EN FINIR AVEC LES IDEES RECUES :
LE LOGICIEL LIBRE EST DEVENU UN MARCHE

Charles-Antoine Schwerer,

Economiste chez Asterès

Avril 2016

ASTERÈS

producteur d'idées

Le logiciel libre offre l'accès au code d'un logiciel et la possibilité de le modifier et de le redistribuer. Initialement conçu comme une alternative au marché, le libre s'inscrit aujourd'hui dans des stratégies commerciales. En créant des barrières à l'entrée, les éditeurs ont marchandisé le logiciel libre. Pour réduire les effets de capture dont ils peuvent être victimes, les clients et acquéreurs de logiciels libres doivent considérer le coût d'utilisation du logiciel tout au long de sa vie, et non son seul coût initial.

L'économie marchande a incorporé le libre

Initialement conçu comme une alternative libertaire et non marchande, le logiciel libre a été progressivement intégré dans les stratégies commerciales d'éditeurs poursuivant un but lucratif. Le logiciel libre se définit comme un logiciel dont le code est ouvert : chacun peut l'utiliser, y accéder, le modifier et la redistribuer sans verser de droits d'auteur à son fondateur. Absence de droits d'auteur ne signifie pas absence de modèle économique et un segment gratuit peut être intégré dans une chaîne de valeur globale. Le développement de Linux repose principalement (à 75%) sur les salariés d'entreprises lucratives, notamment Red Hat, Intel et IBM[1], qui monétisent par ailleurs le développement du logiciel.

Richard Stallman, le logiciel libre et sa marchandisation

Les premiers codes informatiques étaient ouverts et s'échangeaient simplement entre chercheurs. En 1984, un universitaire d'Harvard, Richard Stallman, appelle Xerox pour disposer du code source de son imprimante et la réparer lui-même, comme à son habitude. Surprise, l'entreprise refuse. Le logiciel propriétaire est né et fait rapidement des émules : en quelques mois, les codes de tous les logiciels sur lesquels travaillaient Richard Stallman deviennent confidentiels.

Poursuivant une logique libertaire et philanthropique, le chercheur démissionne rapidement de son poste à Harvard pour développer ses propres logiciels ouverts. Quelques mois après les logiciels propriétaires, les logiciels libres sont nés. En 1991, Stallman rencontre un étudiant finlandais, Linus Thorvalds. Ils développent ensemble un système d'exploitation, Linux, qui prend son essor dans la mouvance libertaire et non marchande.

Comme souvent dans l'histoire du capitalisme, un produit gratuit et idéologique devient rapidement une source de profit et attire de nouveaux acteurs. Pour condamner cette marchandisation du logiciel libre, Richard Stallman se fend en 2007 d'un article intitulé « En quoi l'open source perd de vue l'éthique du logiciel libre ». Le chercheur y oppose la logique libre « mouvement de société » dont les raisons sont éthiques et collaboratives à la logique open source qui repose sur un calcul d'efficacité marchande et constitue une « méthodologie de développement ».

[1] Jonathan Corbet, conférence Linux.conf 2010.

L'économie marchande intègre donc le libre dans ses stratégies : un éditeur peut être rémunéré par un client pour développer un logiciel libre, ou développer gratuitement un logiciel libre pour monétiser des services annexes. Le logiciel peut servir « d'appât » pour le client, comme Gilette donnait à tous les américains majeurs un rasoir pour qu'il achète ensuite leurs lames. Le logiciel peut aussi être valorisé par d'autres services comme l'usage de Facebook ou Google est gratuit et monétisé par ailleurs. **Il convient** donc **d'analyser la complexité du logiciel libre à travers ce prisme : l'accès gratuit au code source ne signifie pas une absence de *business models*.**

Sous une même appellation cohabitent des projets philanthropiques et d'authentiques stratégies commerciales. Loin d'un passe-temps pour *geeks*, le développement de logiciels libres est un marché à part entière et pesait en 2015 en France 4,1 milliards € pour 50 000 emplois[2]. L'open source dispose d'une image non marchande et philanthropique alors que le marché l'a intégré depuis longtemps dans ses modèles d'affaires. **Dès la fin des années 90, des travaux d'économistes dissèquent ainsi les stratégies commerciales du logiciel libre et son inscription dans la sphère marchande[3].**

Les éditeurs monétisent leurs logiciels open source

L'utilisation du libre par des éditeurs professionnels et marchands s'explique par la compétitivité de ce mode de distribution et de production. Le fondateur du mouvement libre Richard Stallman précise que les acteurs de l'open source « envisagent uniquement les enjeux pratiques, en termes de performance : le logiciel non-libre est une solution sous-optimale ». La compétitivité de l'open source repose notamment sur le recours à une communauté de développeurs qui réduit le coût de développement et de test. La disponibilité du code permet en principe de mettre en concurrence les prestataires pour l'amélioration ou la maintenance des logiciels.

Pour un éditeur, la monétisation d'un logiciel libre passe par l'érection de barrières à l'entrée du marché. Les travaux de l'Université de Liège répertorient ainsi trois stratégies emblématiques[4].

- **La dépendance en capital humain : en produisant un logiciel open source volontairement complexe, l'éditeur empêche son client de se saisir lui-même du logiciel** pour en développer d'autres fonctionnalités. Les concurrents éditeurs doivent investir un temps important dans la compréhension du logiciel et le client est dépendant de l'éditeur initial pour tout développement ou actualisation. Les cas des logiciels de *e-learning* Learn'You et de *supply chain* Logistic'In sont évoqués : la complexification du code crée une dépendance de sentier et protège le marché. Le rapport à la communauté de développeur est alors limité par la complexité du logiciel. Ce modèle du « bundle logiciel »[5] génère des revenus par le développement de fonctionnalités, par l'intégration sur les postes de travail ou par la formation des utilisateurs.

- **La faible diffusion du code : le logiciel conçu par l'éditeur est alors développé en open source pour des raisons marketing mais sans faire appel à une communauté**. La société Close'First crée

[2] PAC pour le Syntec et CNLL, *Impact du logiciel libre / open source software en France 2015-2020,* 2015.
[3] D.Raymond, *The Magic Cauldron*, 1999 ou F.Hecker, *Setting Up Shop : the business of open source software*, 2000.
[4] O.Liesen, F.Pichault, J.Desmecht, *Les business models des sociétés de service actives dans le secteur Open Source*, 2009.
[5] L.Dahlander, *Appropriation and appropriability in open source software*, 2005.

ainsi des « systèmes clos » où le logiciel développé vise à être installé chez des clients historiques ayant demandé à passer à l'open source. La faible diffusion du code se retrouve dans les cas de mutualisation entre clients d'un éditeur. IdealX développe ainsi un logiciel de sécurité pour un nombre réduit de clients, réunis dans un Club[6]. Ils sont en capacité de divulguer le code mais sont incités à le conserver comme un avantage concurrentiel. Le rapport à la communauté est alors réduit au minimum pour éviter la divulgation du logiciel.

- **L'ajout de briques propriétaires sur un logiciel libre : l'éditeur utilise des logiciels conçus par d'autres sur lesquels ils développent ses propres fonctions propriétaires, adaptées au client**. Le libre sert de produit d'appel pour développer des logiciels sous licences. Les exemples de CSM'Light ou Little'Admin sont évoqués. L'éditeur peut alors utiliser la communauté de développeur et se comporter en passager clandestin. Le modèle de la « double licence »[7] permet d'imbriquer différemment libre et propriétaire : la société Troll Tech développe une bibliothèque en open source. Cependant, si un éditeur veut intégrer sa bibliothèque dans un logiciel propriétaire, il doit utiliser la version propriétaire du logiciel et payer des redevances.

Le logiciel libre est coûteux pour son utilisateur

L'édition de logiciels libres et propriétaires reposent sur des modèles économiques distincts. Le logiciel libre incite notamment l'éditeur à mettre en œuvre des stratégies de capture plus fines que le logiciel propriétaire. **La comparaison et l'arbitrage entre des offres libres et propriétaires doit ainsi prendre en compte le coût complet d'utilisation du logiciel**. Le coût de la licence (supporté uniquement dans le cas de logiciels propriétaires) représenterait ainsi en moyenne moins de 10% du coût total d'utilisation d'un logiciel[8]. En se concentrant sur le coût initial et en ignorant le coût total, le client est victime d'une asymétrie d'information face à l'éditeur de logiciel qui parvient ainsi à mettre en œuvre des stratégies de valorisation inter temporelles. Le recours à des tarifications à l'usage, tout service compris (Saas, *software as a service*) permet de réduire cette asymétrie.

Gartner et l'invention du TCO

La notion de coût total de détention ou TCO (*Total Cost of Ownership)*, a notamment été popularisée par le groupe Gartner dans les années 1990. Le TCO permettait de calculer le coût total de détention et d'utilisation d'un ordinateur personnel.

Pour un PC acheté 1 200 dollars, le TCO se situait entre 3 400 et 5 700 dollars par an. Sur 5 ans, le coût d'utilisation d'un poste se ventilait entre 13% pour le matériel, 19% pour les logiciels, 29% pour la mise en place et 39% pour le support et l'infrastructure.

La leçon de Gartner est évidente : la prise de décision ne doit pas reposer sur le coût initial de l'investissement mais bien sur le coût complet d'utilisation.

[6] L.Muselli, *Le rôle des licences dans les modèles économiques des éditeurs de logiciel open source*, 2008.
[7] M.Valimaki, *Dual licensing in open source software industry,* 2003.
[8] Benoit Chevalier, *Logiciels libres Open Source, Qu'est ce que c'est ?*, 2005.

Appliquée au logiciel, la logique du TCO doit inclure six sources de coût. Ces coûts allient souvent la prestation de spécialistes (internes ou externes) et une perte de productivité pour les utilisateurs.

- Le coût de **développement** du logiciel
- L'**installation** des logiciels sur les postes informatiques
- La **formation** des utilisateurs
- L'**adaptation** des autres applications et des données
- Le support, la **maintenance** (évolutive en fonction des mises à jour du logiciel) et les pannes
- La **durée de vie** du produit.

Ces sources de coût peuvent rapidement augmenter. Entre 2012 et 2014, le développement du logiciel Si-Paye pour le compte de l'ONP a coûté 346 millions € avant d'être abandonné[9]. En 1993, une erreur dans le développement du logiciel d'un télescope mis en orbite autour de la Terre, Hubble, a nécessité l'intervention d'un astronaute soit un coût de 500 millions $[10]. Plus simplement, la migration des postes de la municipalité de Munich vers des logiciels libres aurait coûté 23 millions € entre 2003 et 2013, sans prise en compte des pannes et de la perte de productivité. Ce montant aurait été compensé par 34 millions € d'économie de licences[11]. Afin que les pouvoirs publics arbitrent selon le coût complet des logiciels, le droit européen a par exemple introduit la notion de « cycle de vie » du produit dans les commandes publiques[12].

[9] Cour des Comptes, *La refonte du circuit de paie des agents de l'Etat, un échec coûteux,* Rapport public annuel 2015.
[10] Benoit Chevalier, *Logiciels libres Open Source, Qu'est ce que c'est ?*, 2005.
[11] CENT.
[12] Directive européenne 2014/24/UE relative à la passation des marchés publics transcrit en droit français par le décret du 25 mars 2016 qui permet d'attribuer un marché public sur la base du coût global du produit.