Méfions-nous du Fauxpen Source !

Et si le plus grand danger qui guettait désormais le logiciel libre n'était pas exogène mais endogène ?

Si, comme le pensent certains, ce n'était plus le logiciel propriétaire qui menace, mais une dilution des valeurs et de la culture du logiciel libre dans une soupe open source de… confusion et d'opacité, au grand dam de l'utilisateur qui ne serait plus alors acteur potentiel d'une communauté ?

On aurait même inventé une expression pour caractériser ces logiciels aux processus de développement privés et fermés qui, et je caricature à dessein[1], finissent par n'avoir d'open source que la licence : les « **fauxpen source** ».

Personnellement j'aime beaucoup ce néologisme. Il a été inventé par Phil Marsosudiro le 2 mai 2009 au cours d'une soirée (arrosée ?) quelque part en Caroline du Nord. Comment le sais-je ? Parce que je me suis rendu sur le site, ou plutôt la page, fauxpensource.org pardi !

Et il est repris ici par Matt Asay[2] dont je ne partage pas forcément l'optimisme (et les contradictions) mais qui évoque une problématique dont nous n'avons pas fini d'entendre parler.

Quand l'open source ne l'est pas assez

(open)

When open source isn’t (open enough)

Matt Asay – 10 novembre 2009 – CNET News (The Open Road)
(Traduction Framalang : Yonnel et Cheval boiteux)

Certains logiciels open source ne sont peut-être pas assez ouverts. Alors que « open source » fait référence à la licence sous-jacente au logiciel et à son adhésion à la définition de l’open source, on trouve de nombreux exemples de projets open source qui offrent une licence ouverte mais un processus de développement relativement fermé.

On a appelé cela « fauxpen source », ou pire encore, mais nous devons peut-être nous y habituer. Cela semble être la nouvelle norme du développement open source. Seules les fondations open source comme Eclipse, Apache Software Foundation et Mozilla semblent en mesure d’y échapper totalement.

Java est souvent cité comme exemple emblématique de « fauxpen source ». Lundi, le directeur technique de SAP, Vishal Sikka, a appelé de ses vœux un processus plus ouvert pour le développement de Java, mettant en avant que Sun exerce un contrôle trop strict sur le développement de Java. C’est un reproche qui mine la communauté Java depuis des années.

Et Java n’est pas le seul. Si Google s’attire les compliments pour ses investissements dans l’open source, certains n’hésitent pas à prétendre que Google garde une communauté Android fermée. Le même genre de plaintes a vu le jour à propos de la gestion de Chrome et de Chrome OS.

Même Red Hat, la quintessence des entreprises open source, est d’abord connue pour ce qu’elle distribue, pas pour ce qu’elle développe. Red Hat, bien sûr, travaille aux côtés d’IBM et d’autres développeurs salariés ou indépendants à l’écriture du noyau Linux, et publie scrupuleusement ses logciels sous licences open source. Mais lorsqu’il s’agit du développement

de sa distribution Red Hat Enterprise Linux, de son middleware JBoss ou d'autres technologies (par ex. KVM), bonne chance si vous voulez trouver des contributions externes significatives.

MySQL ? C'est grosso modo la même chose. L'entreprise (maintenant Sun Microsystems) fait virtuellement tous ses développements en interne, ce qui est vrai pour chaque entreprise privée open source qui me vient en tête. C'est une des raisons pour lesquelles Richard Stallman n'a pas à rougir de s'inquiéter de l'avenir de MySQL, même si c'est sa licence GPL préférée qui est utilisée.

La source est peut-être open, mais le processus pas nécessairement.

Il y a de très bonnes raisons pour que Google, Red Hat, MySQL et d'autres agissent de la sorte sur leurs efforts de développement open source. Ils sont responsables, fiscalement et légalement, devant leurs clients, et doivent être en capacité de garantir qualité et sécurité. On peut comprendre qu'ils exercent un certain contrôle pour s'assurer que les produits qu'ils distribuent protègent l'intégrité de leur marque.

Toutefois ceci témoigne d'un réel fossé entre « l'open source » en tant que licence et « l'open source » en tant que mode de développement et de distribution de logiciels complètement transparent. Le premier modèle est simple à mettre en place, le second beaucoup moins et demande une réelle volonté de la part de l'éditeur.

Les entreprises qui semblent mieux réussir sont celles qui ne comptent pas sur un retour sur investissement direct avec les logiciels libres. En d'autres termes, si vous ne vendez pas de l'open source, il est plus facile d'être ouvert. Doc Searls exprime cela de manière tout à fait pertinente quand il dit que « vous gagnez de l'argent grâce à (l'open source), pas de l'open source ».

Les exemples ne manquent pas. IBM en est un, Google également, bien que je sois d'accord avec ceux qui le critiquent car il peut assurément mieux faire. On peut également citer Facebook, Oracle et quelques autres.

À l'avenir, je pense que nous allons voir cette « fauxpen source » décliner, les entreprises séparant clairement leurs efforts dans l'open source et leurs modèles de ventes. L'open source peut offrir une plate-forme directe de gains, mais ce n'est pas le meilleur moyen pour véritablement générer de l'argent. Pas pour la plupart des entreprises en tout cas.

Notes

[1] Crédit photo : Stevoarnold (Creative Commons By)

[2] Pour un aller plus loin on pourra parcourir ce billet similaire de Philippe Scoffoni que je découvre à l'instant. Il y évoque « l'open source Canada Dry » et la tentation d'un « open source washing ». Extrait : « Faire de l'open source en mode licence est relativement facile alors que le faire en mode communautaire est une tout autre chose (…) les éditeurs qui font le choix de l'open source pour la licence cherchent donc avant tout à profiter de l'effet de mode et à monétiser ce qui apparaît aujourd'hui comme un avantage concurrentiel par rapport au modèle propriétaire ».