CULTURE DU WEB

Les outils de recensement et d'audit open source

" open source compliance tools "

Patrice **Bertrand**
Directeur général

Smile
OPEN SOURCE SOLUTIONS

www.smile.fr • +33 (0)1 41 40 11 00 • contact@smile.fr
www.smile-oss.com • blog.smile.fr • twitter: @GroupeSmile

PREAMBULE

SMILE

Smile est une **société d'ingénieurs experts** dans la mise en œuvre de **solutions open source** et l'intégration de systèmes appuyés sur l'open source. Smile est membre de l'**APRIL**, l'association pour la promotion et la défense du logiciel libre, du **PLOSS** – le réseau des entreprises du Logiciel Libre en Ile-de-France et du **CNLL – le conseil national du logiciel libre**.

Smile compte plus de 650 collaborateurs dans le monde, dont plus de 500 en France (octobre 2012), ce qui en fait *le premier intégrateur français et européen de solutions open source.*

Depuis 2000, environ, **Smile mène une action active de veille technologique** qui lui permet de découvrir les produits les plus prometteurs de l'open source, de les qualifier et de les évaluer, de manière à proposer à ses clients les produits les plus aboutis, les plus robustes et les plus pérennes. Cette démarche a donné lieu à **toute une gamme de *livres blancs*** couvrant différents domaines d'application. La gestion de contenus (2004), les portails (2005), la business intelligence (2006), la virtualisation (2007), la gestion électronique de documents (2008), les PGIs/ERPs (2008), les VPN open source (2009), les Firewall et Contrôle de flux (2009), les Middleware orientés messages (2009), l'e-commerce et les Réseaux Sociaux d'Entreprise (2010) et plus récemment, le Guide de l'open source et NoSQL (2011). Chacun de **ces ouvrages présente une sélection des meilleures solutions open source** dans le domaine considéré, leurs qualités respectives, ainsi que des retours d'expérience opérationnels.

Au fur et à mesure que des solutions open source solides gagnent de nouveaux domaines, Smile sera présent pour proposer à ses clients d'en bénéficier sans risque. Smile apparaît dans le paysage informatique français comme **le prestataire intégrateur de choix** pour **accompagner** les plus grandes entreprises dans l'adoption des meilleures solutions open source.

Ces dernières années, Smile a également étendu la gamme des services proposés. Depuis 2005, un département consulting accompagne nos clients, tant dans les phases d'avant-projet, en recherche de solutions, qu'en accompagnement de projet. Depuis 2000, Smile dispose d'un studio graphique, devenu en 2007 Smile Digital – agence interactive, proposant outre la création graphique, une expertise e-marketing, éditoriale, et interfaces riches. Smile dispose aussi d'une agence spécialisée dans la TMA (support et l'exploitation des applications) et d'un centre de formation complet, Smile Training. **Enfin, Smile est implanté à Paris, Lille, Lyon, Grenoble, Nantes, Bordeaux, Marseille et Montpellier. Et présent également en Espagne, en Suisse, au Benelux, en Ukraine et au Maroc**.

QUELQUES RÉFÉRENCES DE SMILE

SMILE est fier d'avoir contribué, au fil des années, aux plus grandes réalisations Web françaises et européennes. Vous trouverez ci-dessous quelques clients nous ayant adressé leur confiance.

Sites Internet

EMI Music, Salon de l'Agriculture, Mazars, Areva, Société Générale, Gîtes de France, Patrice Pichet, Groupama, Eco-Emballage, CFnews, CEA, Prisma Pub, Véolia, NRJ, JCDecaux, 01 Informatique, Spie, PSA, Boiron, Larousse, Dassault Systèmes, Action Contre la Faim, BNP Paribas, Air Pays de Loire, Forum des Images, IFP, BHV, ZeMedical, Gallimard, Cheval Mag, Afssaps, Beneteau, Carrefour, AG2R La Mondiale, Groupe Bayard, Association de la Prévention Routière, Secours Catholique, Canson, Veolia, Bouygues Telecom, CNIL…

Portails, Intranets et Systèmes d'Information

HEC, Bouygues Telecom, Prisma, Veolia, Arjowiggins, INA, Primagaz, Croix Rouge, Eurosport, Invivo, Faceo, Château de Versailles, Eurosport, Ipsos, VSC Technologies, Sanef, Explorimmo, Bureau Veritas, Région Centre, Dassault Systèmes, Fondation d'Auteuil, INRA, Gaz Electricité de Grenoble, Ville de Niort, Ministère de la Culture, PagesJaunes Annonces…

E-Commerce et Mobile

Krys, La Halle, Gibert Joseph, De Dietrich, Adenclassifieds, Macif, Furet du Nord, Gîtes de France, Camif Collectivité, GPdis, Projectif, ETS, Bain & Spa, Yves Rocher, Bouygues Immobilier, Nestlé, Stanhome, AVF Périmédical, CCI, Pompiers de France, Commissariat à l'Energie Atomique, Snowleader, Darjeeling, Veolia…

ERP et Décisionnel

Veolia, La Poste, Christian Louboutin, Eveha, Sun'R, Home Ciné Solutions, Pub Audit, Effia, France 24, Publicis, iCasque, Nomadvantage, Gets, Nouvelles Frontières, Anevia, Jus de Fruits de Mooréa, Espace Loggia, Bureau Veritas, Skyrock, Lafarge, Cadremploi, Meilleurmobile.com, Groupe Vinci, IEDOM (Banque de France), Carrefour, Jardiland, Trésorerie Générale du Maroc, Ville de Genève, ESCP, Sofia, Faiveley Transport, INRA, Deloitte, Yves Rocher, ETS, DGAC, Generalitat de Catalunya, Gilbert Joseph, Perouse Médical…

Gestion documentaire

Primagaz, UCFF, Apave, Géoservices, Renault F1 Team, INRIA, CIDJ, SNCD, Ecureuil Gestion, CS informatique, Serimax, Véolia Propreté, NetasQ, Corep, Packetis, Alstom Power Services, Mazars…

Infrastructure et Hébergement

Agence Nationale pour les Chèques Vacances, Pierre Audoin Consultants, Rexel, Motor Presse, OSEO, Sport24, Eco-Emballage, Institut Mutualiste Montsouris, ETS, Ionis, Osmoz, SIDEL, Atel Hotels, Cadremploi, SETRAG, Institut Français du Pétrole, Mutualité Française…

Consultez nos références, en ligne, à l'adresse : http://www.smile.fr/clients.

CE LIVRE BLANC

Les aspects juridiques de l'open source ont toujours été d'une grande importance. C'est assez naturel, puisque l'open source se définit en premier lieu par ses licences, qui énoncent les droits et devoirs de l'utilisateur d'un programme.

L'approche moderne du développement d'application consiste pour une part importante à choisir et assembler des composants existants, dont une part croissante est open source. Personne ne peut se priver des bénéfices de cette nouvelle démarche. En évitant de réinventer la roue, souvent une roue moins bonne, on peut concentrer les efforts des équipes de développement sur la valeur ajoutée la plus spécifique, et obtenir d'énormes gains de productivité, mais aussi de qualité et de performances.

Ainsi, les composants open source, petits et grands, sont devenus la principale matière première du développement d'applications. Il devient donc important de gérer leur assemblage dans le cadre d'un processus maîtrisé, pour garder la maîtrise et la connaissance de ce qui constitue son logiciel au final.

Ainsi, l'une des premières recommandations d'une politique open source d'entreprise est, en général, d'identifier les logiciels open source utilisés dans l'entreprise, ou entrant dans la composition d'un programme donné. Le recensement est donc le point de départ d'une politique open source: il s'agit de faire l'état des lieux.

Ce peut être l'occasion d'ailleurs de prendre la mesure des économies que ces logiciels ont déjà permis de faire, et évaluant leur valeur de remplacement par des logiciels propriétaires ou par du développement spécifique.

LES OUTILS DE RECENSEMENT ET D'AUDIT

Depuis quelques années des outils sont apparus sur le marché, qui se proposent d'analyser un patrimoine de logiciel, d'identifier les composants open source utilisés, et leurs conditions de licence.

En anglais, on parle parfois d'outils de *"open source compliance"*, c'est à dire de conformité, ou de respect, s'agissant des exigences des licences des composants open source. On pourrait proposer la traduction en *"Outils de conformité juridique open source"*. Des outils, donc, qui aident les entreprises à assurer le bon respect des exigences juridiques des composants open source qu'elles utilisent. Mais comme on le verra, la conformité juridique n'est que l'une des finalités. C'est pourquoi nous préférons pour cette famille d'outils la dénomination de *"Outils de recensement et d'audit open source"*.

Il existe de nombreux fournisseurs d'outils qui se disputent ce marché, citons en particulier :

- Blackduck Software (www.blackducksoftware.com)
- Protecode (www.protecode.com)
- Palamida (www.palamida.com)
- OpenLogic (www.openlogic.com)
- Antepedia, du français Antelink (www.antelink.com)

Le projet open source Fossology est un peu différent : il dispose d'agents permettant d'identifier les licences et copyrights, mais non d'un référentiel global.

Nous ne chercherons pas ici à comparer ces différents produits, et encore moins à désigner le meilleur. Notre objectif est de présenter leurs cas d'usage, leurs principes de fonctionnement, ainsi que les risques qu'ils adressent.

WWW.SMILE.FR

SOMMAIRE

open source

PRINCIPES DE FONCTIONNEMENT

PRINCIPES GÉNÉRAUX DE FONCTIONNEMENT

Les outils de recensement et d'audit open source fonctionnent en parcourant un référentiel de code, soit directement dans des répertoires de fichiers, soit au travers d'un gestionnaire de sources tel que SVN ou CVS. Pour chaque module identifié, une empreinte est calculée, et comparée à une base d'empreintes disponibles. L'empreinte est un résidu numérique court, qui caractérise le fichier source. Le référentiel ne comporte donc pas l'ensemble des fichiers, mais uniquement leurs empreintes et caractéristiques. Et bien sûr, pour chaque référence de logiciel conservée, un ensemble de métadonnées identifie à quel ensemble il appartient, de quelle version il s'agit, qui en détient le copyright, et sous quelle licence il est distribué et utilisé.

Le calcul d'empreintes numériques utilise des algorithmes tels que SHA ou encore MD5. Ces algorithmes sont construits pour que la moindre modification dans le fichier initial produise une empreinte totalement différente. Un simple espace ajouté dans un commentaire amène une empreinte différente, et donc un fichier qui n'est plus reconnu.

Ce n'est pas un problème, puisque le recensement ne vise pas à déceler une tentative de fraude de la part d'un développeur. Il a pour objectif de rendre service aux développeurs, en les aidant à identifier les composants intégrés dans leur produit. Ainsi, en tout état de cause, si un développeur veut utiliser un composant sans être détecté, il y parviendra sans mal.

Les outils de recensement peuvent travailler aussi bien sur du code source que sur du code objet, ou encore exécutable. Le principe de fonctionnement à base de calcul d'empreintes est compatible avec toutes formes de fichiers, y compris de médias. Concernant les fichiers binaires, il faut ici aussi garder à l'esprit que le simple changement de quelques options de compilation produira un binaire légèrement différent, qui ne sera donc pas reconnu par les outils. Encore une fois, il ne s'agit pas de jouer au détective à la recherche d'un coupable malicieux.

LES BASES DE CONNAISSANCE

Les outils de recensement et d'audits comportent deux parties. Une première application scanne les référentiels de code source disponibles, analyse et consolide les métadonnées relatives au copyright et aux licences. Cette application constitue et met à jour une vaste base d'information décrivant tous les composants open source connus, ou du moins le plus grand nombre possible de composants. La mise à jour de cette base doit être permanente.

Une seconde application audite une base de code particulière pour en recenser les composants, en référence à la base de connaissances, au moyen de comparaisons d'empreintes comme indiqué ci-avant.

Les référentiels de code parcourus incluent typiquement sourceforge.net, github.com, les grandes forges de l'open source, ainsi que les programmes des fondations telles que Apache ou Eclipse. Mais ils incluent aussi les gestionnaires de sources de projets uniques, qu'ils soient de nature communautaire ou d'éditeurs.

Les acteurs du recensement et de l'audit open source revendiquent entre 500 000 et 2 000 000 de composants indexés. Le seul volume n'est sans doutes pas le facteur qualité le plus important, car les composants de qualité, susceptibles d'intéresser les équipes de développement, sont en nombre plus réduit.

Par ailleurs, la difficulté dans l'élaboration de cette base de connaissance n'est pas tant dans le volume, elle est plutôt dans la qualité des métadonnées juridiques associées à chaque composant. La parfaite identification des conditions de licence et de copyright peut être difficile à automatiser, et elle est néanmoins capitale pour la qualité de l'ensemble du fonctionnement. De même, si un composant est identifié dans deux projets distincts, déterminer lequel des projets est l'original pour ce composant est un exercice complexe.

FAUX POSITIFS, FAUX NÉGATIFS

Dans cette recherche de composants réutilisés, deux écueils sont à gérer: les faux positifs, c'est à dire le signalement erroné d'un composant, et les faux négatifs, c'est à dire la non-détection d'un composant présent. Les faux positifs doivent être corrigés par un contrôle supplémentaire, c'est à dire l'examen manuel du composant pointé par l'outil, et sa requalification. Les faux négatifs, en revanche, ne donnent pas lieu à rattrapage. La littérature commerciale des différents fournisseurs évoque abondamment les risques relatifs de faux positifs et faux négatifs. De la même manière que pour les tests biologiques par exemple, un faux positif est une alerte erronée qui donne lieu à un travail supplémentaire, tandis qu'un faux négatif signifie que l'un des cas visé est passé entre les mailles du filet, et ne donnera donc pas lieu au traitement approprié.

Les traitements à base d'empreinte ne peuvent pas réellement amener de faux positifs par un simple aléa technique, qui serait dû à une collision d'empreintes, les seuls cas possibles sont ceux de qualifications erronées dans la base. Une des difficultés en effet dans la qualification de centaines de milliers de composants est de bien identifier celui qui est l'original. Les faux négatifs peuvent résulter soit d'une base incomplète, soit d'une qualification erronée de certains composants, soit de modifications mineures apportées au code.

SNIPPETS

Un simple copier-coller de quelques dizaines de lignes de code implique une œuvre dérivée, et induit donc les obligations de certaines licences, dites copyleft. Détecter un copier-coller est bien plus complexe et bien plus incertain que détecter l'égalité stricte de composants logiciels entiers. Rechercher l'insertion d'un nombre quelconque de lignes coupées au sein de l'un des

milliers voire millions de composants du référentiel est pratiquement impossible à traiter de manière exhaustive. Et plus encore si l'on imagine détecter la manœuvre d'un programmeur qui chercherait à cacher son méfait, et pourrait donc avoir modifié quelques commentaires, voire quelques noms de variables.

Tenter cette identification est donc hasardeux, et peut donner lieu à différents cas de faux positifs. Il faut avoir à l'esprit que même si, au plan juridique, le copier-coller produit une œuvre dérivée, il est dans la pratique beaucoup moins utilisé par les programmeurs, car beaucoup plus complexe à intégrer, et surtout à gérer dans la durée, particulièrement pour des traitements à forte valeur ajoutée. De nombreux outils de recensement et d'audit ne recherchent donc pas les blocs de code.

Outils et base

Certains outils de recensement sont open source, ce qui est de bon aloi s'agissant de faciliter l'adoption de logiciels open source. Toutefois, l'essentiel de la valeur ne réside pas dans le logiciel qui parcourt la base de code, mais dans le référentiel qui a été constitué, et est enrichie de manière continue. Ces bases peuvent porter sur entre 500 000 et 2 000 000 de composants open source, y compris leurs versions successives. Cette dualité logiciel / données, où un logiciel n'est d'aucune utilité sans ses données de référence, est assez courante, certains y voient une limitation des seules conditions de licences open source.

Les implications des licences

BlackDuck Software recense plus de 2000 licences open source utilisées. Ce nombre inclut toutes les variantes et modifications mineures des licences de référence. Dans la pratique, les composants couramment déployés font usage d'un nombre bien moindre de licences. Le standard SPDX, que l'on évoquera plus loin, recense environ 160 licences.

Chacune de ces licences présente ses propres exigences. Les outils de recensement et d'audit évaluent donc l'effet de la combinaison de ces composants, la compatibilité des licences utilisées, et les exigences finales résultantes pour le produit fini, qui d'une manière générale sont le cumul des exigences apportées par chaque composant.

Bill of Material

Selon un principe de traçabilité directement repris de l'industrie, les outils de recensement visent à produire la *"Bill of Material"* ou « BOM » d'un logiciel, c'est à dire un document qui identifie tous les composants entrant dans sa fabrication. De la même manière que pour un ordinateur, intégrant une carte graphique, qui elle-même comprend des circuits intégrés, les composants logiciels peuvent être issus d'une chaîne de fournisseurs comportant plusieurs étages. On peut alors soit traiter le recensement sur le produit final, soit demander à chaque fournisseur de produire la "BOM" qui accompagne son produit, et constituer ainsi la BOM globale par assemblage des BOMs des entrants.

SPDX

Pour que ces manipulations puissent être organisées et automatisées, il faut disposer d'une syntaxe commune pour exprimer les recensements de composants et les conditions de licence associées. C'est l'objet de la norme SPDX, réalisée à l'initiative de la Linux Foundation, et à laquelle tous les fournisseurs d'outils de recensement ont adhéré. Elle est encore récente, et son utilisation effective commence à peine.

Le but est de définir une notation standard pour toutes les informations de copyright et de licence, qui permette l'analyse et la manipulation de cette information par des programmes.

Un document SPDX comporte l'indication, pour chaque package de:

- Nom formel du package
- Identification de la version
- Nom de fichier
- Description
- Site de téléchargement
- Identifiant unique du package
- Licence déclarée pour le package
- License déterminée par le créateur du document SPDX
- Liste de toutes les licenses trouvées dans le package au niveau des fichiers
- Texte des mentions de copyright et dates.

Et pour chaque fichier :

- Nom et type du fichier
- Les informations de licence présentes dans le fichier
- La licence déterminée par le créateur du document SPDX
- Détenteurs du copyright, s'ils sont spécifiés.
- Projet associé, s'il existe.

CAS D'USAGE

POLITIQUE OPEN SOURCE

Les bonnes pratiques de la *politique open source*, encore appelée *gouvernance open source*, mettent toujours en avant une première étape, qui est d'identifier l'existant en matière d'open source dans l'entreprise. Et effectivement, une majorité d'entreprises n'ont pas une vision complète des logiciels déployés au sein du système d'information. L'open source entre souvent par la petite porte, simplement parce qu'un administrateur système, un chef de projet, un développeur, a jugé qu'il pouvait gagner du temps et de l'argent, en intégrant tel composant qui répond à ses besoins. Pas de bon de commande, pas d'autorisation requise.

Or méconnaître son utilisation de l'open source est dommageable, pas uniquement sous l'angle juridique. Peut-être que parmi ces composants, certains seront obsolètes, ou présenteront des failles, ou n'auront pas le niveau de support souhaité. Bref, identifier son patrimoine est toujours une bonne chose, le point de départ de sa politique open source. C'est aussi, comme on l'a dit, l'occasion de mesurer ce qui a été gagné déjà par ces composants, le plus souvent gratuits.

Notons toutefois que le recensement open source ne parcoure pas un système d'information entier, scannant chacun des serveurs pour y découvrir des composants et outils open source qui y auraient été installés. Le recensement porte sur des référentiels de code déjà clairement identifiés.

LE BESOIN DE RECENSEMENT

Pour certains acteurs, le besoin de voir clair dans son patrimoine de code est très réel. Le développement moderne repose largement sur un assemblage de composants, un programme peut en compter des milliers, et il n'est pas aisé de garder précisément la trace de l'origine de chacun d'entre eux et de ses conditions de licences. Les programmeurs ne sont pas tous bien formés sur le sujet, et une partie du logiciel peut être sous-traitée à différents fournisseurs, dont le niveau de maîtrise de la propriété et des licences peut être insuffisant.

Ainsi, avant même de parler de risques, la connaissance précise de son patrimoine logiciel et des *entrants* qu'il comporte est un but en soi. Nous avons évoqué plus haut les bénéfices de toutes natures qu'on peut en attendre: meilleure gestion des impacts de failles de sécurité, meilleure gestion des processus de sélection et d'acquisition, meilleure cohérence.

Le risque juridique supposé, sur lequel nous reviendrons plus loin, n'est pas le seul but du recensement. Différentes autres questions peuvent être posées, en relation aux composants open source déployés.

- Selon quel processus ces logiciels ont-ils été sélectionnés ?
- Y a-t-il eu une étude comparative ?
- Des logiciels différents ont-ils été déployés pour assurer une même fonction ?
- Quelles sont les modalités de support de ces logiciels ?
- Certains présentent-ils des failles de sécurité connues ?
- Quelle version est utilisée ? Y a-t-il un suivi des versions ?
- Quelle est leur pérennité ? Etc.

On voit ici que la pertinence du recensement open source va bien au delà du seul objectif de conformité juridique. Certes, l'argument juridique, et les menaces associées, est plus frappant pour les directions générales que le seul argument d'efficacité et de productivité. Mais les uns et les autres se rejoignent.

Fusions & Acquisitions, audit du patrimoine logiciel

Ces questions surviennent souvent à l'occasion des acquisitions d'entreprises dont une grande partie du patrimoine est fait de logiciels. L'acquéreur se demande naturellement d'une part à qui appartient – de manière détaillée – ce logiciel, et d'autre part ce qu'il est autorisé à en faire. Cela peut concerner un éditeur de logiciel, bien sûr, mais aussi un fabricant de produits physiques incluant du logiciel embarqué, ou bien encore une startup du web. Quelle part du logiciel appartient à l'entreprise, et pour ce qui ne lui appartient pas, dispose-t-elle des droits d'utilisation requis ? L'acquéreur sera-t-il libre de faire un autre usage de ces logiciels ? L'acquéreur court-il le risque d'être attaqué par le détenteur des droits de l'un des composants logiciels ?

Au moment d'évaluer la valeur de l'entreprise, ces questions pèsent très lourd, et l'acquéreur est intéressé à disposer d'un audit externe, basé sur des outils neutres et le plus exhaustifs possibles.

Vulnérabilités

Une fois le recensement des composants établi, y compris leurs niveaux de version, il n'est plus très difficile de consulter une des bases de vulnérabilités existantes, telles que celle du CERT, ou OSVDB, et de lister ainsi les risques de sécurité que présente un logiciel. C'est une valeur ajoutée complémentaire du recensement.

Prévention du risque juridique

La prévention du risque juridique est sans doutes le cas d'usage qui est le plus mis en avant. Entre autres parce que c'est celui qui permet le plus facilement de justifier l'investissement dans un outil de recensement et de conformité juridique open source.

Nous détaillerons au chapitre suivant quels sont exactement ces risques. En résumé, il est clair que utiliser ou distribuer un logiciel en violation des termes de licences de certains de ses composants expose l'entreprise à des conséquences sérieuses. La justice peut tout simplement interdire l'utilisation et la distribution du logiciel, obliger s'il y a lieu la distribution du code source conformément aux exigences des licences et imposer le paiement de droits d'utilisation et de pénalités

Exigences de commercialisation

Aux Etats-Unis, certains grands clients exigent des éditeurs de logiciel qu'ils fournissent la *Bill of Material*, c'est à dire le recensement exhaustif des composants, des entrants, de leur produit. Et ceci, évidemment, pour chaque nouvelle release.

Ce qui amène les éditeurs à mettre en œuvre des outils de recensement, et d'autre part à demander à leurs propres fournisseurs de produire également une BoM, typiquement un fichier au format SPDX. Pour des logiciels d'envergure limitée, une BoM peut être élaborée manuellement, mais lorsque le nombre de composants dépasse le millier, on comprend que cela devient soit impossible, soit du moins très coûteux.

Cette exigence concerne particulièrement les produits open source, ce qui est une anomalie puisqu'il est tout aussi pertinent de s'interroger sur les *entrants* d'un logiciel propriétaire, qu'il s'agisse d'*entrants* open source ou sous licence propriétaire.

Quoi qu'il en soit, la capacité à fournir une BOM pourrait devenir un passage obligé pour distribuer son logiciel outre Atlantique, et c'est donc un cas d'usage qui est amené à s'étendre.

Cette exigence est bien sûr une conséquence indirecte du risque juridique perçu, puisque le client utilisateur d'un produit pourrait également être mis en cause en cas de procès. La fourniture d'une BoM permet au client de montrer qu'il s'est assuré, du mieux qu'il pouvait, de la conformité juridique du produit qui lui était livré.

Les vraies implications juridiques de l'open source

Puisque le cas d'usage principal des outils de recensement et d'audit, du moins celui mis en avant majoritairement par leurs éditeurs, est la protection contre les risques juridiques liés à l'intégration de composants open source, il est bon de dresser une petite synthèse de ce que sont réellement les exigences, contraintes, et risques potentiels dans le déploiement de logiciels open source.

Soulignons que nous ne présentons ici qu'un rapide aperçu de la question. Pour un traitement complet et expert du sujet, nous renvoyons à l'excellent ouvrage de Benjamin Jean « Du bon usage des licences libres », qui fait référence sur le sujet.

Les outils de recensement et d'audit open source et leurs éditeurs sont parfois l'objet de critiques dans le cercle des professionnels de l'open source. On leur reproche un discours commercial qui cherche à faire peur, qui exagère parfois les risques véritables. Sur le terrain, il est possible que les discours commerciaux forcent un peu le trait, mais du moins le discours officiel, celui des sites web et des plaquettes, est très mesuré : il s'agit en premier lieu de *faciliter* l'adoption de l'open source.

Voyons donc ce qu'il en est réellement.

Rappelons que les licences s'appuient sur la propriété intellectuelle et les droits d'auteur ou copyright. Le détenteur des droits peut interdire l'utilisation de son programme, ou bien l'autoriser sous certaines conditions, les conditions constituant les termes de la licence.

Le risque, d'une manière générale, est de ne pas respecter les conditions de licences des logiciels et composants open source utilisés. Voyons donc quelles sont les conditions que peuvent imposer les logiciels open source.

La simple utilisation d'un logiciel open source

La simple utilisation d'un logiciel open source, en l'état, ne présente pas d'exigence particulière.

Un très grand nombre de logiciels open source sont utilisés en l'état, un très grand nombre d'entreprises n'envisage pas d'y apporter des modifications, ni même d'en obtenir les sources. C'est typiquement le cas d'une distribution Linux, du serveur Apache, d'une base MySql ou PostgreSQL, de PHP, Python, VLC, Firefox, et des milliers de logiciels « prêts à l'emploi », qui sont très rarement intégrés comme composants dans une activité de développement.

Peut-on dire pour autant qu'une entreprise qui utilise des logiciels open source sans les intégrer à un développement, donc sans construire d'œuvres dérivées, n'a pas à se préoccuper des questions juridiques ?

La réponse est : oui, presque.

Même dans ce cas, toutefois, il reste un possible risque juridique si celui qui vous a remis le logiciel sous une licence open source n'était en réalité pas autorisé lui-même à le faire. Soit qu'il ne détienne pas les droits qu'il affirmait détenir, soit qu'il ne respecte pas lui-même les conditions de licences des composants intégrés, par exemple en réunissant des composants dont les licences ne sont pas compatibles.

En somme, on peut être contrefacteur, ou enfreindre les termes de licence, même en étant de bonne foi.

C'est pourquoi l'utilisateur final consciencieux est tenu de s'intéresser lui aussi non pas juste aux conditions de licences qui lui ont été données, mais également aux composants intégrés et au bon respect de leurs propres conditions de licence. Pour les logiciels de grandes fondations, toutefois, on peut faire confiance aux processus de contrôle de copyright très sévères déjà mis en place.

Bref aperçu des licences libres, copyleft et non-copyleft

On distingue classiquement deux types de licences open source, les licences dites *copyleft*, et celles dites *non-copyleft*.

Les licences *non-copyleft* permettent de redistribuer des œuvres dérivées sous les termes de licences de son choix, y compris donc sous des licences propriétaires, y compris donc en ne donnant pas accès au code source dérivé. On peut dire qu'elles présentent très peu de risque juridique. Dans cette catégorie, on peut citer les licences Apache ASL, MIT ou encore BSD.

Les licences *non-copyleft* ont malgré tout quelques exigences à respecter, en particulier l'obligation d'inclure la mention du copyright, de l'auteur, et d'un *disclaimer* avec chaque distribution.

Les licences *copyleft*, principalement la GNU GPL, General Public Licence, et ses quelques variantes en particulier LGPL et AGPL, énoncent que *si vous redistribuez une œuvre dérivée*, alors vous devez le faire sous les conditions de licence identiques. Et donc, entre autres, rendre disponible le code source à ceux auxquels vous distribuez l'œuvre dérivée.

Ce sont donc ces licences *copyleft*, dont les exigences méritent une attention particulière, et pour bien le comprendre, il faut préciser ce qu'on entend par « distribuer » d'une part, par « œuvre dérivée » d'autre part.

Distribution de logiciel

En premier lieu il faut rappeler aux entreprises utilisatrices que ces exigences ne portent, classiquement, que sur le cas de *distribution d'œuvres dérivées*. Le déploiement et l'utilisation d'un logiciel, ou d'une œuvre dérivée de ce logiciel, au sein d'une organisation, quelle que soit sa taille, le nombre de serveurs ou de postes, n'est pas une distribution et n'induit donc pas d'exigence particulière. La distribution consiste à remettre le logiciel à un tiers, externe à l'organisation, que ce soit gratuitement ou contre paiement.

Il est important de bien rappeler ceci car de nombreuses entreprises qui ne redistribuent pas le logiciel open source qu'elles utilisent sont mal informées sur ce point.

Peut-on affirmer qu'une entreprise qui ne distribue pas les logiciels qu'elle développe n'aurait pas à se préoccuper des licences open source ?

Ici aussi, on peut dire : Oui, du moins presque.

Il reste quelques considérations :

1) La première, comme précédemment, est d'être assuré que les composants que vous avez obtenus sous telles licences open source pouvaient véritablement vous être distribués sous ces termes. C'est à dire que celui qui a énoncé les termes de licence disposait lui même du copyright sur le code. On en revient à la traçabilité amont.

2) La seconde relève de la temporalité. Un logiciel peut être utilisé pendant une dizaine d'années, parfois plus, et il est courant que l'utilisation d'un logiciel change pendant sa durée de vie. Une branche de l'entreprise, qui utilisait le logiciel, peut être filialisée, puis revendue. Son utilisation du logiciel relèvera alors d'une distribution. Il peut s'avérer qu'il existe un marché pour le logiciel, et l'entreprise peut modifier sa stratégie pour en bénéficier. Bref, il est difficile de présager des modalités d'utilisation d'un logiciel sur la durée.

OEUVRE DÉRIVÉE

Il faut bien expliquer également ce que constitue une œuvre dérivée. Ajouter ou modifier des lignes de codes à un logiciel induit toujours une œuvre dérivée. Invoquer les fonctions d'une librairie externe induit en général une œuvre dérivée de la librairie, mais cela dépend de la licence et des modalités techniques de l'appel.

L'invocation des services d'un logiciel au travers d'un protocole réseau, tel que HTTP ou SOAP, au contraire, n'induit pas une œuvre dérivée. Nous n'allons pas plus avant dans les détails de ce qui fait une œuvre dérivée, d'autant que dans la pratique, tous les cas de figure ne sont pas bien arrêtés. La caractéristique d'œuvre dérivée suppose une réelle originalité de l'œuvre initiale, qui sera appréciée au cas par cas.

Donc, en résumé, les points d'attention particuliers des licences open source portent (a) sur les composants sous licences *copyleft*, (b) dans le cas d'œuvres dérivées, donc dans le cas où l'entreprise a une activité de développement de logiciel utilisant des composants open source, et (c) dans le cas de redistribution, c'est à dire si l'œuvre dérivée n'est pas uniquement utilisée en interne de l'entreprise. C'est lorsque ces trois conditions sont réunies qu'il y a un réel risque juridique si les exigences citées ne sont pas respectées.

COMPATIBILITÉ DES LICENCES

Lorsqu'un logiciel est ainsi « œuvre dérivée » de plusieurs autres logiciels, il doit respecter les exigences énoncées par chacun d'eux. Il peut arriver que ces exigences soient tout simplement incompatibles. Ce peut être le cas en particulier lorsque l'un des logiciels utilisés est sous GPL : l'œuvre dérivée, si elle est distribuée, devrait l'être sous les termes de la GPL. Mais si un autre logiciel utilisé a une exigence semblable quant à la licence finale, alors il n'est pas possible de respecter l'une et l'autre. On parle alors de licences incompatibles.

Il n'y a pas d'issue à l'incompatibilité, autre que de renoncer à l'un ou l'autre des logiciels utilisés, ou encore renoncer à distribuer le logiciel résultant.

QUELLES CONSÉQUENCES ?

Dans le cas où une entreprise n'aurait pas respecté les conditions de licences des composants intégrés, quel est le risque encouru ?

Il y a très peu d'affaires à considérer en référence, très peu de jurisprudence. On peut le voir comme un soucis, car facteur d'incertitude. On peut aussi l'analyser comme traduisant un risque qui, finalement, n'est pas si grand qu'on voudrait le faire croire.

Les quelques affaires que l'on connaît, tant aux Etats-Unis qu'en France, établissent sans l'ombre d'un doute que les exigences de la licence GPL sont soutenues en justice. Il est clair qu'il faut les connaître et les appliquer.

Notons que les entités qui mènent des actions en justice pour défendre la bonne application des conditions de licence ont en général pour pratique de demander une mise en conformité avant d'engager des poursuites judiciaires. Ainsi, même si l'on ne peut présager de la continuation de cette politique, il est raisonnable de dire que, à ce jour, le risque pour un contrevenant est surtout l'obligation de revenir rapidement au strict respect des obligations des licences.

Cependant, si des dommages devaient être décidés par la justice, leur montant pourrait dépendre de l'ampleur des distributions, du bénéfice tiré par le contrevenant, du manque à gagner pour l'ayant droit, du caractère délibéré de la violation, de la bonne volonté dans le retour au respect des licences. On ne peut présager du maximum encouru.

Si les conditions de licences s'appliquent en général à tous les pays sans distinction, le cadre juridique, les usages, et la jurisprudence diffèrent, de sorte que le risque est assez variable. Dans l'ensemble, comme on le sait, il est plus élevé aux Etats-Unis, et donc les entreprises européennes qui distribuent leurs produits aux Etats-Unis seront particulièrement attentives à ces questions.

Dans certains cas, ce qui est perçu comme un risque par l'entreprise, c'est simplement l'obligation de respecter les conditions de licences, et en particulier, s'il y a lieu, de distribuer son œuvre dérivée sous la même licence *copyleft* utilisée par des composants intégrés, et donc

d'en rendre disponible le code source, et d'en permettre la libre modification. Des entreprises considéreront que ce logiciel est porteur d'un avantage concurrentiel, qui sera perdu s'il est rendu disponible à des concurrents.

Un autre risque est que le logiciel intègre des composants dont les licences ne sont pas compatibles. Dans ce cas, il n'y a pas de solution autre que de renoncer à utiliser certains d'entre eux, c'est à dire leur trouver des remplaçants. S'il s'agissait de parties structurantes du développement, le coût peut être très important.

Enfin, le pire des cas peut-être est celui où le logiciel final contient des composants qui ne sont pas open source, et dont l'entreprise n'a pas acquis les droits au travers d'un contrat avec son propriétaire. Ici, le risque est maximal, l'entreprise pourra être attaquée, et devoir payer des dommages qui seraient en proportion des revenus générés par le logiciel, voire en proportion simplement de ses capacités financières. Et les clients de l'entreprise pourraient se voir attaqués également pour avoir utilisé ledit programme sans se préoccuper suffisamment du droit.

On peut voir une grande inéquité sur ces questions entre logiciels propriétaires et logiciels open source: en cachant leur code, les premiers exposent moins les éventuelles violation de copyright et de clauses de licences. Alors que la transparence totale du code open source donne matière aux analyses poussées des auditeurs. Si vous utilisez un peu de code source de Oracle dans votre produit, les outils d'audit ne le détecteront pas, simplement parce qu'ils ne disposent pas du code dans leur base. Utiliser du code propriétaire est infiniment plus dangereux juridiquement que utiliser du code open source, mais ne sera pas un problème pour l'audit.

LA PHILOSOPHIE DE LA GPL

La licence GPL, ou les licences *copyleft* en général, semble poser bien des problèmes, mais pourquoi donc l'a-t-on imaginée ? Il est utile de rappeler, au delà du détail de ses exigences, quelle en est la philosophie.

Une philosophie de donnant-donnant, d'une part, l'interdiction de faire profiter de l'œuvre d'un autre sans transmettre la liberté qu'il a bien voulu accorder. Mais aussi l'idée que l'exigence de propagation, que les mauvais esprits ont appelé de manière assez détestable "contamination", cette exigence stimule l'élargissement, de proche en proche, de la sphère des logiciels open source. Puisque pour utiliser un logiciel reçu sous GPL je dois moi-même fournir mon œuvre dérivée - si je la distribue - sous GPL, alors le patrimoine s'étend de proche en proche. Et l'on peut imaginer un point critique, un "tipping point", où le patrimoine sous GPL est tellement incontournable qu'il n'est plus envisageable de ne pas s'y appuyer, et qu'en conséquence tout ou presque soit GPL. On n'y est pas encore. La licence GPL est effectivement une des plus utilisées, mais une énorme part du patrimoine open source est disponible sous des licences non-copyleft.

UN PEU DE JURISPRUDENCE

CISCO CONTRE FSF

Il n'y a pas des quantités de procès autour de la GPL. L'affaire la plus marquante peut-être remonte à 2003, et porte sur des routeurs Linksys, alors récemment acquis par Cisco. Le firmware des routeurs utilise du code Linux sous GPL. A la suite d'une action menée par Eben Moglen et la FSF, Cisco est obligé de publier le code source intégral du routeur, y compris des parties de code vues comme des secrets de fabrication du constructeur. Cette publication donnera naissance à une communauté ajoutant des fonctionnalités au routeur. Par la suite, des affaires opposeront la FSF à Cisco, sur d'autres produits Linksys jusqu'en 2009.

Le différend Cisco contre FSF au sujet de la GPL est souvent pris en exemple, tant par les défenseurs du Logiciel Libre, que par ceux qui veulent pointer les risques pour une entreprise de l'intégration non contrôlée de logiciels open source.

Quelles vraies leçons peut-on tirer de cette affaire emblématique ?

Tout d'abord celle-ci: avant d'être racheté, Linksys prenait des parts de marché à Cisco grâce à des produits moins coûteux, et ces produits étaient moins coûteux parce que, précisément, ils bénéficiaient d'une énorme quantité de R&D gratuite, sous la forme de code open source sous GPL. Dans le cas de Linksys, clairement, l'open source était un avantage compétitif fort. Du moins l'utilisation de l'open source, le problème étant justement que Linksys prenait les droits sans les devoirs.

En second lieu, Cisco semblait soit ne pas connaître l'utilisation de code open source dans les routeurs de la société acquise, soit du moins ignorer les obligations qui en découlaient. De ce point de vue, l'affaire amena une prise de conscience dans les grandes entreprises: d'une part les logiciels open source ne sont pas dans le domaine public, d'autre part une entreprise doit savoir de quoi sont faits ses produits, produits logiciels comme produits matériels, une forme de traçabilité du code.

Enfin, la mésaventure a peut-être été vexante pour Cisco, mais elle n'a pas causé de dommage sérieux au plan économique. Outre la publication du code, Cisco a dû prendre des engagements fermes concernant son respect des obligations de la GPL et verser une contribution à la FSF. Il faut souligner que tout cela n'a nullement découragé Cisco d'utiliser du code open source dans ses produits, puisque c'est encore le cas aujourd'hui. Et l'on peut noter aussi que Cisco fait partie des grands contributeurs au noyau Linux, ce qui laisse penser qu'elle comprend parfaitement les bénéfices économiques de la R&D mutualisée sous la forme de logiciel open source.

Le procès Jacobsen vs Katzer

Comme le procès Cisco-Linksys, les affaires sont souvent soldées par un accord, dont les termes restent confidentiels. Pour autant, il est clair que si l'entreprise accusée de violer les termes de licence open source accepte cet accord, c'est que l'issue du procès allait lui être défavorable. Mais on ne dispose pas des arguments détaillés du jugement, ni du montant des pénalités.

Plus récemment, le procès Jacobsen vs Katzer est le premier qui ait abouti à un jugement aux Etats-Unis. Le contexte est le suivant : Jacobsen développe un programme destiné au pilotage des trains miniatures. Katzer, une entreprise spécialisée dans le modélisme, utilise ce programme, et réclame des droits à Jacobsen. Jacobsen accuse Katzer de ne pas respecter les clauses de licence de son programme. Katzer se défend en affirmant que les termes de licence open source ne sont pas valables et ne peuvent lui être appliqués.

Le jugement, rendu en 2010, donne raison à Jacobsen, valide les principes de la licence open source, et condamne Kazer à payer 100 000 USD. Un des points relevés par les observateurs est que, bien qu'il n'y ait pas eu de *manque à gagner* pour Jacobsen, puisqu'il n'y avait pas d'activité lucrative, le non-respect des termes de licence causait néanmoins un dommage économique à l'auteur.

Busybox et Free

Un autre procès marquant fut celui de BusyBox, qui a le mérite d'avoir sa déclinaison en France. Comme pour Cisco, il s'agit de logiciel embarqué. BusyBox est un OS compact basé sur Linux, distribué sous GPL, dédié aux systèmes embarqués, et utilisé dans un grand nombre d'équipements, en particulier de set-top box TV.

Aux États-Unis, un procès engagé en 2007 opposait Software Freedom Law Center (SFLC) à différents contrevenants (Best Buy, un distributeur électronique, Samsung Electronics America, Westinghouse Digital, JVC America, Western Digital and Zyxel) accusés d'utiliser Busybox dans leurs produits en violation des termes de licence. Le jugement rendu donne raison au SFLC, et condamne en particulier Westinghouse à 150 000$ de dommage et intérêt ainsi qu'au retrait de tous les équipements du marché cédés à une fondation à but non lucratif.

L'affaire est considérée comme la première portant sur la licence GPL ayant abouti et donc faisant jurisprudence, et le verdict est sévère.

L'affaire est semblable à celle qui a opposé en France l'opérateur Free à la Free Software Foundation. Free utilisant une grande quantité de composants open source sous GPL dans ses Freebox, y compris le même BusyBox, a été assigné en justice fin 2008, et l'affaire instruite lentement jusqu'en 2011, une ordonnance de 2010 exigeant une traduction en français de la GPL. La position de Free, par la voix de son président Xavier Niel, a longtemps été que les Freebox n'étaient pas distribuées, elles étaient un composant terminal de l'infrastructure réseau de Free. Au final, la justice française n'aura pas eu à trancher, Free a finit par céder courant 2011, et a publié le code de tous les logiciels dérivés de GPL présents dans la Freebox.

Comme souvent, le protocole d'accord qui clos le conflit reste confidentiel, de sorte qu'on ignore, au delà du respect futur des licences libres, si d'éventuelles pénalités ont été payées par Free.

CONCLUSION

Pour les entreprises qui ont une activité de développement de logiciel significative, les outils de recensement et d'audit open source répondent à un réel besoin. Ils leur permettent de connaître et de maîtriser les entrants de leurs logiciels, du moins la partie open source de ces entrants. Cette connaissance leur permet de vérifier aisément le bon respect des exigences combinées des licences, et donc de maîtriser le risque juridique. Mais comme on l'a vu, il peut avoir d'autres finalités également.

Le cas d'usage le plus immédiat concerne les entreprises qui distribuent les produits de leur activité de développement, que ce soit sous forme logicielle, ou en embarqué. Néanmoins, nous avons souligné qu'il était difficile de présager des utilisations futures d'un logiciel, et qu'en conséquence, il pouvait être prudent de considérer que tout logiciel pourrait un jour être distribué.

Plus important que le seul déploiement de tels outils, il est essentiel que les entreprises définissent leur politique open source, c'est à dire ce qu'elles veulent faire en matière d'open source, quels critères et quels process elles mettent en place dans le choix de composants.

Les outils de recensement peuvent certainement aider à travailler de manière efficace et sûre avec des composants open source, mais ils ne sont qu'une partie du dispositif. La politique open source doit cibler également les étages amont, c'est à dire la sélection et l'acquisition de logiciels.

Cette politique open source doit impérativement comporter un volet de formation, qui permettra d'assurer que les chefs de départements, chefs de projets et développeurs ont la connaissance minimale requise en particulier en matière de licences. Cette culture générale open source permettra d'assurer qu'ils ne voient pas des risques qui n'existent pas, et qu'ils savent d'eux-mêmes faire les choix les plus pertinents en matière de composants.