

Valorisation de l'open source en tant qu'éditeur

Matthieu BALONDRAGE

5 Février 2020

Table des matières

Glossaire	3
Acronymes	4
1 Etat de l'art	5
1.1 Présentation de l'open source	5
1.1.1 L'open source aujourd'hui	5
1.1.2 Les éditeurs open source	8
1.1.3 Le consommateur	10
1.1.4 Où trouver de l'open source ?	11
1.2 Optimisation des ressources	11
1.2.1 Pareto revisité	11
1.3 Etude du consommateur	11
1.4 Le marché de l'open source	11
1.4.1 Les acteurs de l'open source	11
1.4.2 Chez les éditeurs	12
1.4.3 Chez les prestataires	13

Table des figures

Glossaire

Apache L'Apache Software Foundation (ASF) est une organisation à but non lucratif qui développe des logiciels open source sous la licence Apache, dont le renommé serveur web Apache HTTP Server. Elle a été créée en juin 1999 dans le Delaware aux États-Unis. . 6

Framework Un framework est un ensemble d'outils et de composants logiciels organisés conformément à un plan d'architecture et des patterns, l'ensemble formant ou promouvant un "squelette" de programme, un canevas. Il est souvent fourni sous la forme d'une bibliothèque logicielle et accompagné du plan de l'architecture cible du framework. Un framework est conçu en vue d'aider les programmeurs dans leur travail. L'organisation du framework vise la productivité maximale du programmeur qui va l'utiliser. . 5

Acronymes

BSD Berkeley Software Distribution. 6

CA chiffre d'affaire. 12

FSF Free Software Foundation. 5

GNU GPL GNU General Public Licence. 6

OSI Open Source Initiative. 5

SaaS Software as a Service. 7

SSII Société de Service en Ingénierie Informatique. 14

SSLL Société de Service en Logiciel Libre. 13, 14

Chapitre 1

Etat de l'art

1.1 Présentation de l'open source

1.1.1 L'open source aujourd'hui

Faire de l'open source c'est ...

Un mouvement de pensée

Le logiciel open source est une affaire de liberté. N'importe quel programme devrait être utilisable, modifiable et redistribuable. Les logiciels non libres, "propriétaire", porte atteinte à cette philosophie. Le logiciel libre n'est pas une alternative ou bien un autre business model mais une lutte pour la liberté.

Un grand mouvement est initié par Richard STALLMAN, le père fondateur de la FSF qui soulève un patrimoine extraordinaire de logiciel mis à disposition de tous. Il s'agit d'une véritable révolution pour ce mouvement de pensée profond ou la solidarité sociale et l'entraide en sont les piliers.

Un modèle de développement

L'utilisation d'un modèle de développement open source communautaire est pour Eric RAYMOND un moyen de démontrer la *supériorité* des logiciels réalisés. Plus que des valeurs éthiques, c'est par la création de ce mouvement avec la fondation de l'OSI qu'Eric RAYMOND espère imposer l'open source. Pour certain, ceci apparaît comme une *opération marketing*, pour d'autre comme Richard STALLMAN, il n'est pas permis de jeter les valeurs fondatrices notamment la *liberté*.

Une dimension humaniste et un patrimoine

L'open source permet avant tout d'offrir une chance pour les informaticiens futur de ne pas repartir de zéro, ne pas ré-inventer la roue. Chaque participation de l'Humain apporte sa pierre à l'édifice. Seulement 10% d'un code source est issue de notre création pour 90% de réutilisation de code issus de système d'exploitation, Framework, et autres composants.

C'est ici la valeur ajoutée de l'open source. L'informatique progresse essentiellement car le socle de code qui constitue notre patrimoine s'agrandit.

Respecter des droits

Je ne peux pas vous parler d'open source sans mentionner les licences et les différents droits inhérent à cette mouvement.

Les programmes open source ne sont pas des programmes « sans licences » comme on l'entend parfois. C'est au contraire leur licence qui les fait open source. Ils ne sont pas non plus dans le domaine public, c'est à dire n'appartenant à personne en particulier, ou du moins exempts de droits patrimoniaux.

Lorsqu'un développeur écrit un programme, il en détient les droits d'auteur, le « copyright ». Dans certains cas, ce peut être l'entreprise qui l'emploie qui en détient les droits. Et ce copyright peut être vendu, comme bien immatériel, d'une entreprise à une autre.

Le détenteur du copyright est libre de définir l'utilisation qui peut être faite de son programme :

- Il peut le garder pour lui, en interdire l'utilisation à qui que ce soit.
- Il peut vendre ses droits à un tiers, personne physique ou morale.
- Il peut utiliser son droit d'auteur pour préciser les conditions qu'il pose à l'utilisation de son programme. Il écrit ces conditions dans les termes de la licence d'utilisation.

Il est donc important de bien assimiler la logique suivante : à la base de l'open source il y a la licence, et la licence n'existe qu'à partir du droit d'auteur.

Ainsi tous les logiciels open source ont un propriétaire, ils ne sont pas « à personne », ni même « à tout le monde ». Dans certains cas, ce propriétaire peut être une fondation à but non lucratif, ou bien ce peut être une entreprise commerciale ordinaire. Il peut s'agir aussi de plusieurs coauteurs, en particulier à la suite de contributions successives.

Le détenteur des droits est libre de fixer les conditions de licence, il est libre d'en changer même, et il est libre d'y faire des aménagements ou exceptions, ou de diffuser à certains selon une licence, à d'autres selon une autre licence.

Celui qui reçoit le programme, en revanche, n'est pas libre. Il est lié par les termes de la licence. Certes il n'a pas signé de contrat, mais la licence lui a été bien énoncée, et elle stipule qu'il n'a le droit d'utiliser le programme que sous telles et telles conditions. S'il refuse ces conditions, il n'a pas le droit d'utiliser le programme.

Je vais détailler légèrement cette partie car elle est également une explication des différents modes de pensées et communauté qui ont un impact sur l'évolution de l'open source et mes hypothèse.

Il y a deux grandes familles de licences open source : la famille BSD et la famille GNU GPL. le terme de licence copyleft apparaît pour les premières. « Copyleft » est bien sûr un jeu de mot en référence au « copyright ». Il ne signifie pas pour autant un abandon de droits.

Les licences BSD

La licence Berkeley Software Distribution (BSD) autorise n'importe quelle utilisation du programme, de son code source et de travaux dérivés. Le code sous licence BSD peut en particulier être utilisé, intégré à des logiciels sous licence non open source. Microsoft a repris du code sous licence BSD dans Windows, et que MacOSX est basé sur FreeBSD, une distribution unix BSD.

La seule contrainte spécifique à cette licence est l'interdiction de chercher à tirer avantage de la dénomination de l'auteur, ici l'Université de Berkeley.

Moins de contrainte, plus de liberté : les programmes sous licence BSD sont quasiment dans le domaine public.

Dans la famille BSD, on trouve aussi la licence MIT, et la licence Apache. Les différences entre ces différentes licences sont de l'ordre du détail.

Les licences GNU/GPL

La licence GNU General Public Licence (GNU GPL) est utilisée par 70% des programmes open source.

La licence GNU GPL, se caractérise principalement par son article 2, qui énonce le droit de modifier le programme et de redistribuer ces modifications, qui constituent des œuvres dérivées, à la condition que ce soit sous la même licence GPL.

C'est ce que certains appellent le caractère viral de la licence : elle se communique aux travaux dérivés. Mais je parlerai ici de donnant-donnant.

Qu'est-ce exactement qu'une œuvre dérivée et qu'entend-on par distribuer ?

Vous verrez que la définition est vaste et pas toujours évidente à percevoir.

Est considéré comme une oeuvre dérivée :

- Si l'on prend un programme A, que l'on modifie des lignes de codes pour obtenir un programme B.
- Egalement, utiliser un programme A depuis un programme B, en faisant appel à certaines fonctions (on parlera de «*link*») est considéré comme oeuvre dérivée.
- Comme il existe beaucoup de façon d'appeler un programme A depuis un programme B, on considère donc que si un programme B ne peut pas fonctionner de manière utile sans A alors il est une oeuvre dérivée.

Lorsque l'on « Distribue » un logiciel open source, on livre l'ensemble du code source aux personnes concernées. Commercialiser son programme c'est le distribuer.

A l'inverse il n'est pas considéré comme distribué une oeuvre dérivée qui est utilisé au sein de son entreprise constructrice. Proposer un oeuvre dérivée à travers son service en ligne, comme les Software as a Service (SaaS), n'est pas considéré comme *distribué*. Je vous détaille cela plus précisément dans la licence AGPL

En synthétisant, l'idée de la licence GPL est que, en tant qu'auteur ou propriétaire d'un programme, je vous donne le droit de l'utiliser et d'utiliser ses sources à condition que vous en fassiez autant.

Ceci à pour effet de diviser le monde en deux « camps ». Si vous êtes du côté GPL, alors tout le patrimoine open source sous GPL vous est accessible sans restriction.

C'est donc ce que j'appellerai ici du **donnant-donnant**.

La licence AGPL (Affero)

Comme dit précédemment il est possible de prendre un programme sous licence GPL, le modifier et l'utiliser au sein de son organisation sans en livrer les sources. Rien n'empêche non plus d'offrir un service en ligne construit avec une oeuvre dérivée sans diffuser les sources.

A l'ère des services hébergés de type SaaS, ce type d'usage est régulier or il devrait être considéré comme prohibé étant donné qu'il ne respecte pas vraiment le dogme de la FSF. En effet si le programme est accessible via internet, commercialisé généralement mais que les sources ne sont pas disponibles, on devrait parler de distribution.

Pour répondre à cette controverse, une licence AGPL ou Affero GPL a été créée par la société Affero avec la FSF. Elle ajoute un article qui dit que si le programme initial permettait un accès par le réseau et diffusait ses sources par le réseau, alors le programme dérivé doit en faire de même.

L'article est le suivant :

« Si le programme tel que vous l'avez reçu est prévu pour interagir avec les utilisateurs au travers d'un réseau, et si, dans la version que vous avez reçue, un utilisateur interagissant avec le programme avait la possibilité de demander la transmission du code source intégral du programme, vous ne devez pas retirer cette possibilité pour la version modifiée du programme ou une oeuvre dérivée du programme (...) »

Droits d'auteurs

Qu'en-est-il des droits d'auteurs lors de la réalisation d'une oeuvre dérivée ?

Le droit d'auteur ou copyright sur un programme est une notion quand à elle assez claire contrairement à la notion d'oeuvre dérivée, source d'ambiguïté.

Si un programmeur écrit du code que son esprit conçoit, il n'est pas en train de violer un quelconque copyright. Lui-même, ou son employeur, est titulaire des droits d'auteur **sur son code**.

Plus simplement, on sait quand on enfreint un droit d'auteur.

Statistiques

1.1.2 Les éditeurs open source

Zoom sur l'éditeur

L'éditeur, c'est celui qui détient les droits du produit, en assure le développement, la promotion, la diffusion et le support.

La seule différence avec l'éditeur de logiciels pour l'éditeur open source est qu'il publie son produit sous licence open source. Sinon l'investissement dans le développement du produit et son marketing est le même qu'un produit propriétaire.

Ce modèle a été élu pour permettre de briser les position acquise d'oligopoles sur le marché du logiciel.

Il s'agit donc majoritairement de petites entreprises éditrice qui font du support et du développement du produit leur credos.

Développer un programme open source coute (un peu) moins cher pour ces entreprises car :

1. Elles peuvent s'appuyer sur autant de brique que la licence de son logiciel lui permette.
2. Elles bénéficient de contributions communautaire, que je détaille ultérieurement.
3. Elles possèdent généralement plus de développeur passionné participent à son travail.

Nous parlerons plus en détail du modèle économique des éditeurs dans la partie sur le marché de l'open source.

Généralement, l'éditeur fait le choix de partir sur une licence GPL car elle présente pour eux deux avantages considérables :

1. Du fait de sa popularité, elle est parfaitement lisible et compréhensible ce qui la rend gage de tranparence.
2. Elle empêche les autres de se faire de l'argent sur son dos car elle interdit l'intégration du produit dans un développement propriétaire.

L'aspect communautaire

L'éditeur open source a à sa disposition une communauté qui pourra l'aider non seulement dans le support sur les ressources open source qu'il utilise mais également au développement et au support de son oeuvre.

Au sein de la communauté il est possible de distinguer deux types d'acteurs :

les développeurs indépendants : Qu'il s'agisse de gloire, de monté en compétence sur un domaine ou d'altruisme, il existe des développeurs qui soutiennent le développement de produit et participent au support.

Les contributeurs et entreprises contributrices : Certaines entreprises favorisent l'aide et autorise leurs employés à travailler une partie de leurs temps d'activité sur des projets open sources.

Parmi ces contributeurs, on retrouve beaucoup de salariés d'entreprise IT et ce pour plusieurs raisons comme :

- Le marketing : statuer que l'on a un développeur qui travaille sur un « Grand projet » pour dorer son image et promouvoir son entreprise.
- La gouvernance : car cela permet d'avoir son mot à dire sur les orientations stratégiques d'un produit
- Le socle technique : Plus il y a de contributions à un socle de produit open source dont l'entreprise est utilisatrice, meilleur sera leur business.
- La maîtrise du produit : Monter en compétence et proposer du support sur ce produit.

Les contributions communautaires

Les éditeurs open source comptent en général assez peu sur les apports communautaires, du moins sur le coeur de leur produit. Ils les acceptent car c'est dans la logique de l'open source, mais ne les encouragent guère et l'on peut penser qu'il ne leur déplaît pas de garder la maîtrise de leur produit.

A noter que si son morceau de code est accepté, le contributeur devra généralement signer un accord spécifique qui permet à l'éditeur de disposer librement de son code. C'est assez naturel, car si chaque contributeur pouvait spécifier ses propres conditions de licence, le produit final serait un enchevêtrement de licences indémêlables.

Afin de bénéficier d'une dynamique communautaire, tout en conservant la maîtrise du noyau de leur produit, certains éditeurs mettent en place un dispositif d'extensions, qui permet d'apporter des enrichissements au produit, de manière propre et indépendante du noyau, en assurant la compatibilité avec les versions futures.

Le noyau et les extensions, un écosystème

Le modèle qui semble le plus efficace, et le meilleur compromis, est celui qui distingue le noyau du produit, sous la responsabilité de l'éditeur, et les extensions, réalisées par la communauté.

Les principes de séparation sont les suivants :

- Le noyau doit être d'une grande robustesse, il est certifié par l'éditeur, les contributions externes y sont rares.
- L'interface entre le noyau et les extensions est bien documentée et stable, c'est à dire qu'un changement de version du noyau n'implique pas, du moins le plus souvent, un changement de version des extensions.
- L'éditeur stimule la réalisation d'extensions, car elles donnent de la valeur à son produit et témoignent aussi de l'existence d'une communauté, en soi un gage de pérennité. L'éditeur offre en général une plateforme de mise à

Ce modèle noyau/extensions est celui qui réalise le meilleur point d'équilibre entre les rôles respectifs de l'éditeur et de la communauté, réunissant la garantie et l'engagement de l'éditeur avec le dynamisme et l'énorme capacité de développement de la communauté.

Les supports de l'open source

Le support dans le monde du logiciel, c'est la capacité à apporter de l'aide dans l'utilisation du programme et à corriger le programme le cas échéant.

Le support peut s'adresser aux utilisateurs finaux, comme aux exploitants du programme, ou encore aux programmeurs travaillant sur le programme.

Le déploiement de programmes pour des tâches critiques, en particulier dans des entreprises, requiert absolument un support, car le risque d'une situation de blocage est trop important, cela que ce blocage soit dû à une anomalie ou à un mauvais usage, mauvaise configuration, incompatibilité, etc.

Le support de l'éditeur

Du côté des éditeurs open source (MySQL, eZ Publish, OpenERP, ...), la question est différente : l'éditeur est une société commerciale et son business model est essentiellement basé sur son offre de support. Ici donc, le dispositif de support est très proche de celui des produits propriétaires. Pas identique toutefois car *en parallèle, en complément* au support payant de l'éditeur, il existe souvent un support communautaire, plus ou moins vivace selon les produits. Mais le plus souvent, les corrections touchant au code ne sont assurées que par l'éditeur.

Pour les nouveaux éditeurs de l'open source commercial, le support produit est le fondement du business model, il est leur raison de vivre, leur unique source de revenus. On peut donc s'attendre à un support de grande qualité.

Le support de la communauté

Les produits communautaires bénéficient avant tout d'un support communautaire. C'est à dire basé sur le volontariat de développeurs impliqués, qui répondent aux questions des utilisateurs sur les mailing-lists et forums. Et basé également sur le suivi et la prise en charge des anomalies sur les plateformes de développement communautaires.

Lorsque la communauté est active, comme c'est le cas autour des grands produits, ce support communautaire peut être d'une très grande efficacité, d'une très grande réactivité, très supérieur à un support commercial.

Nous verrons plus tard comment améliorer cette gestion de la communauté dans la partie optimisation des ressources.

1.1.3 Le consommateur

Qui est le consommateur ?

Je classe les consommateurs en deux catégories distinctes :

Les end-users : Ce sont les clients finaux du produit open source. Si le logiciel open source a pour vocation d'être utilisé comme tel par le grand public ou bien parmi les entreprises, on considère ces personnes comme des utilisateurs finaux ou « *end-users* ». Ils utilisent le logiciel, remontent leurs besoins d'amélioration, de correctifs et de support.

Les autres éditeurs et prestataires : Comme nous l'avons aperçu précédemment, une licence open source permet de bénéficier de toutes les ressources sous la même licence. Les développeurs chez les éditeurs et prestataires réalisent des agrégats de différentes solutions open sources à laquelle ils intègrent la leur. Nous le verrons plus en détail dans la partie concernant l'intégration de solution.

Les bénéfices de l'open source pour le client

Bien sûr, les bénéfices économiques sont parmi les premières raisons dans le choix de solutions open source. Même si « libre ne signifie pas gratuit », ces solutions ont toujours un coût de possession sensiblement moins élevé que leurs équivalents propriétaires.

D'autant que les prix de prestations tendent aussi à être moins élevés, car l'ouverture du produit facilite la diffusion de la connaissance.

Mais au fur et à mesure que ces solutions arrivent à maturité, le moindre coût n'est plus le premier critère de choix.

Les principaux arguments sont alors :

- **La non-dépendance**, ou moindre dépendance, par rapport à un éditeur. On sait que changer d'outil peut coûter très cher, et les éditeurs peuvent être tentés de profiter de la vache à lait que constituent ces clients devenus captifs. En anglais, on parle de vendor lock-in, le verrouillage par le fournisseur.
- **L'ouverture** est également un argument de poids. Les solutions open source sont en général plus respectueuses des standards, et plus ouvertes vers l'ajout de modules d'extension.
- **La pérennité** est un autre critère de choix fort, nous y revenons plus loin.
- **Et la qualité** finalement, car dans beaucoup de domaines les solutions open source sont réellement, objectivement, supérieures. Le très grand nombre de déploiements et donc de retours d'expérience, mais aussi leur modèle de développement et leur intégration de composants de haut niveau, permet à beaucoup de surclasser les produits propriétaires souvent vieillissants.

A quoi on peut ajouter le plaisir, pour les informaticiens, d'utiliser des programmes dont ils peuvent acquérir une totale maîtrise, sans barrière ni technique ni juridique.

1.1.4 Où trouver de l'open source ?

Pour conclure

L'open source est donc un mouvement important. Il apporte des valeurs de liberté, solidarité mais aussi des bénéfices tant pour les citoyens que pour les entreprises.

1.2 Optimisation des ressources

1.2.1 Pareto revisité

Une étude sur la participation de développeur et le code total rédigé rapporte que si l'on prend un projet où 200 programmeurs participent, seulement, 10 d'entre eux ont écrit 50% du code.

1.3 Etude du consommateur

1.4 Le marché de l'open source

1.4.1 Les acteurs de l'open source

Il existe dans l'open source 4 grands acteurs :

Les fondations : telles que Apache ou Eclipse, sont des organismes à but non lucratif qui stimulent et pilotent le développement de grands produits open source.

Les distributeurs : Redhat, Canonical (Ubuntu) ou Mandriva sont des distributeurs (très souvent éditeurs par la même occasion). Ils sélectionnent des outils et composants autour d'un noyau Linux, en assurent le packaging, la distribution et le support

Les éditeurs : Diffusent des logiciels sous licence open source, ils réalisent la promotion de leurs produits et proposent du support

Les prestataires : Ils vendent des services sur l'open source. Il peut s'agir de conseil, d'intégration, de support, de la formation, des solutions d'hébergement, etc.

Je m'intéresse particulièrement aux éditeurs et aux prestataires de l'open source qui devront mettre en place des solutions pérennisant financièrement leur activité.

Comment fonctionne le modèle économique ou « business model » de ces entreprises ?

1.4.2 Chez les éditeurs

Même s'ils bénéficient de l'open source pour réduire leur coût de ressources, celle-ci ne comble pas les besoins de financement de la partie développement en interne, hébergement, marketing. Il est donc nécessaire de trouver des sources de revenus pour les éditeurs.

Parmi celles-ci, on distingue 3 principales sources pour l'éditeur :

- Vendre des licences
- Vendre du support
- Vendre de l'intégration

Vendre des licences

Même s'il est interdit de faire payer l'utilisation d'un logiciel open source, les éditeurs ont bien compris comment faire bon usage de la « double licence ».

Double licence commerciale

Il est possible de distribuer une oeuvre dérivée utilisant le programme sans diffuser ses sources à l'aide d'une licence commerciale.

Le programme officiel open source est gratuit mais sa version dérivée elle est payante à travers l'achat de licence commerciale.

Pour l'entreprise MySQL, la vente de licence représente plus de la moitié du chiffre d'affaire (CA). Pour d'autre, ce n'est pas une source de revenu suffisante car leur finalité est de faire un *site* et non un *produit*.

Les extensions payantes

L'éditeur peut proposer des extensions aux fonctionnalités présente dans le logiciel open source. Le logiciel initial est suffisamment de qualité et donne envie de payer quelques extension *optionnelles* supplémentaire pour le confort et le besoin de l'utilisateur.

Un support uniquement sur licence commerciale

Ici aussi, le logiciel est sous licence open source mais si l'on désire avoir le moindre support dessus, il faudra se tourner vers son confrère et sa licence commerciale.

Vendre du support

La prestation de support est une source principale de revenu pour l'éditeur, même s'il doit pour cela faire face à la concurrence potentielle de prestataires tiers que je précise juste après.

Le support d'un logiciel inclut généralement :

- L'accès privilégié aux correctif et ressources spécifique
- Une prise en charge des problèmes (anomalies, utilisation, mise en oeuvre ...)
- Des prestation d'audit, de certification, ou de prise de contrôle à distance, ainsi que la surveillance proactive et les corrections.

On peut distinguer différent modèles de financement concernant la vente de support :

Uniquement le support

Certains éditeurs misent uniquement sur la vente de support. Le logiciel est gratuit mais le support lui est payant. Ce modèle de financement fonctionne pour certaines entreprises comme Tiny(OpenERP), Nuxeo.

Le problème est qu'en cas ou le support n'a pas été utile l'année souscrite pour cause de stabilité du logiciel, le client voudra surement le résilier pour la suite.

Plus le produit est de qualité, moins le support est facile à vendre car le client rencontre moins de difficultés.

L'avantage est que plus on avance dans la technologie et plus la concurrence est présente on doit donc sortir des nouveautés constamment ce qui fragilise le produit et le rend instable. Le support est donc précieux dans le cadre professionnel.

Faire payer la stabilité

Pour d'autres éditeurs, la stabilité d'un logiciel peut devenir source de profit. L'idée est de proposer deux logiciels :

- Celui en licence open source sera classifié de « community edition ». Il sera présenté comme instable, pas entièrement testé, à ne pas déployer en production
- Tandis que le logiciel sous licence non-libre sera la licence sécurisée, une version « enterprise-ready », « fully tested ».

Au final, la version community, c'est celle en cours de développement donc en avance de phase alors que la version entreprise c'est celle qui a été *gelée* dans un état dit « *stable* ».

L'éditeur peut alors diffuser et promouvoir son produit à travers la version community, et apâter les entreprises dans la version payante ou généralement le support est packagé avec.

L'éditeur jongle sur la stabilité de la version community en la rendant suffisamment stable pour donner envie de l'utiliser mais inciter fortement les entreprises à prendre la version sous licence commerciale qui sera accompagnée du support.

Fonctionnalités avancées

Pour l'éditeur, un autre business model et celui de la fonctionnalité avancée. La version community et entreprise n'est pas différente d'un point de vue stabilité mais ce sont les fonctionnalités présentes qui sont réduites sur la version community.

Ceci permet de se dégager du paradigme « libre = instable » totalement infondé.

La difficulté pour l'éditeur va être d'avoir suffisamment de fonctionnalités pour rendre la version libre intéressante mais d'avoir une forte valeur ajoutée sur les fonctionnalités dans la version payante.

Vendre de l'intégration

Pour les éditeurs qui ne sont pas mondialement connus, il est possible de trouver la rentabilité en proposant l'intégration de son produit open source. C'est un moyen de démarrer dans le milieu sans trop de risque mais qui n'est pas « Scalable ». On ne peut s'étendre à l'étranger si l'on est le concurrent direct de ses intégrateurs partenaires. On reste donc sur un marché réduit.

1.4.3 Chez les prestataires

Pour les prestataires de l'open source, le business model est basé sur la vente de prestation et d'expertise sur les produits.

La naissance des Sociétés de Service en Logiciel Libre (SSLL)

L'une des activités du prestataire est la gestion du support de solution open source. Il existe une telle diversité de composants open source qu'il est difficile pour un prestataire unique de se concentrer sur le support dudit logiciel.

Dans certains cas, la prestation est demandé par l'éditeur. Dans d'autre, c'est un prestataire tiers qui assurera du support, sans doute moins légitime et moins expert que l'éditeur lui-même mais aussi moins chère.

Ainsi est né en France le concept de SSLL, une société qui propose d'assurer le déploiement et le support de configuration multi-produits à base d'open source.

Un peu plus tard les Société de Service en Ingénierie Informatique (SSII) généralistes prennent partis en ouvrant des centre de support open source.

Intégrer des solution open sources

Le principe de ce modèle est de construire une application globale, un système d'information, en assemblant des logiciels open sources.

Il faudra donc également assurer le support de ces créations.
La valeur ajoutée ici est dans le choix des solutions à intégrer.

Il y a tellement de possibilité de solution dans l'open source qu'il est nécessaire d'étudier correctement le marché en faisant les bons choix. Ceci passe par une action de veille permanente sur les solutions disponible, afin de déceler les opportunité et produits prometteur pour avoir la création la plus solide et pérenne possible.