François Elie

Des trois communautés aux forges de demain

Complément à
Histoires et cultures du Libre. Des logiciels partagés aux licences échangées.

Framabook
le pari du livre libre

François ÉLIE est agrégé de philosophie et élu local (Angoulême), co-fondateur (et président) de l'Adullact (2002). Il défend depuis dix ans dans le Libre la cause des clients (les utilisateurs qui paient), spécialement publics, et promeut le développement mutualisé de logiciels libres métiers. Il est l'auteur de l'*Économie du logiciel libre* (Paris : Eyrolles, 2009).

Table des matières

Introduction

La compréhension de la galaxie du logiciel libre est difficile, car ses différents acteurs (développeurs, prestataires ou clients) ont des intérêts différents et n'ont pas les mêmes stratégies. Même en observant attentivement on ne gagne pas beaucoup en intelligibilité. Pour éclaircir les choses, nous allons partir de l'hypothèse (au demeurant peu coûteuse) qui est confirmée par les faits que dans l'ensemble la *libération* de l'informatique remonte progressivement des couches basses [1] aux applications métiers en répondant à l'intérêt des acteurs : les développeurs en premier lieu, qui développent pour eux-mêmes, les entreprises du secteur informatique ensuite, qui veulent investir moins en recherche et développement en mutualisant leurs efforts, les utilisateurs métiers enfin, désireux de dépenser moins, en mutualisant le plus en amont possible la recherche de solutions répondant à leurs besoins. Une lecture diachronique permettra, en distinguant des *étapes* successives dans l'histoire du logiciel libre, des *moments*, de distinguer au passage trois communautés d'intérêt et d'anticiper l'évolution probable du mouvement général.

Pour accréditer l'hypothèse de départ il faut faire l'analogie avec la *libération des mathématiques*. Cette analogie permettra de comprendre le premier moment (*free software*) : celui de la prise de conscience initiale au sein d'une communauté proche de l'esprit académique qui connaît depuis des siècles l'efficacité de la *coopétition* [2] : l'efficacité qui est celle de la science *ouverte* [3]. Cette première communauté libère principalement les couches basses de l'informatique, c'est-à-dire en gros les couches systèmes et réseaux, celles qui ne se voient pas. Excusez du peu, l'infrastructure de l'internet sera développée en grande partie dans cet esprit, et avec ces outils.

Au début du XXI^e siècle, la seconde communauté est celle des industriels, qui découvrent qu'ils peuvent trouver leur intérêt en évitant de *réinventer la roue* chacun de leur côté. Ils se mettent à faire des économies de recherche et développement au sein de consortiums où ils collaborent à la production de

1. « Le développement/la maturation de l'*open source* part de l'infrastructure (Linux, Apache…) et « monte » vers l'applicatif (Evolution, Plone…) ». Voir Jean (Aka) Sig, *Qui profitera de l'open source ?*, 2006.

2. Dans leur livre, Brandenburger et Nalebuff décrivent, à partir de la théorie des jeux, l'avantage qu'il peut y avoir à coopérer avec ses concurrents. Voir Adam M. BRANDENBURGER et Barry J. NALEBUFF, *Co-Opetition : A Revolution Mindset That Combines Competition and Cooperation. The Game Theory Strategy That's Changing the Game of Business*, New York : Currency Doubleday, 1997.

3. La science a commencé par être ésotérique, conservée dans des temples, transmise sous forme d'arcanes à des initiés. Bertrand Russell, dans son *Histoire de la philosophie occidentale*, se fait l'écho d'un avis assez largement partagé qui fait du *miracle grec* la naissance de cette science ouverte, qui reprendra avec l'avènement des universités à partir du XII^e siècle.

briques logicielles (du *middleware*) : c'est le second moment, le moment *open source.* Cette nouvelle donne ne change toutefois pas vraiment la situation captive des clients, qui risquent de *changer de maître* : une mutualisation par l'offre des éditeurs qui leur permettait de disposer de *progiciels* est remplacée par une autre mutualisation par l'offre, qui produit des *solutions logicielles* à base de composants *open source*.

La communauté des clients va vouloir tirer bénéfice d'une autre mutualisation des solutions métiers (par la demande), comme un club des utilisateurs qui se révoltent et veulent reprendre la main sur ce qu'ils paient.

En racontant cette histoire, on repérera les trois problèmes à lever pour que s'accomplisse la maturité de l'informatique libre : 1) il y a trop de travail non payé ; 2) le service produit trop peu de code ; 3) les clients (utilisateurs qui paient) n'ont pas encore pris toute leur place active dans l'écosystème de production. Afin de résoudre ces problèmes, et pour que les trois communautés participent d'un même écosystème au lieu de s'épuiser dans de stériles querelles internes, on voit déjà se profiler une solution structurelle autour des forges [4] au sein desquelles s'écrit le logiciel libre.

La révolution industrielle avait ses maîtres de forge. Il est intéressant que nous entrions dans la société de l'information avec des forges : ces *forges à coulée continue* de code, de contenus et de formats ouverts, sont les usines de cette révolution de la production qui donne naissance sous nos yeux à la société de l'information. C'est cette *forge de demain* dont l'idée sera esquissée à la fin de l'article.

1. Mais quelle est donc l'économie du numérique ?

Ce qui intéressait Aristote, lorsqu'il a eu l'idée de la science *économique*, c'était avant tout la manière dont les objets étaient *produits.* Mais dans un monde dominé par le commerce, la production nous intéresse peu d'ordinaire : nous confondons l'économie avec la finance. Nous croyons savoir comment les choses sont produites (pour toutes sortes de raisons : parce que c'est la consommation qui nous intéresse davantage, et aussi parce que nous préférons peut-être *ne pas savoir* comment les choses sont réellement produites, où et à quel *prix*). Nous nous imaginons aussi que la compréhension

4. On désigne par forge un ensemble de serveurs qui permettent de gérer la vie d'un logiciel : *suivis de version* permettant l'écriture du code, *tracker* permettant de gérer l'attribution et le suivi des *bugs* ou des demandes d'améliorations, serveurs *ftp* permettant de gérer les téléchargements, forums, listes de discussions, etc.

de l'économie est à chercher dans l'*échange*.

Il n'est donc pas très étonnant que l'arrivée d'objets numériques, dont le coût marginal de distribution est nul, et qui peuvent être copiés sans être re-produits, laisse perplexe. Plus paradoxal encore, nous observons dans le monde du logiciel que l'échange donne naissance à l'objet, et non l'inverse. C'est un échange non-marchand, un partage, qui est le contexte de création de ces objets numériques, dont l'existence suscite ensuite de l'échange marchand.

➥ 1.1 Rester dans le monde des choses

Il serait pourtant facile de rester dans le monde des choses, et d'ôter aux objets numériques leur caractère propre : il suffit que juridiquement il soit interdit de les copier et le tour est joué ! On restera ainsi dans le monde que l'on connaît si bien, dans le monde des choses où l'original vaut mieux que la copie, où l'on s'imagine que la distribution d'une chose donne forcément lieu à une re-production. Cette *réaction nobiliaire* [5] se crispe sur l'exploitation des droits de propriété, la fameuse *propriété intellectuelle* [6]. Remarquons toutefois qu'elle le fait en sapant les conditions et les motivations de la production, comme on peut l'observer dans l'évolution récente du droit d'auteur [7]. *Les efforts que l'on fait pour ne pas changer précipitent le changement.* La maladresse un peu voyante, un peu lourde, manifestement *réactionnaire* de ceux qui ne veulent pas que ça change, rend parfois d'autant plus sympathique le changement !

Malgré tous les efforts pour rendre péjoratif le mot *téléchargement*, malgré l'assimilation de tout échange non-marchand à du *piratage*, les possibilités offertes par le numérique apparaissent de plus en plus clairement. La vague recouvre les digues.

➥ 1.2 Cultiver le fantasme d'une autre économie ?

Faut-il pour autant considérer que ce paradis de l'échange, où tout pourrait être gratuit et abondant, remet en cause l'économie des choses ? Ceux qui ont

5. L'analogie est de Jean-Claude Guédon, dans un éditorial de *La Recherche*, en novembre 2000. La *réaction nobiliaire* désigne cette résistance au changement, au sein de la noblesse, au début de la Révolution française, qui s'accrochait au *droit*.

6. Recouvrant des régimes juridiques différents aux intentions différentes (le droit des marques, le droit des brevets et le droit d'auteur), la fiction assez récente d'une *propriété intellectuelle* participe d'ailleurs de cette *réaction nobiliaire*.

7. Franck MACREZ, *L'exploitation numérique des livres indisponibles : que reste-t-il du droit d'auteur ?*, Paris : Dalloz, 2012.

ce rêve voient dans les promesses du numérique des promesses qui valent dans le monde des choses, anticipant un nouveau partage de celles-ci.

Ceux qui font ce rêve ne devraient pas oublier que l'on ne peut copier une pizza sans la re-produire. Surtout, ils ne voient pas que si Proudhon ou Marx revenaient, ce ne serait pas pour s'esbaudir que l'on partage de la musique ou de la vidéo : ils s'intéresseraient probablement aux nouvelles formes d'exploitation, aux nouveaux visages de la propriété et aux nouvelles aliénations dans le monde du travail ! Proudhon s'écrierait « le *crowd sourcing*, c'est le vol ». Marx soupçonnerait derrière les *usages* des uns une pente fâcheuse qui va dans le sens de l'exploitation d'un travail non payé et d'un enrichissement insolent [8].

À tout prendre, cela nous ferait presque préférer le monde des choses ! *Ceux qui croient tout faire pour accélérer le changement en sont parfois les pires freins.* Si nous voulons vraiment *libérer* l'informatique, si nous voulons promouvoir l'informatique *libre*, faut-il vraiment promouvoir la posture exclusive de celui qui programme *gratuitement* ?

➥ 1.3 Mais au fait : comment ces objets sont-ils produits ?

Comment et pourquoi les objets numériques sont-ils produits ? C'est en se posant cette question que l'on verra l'intérêt de *distribuer* d'une certaine manière… pour la *production* elle-même. Ainsi se dénoueront les paradoxes, à condition de regarder les choses arriver, au fil du temps.

Souvent, pour présenter le logiciel libre, les conférenciers commencent par énumérer les quatre libertés [9] logicielles. Mais la plupart du temps, les auditeurs ont tendance à considérer cela comme une nouveauté. Si connaître c'est reconnaître, et s'il est vrai qu'on apprend qu'en faisant fond sur ce que l'on sait déjà, et en étant dérangé dans ce savoir même, alors je crois qu'il ne faut pas commencer par les *quatre libertés*, que les gens ne connaissent pas, pour expliquer ce qu'est le logiciel libre. Je commence toujours par parler des mathématiques. L'exemple des mathématiques constitue une meilleure approche, plus familière, qui permet de comprendre très vite, à la fois de quelles libertés il s'agit, et qui dispense d'expliquer trop longuement qu'il

8. À la mode 2.0 : « vous partagez, vous produisez, je deviens richissime ».

9. La liberté d'exécuter le programme, pour tous les usages (liberté 0) ; la liberté d'étudier le fonctionnement du programme, et de l'adapter à vos besoins (liberté 1), l'accès au code source est alors une condition requise ; la liberté de redistribuer des copies, donc d'aider votre voisin (liberté 2) ; la liberté d'améliorer le programme et de publier vos améliorations, pour en faire profiter toute la communauté (liberté 3). L'accès au code source est alors une condition requise. Dans les quatre cas, c'est l'utilisateur qui est libre.

y a une économie du logiciel libre. Comme les auditeurs ont tôt fait de reconnaître dans leur idée des mathématiques les quatre libertés, dérangeons alors au plus vite leur croyance : « Le saviez-vous : Pythagore interdisait à ses disciples de divulguer les théorèmes et leurs démonstrations ». Les mathématiques ont commencé par être propriétaires ! Cela aiguise la curiosité : comment est-on passé du *propriétaire* au *libre*. C'est-à-dire : comment passe-t-on du propriétaire au libre ? L'auditeur *se* pose la question, c'est bien plus important que de lui donner *notre* réponse !

Il ne s'agit pas d'*entrer* dans un monde ouvert, mais d'y rester !

De manière générale – et les mathématiques ne font pas exception – les *Idées* ont été d'abord cultivées entre initiés, dans le secret. Ce serait trop simple de vouloir expliquer cela seulement en fonction d'enjeux de pouvoir (l'alliance du chef et du sorcier, la nécessité pour le *pouvoir* de garder la main sur le *savoir*). Sans doute le savoir est-il utile au pouvoir, et bien sûr qu'il est utile au pouvoir que ce savoir ne soit pas divulgué. Mais c'est surtout tout bêtement pour les *produire* que les *Idées* ont d'abord eu besoin de rester secrètes. Pour ne pas devenir confuses, à une époque où l'écriture est rare. Et c'est ainsi que l'on réinvente la roue, chacun dans son coin, dans le secret, sans trop le savoir.

Chacun comprendra que c'est ensuite pour des raisons d'*efficacité de la production* que l'on *partage* et *diffuse* le savoir. On est passé d'un monde des Temples où l'on vient s'*initier* à un savoir *ésotérique,* à un monde des Universités où l'on cultive, par l'échange le plus ouvert possible, un savoir *exotérique*, qui se développe bien plus rapidement.

Ces deux manières de produire (ésotérique, exotérique) se retrouvent pour le logiciel. Il est assez amusant d'observer que cohabitent, au milieu du gué, deux modèles en crise. L'un se finance et vend encore très bien mais produit de plus en plus mal ; l'autre produit de mieux en mieux mais ne sait pas encore se financer et se vendre. Comment passe-t-on d'un monde à l'autre ? Comment retrouve-t-on, pour la production du logiciel, ce qui a prévalu pour la production des Idées, dans la civilisation du savoir ouvert ?

Essayons de retracer l'histoire de ce passage. À quel moment et pour qui devient-il *intéressant* de produire les choses autrement ?

2. Qu'est-ce qu'une communauté ?

Mais auparavant, il faut expliquer l'usage qui est fait ici du mot « communauté ». Dans le monde du Libre, on entend beaucoup parler de *la communauté* et de *communautés.*

En réalité il y a de très nombreuses communautés. Au sein de projets, autour de bouts de code, autour de traductions, autour de distributions, au sein d'associations. Les individus sont souvent impliqués dans plusieurs communautés. On parlera par exemple de *la communauté Debian*.

Si l'on comprend par *la* communauté l'agrégation de toutes les communautés, alors on se retrouve devant un problème analogue à celui que rencontra Cantor : l'ensemble de tous les ensembles est-il un ensemble ? Pour les ensembles on sait que la réponse est non [10]. Il est probable qu'il en va de même des communautés. On dit *la* communauté pour faire simple, pour faire bref, mais si l'on définit une communauté par des usages, des intérêts, des solidarités, alors *la communauté unique* serait probablement traversée par trop d'intérêts divergents ou de contradictions pour avoir la moindre consistance. À moins que l'on considère les conditions concrètes de la convergence de ces intérêts *apparemment* divergents.

La granularité des communautés réelles explique sans doute en grande partie que l'on parle tant de *la communauté*. Dans la pratique, les communautés sont petites, voire très petites (la taille moyenne d'un projet sur *sourceforge.net* est d'un peu plus de *une* personne…) et l'on se rassure en se disant que l'on appartient quand même à *la* communauté.

J'utilise ici le mot communauté en un sens restreint : celui de communauté d'*intérêts*. Une communauté d'intérêt n'est pas toujours une communauté d'usages avec des solidarités. Dans la société de consommation, la communauté des consommateurs est une communauté d'intérêt, mais vous aurez remarqué que tout est fait pour limiter la prise de conscience de son pouvoir par cette communauté. En ce sens de *communauté* d'intérêt, il y a trois communautés dans le monde du logiciel libre. Certes il y aura des communautés autour de projets transversaux, et là où des intérêts différents peuvent trouver à se satisfaire ensemble : dans la pratique il n'est pas rare que dans la *communauté* d'un projet libre on trouve des développeurs bénévoles, des développeurs payés par des entreprises, et des clients [11]. Mais il est utile de dégager ces trois *intérêts* pour comprendre ce qui se passe.

En somme il faut distinguer trois sens bien différents du mot *communauté*. Il y a (1) des communautés concrètes aux contours plus ou moins flous, faites de gens différents qui *partagent* quelque chose. C'est ce que l'on veut dire quand on parle des communautés. Elles sont plus ou moins informelles, plus

10. Pour une démonstration élégante de cela, voir cette page sur Math En ligne : L'ensemble de tous les ensembles.

11. Nicolas AURAY et Michael VICENTE, "Sociologie du logiciel libre", dans : *Autour du Libre*, 2006, URL : http://www.gral.re/IMG/pdf/Sociologie-LL-MichaelVicente.pdf.

ou moins permanentes. Comme des êtres vivants, ces communautés sont intéressantes à regarder aux membranes, aux interfaces. Il y a ensuite (2) des communautés d'intérêts entre des gens qui n'en sont pas toujours conscients et qui peuvent appartenir à plusieurs communautés concrètes mais d'une manière *particulière* [12]. Il y a enfin (3) *la communauté*, celle que l'on fantasme peut-être en s'imaginant que toutes les *autres* communautés sont comme la nôtre, ou que nous avons tous les mêmes intérêts.

L'idée de cet article est donc simple : nous observons qu'un individu peut participer à une communauté pour trois *raisons* très différentes. Autre manière de le dire : il y a trois espèces de contributeurs au monde du libre. Et il est important de clarifier les choses, afin qu'ils ne se maltraitent [13] point et qu'ils se reconnaissent.

3. Trois moments, trois communautés

Au commencement (parce que nous sommes depuis longtemps dans la civilisation du savoir ouvert), il y avait au sein des universités et des laboratoires de recherche une informatique qui s'échangeait naturellement. Pour les pionniers, l'informatique était évidemment de la même nature que les mathématiques [14]. On achetait du travail, celui d'ingénieurs en régie [15]. Et comme il faut un peu de hasard favorable dans une histoire, on se félicitera qu'une décision américaine de 1956 empêchant AT&T [16] de commercialiser autre chose que des équipements téléphoniques ou télégraphiques ait aidé AT&T à distribuer en 1973 les sources d'Unix aux universités. BSD (*Berkeley Software Development*) est né en 1977 à partir de cette souche à l'Université de Californie.

La collaboration demande des règles du jeu. En mathématiques, et plus récemment en logique formelle, le seul fait d'écrire la même chose, dans une *idéographie* commune (le signe = est assez pratique par exemple...), fait faire de grands progrès. S'il faut tout réécrire et refaire dans *sa* langue, quel intérêt ? Si l'on veut pouvoir collaborer, il faut que les outils soient *portables*. On

12. Encore que les choses soient compliquées : il arrive très souvent qu'un même individu appartienne à des communautés en tant qu'individu développeur sur son temps libre, et qu'il appartienne d'autre part à d'autres communautés dans le cadre de son activité professionnelle.

13. Leibniz, dans son *Discours de métaphysique*, aux paragraphes 19 à 21, s'emploie à montrer le rôle respectif du mécanisme et du finalisme afin qu'ils participent ensemble à la construction de la science, au lieu de s'épuiser à se *maltraiter*.

14. Pour qui s'intéresse à la chose : voir la fiche Wikipédia « Correspondance de Curry-Howard ».

15. Travailler *en régie*, c'est travailler sur un projet, parfois longtemps, chez le client.

16. AT&T ne commercialisera son Unix qu'en 1983.

parle beaucoup aujourd'hui d'*interopérabilité,* la *portabilité* fut le premier effort naturel, ce sera la norme POSIX [17], Portable Operating System Interface est publié en 1984. Un programme peut *tourner* sur des machines différentes ! Les programmeurs se sont mis d'accord sur les interfaces des programmes, sur ce que devait être un programme.

➥ 3.1 Premier moment

Mais au sein des entreprises, qui devaient répondre à une demande croissante de *support* [18], du code se refermait plus ou moins en cherchant une voie commerciale pour se *distribuer*. Dans les universités, au sein des laboratoires de recherche, on s'obstina à profiter des effets de la collaboration. On s'y préoccupe en effet d'abord de *produire*, et pas du tout de vendre. Cela ne doit pas nous surprendre : les institutions académiques, le monde des Universités, c'est l'esprit *du savoir ouvert*. L'usage du savoir (pour aller vite) ce n'est pas leur affaire : c'est la *production* du savoir et sa diffusion qui sont leur souci.

Sans entrer dans les détails, disons que les quinze dernières années du XX^{e} siècle connurent le développement d'un mouvement collaboratif *survivant* dont personne, au début, ne prévoyait les effets. « Ce qui est universel mérite d'être partagé ! » Voilà le principe qui guidait des hommes comme Vinton Cerf et Bob Kahn [19] en 1974 pour partager le protocole IP [20], ou plus tard Tim Berners-Lee en 1989 au CERN pour partager HTML [21]. Lorsque Richard Stallman lance en 1984 le mouvement GNU [22], ou lorsque Linus Torvalds, par un mail resté célèbre, appelle en 1991 à la collaboration pour développer ce qui s'appellera Linux, ce qui les préoccupe tous c'est la *production*.

Les *hackers* sont ainsi : ce qu'ils veulent c'est d'abord comprendre, améliorer, bidouiller, faire exister. La question du *modèle économique* leur est

17. Le nom a été suggéré par Richard Stallman.

18. Ou encore : de *service.*

19. Vinton G. CERF et Robert KAHN, "A Protocol for Packet Network Interconnection", dans : *IEE Transactions on Communications* 22.5 (1974), p. 637–648, URL : http://www.cs.princeton.edu/courses/archive/fall06/cos561/papers/cerf74.pdf.

20. Internet Protocol. Système de routage indéterministe qui découpe l'information à transmettre sur un réseau en *paquets* et les transmet séparément « par où ça peut ». Ce protocole répondait à une commande de l'armée américaine dans le contexte de la guerre froide (l'information devait pouvoir circuler même si une grande partie du réseau téléphonique était hors d'usage – les réseaux téléphoniques utilisent nativement un routage déterministe, pour atteindre le même correspondant on passe par les mêmes standards) ; cela allait se révéler extrêmement utile pour bâtir le réseau des réseaux : l'Internet.

21. Hypertext Markup Language est la norme que suivent les « pages Web ». Elles sont écrites « en html », il s'agit d'un langage à « balises » : pour dire qu'un mot doit être présenté par le navigateur en italiques, on écrit mot.

22. GNU est l'acronyme récursif de « Gnu's Not Unix ».

complètement indifférente.

Il faut poursuivre cette histoire pour comprendre *le libre*, car le passage à l'échelle, la généralisation à tous de cette mentalité de *hacker* est évidemment impossible : on ne peut pas dire aux développeurs du monde entier : « programmeurs de tous les pays, unissons-nous, programmons du libre en mangeant des pizzas ». Fâcheusement, certains ont le mauvais goût d'avoir un loyer à payer.

Tant mieux pour ceux qui sont payés par une institution qui encourage le partage de ce qui y est produit, grand bien fasse à ceux qui ont des loisirs et qui les consacrent entièrement à produire des objets libres. Mais tant que ceux qui vendent du logiciel [23] ou du service n'auront pas vu aussi leur intérêt à ce modèle de production, le libre restera alternatif et confidentiel.

Pourtant, souterrainement, discrètement, le libre domine déjà, par ses productions, le socle sur lequel se bâtit l'informatique. De là vient ce sentiment étrange et triomphant à contempler la domination du serveur Web Apache sur *netcraft*, et navré de voir aussi dérisoire la part des systèmes d'exploitation libres sur les postes de travail.

➥ 3.2 Deuxième moment

Pour d'autres acteurs (au sein des entreprises), dans les premières années du XXI^e^ siècle, le mouvement de portabilité dont il a été question plus haut se prolongea par la *production* de code. Les développeurs des entreprises du logiciel savaient très bien qu'ils travaillaient tous sur la même chose : pourquoi ne pas le faire ensemble ? Seul un système concurrentiel faussé par des pratiques monopolistiques pouvait l'empêcher et le freiner, condamnant tout le monde à réinventer la roue. Remarquons au passage que la *coopétition* est une bonne mesure du rôle régulateur de la concurrence. Lorsque plus personne ne collabore, cherchez la situation de monopole !

Au sein de consortiums comme la fondation Apache ou Objectweb (qui est devenu depuis OW2), des industriels partagent des projets. Leur intérêt ? Faire exister du code plus solide et avec des investissements plus réduits en R&D.

On le voit, l'intérêt de cette communauté n'est ici pas le même : il s'agit finalement de gagner plus d'argent en réduisant ses charges. De l'extérieur, on peut *ne pas croire* que des entreprises puissent ainsi *partager*, mettre en *commun*. Pourtant c'est ainsi : il y a des réelles communautés industrielles autour de l'*open source*. Sur des logiciels qui ne sont pas exactement grand

23. On devrait plutôt écrire *louent*.

public : les logiciels d'infrastructure. C'est d'ailleurs ainsi que se présente le consortium OW2 : « the open source community for infrastructure software ».

Tandis que la première communauté s'est imposée sur les couches basses, c'est donc sur le *middleware* que s'impose la seconde. L'intégration pour des clients se fait *à base d'open source,* mais convenons que ça ne change pas grand-chose au prix qu'ils paient.

L'histoire pourrait s'arrêter là, et pour beaucoup d'acteurs c'est le cas : une histoire figée dans ce conflit entre *free software* et *open source*. Au lieu de reconnaître que l'important est qu'ils ont tous un intérêt (les uns pour le *fun* et les autres pour les économies de R&D) à produire du code en mode ouvert, ils focalisent sur le fait qu'ils n'ont pas le *même* intérêt (donc ce ne peut-être la même ouverture) et l'on se prend à se demander parfois qui est son pire ennemi : celui qui fait *autre* chose, ou celui qui fait la *même* chose *autrement*. . . .

Pourtant comme il serait dommage de s'arrêter en si bon chemin ! Pas pour celui pour qui le libre est gratuit parce que quelqu'un a payé en son temps ; pas pour celui qui a réussi à collaborer avec son concurrent : non ! ce serait surtout dommage pour le client, *l'utilisateur qui paie*. Il continue de chercher (et il a raison !) son intérêt.

➥ 3.3 Troisième moment

Écoutons les membres de ces deux premières communautés (celle des amateurs et celle des industriels) parler de l'*intérêt* du client. Ici l'on évoque la pérennité, l'interopérabilité et la sécurité du code ouvert, là on met en avant les *coûts de sortie* exorbitants d'une solution propriétaire. Tout cela est très juste.

3.3.1 « Un logiciel libre est gratuit une fois qu'il a été payé »

Mais que disent les clients ? Ils vont beaucoup plus loin ! Ils s'avisent qu'ils pourraient faire une communauté, eux aussi, qu'ils pourraient même avoir un intérêt à gouverner la production, à piloter la feuille de route des logiciels qu'ils utilisent et achètent, en particulier les logiciels *métier.* À y bien réfléchir, les clients de logiciels font déjà communauté depuis longtemps, au sein de ce que l'on appelle les *clubs utilisateurs*, où ils tentent de faire pression sur l'éditeur de la solution qu'ils *partagent*.

Qu'est-ce qui distingue un club utilisateur d'une solution libre d'un club utilisateur d'une solution propriétaire ? Autre manière de poser la question :

que devrait être l'attitude d'un *éditeur* d'une solution libre vis-à-vis de ses clients ? Tant que l'intérêt des clients ne s'est pas manifesté très nettement, l'intérêt de l'éditeur est de se comporter encore un peu beaucoup comme un éditeur propriétaire. L'éditeur d'une solution libre devrait encourager ses clients à mutualiser : « Vous avez les mêmes besoins ? Nous ne développerons qu'une fois, et vous en profiterez tous ». Les clients ont très peu l'habitude d'acheter ensemble : c'est pourtant la manière la plus intelligente d'acheter ces objets numériques dont les copies sont identiques à l'original (précisons : acheter ensemble ce qui est générique et paramétrer ce qui est spécifique chez chacun).

3.3.2 « L'argent public ne doit payer qu'une fois »

Parmi les clients, ce sont les personnes publiques[24] qui vont, les premières, s'aviser de cet intérêt à mutualiser. Sans doute, au début, on vise surtout l'*utilisation* de logiciels libres. Mais l'on s'aperçoit vite (1) que pour utiliser un logiciel libre il faut qu'il *existe,* et (2) que s'il n'existe pas il faut le faire exister. Il ne suffit pas d'une délibération d'une assemblée pour que tout *passe au libre*. Dès lors que l'on prend conscience de cela, le problème urgent n'est plus seulement celui de l'*utilisation* des logiciels libres qui existent, mais devient celui du *développement* de ceux qui n'existent pas encore.

Les clients iront donc alors encore plus loin : ils décriront leurs besoins ensemble, dans des spécifications précises, et organiseront et financeront le développement et le partage de solutions libres métier. Le réel intérêt des clients, sur les applications métier est de mutualiser le plus en amont possible. C'est en quelque sorte *la revanche du club utilisateur*, qui reprend la main, qui gouverne (c'est lui qui paie !). La différence avec les progiciels propriétaires, c'est que la solution lui appartient : il en oriente la direction. Autre façon de le dire : l'*éditeur* d'une solution métier, ce devrait être la communauté de ses utilisateurs. La gouvernance de ces projets n'est pas simple à mettre en œuvre.

Une remarque au passage à propos des *appels d'offres*. Peut-être y a-t-il un intérêt (celui du client ?) à vouloir chercher et défendre une égalité dans les appels d'offres de *fournitures* qui ne semblent pas du tout faites pour le logiciel libre. Le problème essentiel est ailleurs : soit la solution existe, il est alors possible de la choisir *sans appel d'offres* puis de mettre en concurrence pour un service dessus (et tout le monde peut répondre !) ; soit

24. Personnes morales de droit public. L'État, les hôpitaux, les collectivités sont des personnes publiques. En fait, depuis 2006, la notion de *personne publique* a été remplacée dans le code des marchés publics français par les expressions « pouvoir adjudicateur » et « entité adjudicatrice ».

la solution n'existe pas et il faut mutualiser avec d'autres pour faire produire cette solution, en s'assurant que l'on en partage la titularité [25] afin de pouvoir la libérer.

Ce mouvement a commencé dans le secteur public, il est en train de naître dans le secteur privé. Il serait très utile d'étudier ce phénomène de près, mais il semble qu'il y ait une double réticence à l'examiner : d'abord cette communauté des clients n'est pas une communauté d'informaticiens, ensuite, à court terme, il n'est pas profitable aux entreprises du logiciel libre que les clients mutualisent trop vite.

Nous étions restés sur un conflit *free software* vs *open source*, voici que s'ouvre un nouveau conflit : mutualisation par l'offre vs mutualisation par la demande, prestataire vs clients. Décidément la vie de *la communauté* n'est pas un long fleuve tranquille !

4. Au milieu du gué

Il se développe, chez les clients qui partagent une solution, cette conscience de la nécessité de *faire* quelque chose (« Ne vous demandez pas ce que ce logiciel peut faire pour vous, demandez-vous ce que vous pouvez faire pour ce logiciel. »). Les clients commencent à comprendre que pour les logiciels métier il est inutile d'*attendre* que des logiciels libres *existent*. S'ils veulent qu'existent de tels logiciels, il faut les faire exister.

Se développe aussi la conscience qu'il y a encore, pour reprendre la formule indignée de Marx, « trop de travail non payé » [26]. Il n'est pas absolument certain qu'il soit *immoral* de vouloir gagner sa vie comme développeur dans l'informatique libre.

Se développe aussi une inquiétude : « le service ne remonte pas assez de code ». L'existence de code libre permet à de nombreuses entreprises de faire du service. Mais trop peu d'utilisateurs comprennent la nécessité de *remonter* du code, de participer effectivement à la communauté des projets.

Des tentatives ont été faites depuis longtemps pour essayer d'organiser le financement de la production de code et sa régulation, pour que contribution et rétribution riment davantage.

25. Lorsqu'un client fait développer une solution informatique pour ses besoins, l'*auteur* en est le prestataire. Si le client veut pouvoir le *libérer*, il faut demander auparavant à en partager la titularité. On trouvera des conseils très utiles dans le *Guide pratique d'usage des logiciels libres dans les administrations* (Adullact).

26. J'observe que je suis *de mieux en mieux* compris lorsque je dis cela, par exemple aux Rencontres Mondiales du Logiciel Libre dans le thème *communautés*.

Voilà les questions cruciales : comment drainer des financements en amont pour la production de code ? Comment rétribuer les contributions ? Comment convaincre les entreprises de remonter/reverser (upstream !) le code qu'elles produisent dans un projet qu'elles utilisent ? Il ne suffit évidemment pas de monter une fondation pour résoudre le problème, et surtout pas une fondation par projet ! Il ne s'agit pas d'un problème administratif mais d'un problème de gouvernance.

Qui paie et pourquoi ? Qui décide ? Qui est payé et pourquoi ? Ces questions cessent de devenir taboues lorsque les clients rentrent dans le jeu.

5. Les *chaudrons magiques*[27]

À la faveur de la mise en réseau de la planète depuis la fin des années 1990, le succès des méthodes agiles [28], les dynamiques de partage et leur redoutable efficacité font que la messe est dite : les méthodes de développement en mode fermé ont vécu. Et ceux qu'elles ont un temps enrichis en sont déjà à essayer des modèles de production plus efficaces. Le monde du logiciel propriétaire, s'il continue (pour combien de temps ?) à *distribuer* comme avant, se tourne vers des manières de *produire* qui viennent du logiciel libre. Indépendamment des modèles de distribution et du statut juridique du code, il est intéressant de voir que tous les développeurs convergent vers les mêmes outils de production [29].

Chez les clients, la question n'est déjà plus de savoir *s'il* faut ou non *utiliser* du libre mais *quand* il faudra opérer telle ou telle migration. Nombre d'éditeurs propriétaires rêvent depuis longtemps de *libérer* leur code et attendent une opportunité.

La convergence qui reste à effectuer, c'est celle des échanges économiques autour du code. Reste en effet à transformer nos façons d'acheter et de vendre. Ce n'est pas si simple à imaginer, car les mécanismes propres de cette économie relèvent presque entièrement des externalités positives, mais il y a fort à parier que cela se fera presque mécaniquement, par le simple *intérêt* des acteurs d'aller vers les circuits les plus courts, les plus pérennes et les plus efficaces. C'est une question qui dépasse le cadre de cet article.

27. Voir Eric S. RAYMOND, *Le chaudron magique*, 1999, URL : http://www.linux-france.org/article/these/magic-cauldron/magic-cauldron-frmonoblock.html.

28. On désigne par *méthodes agiles* des manières pragmatiques d'organiser le développement, en impliquant le client/utilisateur de façon itérative, et en adaptant en continu une maquette.

29. Il est trop tôt pour le faire (voir cet article de Robert MCMILLAN sur *Wired*), mais il faudra observer les effets de GitHub sur le développement logiciel.

Il y a fort à parier que ces lieux de convergence seront les lieux de production : ces forges si bien nommées (qui nous renvoient aux forges à coulées continues qui furent au cœur de la révolution industrielle) où se produit le logiciel libre.

Nous pouvons trouver la confirmation de cette tri-partition de la communauté du logiciel libre en observant qu'il existe trois types de forges logicielles (à condition que l'on entende par forge le lieu d'une production ouverte à la création de projets et au travail collaboratif et non seulement à des dépôts, fussent-ils de référence).

À la faveur de l'accès public à l'Internet au milieu des années 1990, Sourceforge et quantité de forges thématiques regroupent les projets qui s'échangeaient auparavant sur support physique ou poussivement par FTP sur les BBS (Bulletin Board System). Les consortiums d'entreprises firent forge à part. De même, les forges de « clients » [30].

Ce qui produit cette séparation, c'est la volonté de réunir un patrimoine (et de le séparer des autres) et le signe d'un intérêt pour une visibilité, une gestion et une destination particulière de ce patrimoine. Il suffit de regarder d'ailleurs le genre de logiciel que l'on trouve sur ces forges pour s'en convaincre : sur les forges des consortiums on trouve presque exclusivement du *middleware*, sur les forges des « clients » des logiciels métier.

Comment réussir à clarifier et *justifier* les flux économiques autour de la vie du logiciel – naissance, développement, descendance ? Cela ne pourra évidemment se faire qu'en faisant d'une façon ou d'une autre « forge commune » puisque les trois communautés sont interdépendantes et que c'est de la maturité de leurs relations que dépend la maturité du Libre. C'est cette « forge » à inventer qui sera la condition d'un modèle économique stable sur ce modèle technique de production qui devient dominant.

La question centrale de la gouvernance « qui décide ? » renvoie à la mesure des critères de décision.

Ces critères eux-mêmes renvoient à la mesure, la métrique. Il faut parvenir à mesurer la *valeur* d'une contribution (qu'elle soit du code, un retour de bug, un patch, une traduction, une documentation, etc..). Encore une fois ce n'est pas simple, car une récompense trop immédiate est parfois contre-productive [31]. Cette valeur devra surtout prendre *deux* dimensions : une valeur immédiate d'une part (peu ou prou la valeur du *travail* accompli mais mesurée par les pairs), et une valeur acquise par la réutilisation, la pérennité, la

30. Forges telles que Adullact.net (2002), Junta de Andalucia (2005), Osor.eu (2008), qui s'est fondue depuis dans le méta-projet Joinup.

31. Voir par exemple « Crowdsourcing. Les bonnes pratiques : management et motivation », sur *Haute-performance.fr*.

qualité, qui ne peut se mesurer qu'avec le temps. Il faut parvenir à construire un modèle dans lequel il devienne clair qu'en matière de numérique c'est la production même qui est un investissement, mais que la valeur se mesure à l'usage et non au *retour sur investissement*. Développer dans le domaine du numérique, c'est *s'investir*.

Le logiciel libre aura montré la voie, dans ces usines de production, de modèles permettant de produire d'autres objets que du code : des documents, de la musique, voire des films.

> Ce serait peut-être l'une des plus grandes opportunités manquées de notre époque si le logiciel libre ne libérait rien d'autre que du code. [32]

32. Le documentaire sur le libre de Hannu Puttonen (2002), intitulé *Nom de code : Linux* se clôt sur cette formule.

Références

AIGRAIN, Philippe, *Un cadre de réflexion pour comprendre l'impact du choix des licences « copyleft » telle que la GPL, par opposition aux licences « non-copyleft »*, 2002, URL : http://www.adullact.org/article.php3?idarticle=83.

AURAY, Nicolas et Michael VICENTE, "Sociologie du logiciel libre", dans : *Autour du Libre*, 2006, URL : http://www.gral.re/IMG/pdf/Sociologie-LL-MichaelVicente.pdf.

BRANDENBURGER, Adam M. et Barry J. NALEBUFF, *Co-Opetition : A Revolution Mindset That Combines Competition and Cooperation. The Game Theory Strategy That's Changing the Game of Business*, New York : Currency Doubleday, 1997.

CERF, Vinton G. et Robert KAHN, "A Protocol for Packet Network Interconnection", dans : *IEE Transactions on Communications* 22.5 (1974), p. 637–648, URL : http://www.cs.princeton.edu/courses/archive/fall06/cos561/papers/cerf74.pdf.

DE ROY, David et Aurélie VAN DER PERRE, *Aspects généraux de la mutualisation informatique dans le secteur public*, Namur : Facultés universitaires Notre-Dame de la Paix, 2007.

ELIE, François, *Économie du logiciel libre*, Paris : Eyrolles, 2009.

GOSH, Rishab et al., "Guideline for public administrations on partnering with free software developers", dans : *IDA/PGOSS* (2005), URL : http://joinup.ec.europa.eu/elibrary/case/guidelines-public-administrations-partnering-free-software-developers-2005.

GUEDJ, Denis, *Le théorème du perroquet*, Paris : Points, 2000.

LANG, Bernard, "Internet libère les logiciels", dans : *La Recherche* 328 (2000), URL : http://bat8.inria.fr/~lang/ecrits/larecherche/03280721.html.

LE BARS, Christophe, *Le modèle économique des acteurs, nouveau rapport client/fournisseur, appréciation différente des notions de pérennité. Comment s'y adapter*, 2003, URL : http://www.lebars.org/confs/modeles-2003.htm.

MACREZ, Franck, *L'exploitation numérique des livres indisponibles : que reste-t-il du droit d'auteur ?*, Paris : Dalloz, 2012.

PARMENTIER, Philippe et Stéphane VINCENT, *Les plate-formes de mutualisation*, 2007, URL : http://www.oten.fr/spip.php?article3328.

RAYMOND, Eric S., *Le chaudron magique*, 1999, URL : http://www.linux-france.org/article/these/magic-cauldron/magic-cauldron-frmonoblock.html.