Foire Aux Questions (FAQ)
sur les modèles économiques du Libre

Quelles alternatives permettant la co-création existent-elles aux extrêmes 100% propriétaires et 100% libres ?

1. Une permière réflexion peut porter sur l'architecture du logiciel vendu. L'utilisateur final attend, pour une large variété d'aplications, une certaine convivialité. La valeur ajoutée provient dans ce cas de la qualité de l'interfaçage. Dès lors, il est possible de travailler en licence libre pour les soubassements et en licence propriétaire pour l'interface graphique. Apple, avec Darwin, illustre cette stratégie.

2. Dans le cas de librairies notamment, la politique de double licence permet d'associer liberté et propriété. En effet, une librairie sert à la création de logiciels. La valorisation de logiciel passant généralement par la vente de licences, un avantage produit est nécessaire. Les licences libres strictes comme la GPL ne permettent pas une différenciation technologie durable. Dès lors, l'achat d'une licence propriétaire se revèle nécessaire. C'est la stratégie adoptée par Trolltech avec Qt ou MySQL A.B. avec MySQL.

3. Le modèle semi-libre, inauguré par Netscape, permet la cohabitation de gammes de logiciels libres et non-libres ainsi que le transfert du code libre dans les logiciels non-libres. L'entreprise peut ainsi libérer un logiciel offrant un service minimum et proposer parallèlement une version propriétaires plus complète, bénéficiant ultérieurement d'apports de la version libre. Cette stratégie peut être illustrée par Star Office, dérivé propriétaire d'Open Office et intégré à des modules tiers.

4. Le code source de l'application peut être ouvert mais non libre. Diverses restrictions peuvent en effet porter sur l'exploitation de ce code comme l'interdiction d'un usage commercial. La licence communautaire (ni libre, ni Open Source) de Sun Microsystems (SCSL) permet l'ouverture et la modification du code mais s'accompagne de contraintes diverses et variées.

5. Une stratégie propriétaire peut être adoucie par diverses mesures permettant une certaine forme de co-création.

   - Le code peut être disponible à des fins de consultation uniquement, pour une meilleure compréhension de son fonctionnement. C'est ce que propose Microsoft avec son programme Shared Source : seuls quelques privilégiés, triés sur le volet, possèdent le droit d'exécution et de modification.
   - Plusieurs partenaires industriels peuvent s'associer dans le cadre d'un consortium. L'ouverture du code est dès lors possible pour les partenaires, ce qui prévient la détention d'un monopole par un éditeur indépendant. Un exemple de consortium est Symbian (système Symbian OS, ex-EPOC32).
   - Un éditeur peut fournir des trousses à outil permettant d'enrichir le logiciel. Cette stratégie est notamment recommandée par le Professeur Von Hippel du MIT, auteur d'une « théorie » sur les « utilisateurs de pointe ». Il illustre notamment son propos par l'exemple de Sony qui a créé un site Internet mettant à disposition des outils permettant aux fans de Playstation de programmer de nouveaux modules.
   - Le principe du *plug-in*, que l'on retrouve notamment dans les navigateurs Internet et les logiciels de traitement d'images, peut-être vu aussi comme un moyen d'amener des utilisateurs et des entreprises à venir co-développer le logiciel.

L'entreprise peut en outre « panacher ». Ainsi, Symbian développe son système propriétaire Symbian dans le cadre d'un consortium, tout en ayant distribué son langage OPL sous LGPL. Mais alors que les efforts d'ouverture d'une entreprise propriétaire sont généralement bien vus

par la communauté (voir par exemple Symbian), les vélléités liberticides d'entreprises libres sont par contre vivement critiquées et ressenties comme une forme de trahison (voir la polémique autour de YaST). •

Mon entreprise édite un logiciel de prévisions météorologiques[1]. Plusieurs clients et centres de recherche universitaires ont proposé leur participation active aux développements. La liberté d'un tel logiciel peut-elle conduire à un modèle économique viable ?

1. L'entreprise doit peser entre la perte de revenus de licences et les gains liés aux contributions extérieures. La nature spécialisée du logiciel imaginé dans ce cas laisse espérer un prix de vente élevé. Mais la qualité des contributeurs potentiels (à défaut de leur quantité) laisse espérer des contributions de grande valeur, par exemple sur l'amélioration des modèles mathématiques permettant la prévision.

2. L'entreprise doit analyser la provenance de ses revenus. Si ces derniers proviennent essentiellement d'une activité de prestation de services, la perte de revenus de licences sera potentiellement compensée par l'amélioration des services rendus sur base du logiciel amélioré par les pairs.

3. L'entreprise doit analyser la nature de la valeur apportée par le logiciel. Cette analyse peut aboutir à l'identification de modules à valeur ajoutée, qui constituent un facteur de différenciation face aux produits concurrents. L'ouverture peut dès lors porter sur des modules peu différenciants ou sur la création de nouveaux modules par des pairs.

4. « Libre » ne voulant pas dire « gratuit », le code source peut être acheté.. L'activité de co-développement peut aussi s'insérer dans le cadre d'une édition mutualiste rémunérée. IDX-PKI de la société IdealX illustre cette stratégie. Si le logiciel apporte un avantage compétitif à l'ensemble des clients contributeurs, la diffusion peut par ailleurs se limiter *de facto* (mais pas *de jure*) aux seuls clients contributeurs. Dans notre cas, cette stratégie comporte par contre le risque de conduire à la création de notre propre concurrence (par *fork* notamment). •

Robert.Viseur@ecocentric.be

1 Il s'agit d'un cas fictif, proche du cas soumis initialement.