellectuelle est très en vogue. Elle est, dit-on, le meilleur soutien de l'innovation, elle-même créatrice de croissance et d'emplois. Nous voulons montrer ici que la transposition du principe de brevet dans le logiciel est une démarche erronée sur le fond, coûteuse pour la société et nuisible pour l'innovation.

Rappelons avant tout la finalité des brevets. En accordant à un inventeur un monopole sur son invention, les brevets ont essentiellement deux missions. La première est d'ordre moral, une forme de respect de la propriété de l'inventeur : ce qu'il a créé est sa propriété, on ne peut en faire usage qu'avec sa permission. La seconde d'ordre social et économique est d'encourager l'innovation.

Une invention n'apparaît pas spontanément dans le cerveau de l'inventeur. Elle est en général l'aboutissement d'un énorme travail, d'un énorme investissement. Un investissement qui présente un risque sérieux : de ne rien produire d'utile, et donc d'être perdu. Il faut donc, pour le motiver, une réelle perspective d'enrichissement, qui idéalement soit en proportion de l'utilité de l'invention pour la société.

Et c'est précisément en instituant le monopole de l'inventeur sur son invention que l'on motive son investissement. Mais on sait que les monopoles présentent aussi des inconvénients : en éteignant la compétition, ils amènent des rentes de situation, qui peuvent étouffer l'innovation et imposer des prix élevés. Il est donc clair que le législateur doit trouver un équilibre entre ces deux objectifs : récompenser l'innovation sans éteindre tout à fait la concurrence.

C'est cette recherche de compromis qui a conduit par exemple à limiter à 20 ans les brevets dans l'industrie pharmaceutique : suffisamment pour procurer un retour sur investissement motivant, mais non pas pour procurer une rente définitive trop coûteuse pour la société. Le délai limité a aussi pour effet d'obliger à ne pas attendre dans l'exploitation effective de l'invention.

Le cas particulier des médicaments met en lumière un principe de fond : entre valeur morale et valeur sociale du brevet, la première s'incline devant la seconde s'il y a antagonisme, c'est-à-dire que la protection de la propriété intellectuelle ne vaut que dans la mesure où elle contribue au bien-être général, elle est surtout un outil au service du progrès. Ceci étant posé, nous pouvons en venir à la question des brevets logiciels.

L'industrie du logiciel est principalement soumise à la protection du droit d'auteur. Celui qui écrit un programme – ou le cas échéant l'entreprise qui l'emploie – est titulaire du droit d'auteur, qui lui permet d'interdire toute utilisation de son œuvre, ou bien de l'autoriser selon les conditions qu'il définit. La validité du droit d'auteur en matière de logiciel n'est contestée par personne. Et certainement pas par les tenants du Logiciel libre ou Open Source, puisque c'est précisément le droit d'auteur qui fonde la notion de Logiciel libre. On pourrait

s'interroger sur la durée de vie pertinente du droit d'auteur dans le monde du logiciel, mais le progrès est ici tellement rapide que la question est à peine posée.

Certains voudraient appliquer au logiciel une autre forme de propriété intellectuelle, celle des brevets. Or pour ceux qui connaissent le logiciel, il est évident d'une part que les brevets n'y ont pas leur place, et d'autre part qu'en termes d'innovation, ils sont totalement contre-productifs.

Pour les personnes qui connaissent peu l'informatique et la programmation, une analogie avec la sphère littéraire permet de bien comprendre la question. Le droit d'auteur s'applique au texte d'un roman, ou d'un essai, comme il s'applique au code d'un programme : l'auteur est protégé contre des utilisations non autorisées de son texte ou de son programme. Mais le brevet, s'il était applicable au monde des lettres pourrait porter par exemple sur "un roman mettant en scène des policiers menant une enquête", ou encore "un procédé par lequel l'auteur fait parler à tour de rôle différents personnages évoquant une même scène". Et ainsi de suite.

Mais ce n'est pas tout : pour bien se représenter la contrainte, il faut imaginer encore qu'il existe des centaines de milliers de brevets de ce genre. Aucun auteur ne pourrait donc raisonnablement les connaître, et a fortiori essayer de les respecter.

Nos responsables politiques ont souvent une culture plus littéraire que scientifique, sans parler même d'informatique. Cette analogie devrait les aider à comprendre en quoi les brevets sont totalement inadaptés au monde du logiciel. Un programme informatique n'a aucune ressemblance avec la molécule d'un médicament, ou le principe d'un nouveau réacteur, il est bien plus semblable à un roman, un énorme roman puisqu'un programme sérieux peut compter plusieurs millions de lignes de code. Et un roman écrit par de nombreux auteurs coordonnés.

C'est donc une chose que tous les informaticiens savent : il est absolument impossible d'écrire un programme en se préoccupant de centaines de milliers de brevets potentiels. Les entreprises du logiciel, si elles ne renoncent pas, devraient avancer dans un immense champ de mines, qui devient plus dense chaque jour.

On ne peut pas croire que l'informatique aurait épuisé son sujet au point que n'importe quel programme nouveau ne ferait que reprendre les inventions passées. À l'évidence, ce n'est pas le cas. Le problème est surtout qu'une immense majorité de ces brevets ne sont porteurs d'aucune innovation. Ils portent sur des procédés que n'importe quel expert jugerait évidents, et qui dans bien des cas ont été mis en oeuvre depuis des années sans prétendre au titre d'invention.

À titre d'exemple, il existe un brevet portant sur la mise en place d'une liste de tâches assignées à un développeur. Dans beaucoup de cas, la validité de ces brevets ne serait pas reconnue au final si un procès avait lieu. Mais les procès coûtent très cher et leur issue est incertaine. De sorte que la menace des brevets est une arme puissante, qui permet dans certains cas d'intimider un concurrent plus innovant, dans d'autres cas de prélever une rente sur son activité.

Au final, les brevets logiciels étouffent l'innovation, et apportent un surcoût énorme, qui se diffuse à toute l'industrie du logiciel, détournant des richesses qui pourraient être utilisées pour une réelle innovation. Et bien sûr, le coût au final est toujours payé par le consommateur et par la société. Dans ce contexte, on pourrait penser que les entreprises du logiciel seraient unanimement opposées aux brevets logiciels. Elles le sont effectivement, à l'exception d'une petite poignée d'entre elles, qui ont perçu qu'elles pouvaient en faire usage pour verrouiller le marché. L'évolution de Google sur ce sujet est instructive : d'abord attaqué en tant que challenger sur des motifs douteux, l'entreprise conteste la pertinence des brevets.

Puis, en désespoir de cause, Google rachète Motorola Mobile pour 12 milliards d'euros, principalement pour disposer de ses 17 000 brevets, qui lui serviront à dissuader les attaques des quelques autres géants du logiciel. Ayant investi autant de milliards dans les brevets, pour des raisons non pas d'innovation technologique, mais d'autodéfense juridique, on peut imaginer que Google militera désormais aux côtés des autres géants pour préserver l'environnement légal qui donne leur valeur à ces brevets. De la même manière Microsoft jeune entreprise innovante n'était pas favorable aux brevets logiciels, mais Microsoft géant mondial en est le principal avocat.

C'est ainsi que l'emprise des brevets sur l'industrie du logiciel s'est accrue par cet effet boule de neige, où les victimes d'hier deviennent les parasites de demain. Aujourd'hui, les géants du logiciel qui ont dû investir dans l'acquisition de brevets même bidon entament une énorme campagne mondiale de lobbying pour faire adopter les mêmes principes par les autres pays. On lira avec intérêt, mais avec angoisse aussi, le récit d'un programmeur de Nouvelle-Zélande, expliquant avec quelle persévérance Microsoft a tenté de faire breveter dans son pays le procédé consistant à stocker un document bureautique sous la forme d'un fichier XML. Ce que Open Office faisait depuis des années.

Ce n'est pas anecdotique. Une guerre est engagée au niveau mondial, une guerre dans laquelle des armes de destruction massive – les centaines de milliers de brevets bidon – ont été accumulées de l'autre côté de l'Atlantique, et qui pourrait paralyser l'industrie du logiciel au profit de quelques acteurs.

Pour y parvenir, il faut amener les législateurs de tous les pays, et en premier lieu européens, à calquer leur droit en matière de brevets logiciels sur celui des États-Unis. De gros moyens sont engagés dans cette bataille. Le législateur connaissant mal cette industrie, encore moins la manière dont est produit un programme informatique, il semble facile de lui faire croire que la question des brevets logiciels serait apparentée à celles du droit des marques, ou encore du droit d'auteur, le tout amalgamé dans la notion générale de "propriété intellectuelle".

De cette manière, les opposants aux brevets logiciels sont placés dans le camp de la contrefaçon des produits de marques, de la violation du droit d'auteur, du piratage de musiques et de films. Essayer de raccrocher les brevets logiciels à la défense de la propriété intellectuelle en invoquant la défense de l'innovation est une véritable imposture.

D'un point de vue pratique, les brevets ne conviennent simplement pas au monde du logiciel, de même qu'ils ne conviennent pas à la sphère littéraire. Sans parler même de leurs effets

néfastes, il faut souligner qu'ils sont tout simplement inappropriés dans ce contexte, qui est du ressort du droit d'auteur, où l'innovation n'est pas une invention. Les brevets ont pour seul effet de verrouiller le marché du logiciel entre les mains de quelques géants, de paralyser une industrie, de réduire l'innovation et d'augmenter les coûts pour les utilisateurs. Il est urgent que les politiques français se saisissent de la question.

KELORA : UNE HISTOIRE DE BREVETS FORT INSTRUCTIVE

Les défenseurs du logiciel libre et open source luttent avec acharnement contre les brevets logiciels, et c'est l'un des combats les plus difficiles à expliquer aux non-experts, et à ceux qui nous gouvernent. Voici une petite histoire qui pourrait aider à en faire comprendre l'enjeu.

L'histoire commence fin 2010. Plusieurs milliers de sites de e-commerce reçoivent un courrier adressé par les avocats d'une boite dénommée Kelora. Le courrier fait état de brevets que détiendrait Kelora sur un procédé consistant à affiner une recherche en sélectionnant successivement des critères. C'est ce qu'on appelle la recherche par facettes, et qui s'est généralisé ces dernières années, à peu près en même temps que la comparaison de produits, ou encore les commentaires et évaluations de clients.

Le courrier demande au propriétaire du site 70 000 dollars pour avoir le droit d'"utiliser sa technologie brevetée". S'il accepte de payer sous 15 jours, alors le prix sera exceptionnellement réduit à 35 000 dollars. S'il refuse de payer, il est menacé d'un procès et pourrait se voir réclamer plusieurs millions de dollars.

Vous imaginez peut-être que Kelora serait l'un des grands sites de e-commerce américains, qui aurait innové en introduisant la recherche par facette ? Ou sinon, un éditeur de logiciel spécialisé dans le e-commerce ? Un prestataire informatique alors ? Non, rien de tout cela. En fait, Kelora n'a aucune activité industrielle. C'est ce qu'on appelle une "Non Practicing Entity", c'est-à-dire une entité qui n'a aucune autre activité que d'extorquer de l'argent aux vraies entreprises.

Ces "NPE" sont aussi appelés "patent-trolls", et ils constituent une des plus graves menaces pour l'innovation, et plus largement, pour les petites et moyennes entreprises de nouvelles technologies. Kelora n'a pas même fait travailler l'ombre d'un chercheur qui aurait inventé la recherche par facette, en fait Kelora a seulement flairé la bonne aubaine en rachetant ce brevet à une autre boite, qui l'avait déposé en 1994. On notera également que les premières mises en œuvre de recherche par facette remontent en fait aux années 80.

Cette affaire-là se termine bien. Kelora a commis une petite erreur tactique : celle de s'attaquer à quelques géants, incluant eBay, Microsoft, Dell, Office Depot, Target, et d'autres encore. Contrairement aux petits acteurs, ceux-là avaient les moyens de payer des avocats pour utiliser la recherche par facette sans se soumettre. Après 2 années de procès, l'affaire a été jugée en mai 2012, et les prétentions de Kelora ont été massivement rejetées.

L'affaire se termine bien, mais il faut absolument lui donner un écho immense, pour plusieurs raisons.

Dans ce type d'affaires, la victime qui accepte de payer est systématiquement obligée de signer un protocole qui l'oblige à ne jamais parler de la transaction, sous peine d'une pénalité plus lourde encore. C'est pourquoi le procédé reste incroyablement ignoré ; la victime enrage, mais est contrainte à ne pas faire connaître sa mésaventure.

Le scénario de l'affaire Kelora est typique de l'action mafieuse des "patent-trolls", tant dans la loi du silence qui est exigée, que dans la technique du "payez dans les quinze jours, sinon c'est le double".

Les premières victimes de cette prédation sont les PME. Le troll leur demande 70, 100, 200 milles dollars, ils ont paniqué et appellent un avocat, qui leur dit généralement : "Mon pauvre ami, vous voilà victime d'un phénomène bien connu. Ce soi-disant brevet ne vaut rien bien sûr, nous pourrions aller en justice, mais prévoyez pour cela au moins un million de dollars". Et la PME paye l'impôt du pseudobrevet, le plus souvent, et signe le pacte du silence, qui permettra au prédateur de continuer sa triste et lucrative besogne.

Cette histoire se passe aux États-Unis. Fort heureusement, il n'y en a pas de semblable en France en particulier, car nous ne reconnaissons pas les brevets logiciels. Quoique dans le cas cité, le brevet est plus sur un procédé d'interface que sur du logiciel proprement dit. Mais du moins, nous avons une culture moins judiciarisée, et l'on peut s'en féliciter.

On pourrait donc se croire à l'abri, et laisser les pauvres PME américaines à leur triste sort. Mais les sociétés françaises sont en réalité très concernées, aussitôt qu'elles essayent de faire du business de l'autre côté de l'Atlantique. Ce n'est pas une inquiétude théorique : nous en connaissons plusieurs, de ces entreprises françaises qui, à peine leur bureau ouvert à Seattle, Miami ou Denver, ont reçu le sinistre courrier d'avocat. Et qui au final ont été contraintes de payer cette taxe mafieuse et contraintes de se taire depuis.

Il est important, donc, que nos entreprises qui ont des ambitions d'expansion soient mises en garde. Et il est important surtout d'affirmer que nous ne voulons pas de telles arnaques en Europe, et qu'il faut pour cela lutter avec la dernière énergie contre les brevets logiciels. Il est important enfin que nos dirigeants politiques soient informés du sort que connaissent nos fleurons informatiques aussitôt qu'ils s'essayent sur le marché américain. Les victimes n'ont pas le droit de parler, je parle ici pour elles.

Les brevets logiciels, mais aussi les brevets de procédés d'interface, tels que le fameux "rebond en butée de scroll" qui était au cœur du procès de Apple contre Samsung, sont fondamentalement une arme de dissuasion des grands contre les petits. Un petit nombre de procès entre géants ont fait la Une des médias, mais en réalité les plus grands acteurs visent le plus souvent le statu quo, la dissuasion nucléaire : j'ai des milliers de brevets, tu as des milliers de brevets, restons-en là et partageons-nous le marché.

On voit sur l'affaire Kelora que les vraies victimes de ce type de brevets, ce sont les PME : dans certains cas pour leur extorquer de l'argent, mais dans d'autres cas pour les dissuader d'entrer en compétition contre l'oligopole en place. C'est ainsi que les brevets logiciels sont avant tout une arme anti-innovation, anti-concurrence. Et les mêmes grandes entreprises qui se sont défendues et ont gagné contre les brevets de Kelora, sont capables demain d'attaquer elles-mêmes de plus petits acteurs qui leur feraient concurrence, en invoquant à peu près le même genre de brevets, qu'ils déposent par milliers chaque année.

Une étude de la Boston University School of Law évalue à 29 milliards de dollars annuels la ponction des NPE sur les entreprises américaines. Non pas les affaires liées aux brevets en général, mais spécifiquement les coûts liés aux NPE ou "patent-trolls". Et si une moitié de

ces coûts était portée par de grands groupes, une majorité des victimes étaient de petites et moyennes entreprises.

Il faut faire connaître l'affaire Kelora, car elle est la partie émergée d'un iceberg immense et puant qui chaque jour saigne des entreprises, y compris françaises.

GENIVI : R&D MUTUALISÉE DANS L'AUTOMOBILE

Les projets open source d'envergure ne manquent pas. Et nombreux sont ceux auxquels participent activement de grandes entreprises. Le plus célèbre d'entre eux est bien sûr le noyau Linux lui-même, auxquels contribuent plusieurs dizaines de grandes entreprises, de tous domaines d'activité.

Sony, Nokia, Samsung ou même Volkswagen, sont au nombre des contributeurs actifs. Pour ces entreprises, Linux fonctionne sur une logique de coopérative agricole, une logique de mutualisation des moyens donc de division des coûts. Une variante de ce qu'on appelle parfois "coopétition", c'est-à-dire que l'intérêt bien réfléchi de chacun conduit à identifier un terrain de coopération dans un contexte général de compétition.

Il y a quelques jours le Comité Open Source de Syntec Numérique a invité Philippe Colliot et Fabien Hernandez, de PSA Peugeot Citroen, à présenter le projet Genivi. Le "IVI" de Genivi signifie "In Vehicule Infotainment", et fait référence à l'informatique embarquée dans un véhicule pour des fonctions non critiques, en particulier autour de la fonction de navigation GPS.

L'INFORMATIQUE EMBARQUÉE, UN CYCLE DE VIE RAPIDE

Il y a deux ans, quelques constructeurs automobile ont fait le constat qu'une partie de l'informatique embarquée sur les véhicules allait être de plus en plus du ressort du consumer electronics, un monde très différent du leur.

La navigation GPS proposée en option par les constructeurs souffrait de plus en plus de la concurrence de petits équipements ventousés sur le pare-brise, fabriqués en très grande série. Au delà du prix, c'est la problématique du time-to-market et de cycles de vie très différents: 4 à 5 ans pour la conception et la production d'un véhicule, 6 mois à 1 an pour l'électronique grand public.

Il deviendrait de plus en plus difficile pour un constructeur automobile, aussi puissant soit-il, de développer seul une plateforme "IVI" complète, et de la maintenir au meilleur niveau sur la durée, avec des attentes toujours croissantes de la part des clients. Et par ailleurs, on sait qu'une part de plus en plus grande de l'informatique embarquée, dans l'automobile comme ailleurs, s'appuie sur un noyau Linux et des couches open source.

Ce qui est particulièrement intéressant, c'est que devant ce constat, des grands constructeurs automobiles ont fait le choix de lancer un ambitieux projet open source, visant à définir et mettre en place un middleware commun, portant tout ce qui n'est pas véritablement différenciant.

COMPOSANTS OPEN SOURCE

Le projet s'appuie exclusivement sur des couches et composants open source, dont il est prévu que environ 80% soient utilisés en l'état, 15% sélectionnés puis adaptés, et 5% développés et maintenus par l'alliance. Et l'ensemble des logiciels produits par Genivi seront distribués sous licences open source.

Nous n'entrerons pas ici dans le détail des choix techniques, des modalités de gouvernance, ni même dans la roadmap du projet. Ce serait passionnant, mais l'essentiel pour nous est l'illustration d'une démarche de R&D mutualisée, déployée par de grands groupes industriels.

De nombreuses questions furent posées à nos intervenants. Pourquoi les entreprises n'étaient-elles pas parties sur une démarche de type consortium, sans dimension open source? Un dispositif où les partenaires formeraient un club fermé, où seuls les contributeurs pourraient bénéficier du produit des travaux? Étonnamment, la question semble avoir été tranchée très vite: il fallait pouvoir à la fois s'appuyer sur l'énorme patrimoine open source disponible, mais aussi inscrire l'ensemble du projet dans une démarche articulée avec les grandes communautés existantes.

Ne pouvait-on pas craindre que certains jouent les "free-riders", prenant sans donner? Non, le constructeur qui utiliserait la technologie sans avoir construit sa propre expertise se mettrait dans une situation de dépendance insupportable.

UN VECTEUR DE COMPÉTITIVITÉ PUISSANT

Le projet Genivi mérite d'être connu bien au-delà du monde de l'automobile car il est un exemple superbe d'une démarche de R&D mutualisée fondée sur l'open source, une forme de coopétition. Les acteurs du projet Genivi ne sont pas de doux rêveurs empreints d'humanisme. Ils ne visent pas non plus une communication favorable qui mettrait en avant leur démarche d'ouverture ou leur contribution à l'open source. Ce sont des acteurs industriels qui ont bien analysé l'intérêt économique et stratégique de leur démarche. C'est en cela que le projet est emblématique, et pourrait inspirer bien d'autres acteurs, de toutes tailles: l'open source, dans sa logique mutualiste, est un vecteur de compétitivité puissant pour les entreprises.

On savait déjà que l'open source était choisi par les entreprises pour des raisons très pragmatiques, en tant qu'utilisateurs, consommateurs et clients, de logiciels. Ici, nous avons la démonstration que d'autres raisons, tout aussi pragmatiques et intéressées, peuvent aussi conduire des entreprises à initier un projet open source, en stimulant une communauté d'acteurs partenaires, et à y investir sous la forme de contributions.

Au final, les participants d'un tel projet obtiennent une plateforme logicielle de qualité pour une fraction du coût. C'est le modèle qui est l'essence même de l'open source, et si PSA et BMW en ont identifié les bénéfices, nul doute que bien d'autres pourraient s'en inspirer.

SANS FIL ET SANS CHAÎNES, LA LIBERTÉ VIENT AUX MOBILES

QUELLE PLACE POUR LE LOGICIEL LIBRE?

Le logiciel libre est trop peu présent dans l'univers des mobiles où, il faut bien le reconnaître, applications et outils open source sont très minoritaires. Pourquoi? Parce que l'open source a besoin d'universalité et de standards ouverts pour s'épanouir, et ils ont longtemps manqué, dans ce monde surtout caractérisé par des plateformes fermées, et des modèles économiques fermés également.

Dans l'actualité, le sujet des mobiles et de l'open source est souvent réduit à la part de marché d'Android, et l'analyse de son degré d'ouverture réel.

Pourtant, il y a bien d'autres préoccupations, peut-être plus essentielles. Et nous voulons évoquer ici deux d'entre elles.

- La première, la Liberté!
- La seconde, les standards ouverts.

LIBERTÉ !

Depuis le temps qu'on vous le dit: le logiciel libre, c'est avant tout affaire de Liberté!

Si on l'avait oublié, le monde du mobile redonne tout son sens à cette maxime.

En février 2011, Apple a décrété que les applications disponibles sur l'AppStore ne pourraient ni demander un paiement, ni même proposer un lien vers un site web permettant de faire un paiement. Les paiements ne pourraient être opérés que sur la plateforme d'Apple, et moyennant un prélèvement de 30%. 30%, pour bon nombre d'acteurs, c'est le niveau de leurs marges espérées.

Il n'y a pas de bons ni de mauvais acteurs, pas d'entreprise qui cherche à faire le bien de l'humanité et peu qui soient nuisibles par essence. Mais une chose est claire et nette: l'acteur dominant cherche toujours à créer son propre standard de fait pour assoir sa position, et à convertir cette position dominante en un prélèvement. Et les standards ouverts, les vrais, sont notre meilleure arme contre les acteurs dominants.

En considérant le monde merveilleux de l'iPhone, les plus de quarante ans ne pourront s'empêcher de penser à l'ère du Minitel, une époque où le même industriel pouvait contrôler le terminal, les tuyaux, l'accès aux contenus, et pouvait prélever une dîme sur tous les services proposés par des tiers. Ce monde merveilleux a été balayé par l'Internet, la neutralité du réseau, la distinction des rôles, un monde de standards ouverts, pour le plus

grand bénéfice des utilisateurs, mais aussi de la majorité des acteurs, créant une diversité d'industries nouvelles et florissantes.

Au-delà des aspects économiques, c'est l'existence même d'un pouvoir absolu et arbitraire qui est inadmissible. L'acteur dominant définit ses propres règles, et peut décider de ce que chacun a le droit de lire et de faire avec son mobile: tant des limites de l'érotisme que du politiquement correct, ou des modèles économiques compatibles avec le sien. Big Brother cool et smart peut-être, mais Big Brother quand même.

On pourrait penser que les entreprises seraient moins éprises de liberté que les citoyens, qu'elles ne songeraient qu'aux parts de marché et aux qualités techniques des plateformes. Mais imaginons qu'une grande entreprise française ait investi dans d'importantes applications mobiles, devenues vitales pour son activité. Imaginons qu'un jour, quelqu'un à Cupertino trouve que le modèle économique de cette entreprise lui pose un problème. Du jour au lendemain, les conditions de l'Appstore se trouvent changées, et cette application est interdite. Ou alors, elles est autorisée sous condition d'un prélèvement de 30% sur les ventes de l'entreprise. Ce n'est pas un délire paranoïaque, c'est déjà là. Or les entreprises n'aiment pas l'incertitude lorsqu'elles doivent investir.

L'ALTERNATIVE ?

Dès le mois de juin 2011, le Financial Times, un des principaux médias en ligne payants, refusait le carcan économique de l'AppStore, et proposait sa version payante sous HTML5.

Au mois d'août dernier, C'est Amazon qui a lancé une version totalement HTML5 de son lecteur ebook Kindle. Elle est compatible avec toutes plateformes mobiles et tablettes, offrant une ergonomie soignée sur l'iPad, sans le moindre compromis par rapport aux applications natives. Et bien sûr, elle permet d'accéder au store d'Amazon pour acheter un livre.

Ces quelques grands acteurs auront un rôle déterminant dans le basculement général hors des plateformes propriétaires. Ils ont les moyens de démontrer ce qui peut être fait avec la plateforme standard du HTML5. Ils ouvriront la voie à des acteurs plus petits, qui verront qu'il n'y a pas de différence d'expérience utilisateur entre une application HTML5 bien conçue et une application native. Or la première permet de cibler tous les mobiles du marché, la seconde une petite partie seulement. Mais plus important : seule la première permet d'élaborer son modèle économique en toute liberté, et de garder pour soi les revenus que l'on parvient à collecter de ses abonnés.

LA REVANCHE DES STANDARDS ?

La guerre des plateformes va prendre fin, la revanche des standards est pour bientôt.

On connait bien l'effet boule de neige qui conduit à la domination d'une plateforme. La plateforme dominante attire davantage de développeurs, offre donc plus d'applications, ce qui la rend plus dominante encore. Et les utilisateurs ont un intérêt réel à faire comme tout le monde, à être dans le même univers que la majorité, être eux-mêmes compatibles.

C'est ce qu'on appelle l'effet réseau, qui mène rapidement à un quasi-monopole. Il a joué pendant 20 ans en faveur de Windows. Mais Microsoft n'avait jamais osé – ou alors jamais été en situation de pouvoir – imposer un prélèvement sur les revenus des fournisseurs d'applications et de services basés sur Windows. Personne n'aurait imaginé un prélèvement de 30% pour l'achat d'un logiciel comptable, d'un billet d'avion ou d'un abonnement à votre journal préféré.

Il semble aujourd'hui que HTML5 en tant que plateforme mobile neutre, unificatrice, puisse être le grand vainqueur de la guerre des plateformes, et faire entrer le mobile dans l'ère des standards.

De manière ironique, Steve Jobs avait été l'un des premiers à parler de HTML5 comme d'une plateforme crédible pour l'avenir. C'était à l'époque de sa guerre à lui, la guerre du Flash, et il s'agissait de démontrer que l'on n'avait pas besoin de Flash sur les iPhones. Néanmoins, la suite lui a donné raison au delà de ses espérance: si HTML5 est une alternative à Flash, et une alternative qui a le mérite d'être un standard ouvert, il pourrait être une alternative à la plateforme iOS elle-même.

Plateformes et outils de développement: le cross-platform est open source

Les chiffres de parts de marché tombent mois après mois, et les positions changent rapidement. Aux dernières nouvelles l'iPhone a encore près de 70% de parts de marché en France, mais la montée d'Android est rapide. Aux États-Unis on a Android à 50%, iOS autour de 30%, et Blackberry autour de 10%. Dans l'ensemble du monde, on estime que Android est autour de 40%, iOS de 20%, et Symbian encore près de 20%.

Une chose est certaine: il n'y a plus de plateforme dominante, et c'est une excellente chose.

Ce grand rééquilibrage des parts de marché dans le mobile amène une nouvelle donne pour tons les acteurs. Alors qu'il y a 2 ans encore beaucoup se contentaient d'une appli iPhone, tout le monde a compris que désormais, il faudrait cibler au moins 3, et si possible 5 plateformes pour couvrir toutes les cibles. Dans ce contexte nouveau, personne ne veut développer et maintenir 3 à 5 applications natives différentes.

C'est pourquoi le développement cross-platform a connu un essor extraordinaire depuis 18 mois. Développer une application unique pour cibler tous les mobiles du marché, c'est un rêve qui est à portée demain.

Et dans ce domaine, les outils de développement et frameworks open source dominent ce sujet.

Citons rapidement les outils tels que PhoneGap, Rhodes Mobile ou encore Titanium.

Il y a deux grandes familles d'outils: ceux qui à partir d'un développement unique génèrent des applis natives pour différentes plateformes, c'est le cas de Titanium, et ceux qui sont en fait des outils de développement ciblant la plateforme Html5, c'est le cas de PhoneGap, ou

Rhodes par exemple. Outre la portabilité et le respect des standards, le HTML5 a aussi l'avantage de rapprocher le développement web et le développement d'applis.

Et les applis mobiles open source, c'est pour bientôt ?

Dans le monde des mobiles, il y a encore trop peu d'applications open source. L'une des raisons en est justement l'absence de standard et la multiplicité des plateformes.

A quoi on peut ajouter les conditions incroyables imposées par Apple aux développeurs quant à la non-publication de leur code.

Le plus souvent, les contributeurs de programmes open source ne visent pas un marché, ils visent à l'universalité, ils ont besoin de croire que leur programme pourra changer le monde, au moins un peu. Mais s'il ne peut être destiné qu'à 35% des mobiles, il sera difficile de changer le monde. L'open source a explosé avec le web, car un programme destiné au web pouvait atteindre l'humanité entière, l'humanité connectée du moins.

C'est pourquoi, avec HTML5, le retour des standards ouverts permettra un retour en force de l'open source dans la sphère des mobiles.

COMMENT UNE FILIÈRE S'EST STRUCTURÉE : LE LOGICIEL LIBRE ET OPEN SOURCE

Les entreprises ont toujours une double relation avec leurs confrères et concurrents : une relation de compétition, parfois virulente, dans la lutte pour les mêmes marchés, et une relation de coopération dans la défense d'intérêts communs, ou l'élaboration de partenariats.

On retrouve naturellement cette dualité parmi les entreprises agissant dans le logiciel libre et open source.

Elles ont des intérêts multiples à travailler ensemble. Les entreprises du logiciel libre sont majoritairement de petite taille. Elles ont donc souvent besoin de s'associer pour répondre à des marchés trop grands pour l'une d'elles. Elles ont aussi des métiers différents. On distingue en particuliers les éditeurs de logiciels, qui créent des produits, et les intégrateurs et prestataires de service, qui déploient ces produits au service de leurs clients. À l'évidence, ils doivent travailler ensemble. Et pour cela, se connaître et tisser des liens.

Par ailleurs, les acteurs du logiciel libre sont souvent impliqués sur des sujets sociétaux relevant de la sphère politique. Ils se sont battus notamment pour la reconnaissance des logiciels libres par les pouvoirs publics, contre la brevetabilité du logiciel ou contre les abus de position dominante de certains éditeurs américains. Au travers de tous ces combats, ils ont appris à se connaître, et ont apprécié l'utilité d'un effort commun, la force du nombre. Il faut dire que le travail commun, en réseau, au sein de communautés, est dans les gènes du logiciel libre, qui est le plus souvent développé par les contributions de nombreux développeurs qui non seulement s'intègrent de manière organisée, mais sont pilotées aussi par une gouvernance fondée sur la méritocratie. Bref, les acteurs du logiciel libre savent travailler ensemble, et connaissent la force de l'union.

L'écosystème du logiciel libre est donc organisé en de nombreuses associations, de nature très diverse. L'APRIL, association de promotion et la défense du Logiciel Libre, fondée en 1996 ; l'AFUL, Association Francophone des Utilisateurs de Logiciels Libre, fondée en 1998. L'une et l'autre ont pour mission de promouvoir le logiciel libre et les valeurs qui lui sont associés, tant dans la sphère sociétale, réglementaire et juridique, citoyenne, que dans celle des entreprises.

Les entreprises agissant dans le domaine du logiciel libre peuvent adhérer à l'APRIL ou à l'AFUL, et nombreuses sont celles qui le font, mais ces associations n'ont pas pour vocation de les représenter spécifiquement et les enjeux économiques particuliers des entreprises ne sont pas dans leurs priorités. De plus, les entreprises de la filière ont aussi en commun des préoccupations qui ne sont pas liées seulement au logiciel libre : le soutien aux TPE et PME en général, mais aussi les dispositifs d'aide à l'innovation.

C'est pourquoi les entreprises de la filière, appréciant toujours la démarche associative, se sont depuis longtemps constituées en associations, principalement au niveau régional. On peut citer par exemple Alliance Libre en région Pays de Loire ou encore Libertis en région

PACA. Elles ont un rôle de terrain avec des animations permettant aux membres de se connaissent entre-eux, d'être reconnus dans leur écosystème, de monter des partenariats. Certaines perçoivent des cotisations de leurs membres, parfois des subventions de collectivités locales, et peuvent avoir quelques permanents, d'autres ne fonctionnent que par le seul bénévolat. Elles sont reconnues dans leur rôle d'animation régionale de la filière, organisent des colloques, salons, ou rencontres, en partenariat avec les pouvoirs publics régionaux. Ces associations ont, pour la plupart, une vraie mission de représentation des entreprises de leur bassin, et une vraie légitimité à cela, mais au seul niveau régional.

En parallèle, l'État a créé en 2005 les Pôles de Compétitivité. Dans un périmètre régional, ces associations loi de 1901 réunissent des entreprises de toutes tailles et des laboratoires de recherche, avec pour finalité de promouvoir l'innovation et la croissance de leurs membres en stimulant le travail en réseau, en conduisant des actions d'animation, de mutualisation ou d'accompagnement sur des thématiques telles que l'accès au financement privé, le développement à l'international, la formation, etc. Les technologies de l'information sont au coeur des thématiques de nombreux pôles, et naturellement le logiciel libre et open source y tient donc une place. Ainsi, le Pôle de Compétitivité "Systematic Paris Region" est organisé actuellement en 6 groupes thématiques, dont l'un, le "GTLL", est consacré au logiciel libre. Il réunit environ 70 entreprises agissant dans le logiciel libre en région parisienne et a permis en 5 ans le montage et le financement de 33 projets impliquant plus d'une centaine de partenaires, représentant un effort de R&D total de plus de 140 millions d'Euros.

Pour être complet, il faut citer également des associations ayant une mission plus ciblée, en relation avec le logiciel libre. Par exemple l'Adullact, fondée en 2002 et dont la mission est de développer et promouvoir un patrimoine commun de logiciels libres métiers, particulièrement à destination des collectivités. Ou encore Framasoft, fondée en 2001, qui vise à promouvoir et développer des logiciels libres et des ressources libres, principalement orientées vers l'utilisateur final. Ici aussi, ce type d'associations peut accueillir des entreprises, et oeuvrer avec des entreprises, mais ne prétend certainement pas les représenter.

Pour résumer, le paysage associatif du logiciel libre peut se partager en:

- Des associations de promotion à large spectre, agissant au niveau national, qui accueillent de nombreuses entreprises, mais ne se consacrent pas particulièrement à leur défense et ne visent pas à les représenter.
- Des associations ayant une mission plus ciblée.
- Des associations d'entreprises du logiciel libre de niveau régional.
- Des Groupes thématiques au sein des pôles de compétitivité, qui accueillent aussi de nombreuses entreprises, avec une finalité de promotion de l'innovation et de la recherche.

Jusqu'en 2010, il manquait donc une représentation nationale de la filière.

Comme toutes les filières, celle du logiciel libre et open source a besoin de se faire entendre des pouvoirs publics. Et ceux-ci, réciproquement, ont besoin de représentants lorsqu'ils prennent des décisions ou élaborent des lois qui impactent cet écosystème. Ils ont besoin

d'interlocuteurs qui témoignent de leur vision de chefs d'entreprises, qui soient experts du domaine, qui soient respectés de leurs pairs et légitimes, c'est-à-dire désignés au travers d'une gouvernance transparente et démocratique. C'est pourquoi il appartient à chaque filière de s'organiser, de définir la gouvernance qui lui convient, et de désigner par ce moyen ses représentants.

C'est ce qu'a fait la filière des entreprises du Logiciel Libre et open source en 2010. Dix grandes associations régionales et groupes thématiques se sont réunis et ont posé les bases du Conseil National du Logiciel Libre (CNLL), une association fédératrice, avec pour mission d'une part d'être porte-parole de la filière et le relai de leur communication au niveau national, et d'autre part de promouvoir les échanges entre associations, échanges de bonnes pratiques, d'information, voire de collaborations. En décembre dernier l'association Aquinetic est devenue le onzième membre du CNLL.

Le CNLL fraichement créé s'est doté de statuts et d'une gouvernance ouverte, transparente et démocratique. Ainsi les présidents d'associations membres tiennent une réunion du bureau une fois par mois, et font le point sur les actions engagées par chacun localement, et sur les actions et communications menées au niveau national.

L'une des premières tâches que s'est donné le Conseil a été de bien recenser, connaître et faire connaître, les entreprises du logiciel libre. Elles sont majoritairement de petite taille, se portent plutôt bien, travaillent plutôt pour de grands clients, et sont très engagées.

Lors des élections de 2012, le CNLL a pu faire entrer le logiciel libre, et les sujets tant réglementaires que de société qui s'y rattachent, dans la campagne électorale, invitant les candidats à prendre position sur 8 thématiques essentielles. Les réponses des principaux candidats étaient claires et presque exemptes de langue de bois. Elles étaient dans l'ensemble assez positives, allant dans le sens des demandes du CNLL sur la plupart des sujets. Bien sûr, il reste pour le candidat François Hollande devenu président, à déployer une politique conforme à ces promesses. Le CNLL est aussi là pour y contribuer.

La représentation légitime d'une filière suppose un fonctionnement fondé sur la démocratie. Le CNLL étant une forme de fédération, ses votants sont les membres du bureau, le plus souvent présidents des grandes associations régionales, eux-mêmes démocratiquement élus par leurs membres. On a donc une démocratie à deux niveaux, où les présidents d'associations jouent un rôle de Grands Électeurs en quelque sorte. À l'usage, ce type de fonctionnement s'avère bien plus ouvert, animé et au final plus démocratique qu'un suffrage direct, car le nombre réduit de membres du bureau permet des échanges de vues constants, denses, rapides, et le passage au vote fréquent sur chaque sujet abordé.

Dans un tel fonctionnement, le président est véritablement un animateur et un porte-parole, voire porte-étendard en quelques occasions, et non un chef. À l'inverse, une désignation au suffrage direct, qui supposerait que les 250 à 300 entreprises du logiciel libre soient impliquées de manière directe dans l'instance nationale, aurait pour effet la désignation d'un président plus autonome, mais plus isolé, potentiellement plus autocratique dans l'exercice de sa fonction. Au-delà des réflexions de constitutionnalistes, c'est surtout dans la pratique quotidienne que l'ouverture de la gouvernance se révèle, et à l'évidence, elle dépend beaucoup des hommes, et femmes, plus sans doute que des institutions.

En conclusion, le monde du logiciel libre apprécie la démarche associative, le travail collaboratif et la force de l’union. Certains imaginent que l'écosystème du logiciel libre et open source serait encombré par des dizaines d'associations rivales, revendiquant toutes de représenter les acteurs du domaine. C'est une vision très erronée. En fait, comme on a essayé de le montrer ici, la réalité n’est pas si complexe : il y a de nombreuses associations, aux missions distinctes, qui travaillent assez bien ensemble, sans réelle rivalité. La plupart ont un rayon d’action régional. Et celles qui visent à représenter les entreprises se sont donné une représentation nationale unique avec le Conseil National du Logiciel Libre.

L'Open Source facteur de compétitivité par la mutualisation de R&D

Lorsqu'on parle des business models de l'open source, on pense le plus souvent à ses acteurs économiques, éditeurs de logiciel, intégrateurs, distributeurs, prestataires de support, tous entreprises à but lucratif si l'on peut dire.

On imagine que pour le reste, l'open source est affaire de bénévoles, travaillant soirs et week-ends sur des projets communautaires, pour la gloire ou pour le bien de l'humanité.
Et on oublie bien souvent le troisième modèle de l'open source, celui qu'on pourrait appeler "coopératif", ou encore "mutualiste". Il a pourtant une importance extraordinaire, puisqu'on y trouve Linux soi-même, les projets de la fondation Apache ou de la fondation Eclipse. Du beau monde...

De quoi vivent ces projets ? Principalement de dons en nature, soit de code source directement, soit de temps de développeurs, payés par leur employeur, et travaillant sur les projets de la fondation. Pourquoi les entreprises font-elles ces contributions ? Pour l'essentiel, ce n'est pas affaire de marketing, il ne s'agit pas d'avoir son nom sur le spi d'un voilier géant. Elles sont au contraire relativement discrètes sur leur implication. Ce n'est certainement pas non plus par philanthropie ou dans une logique de mécénat. Leurs motivations sont de différentes natures. Participer à la gouvernance des projets, c'est à dire avoir son mot à dire dans les orientations qui sont prises, pour les diriger dans le sens de ses besoins. Construire une expertise interne sur ces projets, qui permettra de bien les intégrer à ses propres développements. Acquérir une légitimité sur le marché. Mais surtout, au final: disposer pour soi même d'un outil de qualité en partageant les coûts.

Il est très instructif de regarder la liste des plus grands contributeurs du noyau Linux.

Sony, Nokia, Samsung, Volkswagen, sont au nombre des contributeurs actifs. Pourquoi Volkswagen payerait ses salariés à améliorer le noyau Linux ?

Pour ces entreprises, Linux fonctionne sur une logique de coopérative agricole, une logique de mutualisation des moyens donc de division des coûts. Créer seul un système d'exploitation solide et complet serait hors de portée, utiliser un système propriétaire serait trop coûteux et pas toujours adapté à leurs besoins spécifiques. Reste la mutualisation: chacun "met au pot" quelques développeurs, et ensemble ils disposent d'un OS de qualité exceptionnelle, sur lequel ils ont une certaine maîtrise. Bien sûr, cela signifie qu'aucun d'entre eux ne dispose d'un OS meilleur que celui de son concurrent: cet aspect là de la concurrence a été pratiquement neutralisé. Mais il leur reste encore bien assez de domaines dans lesquels ils peuvent porter le combat, être meilleur, écraser les concurrents. Au global, ce domaine de coopération apporte bien sûr de moindres coûts à chacun, et donc une meilleure compétitivité.

On retrouve cette logique coopérative sur la plupart des projets de fondations. Le moteur de recherche Lucene en est un bel exemple. Depuis 2002, Yahoo a été un important sponsor

des travaux de son créateur, Doug Cutting, qui a ensuite été salarié de Yahoo, de 2006 à 2008, tout en continuant ses travaux sur ce projet de la fondation Apache. En 2006, CNET donne à la fondation Apache le source de SolR, initialement un outil de recherche interne du groupe de média, qui rejoint le projet Lucene. Et Yahoo continuera de financer les projets qui gravitent autour de Lucene, en particulier Hadoop, devenu l'outil de référence pour la répartition de tâches à très grande échelle. On est bien ici dans une logique coopérative: CNET bénéficie de produits très supérieur à ce qu'il aurait pu financer seul, et de même pour chaque jour financé par Yahoo, le bénéfice est décuplé puisqu'une vingtaine de commiters entourent Doug Cutting, payés par différents employeurs.

C'est une exemple particulièrement intéressant, qui pourrait être pris en modèle dans bien d'autres domaines. Dans un nombre extraordinaire de secteurs, des fédérations interprofessionnelles pourraient définir des terrains de coopération logicielle, stimuler des projets mutualistes...

D'une certaine manière, le modèle économique de l'éditeur logiciel traditionnel peut aussi s'assimiler à un financement coopératif: différents clients de l'éditeur payent un droit d'usage, et/ou un support, et la somme de ces contributions individuelles finance le développement du produit. La différence entre ce modèle éditeur standard et le modèle coopératif est dans la prise de risque et dans le partage des bénéfices. L'éditeur prend des risques en investissant pour construire son produit, bien avant d'avoir des clients. En contrepartie de cette prise de risques, il escompte un profit important si son produit est un succès. Et en particulier, il espère vendre de nombreuses licences aussi cher que le permettra le marché, sans laisser croître ses coûts de développement en proportion. Dans un développement coopératif, les clients partagent le risque, et sont assurés en retour de coûts au plus bas et surtout, de tirer eux-mêmes les bénéfices d'échelle.

Pour autant, l'approche coopérative du développement n'est pas indissociable de l'open source. Dix entreprises peuvent s'associer et cofinancer le développement d'une application, qui servira à chacune d'elles. Elles peuvent créer une petite structure pour en gérer les développements et les évolutions. Le domaine des ERP "verticalisés", des applications qui s'appuient sur un socle ERP pour en adapter les fonctionnalités aux besoins d'un secteur d'activité particulier, serait typiquement le domaine de prédilection de cette approche coopérative de l'open source.

L'application résultant de cet effort coopératif peut ne pas être open source. Et même, certains pourraient craindre qu'elle le soit, car elle pourrait alors être utile à d'autres qui n'auraient pas contribué à son financement.

L'open source n'accepte pas de frontière, pas de restriction dans la diffusion: l'application, si elle est open source, ne pourra pas être réservée à ses seuls sponsors initiaux. Il faut avoir les nerfs solides, pour voir un concurrent non coopératif reprendre l'application sur laquelle on a investi.

Mais dans la pratique, on sait bien qu'il n'est pas si facile de "prendre" une application. Ayant démonétisé le copyright, l'open source a mis la connaissance, l'expertise, la maîtrise, au premier plan. L'entreprise qui n'aura pas participé aux travaux aura, dans la pratique,

beaucoup de mal à profiter de l'application sans avoir construit une certaine maîtrise, et sans avoir son mot à dire quant aux orientations.

C'est pourquoi, pour un groupement d'entreprises dans une logique coopérative, la crainte de voire certains profiter sans avoir contribué, peut passer au second plan. Ce qui prime, c'est la croyance que d'autres au contraire viendront rejoindre les rangs des sponsors, apporter leurs propres idées, des moyens renforcés, des coûts encore divisés, pour viser une application encore meilleure. Bien sûr, il faut une certaine foi, mais quelques belles réussites ont ouvert la voie.

Conduire un projet open source

Vous voulez lancer un projet open source ? Bravo ! Vous allez être utile, vivre une aventure extraordinaire, vous faire des amis, contribuer au progrès d'un monde de plus en plus numérique, connaître l'excitation des startups, et peut-être même vous enrichir, qui sait.

Sur quoi portera votre programme et à qui s'adressera-t-il ? Ce n'est pas le sujet ici. Nous supposerons que vous avez une idée originale et lumineuse. Open source ou pas, ça pourra aider.

La première question à se poser est évidemment celle-ci : Que cherche-t-on vraiment ? A devenir riche ? A concrétiser une idée géniale, qui pourra changer le monde ? A se faire plaisir dans le développement ? A se faire connaître, pour recevoir des offres d'emploi ? A monter une startup et vivre l'aventure entreprenariale ? Ce projet open source est-il une aventure dans laquelle on se lance corps et âme, ou bien un passe temps pour le week-end ? Est-ce une histoire individuelle ou un travail d'équipe ? Beaucoup de questions pour démarrer... Nous ne traiterons pas ici de toutes les étapes standard d'un lancement d'activité : analyse de marché, panorama de la concurrence, plan stratégique, business plan, etc. Nous resterons focalisés sur les particularité d'un projet open source.

Faut-il un nouveau projet ?

N'y a-t-il pas déjà un projet semblable auquel vous pourriez contribuer ? D'une manière générale, il y a déjà beaucoup de projets open source, et donc trop de projet qui manquent d'activité. Chacun veut avoir son propre projet, être seul maître à bord. Chacun espère recevoir des contributions, plutôt que de contribuer aux autres projets. Le foisonnement de nouveaux projets a des effets favorables : il amène une concurrence motivante, et une forme de darwinisme qui globalement améliore l'offre. Mais il peut représenter aussi beaucoup d'énergie perdue sur des travaux parallèles, ce qui n'est pas tout à fait dans l'esprit de l'open source, celui de l'enrichissement collectif de biens communs. Donc, si votre objectif n'est pas de lancer une startup mais de construire le patrimoine logiciel de l'humanité, voyez d'abord s'il ne serait pas plus utile de contribuer à des projets existants. Vous pourrez, petit à petit, y faire reconnaître vos talents, et, selon le principe de méritocratie qui prévaut en général, avoir votre mot à dire dans la gouvernance du projet.

Créer le buzz en même temps qu'on crée le programme.

D'une manière générale, quand on se lance dans un logiciel et qu'on espère en faire un business, la tendance du technophile est de passer 6 mois à développer dans son garage en buvant des litres de café, puis enfin sortir de la maison pour aller chercher des clients. C'est précisément la démarche qu'il faut éviter. Pour plusieurs raison. L'une d'elles étant que votre développement doit impérativement être confronté à quelques cas concrets

d'utilisation. L'idéal est de trouver un ou plusieurs premiers clients, qui acceptent d'être des utilisateurs pilotes, et dont le besoin permettra de guider la conception.

Mais ce n'est pas la seule raison. Dans l'open source, davantage que dans le logiciel propriétaire, la clé du succès est dans la qualité du produit : aucun marketing ne sauvera un produit médiocre, et un produit exceptionnel fera son propre marketing. Mais votre produit ne sera peut-être pas tout à fait exceptionnel, et aura besoin d'un peu d'aide tout de même pour se faire connaître.

Votre projet doit avoir son site avant même que la première ligne de code ne soit écrite, aussitôt que le concept est posé. Un site au moins en anglais bien sûr. Pendant toutes la phase de développement, le site annoncera les avancées, d'abord les principes, ensuite la roadmap, puis même les difficultés rencontrées, les choix techniques, etc. Une forme de teasing : toute la vie du développement, en quasi direct, façon loft-story. Et bien sûr, vous accompagnez cela de twitts quotidiens, de commentaires posés dans les blogs abordant votre thématique, vous prenez le temps de répondre à toutes les questions, de discuter avec chaque personne qui remonte un bug, etc. Pendant que vous développez, le site commencera à éveiller l'intérêt, à créer le buzz, et à tisser des liens qui amélioreront son référencement.

La communauté

Voulez-vous faire naître une communauté de développeurs participant à votre développement ? C'est la démarche la plus naturelle pour un logiciel open source. Clairement, créer un logiciel open source sans accepter et stimuler les contributions de tiers serait un non-sens. Il existe pourtant quelques éditeurs qui sont dans cette logique, mais nous supposerons ici que ce n'est pas votre cas.

Une communauté permettra de partager la charge du développement, donc de dynamiser le projet, d'apporter des regards neufs quant à la roadmap, de stimuler la promotion et la diffusion du produit, de lui donner aussi de meilleurs gages de pérennité, condition à l'adoption par des utilisateurs exigeants. Bref, une communauté est une bonne chose. Et soulignons qu'une communauté n'est pas faite que de développeurs. L'utilisateur qui parle du produit sur son blog, celui qui remonte un bug, qui propose une nouvelle fonctionnalité, qui enrichit la traduction, qui corrige la doc sur un wiki, tous ceux-là participent de la communauté, et sont précieux.

Mais comment lance-t-on une communauté ? On connaît quelques créateurs de logiciel qui décident un jour de mettre leur œuvre en open source, déposent les sources sur une forge, font un petit communiqué, puis attendent désespérément les contributeurs. Une communauté ne se décrète pas. On sème la graine qu'est le code source, on veille aux conditions d'éclosion, on arrose chaque jour. La communauté va germer un jour, doucement, puis prendre racine, petit à petit. Et quelles sont ces conditions optimales à réunir ? Un bon produit, évidemment, mais aussi l'écoute, l'échange, la communication, une gouvernance équilibrée et transparente. Et des outils, bien sûr, comme support de tout cela.

En général, une communauté naît plus facilement lorsque votre projet n'a pas une finalité commerciale, car les contributeurs peuvent être réticents à l'idée que leur travail bénévole

sert à vous enrichir. Dans le cas de produits commerciaux, un modèle noyau-extensions est parfois le bon moyen d'articuler un produit central dont le développement est centralisé, entouré par un foisonnement d'extensions, qui sont du ressort de la communauté.

Quels outils, quelle forge ?

Si vous avez choisi un modèle de développement communautaire, alors il vous faut choisir une forge pour accueillir vos sources, vos travaux, vos échanges. Nous ne détaillerons pas ici des outils de développement proprement dits, outils d'infrastructure, environnement de développement, gestionnaire de source, tracker et compagnie. Si vous souhaitez ouvrir le développement et avez de petits moyens, le plus simple est de gérer votre projet sur l'une des forges disponibles gratuitement pour les projets open source. L'ancêtre est le célèbre SourceForge (sourceforge.net), mais les développeurs tendent à préférer maintenant des forges plus faciles à utiliser, mieux intégrées, telles que Google Code (code.google.com), Github (github.com/) ou encore Gitorious, une déclinaison plus communautaire. Par rapport au classique SVN, Git est plus encore tourné vers le développement collaboratif. Chacun peut forker (amicalement) le programme pour y ajouter ses développements et l'auteur initial est libre de choisir les travaux qu'il intègre. C'est une approche qui abaisse la barrière à l'entrée pour les contributeurs, tout en préservant une certaine maîtrise d'ensemble.

Outre les outils qu'elles proposent, ces forges contribuent aussi à faire connaître votre projet, puisque beaucoup vont y faire des recherches. Votre projet sera également référencé sur Ohloh.net, où il attirera peut-être l'attention de nouveaux contributeurs. Si vous avez plus de moyens, vous pouvez opter pour une forge dédiée, telle que Tuleap par exemple, ou des outils très simples tels que Redmine. Mais l'initiateur d'un projet open source a souvent tellement à faire avec son propre développement qu'il appréciera les outils prêts à l'emploi en mode SaaS, bien que les plus connus ne soient pas open source eux-mêmes.

Si votre projet a une finalité suffisamment générale, qu'il a un socle solide et déjà de multiples contributeurs, que vous acceptez les règles de gouvernance associées, alors vous pourrez un jour le soumettre à l'incubateur de la fondation Apache. Indépendamment des ressources et outils de la fondation, c'est évidemment une consécration pour un projet open source, et l'assurance d'attirer des contributeurs de qualité.

Choisir sa licence

Le choix de la licence peut s'avérer déterminant pour votre projet. Certes, si vous détenez le droit d'auteur sur l'ensemble de vos sources, vous pourrez toujours changer de licence ultérieurement, mais il est grandement préférable de faire le bon choix dès le départ.

Les utilisateurs, particulièrement en entreprise, n'aiment pas les licences exotiques. N'inventez pas des termes de licences qui vous seraient propres : il existe suffisamment de licences bien connues pour que vous soyez certains d'y trouver votre bonheur.

Il y a deux grandes familles de licences open source, qu'on appelle parfois « copyleft » et « non-copyleft ». Les licences copyleft, essentiellement la GNU GPL et ses variantes, se caractérisent par l'obligation qui est faite à ceux qui voudraient redistribuer le logiciel, éventuellement modifié, de le faire sous les mêmes termes. Autrement dit, un tiers ne peut

pas refermer le code, ne peut pas construire un produit non-Libre à partir de votre produit Libre. Cela réduit donc les risques de devoir subir quelques concurrents profiteurs. C'est la raison pour laquelle la GPL est souvent choisie par les éditeurs à vocation commerciale. A contrario, les licences « non-copyleft » exigent peu de choses, et sont choisies en général par ceux qui sont motivés par la plus large utilisation possible de leur logiciel. Pour être précis, entre les deux existent aussi des licences dites « weak-copyleft », qui ont une exigence intermédiaire.

Donc pour faire simple, si votre but est de voir votre logiciel utilisé le plus largement possible, quitte à ce que d'autres en fassent une version non-Libre, alors vous opterez pour une licence non-copyleft, telle que Apache, MIT ou BSD. Si vous avez un objectif économique, vous choisirez sans doutes davantage une licence GPL, voire LGPL. Attention, on ne dit pas ici que la licence GPL serait le moyen le plus sûr de gagner de l'argent ! En soi, elle ne vous rapportera rien, et en particulier ne vous permettra pas de réclamer un quelconque droit d'utilisation. Et l'objectif des licences copyleft n'est pas celui-ci, il est plutôt de promouvoir une logique de donnant-donnant, qui stimule de proche en proche l'utilisation de licences Libres. Du moins les licences copyleft permettent d'éviter des œuvres dérivées qui seraient propriétaires, et probablement concurrentes. Citons également ici la licence Affero GPL (AGPL), qui oblige celui qui offrirait un service de type SaaS utilisant votre logiciel à publier son propre code source sous la même licence.

Une autre considération à prendre en compte en matière de licence est l'intégration d'autres composants open source, qui vous feront gagner du temps et de l'argent, mais qui feront de votre logiciel une œuvre dérivée, soumise donc aux obligations de licence de chacun de ces composants. Vous devez ainsi vous intéresser vous-mêmes à ces exigences, et vérifier que la licence que vous souhaitez utiliser est bien compatible avec celles de tous les composants intégrés.

Quel modèle économique ?

Comme on l'a dit en introduction, votre but n'est peut-être pas de vous enrichir. Si votre effort est purement bénévole et humaniste, alors vous pouvez passer ce paragraphe.

Il existe de très belles success-stories commerciales dans le logiciel open source, parmi lesquelles ont peut citer Redhat évidemment, MySQL, JBoss, OpenERP, Talend ou encore Magento. Mais bien d'autres également. A partir de 2000 environ, il semblait que toutes les startups du logiciel choisissaient un modèle en tout ou partie open source, ce qui a donné un formidable enrichissement de l'offre dans la dernière décennie.

Pourtant, il n'est pas aisé de gagner de l'argent avec un logiciel que chacun peut redistribuer librement, sans contrepartie. Il existe une diversité de modèles économiques pour le logiciel libre. Certains fondés sur la seule offre de support, et de quelques prestations associées. D'autres sur une politique de double licence avec – pour caricaturer – une licence Libre pour faire connaître le produit, une licence non-Libre pour gagner de l'argent.

Même s'ils partagent, en général les valeurs du Logiciel Libre, des valeurs de liberté et de biens communs, certains éditeurs choisissent un modèle open source à des fins marketing :

ils comptent sur le bouche à oreille et la libre diffusion du logiciel pour créer une base installée rapidement. Et si le produit est bon, ça peut marcher. Et puis, une fois leur logiciel largement diffusé et utilisé, ils se demandent comment transformer ce succès d'estime en revenus.

Une chose est sûre : il ne faut malheureusement pas compter sur le sens du devoir ou la grandeur de l'âme humaine pour expliquer aux utilisateurs qu'ils ne sont pas tenus de payer, mais que ce serait sympa quand même. Ce serait certainement légitime, mais ça ne marche tout simplement pas. Pour convaincre vos utilisateurs de vous payer alors qu'ils n'y sont pas contraints, il faut leur faire une promesse convaincante, leur offrir une réelle valeur ajoutée.

Si votre produit est entré dans des grandes entreprises, intégré à des utilisations critiques, alors la demande de support viendra naturellement : les DSI sérieuses n'imaginent pas qu'une fonction critique dépende d'un logiciel sans contrat de support.

Mais plus largement, il peut y avoir un équilibre délicat à trouver entre utilisations payantes, et utilisations gratuites. Si un jour vous comptabilisez les utilisateurs de votre produit qui ne payent rien, et que ce décompte vous fait enrager, inutile d'adresser des messages de supplication ou d'injure à votre base installée, l'open source n'était peut-être pas le modèle qui vous convenait !

Dans les années 2000, MySQLétait la figure emblématique de l'éditeur open source. Moins de un déploiement sur 1000 de la base de données qui donna son nom à la plateforme LAMP était source de revenu pour l'entreprise MySQL AB. Mais par la loi des grands nombres, ce petit pourcentage suffisait à faire vivre une entreprise qui comptait 400 employés en 2008. Votre produit ne vise peut-être pas cette échelle, mais du moins il faut comprendre que, pour un modèle économique viable, il faudra soit une petite base installée et un fort taux d'utilisations payantes, soit une faible taux de payant mais une grande base.

INTÉGRATEUR DE SON PROPRE PRODUIT

Il est probable que votre produit aura besoin d'intégrateurs, de prestataires capables de le mettre au service de ses utilisateurs finaux. Pour un logiciel libre et gratuit, la prestation d'intégration peut représenter la plus grande part de la facture du client final. Or personne mieux que vous ne saura faire le meilleur usage de votre produit. Pourquoi ne pas en proposer l'intégration vous-mêmes ? Vous en serez le meilleur ambassadeur, ça vous fera des revenus rapides, ça vous permettra d'être en relation directe avec vos utilisateurs, et donc de bien sentir les forces et faiblesses du produit. Bref, presque que des avantages. Et c'est ainsi que de nombreux éditeurs de logiciels, tout particulièrement open source, commencent par être leur propre intégrateur. Mais en grandissant, les quelques inconvénients apparaissent. L'intégration est un métier qui n'est pas « scalable », comme disent les investisseurs, c'est à dire que les coûts (les salaires) augmentent de manière linéaire quand augmente le chiffre d'affaire, de sorte que les bénéfices sont peut-être plus faciles au début, mais structurellement limités. Tandis que le métier d'éditeur pur, pour un produit à succès, peut amener des marges de plus de 50%. Même si, dans l'open source, ce type de marge est difficile à atteindre sans quelques entorses au modèle.

Il est difficile aussi d'être intégrateur de son propre produit à l'international ; Et c'est souvent lorsqu'il cherche à sortir des frontières de son pays d'origine que l'éditeur abandonne l'intégration pour se concentrer sur son cœur de métier. Les logiciels, surtout open source, franchissent facilement les frontières, pour peu qu'ils aient un site et une interface en anglais, voire multilingue. Sans y avoir œuvré particulièrement, on peut s'apercevoir un jour que son logiciel est déployé du Brésil aux Philippines. Il est temps alors d'organiser un véritable réseau de partenaires intégrateurs, de valider leur niveau d'expertise, de s'assurer qu'ils sont motivés à vendre votre offre.

LEVER DES FONDS ?

La recherche de financements sort un peu du cadre de cet article, mais on doit en dire quelques mots. On distingue au moins deux phases, l'amorçage et le développement. La phase d'amorçage, qui peut représenter de 100 à 300 k€, est souvent financée par les économies des fondateurs, des investisseurs amis ou de la famille, ou des Business Angels.

Plus tard, se posera peut-être la question de faire entrer de vrais fonds de capital risque, pour des montants plus élevés. Ça peut être une bonne idée, une nécessité parfois. L'open source est toujours un de ces mots-clés qui met en appétit les investisseurs. Plus vous aurez pu démontrer déjà la solidité de votre projet – idéalement, même, sa rentabilité – plus vous pourrez négocier cette entrée dans des conditions favorables. Sachez aussi que le fonds, s'il peut accepter d'attendre quelques années, aura un objectif de rentabilité élevé, qui a des chances de modifier votre modèle économique, voire même votre fonctionnement. Si c'est un objectif que vous partagez, alors tout va bien. Mais pour certains entrepreneurs du Libre, les implications opérationnelles d'une rentabilité aussi élevée peuvent apparaître moyennement compatibles avec leurs idéaux.

On se trouvera ramené à la question que nous posions en introduction : quel était votre objectif ?

L'INTERNATIONAL

Le marché du logiciel est global, mondial. Pour un logiciel propriétaire, le penchant naturel serait de se contenter d'abord de son marché national, voire même local. Il est vrai que s'il faut embaucher des commerciaux, faire de la prospection, des rendez-vous, des démos, des salons, … pour arriver à vendre son produit, on ne peut pas faire tout cela à grande échelle dès le démarrage.

Mais ce sont là de vieilles méthodes, dont un logiciel open source peut se passer. Pas éternellement sans doutes, mais certainement au démarrage. Si le commerce se fait par bouche à oreille, si le libre téléchargement, et la libre utilisation du programme permet à chacun d'en mesurer les atouts, si les réseaux sociaux propagent la bonne nouvelle, … le produit peut prendre son essor à l'international pratiquement sans effort.

La condition première, c'est que toute sa communication, son interface utilisateur et toute sa documentation soient au minimum en anglais, et autant que possible traduites en quelques langues. Ainsi, de nombreux créateurs de logiciels open source s'émerveillent de voire leur produit utilisé à l'autre bout de la terre. Même si ces utilisateurs lointains n'envoient pas de bon de commande, leur impact est énorme. Dans un an ou deux, peut-être commencerez

vous à tisser un réseau d'intégrateurs à l'international, et ces utilisateurs de la première heure seront alors des atouts précieux.

Conclusion

Nous espérons avoir couvert ici les principales questions et démarches à intégrer pour lancer un projet open source. Rappelez vous que le succès de votre projet dépendra pour beaucoup de votre code, mais pas uniquement de votre code. De votre concept, bien sûr, de votre capacité à faire connaître votre projet, à faire naître une communauté, à piloter une startup aussi, et le cas échéant à convertir votre succès en revenus, qui pourront assurer la pérennité de votre aventure.

Libre et Gratuit: le logiciel libre et l'argent

Dès l'origine, les défenseurs du logiciel libre ont lutté contre l'association entre libre et gratuit. Elle venait en partie du double sens du mot free en anglais, et l'on a trouvé cette formule bien connue pour y répondre: « free as in free speech, not as in free beer », pour dire que le free de free software parlait de liberté, non de gratuité.

Mais si libre et gratuit se trouvaient ainsi associés, ce n'était pas qu'une affaire de mots. Revenons sur la relation complexe entre libre et gratuit, entre le Logiciel Libre et l'argent.

Free software, not freeware

Bien sûr, il faut commencer par rappeler que la gratuité ne fait pas un Logiciel Libre. Il existe de nombreux logiciels gratuits, dits freeware ou shareware, qui en général ne sont pas des Logiciels Libres. Existe-t-il à l'inverse des Logiciels Libres qui ne sont pas gratuits ? Nous essayerons d'y répondre plus loin.

Ce qui définit un Logiciel Libre, comme on le sait, c'est la liberté de l'utiliser, de l'étudier, de l'améliorer et de le redistribuer, sans restriction. On peut se référer également aux 10 conditions requises par l'Open Source Initiative pour qu'une licence puisse se dire « open source ». Mais que l'on prenne l'une ou l'autre de ces définitions, on parle à peu près de la même chose.

Rien d'immoral à être payé !

Une première chose à souligner est que, même pour les plus intransigeants défenseurs du Logiciel Libre, l'argent n'est pas sale, il n'y a rien d'immoral à faire payer du logiciel puisque payer le logiciel, c'est payer les programmeurs qui l'ont écrit. Certains pourraient souhaiter un « juste prix », qui correspondrait à des marges « raisonnables », mais personne ne saurait dire précisément ce que cela signifie.

Non seulement il n'y a aucun mal à faire payer le Logiciel Libre, mais dans beaucoup de cas, c'est véritablement une nécessité vitale, la condition de sa survie, de son caractère durable.

Pas de problème donc, à faire payer le Logiciel Libre, mais dans la pratique comment s'y prend-on ?

Payer, un devoir moral ?

On peut répondre à la question sur le plan moral, justement. Le logiciel libre est porteur de valeurs profondes qui vont au delà du logiciel, et parmi celles-ci, des valeurs de responsabilité sociale. Qu'est-ce que cela signifie ? En premier lieu, que l'on doit parfois agir pour une meilleure société sans que cela soit une obligation au plan juridique.

Il n'est donc pas anormal de s'attendre à ce que chacun rende durable le modèle, sans y être contraint. Nous pouvons aider à la reconstruction de Haïti, ou bien privilégier les produits du commerce équitable sans y être contraints, parce que cela nous semble bon.

Les anglo-saxons utilisent parfois une autre expression, en relation avec le logiciel libre, en parlant de la nécessité d'un « quid pro quo ». Ce qui ne signifie pas un malentendu, mais une logique de donnant-donnant: si l'un donne le droit d'utiliser un logiciel, l'autre doit donner quelque chose en retour.

MySql a été parmi les premiers à invoquer cette formule, dès les années 90, en laissant entendre que les clients qui font un usage professionnel du produit se doivent de l'utiliser sous une licence commerciale. Il faut souligner qu'en mettant en avant un devoir moral de l'utilisateur, on s'éloigne des termes de la licence, qui ne réclament rien en retour.

Devoir moral, certes, et pourtant l'âme humaine est ainsi faite que si payer est optionnel, on paie peu en général. Et il est clair que l'âme des dirigeants d'entreprises ne vaut pas mieux !

Comment faire payer ?

Comme on l'a tant répété, « Libre ne veut pas dire gratuit », et pourtant … Pourtant, s'il est possible de vendre une multitude de services environnant un Logiciel Libre, il est pratiquement impossible de vendre le logiciel lui-même ou le droit de l'utiliser.

Ce qui rend la vie fort difficile pour un grand nombre d'éditeurs du Libre, surtout les plus petits d'entre eux. Combien de fois lit-on ceci dans les forums « je fais du Libre parce que j'y crois, mais personne ne fait le moindre effort pour financer mes travaux. Ils osent même me demander avec impatience telle correction ou telle fonctionnalité, sans être prêts à y mettre 100 €. Au final, je n'y arrive plus. » Le libre non gratuit, ce n'est pas simple.

Pour être clair, ce que l'on fait payer, ce n'est pas un prix d'acquisition, puisqu'il n'y a pas transfert de propriété. Ce n'est pas non plus un droit d'utilisation, puisque la licence autorise au contraire une libre utilisation. Bien sûr, il n'est pas interdit de vendre le droit d'admirer le soleil couchant ou de respirer l'air marin, si l'on trouve un client disposé à payer. Mais on en fera difficilement un business.

Ni prix d'acquisition, ni droit d'utilisation donc. Dans les années 80 et 90, on pensait faire payer principalement la distribution du logiciel, c'est à dire le fait de l'amener jusqu'à son client sous une forme qu'il pourra utiliser. A cette époque, il s'agissait souvent d'envoyer des disquettes par la poste. Mais le simple acheminement ne vaut plus grand chose aujourd'hui. Il reste bien sûr la valeur ajoutée d'une sélection et d'une validation des produits distribués, mais ici aussi, c'est un peu insuffisant pour un modèle économique.

Combien ? Et combien de fois ?

Combien peut-on faire payer la distribution d'un Logiciel Libre ? Une première réponse serait « le prix que l'on veut », ou du moins, « le prix que l'on peut, pour autant que l'on trouve un acheteur ».

Si j'ai développé un logiciel particulièrement utile, que je trouve un client très intéressé, il n'y a évidemment pas de limite particulière au prix que je peux demander pour la distribution de mon logiciel à mon client, y compris sous licence Libre. D'une part c'est moi qui choisit d'utiliser une licence Libre, et même une fois cette décision prise, rien ne m'oblige encore à le distribuer à qui que ce soit. On voit donc que je suis libre de fixer mon prix, du moins pour cette première vente.

Néanmoins, aussitôt après cette première vente, ce premier client est lui-même autorisé à redistribuer mon logiciel, selon les termes de licence que j'aurai précisés, sans qu'il soit possible d'exiger une quelconque rétribution à l'égard du détenteur des droits que je suis. Ainsi, ce premier client pourrait bien être le dernier, ou du moins être mon premier concurrent, puisque je l'ai autorisé irrévocablement à diffuser mon logiciel gratuitement et sans limite (du moins la version qu'il a reçue). On voit que dans ces conditions, mon deuxième client sera plus difficile à trouver.

Gratuit une fois payé ?

Comme on le sait, la plus grande partie du Logiciel Libre que nous utilisons n'est pas écrit par des bénévoles, le soir après leur travail, mais par des développeurs qui ont une famille à nourrir, ou sinon du moins un loyer à payer.

« Le logiciel libre est gratuit, … une fois qu'il est payé ! » selon une formule pleine d'esprit, que l'on doit à François Elie, président de l'Adullact.

J'aime cette formule, qui affirme la gratuité tout en soulignant la nécessité d'un financement. Mais comment traduire cet énoncé dans les faits ? Le « une fois que » laisse entendre qu'il y aurait deux phases: dans un premier temps il faut payer le développement, puis seulement le logiciel pourrait devenir gratuit. Cette dépendance temporelle est difficile à traduite au plan juridique. Sauf à ce qu'un logiciel ait une phase non Libre, pendant laquelle un droit d'utilisation sera collecté, suivie d'une phase Libre. Mais c'est une logique que personne n'a retenue: d'une part la phase non Libre priverait le logiciel de l'essor, des contributions, et de la diffusion rapide que seul permet le Libre, et d'autre part si malgré tout il est un succès, on peut craindre que, au contraire, il ne soit jamais libéré !

On pourrait modifier la formule de François Elie : « le logiciel libre est gratuit... et pourtant il faut qu'il soit payé ! ». Ce qui certes accentue la contradiction, mais décrit peut être mieux encore la situation.

Support et services

Vendre du support, ou plus largement du service, est la manière la plus simple, et la plus claire, de faire payer le logiciel libre.

Tout le monde comprend assez bien que, si l'on demande à un ingénieur de travailler 3 jours, il convient de le payer, qu'il s'agisse de logiciel libre ou propriétaire.

La vente de support est donc la base de tous les modèles économiques des éditeurs du Libre. Mais c'est un modèle qui a quelques limites.

L'une d'elles est que le contrat de support doit financer non pas seulement la prestation de support, mais aussi le développement du produit, lorsqu'il y a eu un véritable investissement d'éditeur en amont.

L'autre est qu'un modèle économique fondé sur le support est jugé « non extensible », par les investisseurs, c'est à dire les revenus ne pourront croître qu'avec une croissance corrélée des coûts salariaux. Ce qui au final ne permettra pas les niveaux de marges extraordinaires des éditeurs propriétaires, lorsqu'ils parviennent à une situation d'oligopole. C'est sans doutes la raison pour laquelle un modèle économique fondé sur le seul support est rarement retenu par les éditeurs qui ont des « VCs » à leur capital.

Le troisième inconvénient du seul support pour le développeur d'un produit, est que le besoin de support est inversement proportionnel à la qualité du produit. Si le produit atteint une stabilité exemplaire, il est d'autant plus précieux pour ses utilisateurs, qui peuvent pourtant d'autant plus se passer de support. Du moins pour des utilisations peu critiques.

Et une difficulté supplémentaire est que le développeur d'un logiciel, même s'il a une légitimité particulière à offrir du support sur son produit, n'a pas pour autant un droit exclusif à cet égard, ce qui l'obligera donc à pratiquer des prix raisonnables. Pour l'utilisateur, c'est bien sûr l'un des atouts particuliers du Libre.

LE LIBRE GRATUIT EXISTE ET SE PORTE BIEN

S'il n'y a pas de problème, et souvent une vraie nécessité, à ce que le Logiciel Libre soit payant d'une manière ou d'une autre, il ne faudrait pas à l'inverse renier le Logiciel Libre gratuit.

Au sein du patrimoine open source, la famille des produits de fondations, constitue un gigantesque vivier de grands produits.

On y trouve tout le catalogue de la Fondation Apache, des produits d'une extrême robustesse, dont certains ont pris une part de marché dominante: Tomcat, SolR, mais aussi Perl ou Struts, et bien d'autres standards. Et on peut y ajouter encore ceux issus des autres grandes fondations, Eclipse, FSF, Linux, Gnome ou encore Mozilla.

Les produits de fondation sont dans une catégorie « déjà payés »: leur développement a été financé par des dons, en code source, ou en hommes-mois. La fondation elle-même n'est pas à but lucratif, et d'ailleurs elle ne propose aucune forme d'activité susceptible de lui apporter de l'argent, si ce n'est quelques conférences. On est bien dans de l'open source gratuit.

Bien sûr, le TCO, le coût total de possession, ne sera pas nul: il faut bien installer le produit, lui donner un serveur, parfois le paramétrer, former des utilisateurs, et assurer un certain support. Mais on parle de produits dont l'équivalent propriétaire peut coûter très cher, et il est clair que le coût total sera très sensiblement inférieur. L'open source gratuit existe, et il est même en pleine forme.
Le modèle économique de l'open source des grandes fondations est un modèle mutualiste, un modèle de coopérative: différents acteurs, entreprises principalement, contribuent aux projets en donnant du temps de leurs ingénieurs, et en cédant à la fondation la propriété intellectuelle résultant de ces travaux. En contrepartie, les donateurs ont un mot à dire quant

aux orientations du projet, et bénéficient de logiciels de qualité, que leurs équipes maîtrisent techniquement, qu'ils peuvent utiliser librement. Beaucoup utilisent sans contribuer, sans que cela ne mette en péril le modèle.

Le modèle des fondations est dans l'esprit d'une gestion de biens communs, un patrimoine partagé, dont tous peuvent bénéficier, mais qui doit néanmoins être entretenu et enrichi.

LE LIBRE PAYANT RESTE MOINS CHER

Venons-en maintenant aux produits d'éditeurs.

Les éditeurs open source sont des sociétés à but lucratif qui ont choisi de distribuer leur produit sous une licence open source. Elles ont fait un investissement, parfois énorme, dans le développement d'un produit, en prenant souvent un risque important. Elles espèrent bien que leur investissement soit remboursé, avec même un beau bénéfice si possible. Nous avons décrit les différentes variantes de business models des éditeurs, et nous avons expliqué aussi comment le marketing était parfois l'un des fondements du modèle.

Aucun éditeur open source ne fonde son modèle économique ni sur un droit d'utilisation, ni sur un prix de distribution. Ce qui est valorisable, c'est la connaissance, l'expertise sur le logiciel.

Ainsi, les éditeurs open source fondent tous leur modèle en partie au moins sur le support: soit support pur, par souscription annuelle, soit support associé à une version non Libre, offrant des fonctionnalités plus avancées, ou bien une plus grande stabilité. Dans certains cas, on peut dire froidement que la version Libre de leur logiciel est un outil marketing, aidant à la vente de la version non Libre. Une idée que certains trouveraient révoltante. Mais quand bien même ce serait le cas ? Il faut malgré tout qu'il existe une version Libre et qu'elle soit d'une qualité suffisante pour donner une image favorable du produit et de l'entreprise... C'est donc toujours une contribution favorable au patrimoine logiciel Libre de l'humanité.

Bien évidemment, les éditeurs open source sont les premiers à combattre l'association d'idées entre open source et gratuité. Certains vont même plus loin, et luttent contre l'idée que l'open source soit moins cher. Ils souhaitent, et on peut le comprendre, mettre en avant surtout les qualités distinctives de leurs produits. Mais l'honnêteté oblige à reconnaître qu'il existe aussi de très bons produits propriétaires sur beaucoup de créneaux. Les solutions open source ont de vrais atouts qui leur sont propres: elles sont en général plus respectueuses des standards et plus ouvertes, et sont souvent plus récentes, et donc bâties sur des principes d'architecture plus modernes. Enfin, le modèle noyau / extensions, qui s'est généralisé, leur permet d'offrir à la fois la solidité et la cohérence d'un noyau maîtrisé par l'éditeur, et une dynamique, un foisonnement que seul le développement communautaire peut apporter. Ce sont de vrais atouts, uniques. Mais les produits open source sont aussi presque toujours moins chers, tant en termes de licence que de déploiement, et c'est une erreur tactique, à mon avis, que de vouloir le nier.

Conclusion

Comme on l'a vu, un Logiciel Libre ne peut pas réellement être payant. Mais si l'on veut qu'un développeur gagne sa vie, voire même qu'il s'enrichisse, qu'il ait l'envie et les moyens d'en écrire davantage, bref si l'on veut que le modèle soit durable, alors il ne faut pas rêver d'un Logiciel Libre qui soit à l'écart des mécanismes économiques. Il est important au contraire d'en voir tous les rouages, de les analyser, de les comprendre.

Nous espérons y avoir contribué.

LE MODÈLE NOYAU-EXTENSIONS, L'ARME FATALE DES SOLUTIONS OPEN SOURCE

Coût, liberté, respect des standards, ouverture, ... les atouts des solutions open source sont bien connus, mais il est une force toute particulière de ces produits qui est moins connue et pourtant déterminante: le modèle noyau-extensions, massivement adopté par les éditeurs de solutions open source. En quoi consiste ce modèle ? C'est à la fois un modèle d'architecture, un modèle économique et un modèle d'écosystème.

UN MODÈLE D'ARCHITECTURE

En termes d'architecture logicielle, il s'agit simplement d'identifier un programme noyau offrant un premier niveau de services, et de mettre en place des points d'attache permettant d'y intégrer des extensions, qui apporteront des fonctionnalités supplémentaires, ou adapteront des fonctionnalités d'origine. Plus ces points de branchement d'extensions sont nombreux, plus le programme offre de possibilités d'extensions.

Au niveau technique, ce modèle a de nombreux bénéfices. Il permet en particulier de ne pas faire grossir inutilement le programme. On sait que version après version, sous la pression de la concurrence, les programmes intègrent de plus en plus de fonctionnalités, pas toutes nécessaires, qui les font grossir jusqu'à l'obésité. Cet embonpoint pèse sur les coûts de développement et de maintenance, voire même sur les performances, et peut nuire aussi à la facilité d'utilisation. Comme pour les systèmes d'exploitation, la solution est dans la flexibilité, et le modèle noyau-extensions permet à chacun d'installer les extensions correspondant à son besoin, sans pour autant alourdir le programme pour tous. Firefox est un des précurseurs du modèle, et un excellent exemple de l'équilibre que permet un noyau léger, associé à la richesse fonctionnelle des extensions.

Sur le plan technique toujours, le modèle noyau-extensions a un autre avantage important: il permet de modifier des fonctionnalités sans modifier du code source du noyau. Modifier le code source d'un programme pour l'adapter à son besoin est une opération délicate, qui peut déstabiliser le produit, mais surtout, qui sera à réintégrer manuellement aux versions suivantes du produit. Le modèle noyau-extensions offre la même flexibilité que la modification de code, mais dans une approche plus structurée, et traitant mieux les problèmes de montée de versions, puisque le plus souvent, les extensions restent compatibles avec une nouvelle version du produit.

UN MODÈLE D'ÉCOSYSTÈME

Mais le modèle noyau-extensions joue un rôle fondamental également dans la bonne cohabitation entre l'éditeur d'un produit et son écosystème.

Les licences Open Source permettent à chacun de modifier le code source d'un programme. Mais on sait bien que ce n'est pas toujours une bonne idée, et en particulier c'est une chose à

éviter s'il s'agit uniquement d'adapter un produit à son seul besoin. La seule bonne manière de modifier le code d'un produit Open Source est de s'impliquer dans la communauté, ce qui suppose d'une part d'avoir l'expertise requise pour être accepté comme commiter, et d'autre part que les commiters tiers soient bienvenus.

Le modèle noyau-extensions lève cette barrière, il permet à chacun d'enrichir un produit en écrivant ses propres extensions, sans modifier le code d'origine. Selon les cas, ces extensions peuvent ajouter une petite fonctionnalité en se branchant sur une API, mais elles peuvent aussi redéfinir entièrement une classe, et donc une fonctionnalité existante.

Pour les éditeurs Open Source, le modèle noyau-extensions a donc un bénéfice important: il permet de tracer une frontière entre le domaine de l'éditeur et le domaine de la communauté. Le noyau, c'est l'éditeur, les extensions, c'est le terrain de la communauté et des intégrateurs. L'éditeur s'engage sur la stabilité du noyau, avec des équipes compactes et structurées, tandis que différents acteurs de l'écosystème enrichissent le produit par des extensions.

La frontière existe aussi en termes de propriété intellectuelle. Lorsque des tiers contribuent directement à son programme, l'éditeur s'assure en général que chaque contributeur lui cède la propriété de ses travaux en signant un contributor licence agreement, car il ne veut pas que la propriété du programme soit parcellisée, ce qui rendrait difficile un changement de licence. Mais cette disposition juridique complique la contribution, tandis que la contribution au moyen d'extensions n'a pas cette contrainte.

UN MODÈLE ÉCONOMIQUE : LA PLACE DE MARCHÉ

Un des aspects les plus intéressants du modèle est la naissance de véritables bourses d'échange, des référentiels permettant de rechercher et de télécharger des extensions. Les éditeurs ont bien compris l'importance des extensions, et stimulent l'offre en mettant en place ces plates-formes d'échange, avec classement thématique, outils de recherche, mais aussi dans certains cas, des commentaires d'utilisateurs, voire des notations. Ils permettent souvent de classer les extensions par niveau de popularité, selon le nombre de téléchargements. Community managers, developper days, tutoriaux, promotion des meilleures extensions... nombreuses sont les actions stimulant le modèle.

Les extensions sont d'origines diverses, toutes ne sont pas de qualité irréprochable, et c'est le savoir-faire de l'intégrateur que de sélectionner les meilleures extensions, en scrutant les forums mais aussi en menant de véritables audits.

Il existe quelques variantes de ces plates-formes d'échange: dans certains cas, les extensions sont toutes gratuites, pour d'autres produits le gratuit cohabite avec le payant. De même certains éditeurs restent dans un rôle de facilitateur, tandis que d'autres mettent en place une certification qualité des extensions. Enfin pour certains, les extensions font partie du business model, l'éditeur prenant une commission sur chaque vente, à la manière de l'Appstore.

Un atout concurrentiel

Dans une logique ancienne, un éditeur pourrait voir les extensions comme une petite part de valeur ajoutée qui lui échappe, qu'il pourrait revendiquer. Mais les éditeurs modernes ont bien compris que, au contraire, chaque extension, gratuite ou payante, l'enrichit en enrichissant son produit. Si son produit peut être enrichi par des centaines de programmeurs dans le monde sans qu'il n'ait rien à payer, il ne peut être que gagnant. Une sorte de variante du crowdsourcing, qui décuple la capacité de développement autour du produit.

Pour les utilisateurs, bien sûr, la quantité d'extensions disponibles est aussi l'assurance de trouver une réponse à un besoin particulier, prête à l'emploi, sans le moindre développement. Donc un facteur d'économie et de flexibilité. Mais, le nombre d'extensions est aussi devenu une mesure de la dynamique d'un produit, de sa popularité, de sa part de marché.

Et le propriétaire ?

En principe, ce modèle noyau-extensions n'est pas réservé aux produits Open Source, ni au plan technique, ni au plan des licences. Le libre accès aux sources du produit facilite malgré tout la mise en place du modèle, mais ce n'est pas une condition nécessaire, et rien n'interdit à un éditeur propriétaire de mettre en place un tel schéma d'extensibilité. Pourtant ils sont peu nombreux à l'offrir, et lorsqu'ils le font, c'est à une échelle très différente. En fait, quelques éditeurs propriétaires sont également parvenus à embarquer des offres de tiers environnant leur produit, et peuvent aussi revendiquer une sorte d'écosystème. Mais généralement il ne s'agit pas de petites extensions, plutôt de développements d'envergure, et bien sûr, jamais gratuits. Ainsi il y a 25 extensions d'origine tierce annoncées sur le site de SAP, contre plusieurs milliers pour les solutions Open Source les plus actives.

Pourquoi ? Sans doutes en partie parce que l'éditeur revendique toute la création de valeur possible autour de son produit. Mais, la principale explication est plutôt du côté des clients et utilisateurs de ces produits. En effet, une grande majorité des extensions Open Source ne proviennent pas de l'investissement d'un éditeur tiers, mais d'un développement initialement réalisé pour un déploiement particulier, puis rendu générique et mis à disposition de tous. Or, le client d'une solution propriétaire n'est pas dans un logique de partage, il n'imagine pas qu'un développement fait pour lui, qu'il a chèrement payé, puisse être utilisé gratuitement par d'autres. Il s'attendrait à en tirer un bénéfice économique, mais comme son métier n'est pas l'édition de logiciel, ce serait trop compliqué, de sorte qu'il préfère simplement garder son petit développement pour lui.

A l'inverse, en tant qu'intégrateur de solution Open Source, nous constatons couramment que nos clients sont très ouverts à l'idée de partager leurs développements. Il y a même un véritable effet boule de neige: plus l'on a trouvé d'extensions utiles fournies par d'autres, plus on est disposé à en reverser.

Quelques exemples.

Magento, une solution d'e-commerce Open Source très en vogue, compte plus de 2400 extensions, environ un tiers sont gratuites, les autres ont un prix souvent de l'ordre de 100 dollars. La place de marché Magento-Connect présente un système de commentaires (reviews), pas de notation, ni de certifications, mais un classement par nombre de téléchargements.

Environ 50% des extensions sont des jeux de templates, permettant de mettre en place une charte graphique prédéfinie, les autres portent sur des ajouts ou modifications de fonctionnalités. Un déploiement typique de Magento peut n'utiliser que très peu d'extensions, par exemple une pour la localisation, une pour l'optimisation du référencement, et une pour la logistique. Pour Magento, les extensions contribuent à la dynamique du produit, mais non au modèle économique. Pour Prestashop, autre solution d'e-commerce Open Source, la vente d'extensions est une source de revenus, l'éditeur prélevant 30% du prix de vente des extensions, à la manière de Apple.

Dans le monde de la gestion de contenus, Drupal est un parfait exemple du modèle noyau-extensions, il se caractérise par un tout petit noyau, qui se réduit à un moteur de blog, et qui n'est pratiquement jamais déployé seul. La gestion de contenus structurés, par exemple, qui est native dans beaucoup de CMS, est implémentée par le module CCK, pratiquement toujours déployé. Drupal compte de l'ordre de 6000 extensions, toutes gratuites. Il n'y a pas de certification, ni d'évaluation, ni même de reviews par la communauté, mais le classement par défaut est celui du nombre de downloads, qui reflète la popularité.

Dans le domaine de l'ERP, l'extensibilité est bien plus importante encore que dans le contexte du e-commerce ou de la gestion de contenu, car la diversité du besoin est immense. Ici, il ne suffit pas d'ajouter une petite fonctionnalité, il est véritablement question de redéfinir des classes métier, qu'il s'agisse de produits, de clients ou de factures, pour les adapter à son besoin. Ainsi le produit OpenERP compte près de 700 modules d'extension, dont 200 proviennent de l'éditeur, 400 de partenaires intégrateurs, et une centaine sont d'origine communautaire. Elles sont toutes Open Source, mais certaines relèvent d'un programme dit de "shared funding", qui vise à amortir leur développement sur les premiers déploiements.

Conclusion

Comme on l'a vu, le modèle noyau-extensions, parfois aussi appelé "hub-and-spoke" (moyeu et rayons) est bien l'arme fatale des solutions Open Source. Il joue surtout un rôle essentiel en permettant de combiner la logique centralisée de l'éditeur avec le foisonnement darwinien du développement communautaire. Les éditeurs modernes l'ont parfaitement compris, et autant pour des raisons techniques qu'économiques, ils ont pratiquement tous adopté ce modèle, qui réconcilie parfaitement l'Open Source dit commercial, avec la démarche communautaire, apportant aux clients et utilisateurs les bénéfices combinés de l'un et de l'autre.

OPEN SOURCE ET BIENS COMMUNS

L'un des aspects du logiciel libre les plus essentiels à mes yeux est de constituer un patrimoine de code source, comparable au patrimoine du savoir scientifique, un bien commun disponible à tous, et qui est la condition du progrès.

Il est intéressant de pousser un peu cette analyse, en voyant comment le logiciel open source s'intègre à la notion plus générale de biens communs. Les biens communs ont intéressé des économistes de renom, et nous n'aurons pas la prétention ici de contribuer à la réflexion, mais seulement d'en faire une brève synthèse, en l'appliquant au domaine particulier du logiciel, du logiciel libre et open source en particulier.

En 1968, Garret Hardin publiait un article dans la revue Science, intitulé *« the tragedy of the commons »*, la tragédie des biens communs. Il y décrivait comment différents acteurs économiques profitant d'un bien commun peuvent le consommer jusqu'à épuisement, sans qu'aucune régulation n'intervienne. L'exemple choisi est celui d'une prairie partagée, sur laquelle différents éleveurs peuvent faire paître leurs bêtes. Lorsqu'un éleveur unique fait paître son troupeau sur une pré qui lui appartient, chaque bête supplémentaire lui apporte un bénéfice, mais également un coût, à savoir la disparition d'une partie de l'herbe. Lorsque de nombreux éleveurs font paître leurs troupeaux sur une même prairie, chaque bête supplémentaire apporte un bénéfice à son propriétaire, tandis que le coût, lui, est partagé par l'ensemble des éleveurs. Il s'ensuit que l'intérêt de chaque éleveur est de maximiser son troupeau, jusqu'à ce que ces éleveurs ensemble consomment la prairie et collectivement détruisent leur bien commun et leur revenu.

L'échec des récentes négociations sur la protection du thon rouge est malheureusement caractéristique de ce drame des biens communs : parce que chaque pêcheur, mais aussi chaque pays, individuellement, n'a pas d'intérêt à réduire sa propre part de capture, l'ensemble des acteurs, ensemble, ne savent s'empêcher la surexploitation et la destruction du bien commun, jusqu'à même qu'il soit impossible à reconstituer.

Différents mécanismes peuvent éviter la tragédie. Pour Hardin, l'un d'entre eux est le découpage du terrain, et la privatisation des parcelles, l'évacuation du bien commun, donc. Une autre solution peut être au contraire la nationalisation du bien, ou du moins l'organisation de sa gestion à un certain échelon de collectivité, où les acteurs définiront des règles régissant l'utilisation commune des biens et se donneront les moyens de faire appliquer ces règles.

La gestion des biens publics a été le principal thème de recherche de Elinor Ostrom, qui a reçu le prix Nobel d'économie en 2009 pour ses travaux. Elle a étudié en particulier comment différentes communautés à travers le monde ont géré la question des biens publics, souvent avec succès. Ni en privatisant, ni en nationalisant, mais en établissant ensemble les modalités d'une gestion collective équitable et efficace. Car si privatisation ou nationalisation peuvent être des solutions, ce ne sont pas toujours les solutions optimales au plan économique. Il existe des domaines où une régulation communautaire est préférable, non pas en termes éthiques mais bien en termes économiques.

Mais si les biens communs sont parfois gérés avec sagesse par de petites communautés, force est de constater qu'à une échelle supérieure, ils sont bien souvent mal gérés. Dans le cas du thon rouge, ni la privatisation, ni même la nationalisation ne sont envisageables.

Certains économistes abordent la gestion des biens communs par la théorie des jeux, qui modélise les interactions entre individus, susceptibles chacun de choisir la collaboration, un intérêt collectif, ou la défiance, un intérêt personnel. Un cas particulier est le dilemme du prisonnier, un jeu qui montre comment l'intérêt particulier de chaque intervenant ne permet pas d'atteindre la situation qui serait optimale pour tous. La théorie des jeux montre toutefois qu'en présence d'interactions répétées entre les intervenants, une attitude de coopération peut être récompensée, et une attitude égoïste punie, ce qui permet d'atteindre un optimum global.

En cette semaine du développement durable, il est utile de se rappeler que le premier des biens communs est la terre, et l'ensemble de ses ressources. Et le CO2, comme la pollution de l'air, et le réchauffement climatique dans son ensemble, sont des formes de biens communs en négatif, c'est à dire non pas des ressources à partager, mais des dommages à éviter: produire plus que sa part du dommage dont tous souffriront est équivalent à consommer plus que sa part d'un bien commun.

Les logiciels open source (on y arrive...), qui sont par essence, à la disposition de tous, sont également des biens communs, une forme de patrimoine de l'humanité. D'ailleurs, le rapport intitulé *« Floss Roadmap »*, un document élaboré dans le cadre de l'Open World Forum, compare le logiciel open source à une forêt, en tant que bien commun dont différents types d'acteurs ont la charge.

Mais prendre image sur la forêt, aussi bien d'ailleurs que sur les ressources halieutiques, ou encore l'eau douce des nappes phréatiques, ne permet pas de bien analyser le cas de ces biens immatériels particuliers que sont les programmes.

Les économistes parlent de biens rivaux, ou non-rivaux. Un bien rival est un bien dont chaque consommateur prive nécessairement les autres. Les biens matériels sont en général des biens rivaux, mais tous les biens immatériels ne sont pas non-rivaux, typiquement les noms de domaine.

Le logiciel libre est un bien immatériel non-rival: lorsqu'une personne installe le navigateur Firefox sur son poste de travail, il ne prive aucune autre personne, il ne réduit pas le stock de navigateurs disponibles

En matière de biens rivaux, les profiteurs, les *« free riders »*, nuisent directement à la communauté, et peuvent même directement mener le système à sa fin. En matière de biens immatériels, chaque profiteur n'ajoute pas véritablement à la nuisance, mais du moins ne contribue pas à créer ou à renouveler la ressource, ce qui peut avoir la même conséquence. L'intérêt de chaque utilisateur d'un logiciel libre est d'attendre que les autres améliorent le programme. Du moins, c'est son intérêt perçu, à la manière du prisonnier qui dénonce son complice, car son intérêt véritable est sans doutes de collaborer à l'élaboration d'un programme meilleur.

Le logiciel est à certains égards un bien inépuisable car indéfiniment duplicable. Mais il est aussi un bien périssable, car un programme qui n'évolue pas perd une bonne part de son utilité, donc de sa valeur avec les années. On a donc un bien qui peut être consommé sans limite, mais doit néanmoins être entretenu et renouvelé. Ainsi, même pour des biens immatériels et non-rivaux, la question du développement durable reste pertinente. D'autant qu'à la différence de la prairie ou du poisson, il ne suffit pas de diminuer son prélèvement pour espérer un renouvellement.

Avant de conclure, il est intéressant d'évoquer aussi la notion d'anti-bien commun, *« anticommons »*, et l'expression *« tragedy of the anticommons »*, que l'on doit à Michael Heller. Elle fait référence à une situation où au contraire, le fait de considérer un bien comme privé et non commun, nuit à l'intérêt de chacun. Et l'un des exemples utilisés par Heller est dans le domaine des brevets. Il montre que, dans certains cas, un nombre excessif de détenteurs de brevets intervenant dans un même domaine rend leur utilisation si complexe qu'ils sont dans les faits inutilisables, de sorte que les titulaires des brevets ne peuvent en tirer aucun profit. Même s'il y a bien d'autres bonnes raisons de luter contre les brevets logiciels, c'en serait une supplémentaire: en matière de logiciel, la tragédie des anti-biens-communs est bien de nature à bloquer la mise en œuvre de l'innovation.

En synthèse, nous avons vu ici en quoi le patrimoine logiciel open source est un bien commun de l'humanité, un patrimoine. Un bien commun immatériel, duplicable à l'infini, mais qui n'est pas inépuisable pour autant, c'est son grand paradoxe. Comme les pêcheurs de homards de Nouvelle-Angleterre étudiés par Elinor Ostrom, les communautés de développement se donnent des règles pour récompenser les contributeurs, dissuader les tricheurs, et faire prospérer leur bien commun. L'open source a connu de grandes victoires, mais les menaces sont nombreuses, et il est bon de se rappeler parfois que les biens communs ne sont pas faciles à entretenir par les seuls mécanismes économiques. On ne voudrait pas voir le logiciel libre finir en sushi.

LE MARKETING, FAIBLESSE OU FORCE DE L'OPEN SOURCE ?

Les solutions Open Source disposent d'armes différentes pour se faire connaître. Elles n'ont pas les mêmes budgets marketing, mais elles sont nées avec le Web, dans les communautés, avec le buzz. Et pour certaines, le marketing Open Source est même un des fondements du business model.

Dans un monde de communication, dans un monde où chaque entreprise doit payer sa place sous les feux des projecteurs, où chaque parcelle de visibilité se monnaye chèrement, l'Open Source souffre d'un déficit de marketing. "Si ce produit est aussi bon que vous le dites, pourquoi ne l'ai-je pas vu au grand salon des solutions, le mois dernier ?"

Comme souvent, quand on parle d'Open Source, il faut distinguer les trois grandes typologies de produits et d'acteurs : fondations et communautés d'une part, éditeurs d'autre part.

Voyons d'abord, l'Open Source des grandes fondations, Linux, Apache, Eclipse, etc. Ces institutions sont à but non-lucratif, fonctionnent principalement par des dons en nature, soit de code source, soit de temps-homme. Elles gèrent de tous petits budgets, généralement consacrés aux plates-formes de développement. Elles n'ont clairement pas vocation, ni argent, pour acheter des encarts dans la presse IT.

Mais ces institutions jouissent d'une phénoménale renommée, d'une forte visibilité, et leurs sites bénéficient d'un référencement extraordinaire. De plus, le caractère non économique de leur fonctionnement, et le parrainage des plus grands acteurs du marché, leur donne une très forte crédibilité.

On a vu comment, sans budget de communication, Mozilla pouvait prendre des parts de marché à Internet Explorer, Linux à Windows ou Eclipse à JBuilder. Les produits phares sont sur le devant de la scène quoi qu'il advienne. Mais les autres ? Nous voyons par exemple tous les jours des entreprises déployer des moteurs de recherche propriétaires coûteux, dont les armées de commerciaux ont fait leur travail comme il se doit, en évitant de dire qu'il existait un produit Apache, SolR, faisant la même chose, et souvent mieux, sans coût de licence.

Ces produits Open Source de moindre visibilité, présents sur un marché où le marketing et l'action commerciale sont puissants, ne sont connus que de ceux qui peuvent et savent faire des recherches, évaluer, prendre la maîtrise des outils. Car il n'y a pas de commercial chez Apache pour vous faire une démo de SolR, et vous faire l'article.

Avec les produits de fondations, comme avec les produits communautaires, c'est en fait aux prestataires de prendre à leur charge l'essentiel du marketing. C'est la moindre des choses, d'ailleurs, puisque les mêmes prestataires récupèreront en retour l'essentiel du budget de déploiement. C'est la logique que nous avons adoptée chez Smile, avec la mise à disposition et la promotion de livres blancs, pour à la fois présenter et analyser l'offre du marché sur

chaque segment de produit, mais aussi, clairement, faire le marketing des solutions Open Source en général. C'est une mission qui incombe à l'intégrateur, car nul autre ne prendra ce flambeau.

Voyons maintenant les éditeurs Open Source, ces sociétés "normales", c'est-à-dire à but lucratif, qui ont choisi de distribuer leur produit sous licence Open Source, en tout ou partie. Bon nombre de ces éditeurs sont encore des sociétés de relativement petite taille, dont l'essentiel des ressources est consacré au développement produit. La plupart n'ont pas les moyens de payer un stand de 30 m2 dans les plus grands salons professionnels, ni la quatrième de couverture de la presse magazine.

Les solutions Open Source, du moins sur certains marchés, sont un peu dans la situation des lessives de distributeurs: pas de publicité à la télé, pas de présence en tête de gondole, pas de marketing, packaging minimaliste, pas de publi-reportage... et donc un produit moitié moins cher, de qualité identique. L'analogie a ses limites, car sur bien des terrains, les solutions Open Source ne sont pas de qualité identique mais de qualité supérieure, en termes de robustesse, ouverture, pérennité, dynamique de développement, extensibilité, etc.

La faiblesse de leurs dépenses marketing pourrait réjouir leurs clients: chaque euro dépensé en contrat de support n'ira qu'à des choses directement utiles: développement produit, maintenance, assistance utilisateurs. Ils ne financeront pas sans le vouloir la prochaine Coupe de l'America. Alors que chez certains grands éditeurs de solutions propriétaires, lorsqu'ils sont en situation proche du monopole, c'est souvent plus de 60% des recettes qui passent en commerce et marketing, sans bénéfice direct pour le client.

Mais on ne peut en rester à cette analyse, en se réjouissant de la saine utilisation des revenus. Si les sites professionnels sont remplis d'encarts ventant les solutions propriétaires, si l'on ne croise sur les salons que ces mêmes solutions, et si par un effet en cascade, le Gartner et les médias professionnels parlent surtout de ces mêmes produits, alors le décideur sera tenté de penser que c'est ce qui se fait de mieux.

Or comme on le sait, le Web révolutionne le marketing comme bien d'autres aspects de nos business. Au jeu du marketing viral, du marketing social, l'Open Source a aussi des atouts spécifiques. Parce qu'il est par nature directement appuyé sur des communautés, l'Open Source n'a pas besoin de payer des boites de buzz-marketing à mettre des faux posts dans tous les blogs de la planète. Parce qu'elles sont directement appréhendées et appréciées par de vrais leaders d'opinion, les solution Open Source de qualité sont immédiatement relayées dans une nuée de twits d'influence. L'Open Source est né dans les communautés et dans les réseaux, et n'a donc nul besoin d'y payer sa place.

Et d'ailleurs, une manière purement cynique d'analyser le business model d'éditeur Open Source serait de le voir comme un deal un peu étrange : marketing gratuit contre code source libre. Même si une majorité des acteurs adhère probablement aux valeurs profondes, philosophiques et de responsabilité sociales, de l'Open Source, il n'empêche que c'est bien la logique économique sous-jacente: le caractère Open Source va permettre la diffusion ultra-rapide du produit – pour autant qu'il soit bon – et son adoption par une immense base

d'utilisateurs. Certains ne seront qu'utilisateurs, d'autres seront testeurs bénévoles, d'autres développeront des extensions qui nourriront un écosystème favorable au produit. Mais tous en parleront à leurs collègues, écriront sur leurs blogs, posteront des commentaires, twitteront … et lanceront un buzz à l'échelle mondiale.

Le produit Magento en est une belle illustration, qui en l'espace de deux ans seulement est devenu le produit e-commerce dont tout le monde parle, le produit leader dans le monde entier, le produit qui concentre la réalisation d'extensions. Il fallait pour cela un excellent produit, bien en phase avec les attentes du marché. Mais il fallait aussi qu'il soit Open Source. Jamais une solution propriétaire n'aurait pu s'imposer de la sorte, à l'échelle mondiale, même avec un budget publicitaire de dizaines de millions d'euros. La campagne n'a pas été chère payée, le deal fondateur est évidement gagnant.

C'est tout le paradoxe, et ce sera notre conclusion : si le marketing ordinaire est la faiblesse de l'Open Source, le marketing moderne, 2.0 si l'on peut dire, est sa force, voire son fondement.

BANALISATION DE L'OPEN SOURCE, BONNE NOUVELLE ?

Il n'est pratiquement plus une entreprise dont une part du système d'information ne soit construit avec des composants et solutions Open Source. Est-ce la consécration ? Oui, d'une certaine manière. Est-ce la fin de l'histoire ? Certainement pas.Certainement pas.

L'Open Source continue de monter en puissance, et il n'est pratiquement plus une entreprise dont une part du système d'information ne soit construit avec des composants et solutions Open Source. L'Open Source est partout, et tous les acteurs se disent désormais Open Source. Les anglo-saxons disent « Open Source is going mainstream », c'est à dire, en quelque sorte, entre dans la normalité, voire même devient le standard. Est-ce la consécration ? Oui, d'une certaine manière. Est-ce la fin de l'histoire ? Certainement pas.

On peut analyser cette banalisation de l'Open Source sous plusieurs angles.

Le plus positif, bien sûr, concerne les entreprises adoptant l'Open Source : pour une grande majorité de clients, de DSI, quelle que soit la taille d'entreprise ou son secteur d'activité, les solutions Open Source sont désormais pleinement acceptées, et leurs bénéfices parfaitement appréciés. C'est énorme. La banalisation ici, est bien une forme de consécration. Que de chemin parcouru ! Du côté des prestataires, ce marché florissant a évidemment attiré du monde, au-delà des experts initiaux. Dans les années 95-2000, les SSII généralistes avaient clairement raté la première vague du Web, laissant le marché aux "web agencies" de l'époque. A partir de 2000, les généralistes s'y étaient finalement lancés. "Le web, c'est devenu banal, tout le monde fait du Web, vive le Web, nous aussi nous pouvons être experts en Web". Le Web s'est généralisé, certes, mais pas réellement banalisé, de sorte qu'aujourd'hui encore on compte très peu de grands sites de l'Internet réalisé par ces généralistes. Même scénario dans l'Open Source, 10 ans plus tard. "Tout le monde fait de l'Open Source, vive l'Open Source, nous aussi nous voici experts en Open Source », c'est le nouveau cri de guerre des SSII généralistes. Mais ici aussi, généralisation ne veut pas forcément dire banalisation, car l'offre Open Source est trop dense et foisonnante, et bouge trop vite pour ces grands acteurs. Il leur reste à comprendre que l'Open Source n'est pas une technologie, mais plutôt une manière d'aborder la technologie.

Pourtant, même si elle ne réussit qu'à moitié, la banalisation côté prestataire est plutôt une bonne chose. Elle élargit l'offre de services, aide à convaincre les DSI encore hésitantes, et peut ouvrir de nouvelles portes aux solutions Open Source.

Une certaine forme de banalisation est liée tout simplement au prix. Dans les années 90, les produits Open Source étaient majoritairement issus de fondations ou de communautés, et donc gratuits. La gratuité était alors tellement associée à l'Open Source qu'elle en était devenue pour certains une caractéristique implicite. A tel point qu'il a fallu expliquer, rabâcher presque, que Libre ne voulait pas dire gratuit.

Petit à petit, à partir des années 2000, les éditeurs Open Source commerciaux sont montés en puissance, et ont adopté des modèles économiques divers, basés sur le support pur ou sur un support associé à des versions spéciales entreprises, rendant l'usage le plus souvent payant.

Pour ceux qui ne se souciaient pas de la liberté de modifier les programmes, ou de leurs autres qualités uniques, l'Open Source non gratuit est une forme un peu décevante de banalisation.

La banalisation et l'oubli du sens : une grande menace qui pèse sur l'Open Source

Dans ce domaine, les éditeurs qui réussissent sont ceux qui ont réellement compris les spécificités de l'Open Source, ceux qui savent s'appuyer sur des communautés pour, au minimum, environner leur produit d'un foisonnement d'extensions, ceux qui sont nés dans le berceau de l'Open Source, et portent un minimum de croyances.

Mais aux côtés de ceux-ci, on voit apparaître aussi de l'Open Source de pur opportunisme. Des éditeurs venus du propriétaire, parfois en difficulté sur leur marché, et qui se disent un jour "bon, s'il faut être Open Source maintenant pour arriver à vendre, alors nous aussi on est Open Source. Comme tout le monde, finalement. Si vous me dites que je vais pouvoir faire payer autant... je n'ai pas de problème avec l'Open Source. Surtout si la version qui marche réellement, celle que les gens installeront vraiment, n'est justement pas sous licence Open Source." Bref, pour ceux-là, la banalisation c'est le paradis. Un peu artificiel.

On est partagé devant ces acteurs nouveaux venus. D'un côté, il faut malgré tout qu'une version au moins de leur produit soit sous une licence Open Source, le plus souvent GPL. Même modeste, c'est toujours une contribution appréciable au patrimoine logiciel disponible pour tous, y compris pour d'éventuels forks. Et puis, d'une manière plus large, l'essor phénoménal de l'Open Source commercial a fait naître une typologie de programmes qui n'auraient pas pu voir le jour dans l'Open Source communautaire ou de fondations.

Le fait qu'une majorité des utilisateurs de leur produit soient sous une licence non Open Source n'est pas un problème en soi. Ils reçoivent un support de qualité, et ils financent un développement dynamique du produit. Non, le regret n'est pas au plan économique, ni même éthique, mais plutôt dans le domaine sémantique: le sens des mots s'estompe doucement, se perd parfois. C'est le côté sombre de la banalisation.

Or justement, un danger plus grave encore est la banalisation orchestrée par les acteurs du monde propriétaire. Car de ce côté, c'est une vraie stratégie d'étouffement qui est à l'œuvre. Dans les années 90 déjà, on avait donné un nom à cette stratégie déployée par Microsoft principalement : "Embrace, Extend, Extinguish", Adopter, Étendre, Éteindre. A l'époque, il s'agissait des standards de l'Internet. Plutôt que de combattre les standards dominants, la stratégie consiste à les adopter, puis les étendre, c'est-à-dire ajouter des possibilités et des paramètres, puis finalement les éteindre, c'est-à-dire que le standard au final n'en est plus un, il a été pourri par une multitude d'implémentations divergentes.

La même stratégie est à l'œuvre aujourd'hui concernant l'Open Source. Microsoft a créé une (relative) surprise en étant bien en vue au salon Solutions Linux. Quoi ? Microsoft ne combat plus l'Open Source ? Oui, vous avez compris, désormais Microsoft "embrasse" l'Open Source. La banalisation est donc actée : si maintenant Microsoft fait de l'Open Source, c'est vraiment

que l'Open Source serait devenu aussi transparent que l'air que nous respirons: vital si l'on veut, omniprésent sans doutes, mais surtout oublié. On est passé au stade "extend": puisque tout est plus ou moins Open Source, le concept même se perd dans la banalité. Mais l'air que nous respirons s'appauvrit peu à peu en oxygène. L'étape finale est l'extinction: tout le monde est gentil, c'est merveilleux, mais c'est donc que gentil ne veut plus rien dire.

Les pères fondateurs s'étaient attachés à définir avec précision ce que voulaient dire les mots "Logiciel Libre" (les 4 libertés) ou "Open Source" (les 10 conditions). Ils avaient senti certainement qu'il y avait un danger dans le flou, que si l'on pouvait se dire "plus ou moins Open Source", alors tout le monde allait arborer cette étoffe vaporeuse, et l'on risquait de ne plus savoir de quoi on parle. C'est un peu ce qui se passe aujourd'hui.

A cet égard, il ne fait aucun doute que la banalisation, et l'oubli du sens, est la seconde grande menace qui pèse sur l'Open Source aujourd'hui, juste après le Saas, et l'oubli du programme.

LE SAAS, AMI OU ENNEMI DE L'OPEN SOURCE ?

Le monde de l'open source est partagé face à la montée en puissance du Saas (Software as a Service). Père fondateur, Richard Stallman y voit " *pire qu'une imbécillité* " : la perte de contrôle de l'utilisateur non seulement sur les programmes, mais, plus grave, sur les informations. A l'inverse, nombre d'éditeurs open source discernent dans le Saas la possibilité d'un business model enfin limpide, qu'ils cherchent encore à mettre au point.

LA DISPARITION DU CODE SOURCE

A y regarder de plus près, le Software as a Service, comme son nom l'indique, substitue le service au logiciel : le programme n'est plus, vive le service. Si le programme disparaît, le code source du programme disparaît plus encore, et le fondement même de l'open source s'en trouve aboli. En première analyse, le Saas serait une forme de bundle mêlant application et hébergement, le tout confondu pour offrir un service. La vraie révolution tient alors en partie dans la démarche " DIY " (Do it Yourself) : l'application arrive totalement prête à l'emploi, en self-service. Il reste si peu de configuration, et d'une telle simplicité, que le client la prend en charge sans aucune expertise. Le Saas entraîne alors avec lui la fin du prestataire informatique. Du moins, pour les catégories d'applications éligibles, car on ne pourra pas tout faire en mode DIY. Les fournisseurs d'offres Saas s'appuient massivement sur l'open source. Ils veulent construire des applicatifs web de grande envergure sur des socles modernes et extensibles, et l'open source offre le meilleur des socles, une manne. Problème, même une société telle que Google considérée, à certains égards, comme le numéro 1 du Saas, et qui raffole d'open source, se trouve parfois accusée de ne pas assez reverser ses œuvres dérivées de l'open source. Face aux exigences de la licence GPL(*), mettre à disposition un service, gratuit ou pas, ne s'assimile pas à distribuer un programme, de sorte que l'on peut partir d'un code sous GPL, construire une œuvre dérivée et la proposer en mode Saas, sans se voir obligé de diffuser cette œuvre sous GPL. Cette faille, sérieuse, a été comblée par la licence AGPL, qui tient compte ce cas de figure. Mais celle-ci reste modestement utilisée pour le moment.

Cependant, le Saas semble une opportunité pour l'écosystème open source. Les éditeurs du secteur se tournent vers d'autres sources de revenus : maintenance, versions " entreprise " non libres, plus stables ou bien aux fonctionnalités élargies. Or, réunir application et hébergement dans un bundle Saas apporte une réponse limpide à la question : " Comment sera financé un logiciel libre ? " Le client ne payera ni un droit d'utilisation, ni un serveur, ni un hébergement, ni les jours d'un prestataire : il paie un service. Ainsi, pour les éditeurs open source, le Saas offre deux attraits : une simplicité de déploiement et d'exploitation qui répond à une réelle attente du marché, et un modèle économique, enfin, parfaitement lisible.

UN VERROUILLAGE PIRE QUE CELUI D'UN LOGICIEL PROPRIÉTAIRE

Parfait, mais que deviennent alors les codes sources ? Le risque est réel que les programmes, une fois invisibles, ne deviennent insignifiants. Les sources de l'application, mise en Saas, resteront-elles disponibles ? Voilà la question que doit se poser le client. Car à certains égards, le verrouillage (lock-in) dans le Saas pourrait s'avérer pire que celui d'un logiciel propriétaire : non seulement changer de programme deviendra compliqué, mais même la récupération des données posera problème. Et si récupérer ses données est surtout affaire contractuelle, pour qu'elles soient utilisables, il faut encore que le programme reste disponible, et libre.

Les vraies caractéristiques des solutions open source, si elles ne tiennent pas à la gratuité, s'inscrivent dans le respect des standards, dans l'ouverture, dans la dynamique de développement, les contributions communautaires, la rapidité de maturation. Que deviendront ces atouts en mode Saas ? Pourra-t-on faire cohabiter le foisonnement d'une offre communautaire avec l'engagement de l'éditeur du Saas en termes de qualité de service ? C'est l'enjeu. Mais il serait risqué d'oublier que derrière le Saas, il y a des programmes.

IL FAUT ENSEIGNER LE LOGICIEL LIBRE

[Tribune parue dans le journal Le Monde du 18 octobre 2012. Co-écrite par Stéfane Fermigier, Roberto di Cosmo et Patrice Bertrand]

Presque tous les aspects de notre vie dépendent directement ou indirectement d'une multitude de logiciels, et une grande partie de ces logiciels sont aujourd'hui des logiciels libres.

Alors que, par une circulaire du premier ministre Jean-Marc Ayrault, le gouvernement vient d'affirmer l'importance du logiciel libre dans les systèmes d'information de l'Etat et affiche une politique volontariste pour en accompagner l'utilisation, nous croyons indispensable d'intégrer l'étude des logiciels libres dans la formation des futurs ingénieurs. Pour la grande valeur pédagogique des logiciels dont le code source est disponible, pour les valeurs éthiques de partage qui les mettent en adéquation avec les missions de l'enseignement public, mais, plus encore, parce que les logiciels libres forment désormais la base de l'informatique moderne.

L'INFORMATIQUE EN PLEINE RÉVOLUTION

Le centre de gravité de l'informatique s'est déplacé au cours des dix dernières années. On passe progressivement de l'ère des ordinateurs individuels à celle de l'après-PC : navigateurs Web et objets communicants – smartphones, tablettes – donnent accès aux ressources de millions de serveurs qui constituent le "cloud".

Or l'immense majorité des systèmes d'exploitation, langages et outils de programmation utilisés depuis plus de dix ans pour développer les services des géants du Web mondial comme Google, Facebook ou Twitter, mais aussi des start-up, petites ou grandes, sont des logiciels libres. Il en va de même pour une grande partie des briques logicielles embarquées dans les objets qui nous entourent, des téléphones aux tablettes, des "box" des fournisseurs d'accès à Internet au système nerveux de nos automobiles.

Le logiciel libre est au coeur des technologies qui accompagnent cette nouvelle ère, et il a joué un rôle pionnier dans l'émergence des modes de collaboration qui s'imposent aujourd'hui comme des gisements de productivité et d'innovation pour l'ensemble de l'industrie et des services : l'innovation ouverte, qui implique le partage et la collaboration des entreprises avec des partenaires externes ; les wikis et les réseaux sociaux d'entreprise, qui décloisonnent l'entreprise traditionnelle ; le développement "agile", qui permet d'accélérer la mise sur le marché de produits mieux adaptés aux besoins.

Assurer en France une bonne connaissance des logiciels libres et une présence forte dans les projets libres les plus importants est un enjeu majeur pour maintenir la position française dans ce domaine stratégique. Si l'on veut que davantage de jeunes geeks français lancent leurs entreprises sur le Net, que la France tienne une place plus grande dans la (plus si) nouvelle économie et, plus largement, dans l'industrie informatique, il est indispensable que

le système éducatif leur apprenne à manipuler cette nouvelle matière première de l'informatique que sont les logiciels libres ou open source.

Le besoin d'une formation adaptée

L'écosystème du logiciel libre est le premier à avoir montré qu'il était possible de fédérer le travail de communautés de plusieurs dizaines à plusieurs milliers de développeurs répartis dans le monde entier, sans autre moyen de communication que l'Internet et des outils de collaboration nouveaux, pour réaliser des logiciels de qualité industrielle, dont beaucoup sont devenus des standards.

C'est en enseignant les programmes et les technologies du logiciel libre, mais aussi en associant les étudiants à son développement selon les modes d'organisation et de collaboration qui lui sont propres que l'on formera les jeunes ingénieurs à ces méthodes collaboratives, à ces approches ouvertes. Au-delà même de la sphère informatique, ces savoirs seront la clé de la compétitivité de nos entreprises au XXIe siècle.

Enseigner le logiciel libre nécessite un effort spécifique : il ne suffit pas d'utiliser des logiciels libres à la place de logiciels propriétaires, il faut expliquer les mécanismes employés pour permettre à des centaines de programmeurs éparpillés sur la planète de coopérer de façon cohérente sur des logiciels de plusieurs millions de lignes de code ; on doit apprendre les notions juridiques, organisationnelles et économiques qui sont à la base de l'écosystème du logiciel libre. Il convient aussi de mettre en contact les étudiants avec les communautés de développeurs.

Pour cela, il faut un effort pédagogique important, qui doit être soutenu par des mesures incitatives et non plus laissé simplement à la bonne volonté de quelques précurseurs. On doit aussi encourager la recherche qui se développe autour des logiciels libres et fournit des outils nouveaux pour accompagner leur essor.

Un gisement d'emplois futurs

Le logiciel libre porte des valeurs humanistes fortes, en considérant que le logiciel doit faire partie du patrimoine de connaissances de l'humanité, un bien commun qu'il convient de cultiver en commun.

Mais le logiciel libre est aussi au coeur d'une activité industrielle importante, encore souvent méconnue. Selon les études du cabinet Pierre Audoin Consultants, cette économie représente un chiffre d'affaires de 2,5 milliards d'euros en France, soit environ 30 000 emplois locaux. Le dynamisme de ce secteur, 30 % de croissance par an, signifie aussi un important gisement d'emplois futurs qui ont du mal à être pourvus par le système éducatif actuel, pour qui les développements logiciels sont encore perçus comme une voie inférieure.

Voilà une raison supplémentaire pour enseigner davantage le logiciel libre aux futurs jeunes diplômés : ils s'assurent des débouchés dans un secteur de pointe au dynamisme exceptionnel. Et il ne s'agit pas seulement des quelques centaines de sociétés spécialisées dans le logiciel libre en France puisque, selon une étude menée auprès de plus de 500

entreprises dans onze pays, plus de la moitié d'entre elles ont intégré le logiciel libre et open source à leur stratégie en matière de système d'information.

Ces entreprises ont besoin de compétences : il est important qu'elles les trouvent en France.