

Interview Java-Champion Vincent Brabant par rapport à l'annonce de Sun de rendre Java Open Source

par [Brabant Vincent](#)

Date de publication : 23 Août 2006

Dernière mise à jour : 23 Août 2006

Suite à la publication récemment faites par Sun de sa ROADMAP concernant L'Open Sourcing de Java, nous avons demandé à notre Java Champion, Vincent Brabant, de nous accorder un interview à ce sujet. Le voici.

Que pensez de l'annonce faite par Sun de rendre Java Open Source ?
Quelle licence pense-tu que Sun choisira ?
Sun a toujours voulu éviter le Fork de Java. Comment va-t-il s'y prendre pour éviter cela ?

Cet article est également disponible au [format PDF](#) ([mirroir http](#))

Que pensez de l'annonce faite par Sun de rendre Java Open Source ?

Le plus grand problème, comme toujours, c'est que lorsque Sun parle de Java, il n'y a plus personne qui sait de quoi il parle exactement. Est-ce que lorsque Sun parle de Java, il veut dire

- la [plateforme Java](#),
- le [langage Java](#),
- la [JVM](#),
- la [JRE](#),
- le [JDK](#)

?

Est ce que cela se limite à Java SE ou également à Java EE, Java ME ? Et quid de Sun Java System, de Java DB, etc ?

Sun utilise tellement le terme Java pour quasi n'importe quoi finalement (le Sun Java System est un OS. Et il porte le nom de Java, car il permet de faire tourner des applications Java. A la bonne heure. Ce n'est pas le seul OS au monde qui permet de faire tourner des applications Java) qu'on ne sait plus trop à quoi il fait référence.

Car c'est tout de même un drôle d'annoncer que Java va bientôt être rendu Open Source, alors que Sun a annoncé officiellement, lors de JavaOne 2006, la disponibilité de Glassfish, l'implémentation de Java EE 5, sous licence CDDL, que JBoss, est disponible depuis toujours sous licence LGPL, et que la [fondation apache](#) a également le projet Geronimo, implémentation [J2EE 1.4](#) disponible en Open source.

Pour ceux qui l'ignorerait, Java EE est une surcouche au dessus de Java SE. Donc, c'est peut-être de Java SE que Sun voulait parler. Mais il existe également des implémentations Java SE qui sont sous licence open source, comme [gcj](#) (licence GPL), [kaffe](#), [harmony](#) (licence Apache), etc.

Mais ce sont bien souvent que des implémentations partielles, voire incomplètes, et très souvent en retard (laquelle de ces trois implémentations permet d'exécuter du code [J2SE 1.4](#) ?)

Dans un certain sens, on peut également affirmer que le code source de Java est disponible, et même sous licence open source. Il suffit d'aller sur la page de download: <http://java.sun.com/javase/downloads/> et de choisir de télécharger le code source sous licence Sun Community Source Licence (SCSL). Le hic, c'est que cette licence est reprise sur le site de la FSF comme étant une licence non-libre, et n'est pas non plus agréée par l'OSI.

Depuis peu, Sun permet même aux entreprises de fixer eux-mêmes les bogues de la JVM et de la déployer sur leurs machines. Mais pas de redistribuer cette JVM au public.

Pour résumer, lorsque Sun dit qu'il va rendre Java Open Source, cela signifie que Sun va rendre son implémentation Open Source. Ok, mais l'implémentation de quoi ?

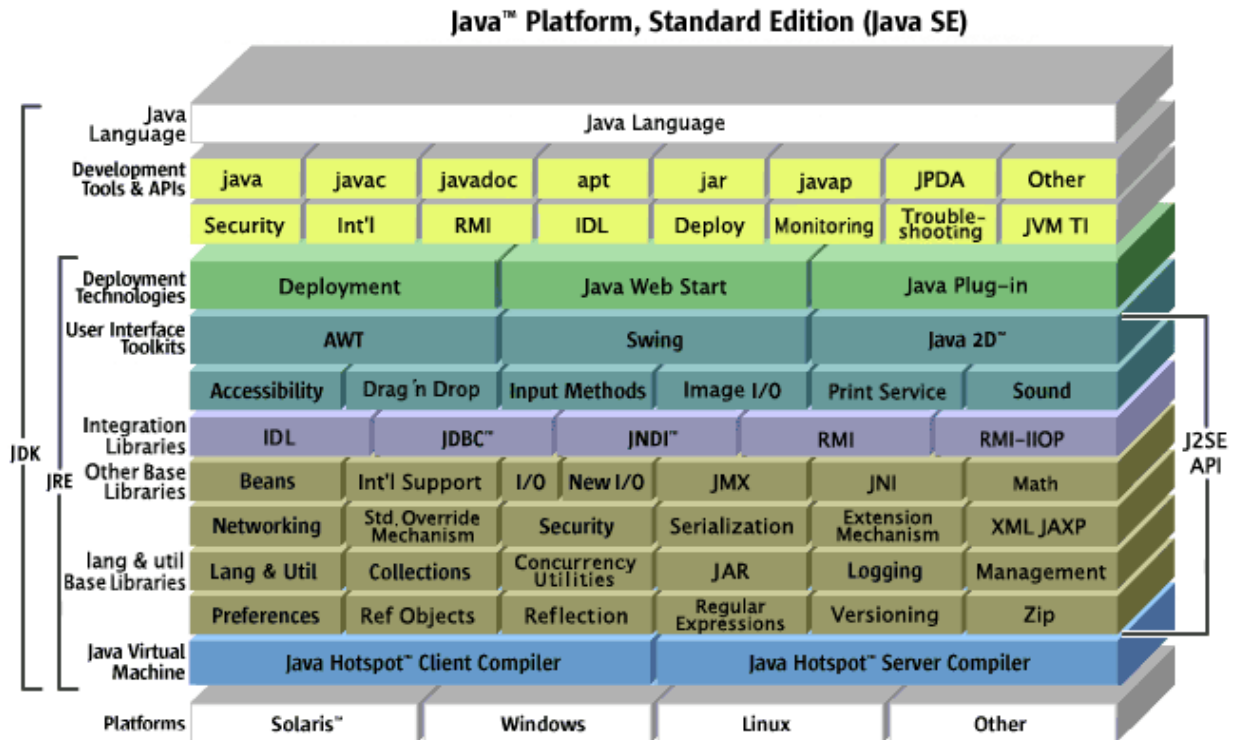
Java EE est déjà disponible sous une licence open source, donc il ne fait pas référence à cela. Reste Java SE et/ou Java ME.

Mais là, ce n'est pas encore fini.

Lorsque vous allez sur la page <http://java.sun.com/javase/#javasefamily>, vous voyez qu'il y a une distinction à faire

entre la JVM, la JRE et le JDK.

Voici un magnifique tableau, contenant tous les composants qu'on retrouve dans le JDK et la JRE:



(c) Sun Microsystems 2006

Or, il est dit que pour le mois d'octobre, Sun mettra la technologie Java Hotspot (la JVM, donc) ainsi que le compilateur javac en Open Source. Mais une JVM, sans les API (le tout formant la JRE), ne sert à rien.

Je sais bien qu'il faut commencer par quelque chose. Et que commencer par la JVM, c'est le plus logique. Mais bien que cela sera rendu open source en octobre, cela ne va rien révolutionner du tout, pour l'utilisateur, et non plus pour le programmeur d'applications Java. Il nous faudra attendre plus longtemps.

Les utilisateurs Linux, qui auraient espérer avoir la JRE (Java Runtime Engine) de Sun, en Open source avec leur distribution Linux pour la fin du mois d'octobre en seront pour leur frais. Ils vont devoir attendre plus longtemps.

Les développeurs Java devront également attendre plus longtemps. Car ce n'est pas parce que le compilateur Java sera disponible en open source que tout le JDK le sera pour fin octobre.

En fait, ceux qui devraient pouvoir tirer profit en premier lieu, ce seront les développeurs des autres implémentations open source de Java. En effet, cela devrait leur permettre de reprendre, par exemple, entièrement le compilateur Java. Mais pour cela, il faudrait encore que la licence choisie par Sun le permette. Et voir également ce que Sun va faire de la marque déposée Java. Et là, c'est une autre histoire.

Concernant le fait que Sun possède la marque Java, je voudrais vous rafraichir la mémoire avec un événement qui s'est produit il n'y a pas très longtemps finalement

On affirme toujours que JBoss est disponible sous licence LGPL: Une licence Open Source, et Libre. Le code de

JBoss devrait donc être disponible pour tout le monde. Bien que la licence LGPL soit compatible avec la licence Apache, la fondation Apache n'a pas pu reprendre le code de JBoss dans son projet, Geronimo, tel quel. Car les noms des packages contenaient JBOSS, et JBOSS est une marque déposée (TradeMark, TM). Appartenant à qui ? Même pas à la société JBoss, mais à son fondateur. Et étant opposé à l'initiative d'Apache, il a joué de tout son poids pour qu'on ne retrouve pas une seule trace du nom JBoss dans le code source du projet Geronimo.

Imaginez un seul instant que Sun fasse également jouer cela. Cela voudrait dire qu'aucune implémentation de JVM/JRE ne pourrait avoir des apis du style java.lang, java.io, java.awt, etc. On comprend bien à ce moment là que ces implémentations ne pourraient pas faire tourner un seul programme Java.

Aussi, GCJ est sous licence GPL. Et donc ne pourrait pas reprendre le compilateur de Sun si la licence utilisée par Sun ne serait pas sous une licence compatible avec GPL. Or, il est à noter qu'il n'existe pas beaucoup de licence open source qui soient compatible avec la licence GPL. cfr http://www.fsf.org/licensing/licenses/index_html#GPLCompatibleLicenses et http://www.fsf.org/licensing/licenses/index_html#GPLIncompatibleLicenses

Aussi, Sun a déclaré (cfr http://blogs.sun.com/roller/page/mr?entry=yes_we_really_are_going) que la licence qu'elle allait choisir, serait une [licence approuvée par OSI](#).

Mais attention aussi ici.

Car toutes les licences approuvées par OSI, et donc reconnue comme Open Source, ne sont pas des licences reconnues comme étant des licences libres par la Free Software Foundation. Ainsi, que se passerait-il (cas improbable) si Sun choisissait de sortir le compilateur Java sous une [licence APL \(Apple Public Licence\)](#) qui est bien reconnue par l'OSI comme étant une licence Open source, mais qui est cataloguée par la Free Software Foundation comme étant une [licence non libre](#)

On voit donc bien ici tout le problème que va causer le choix de la licence:

Certaines sont compatibles GPL, d'autres ne le sont pas.

Certaines sont compatibles Apache, D'autres ne le sont même pas.

Imaginez le choix d'une licence OSI qui ne serait compatible ni GPL/LGPL ni Apache. Personne, finalement ne serait gagnant. Mais Sun aura tenu ses promesses. Le code source sera bien disponible sous licence open source reconnue par l'OSI.

Avant de crier victoire, il va donc falloir attendre quelle sera la licence choisie par Sun.

Quelle licence pense-tu que Sun choisira ?

Franchement, je ne sais pas trop.

La licence Apache me paraît une bonne licence. Et elle a une bonne renommée auprès des développeurs Java.

Mais Sun pourrait choisir la CDDL, licence dont elle est l'auteur.

Cela pourrait faciliter l'échange de code entre divers projets comme Glassfish, NetBeans, JAXB, et son implémentation de Java SE.

Un autre option, toujours possible, serait de créer une nouvelle licence et de la soumettre à l'OSI. Une nouvelle licence qui inclurait des clauses protégeant quelque peu le forking de Java. Mais je pense plutôt que Sun a un faible pour la CDDL.

Ce qui est rassurant de voir, c'est que Sun est à l'écoute des autres parties, et de la communauté Java. Il suffit de voir le sondage qu'ils ont lancé sur java.net à ce sujet. (cfr <http://java.net/pub/pq/116>) et le forum pour recueillir les remarques.

Sun va également demander aux [Java-Champions](#) leur avis sur cette question. Mais, c'est tout de même Sun qui aura le dernier mot. Personne ne pourra les obliger à choisir une licence plutôt qu'une autre.

Sun a toujours voulu éviter le Fork de Java. Comment va-t-il s'y prendre pour éviter cela ?

Le fait que Sun a déclaré qu'il allait choisir une licence Open Source agréée par l'OSI est assez étonnant.

En effet, il est clairement mentionné dans le point 3 (cfr <http://www.opensource.org/docs/osd.pdf>) que pour être agréée par l'OSI, une licence doit permettre les modifications et le travail dérivé (ce que j'appelle le fameux forking). Mais il y a également le point 4 de ce même document qui dit clairement que l'auteur du code source peut exiger que le travail dérivé (le fork) porte un nom différent.

Ainsi, pas question, pour IBM, par exemple, de sortir sa version de JRE contenant l'API SWT, tout en continuant à l'appeler Java. Ça rappelle un peu ce qui s'était passé à l'époque avec Microsoft, qui a finalement sorti son J#.

Sun espère donc clairement que sa marque déposée, Java, soit assez forte que pour les gens ignorent toute autre implémentation qui ne s'appellerait pas Java. Une autre solution, assez simple pour Sun, serait de publier sur un site le résultat du TCK pour chacune des implémentations.

Et donc, on verrait de suite si l'implémentation est vraiment bien compatible. Mais pour cela, il faudrait que Sun garde le contrôle sur le TCK. Ou le donne à un organisme qui devrait superviser cela.

Et pourquoi pas créer une Java Foundation, tout comme IBM avait créé une Eclipse Foundation, qui superviserait le JCP, et le TCK.

Et puis, peut-être que Sun a compris que la communauté Java était assez mature que pour ne pas créer de Fork. Mais j'espère qu'IBM a également compris cela.

Aussi, si Sun désire éviter le forking de Java, il devrait donner des droits d'accès suffisant pour être sûr que la communauté Java puisse fixer / améliorer le code source.

Imaginez que Sun rende Java Open Source, mais que seulement les employés de Sun aient accès au repository CVS ou SubVersion. On peut être sûr qu'on verra alors surgir un fork de l'implémentation Java.

Pour conclure, je dirais que le risque de fork de Java ou pas va également dépendre de la façon dont Sun va gérer les accès en écriture au repository du code source.

Merci beaucoup, Vincent, de nous avoir fait partager ton point de vue.