Introduction

Avant de commencer, il est primordial de définir le concept d'***Open Source*** et expliquer pourquoi il ne faut pas le confondre avec le **logiciel libre**, ou, encore pire, avec les **logiciels gratuits**.

Qu'est-ce que l'Open Source ?

Le terme « *Open Source* », ou « *code source ouvert* » en français, renvoi à l'origine au fait de **rendre accessible aux utilisateurs le code source du logiciel** que l'on distribue. Si cette idée ne parle pas à la plupart des utilisateurs d'aujourd'hui, cette pratique est à **toujours existé en informatique**. Dans les années 60, les ordinateurs étaient livré avec des logiciels dédiés et destinés, la plupart du temps, à des utilisateurs avisés qui n'acceptaientt pas forcément dse faire tourner des logiciels dont il ne pouvait **vérifier et comprendre le fonctionnement en détails**. Ainsi, les éditeurs fournissaient les codes sources gratuitement sur demande des utilisateurs, qui pouvait alors **explorer et modifier le code** selon leurs besoins.

Plus formellement, on parle de **logiciel *Open Source*** lorsque la licence du dit-logiciel respecte l'***Open Source Definition*** établit par l'***Open Source Initiative***, à partir des ***Debian Free Software Guidelines*** (*DFSG*) considérées elles comme appartenant au **logiciel libre**. Cette définition inclut **dix principes** :

1. **Redistribution libre :** *N'importe qui peut redistribuer, gratuitement ou non, le logiciel en tant que composant de son propre logiciel. L'auteur du logiciel redistribué ne peut exiger de redevance.*
2. **Garantie d'accès au code source :** *Une distribution du logiciel doit toujours inclure le code source, ou bien indiquer comment se procurer le code source. Celui-ci doit être accessible gratuitement et ne doit pas être obscurci (rendu compliqué volontairement) de quelques manières.*
3. **Oeuvres dérivées :** *N'importe qui doit pouvoir créer une version dérivée du code source et le distribuer sous une licence respectant les termes de la licence du logiciel original.*
4. **Clause d'intégrité du code :** *La liberté de distribution de version dérivée peut-être restreinte uniquement s'il est possible de distribuer des versions correctives du code source.*
5. **Non-discrimination contre des personnes ou des groupes :** *La licence ne doit pas discriminer des personnes ou des groupes de personnes (**ex :** interdire l'utilisation de son logiciel à Google).*
6. **Non-discrimination à l'égard des champs d'applications :** *L'utilisation du logiciel ne peut pas être limité à certains champs d'applications (**ex :** bannir l'utilisation à des fins militaires).*
7. **Distribution de la licence :** *Les droits attachés au logiciel doivent*

s'appliquer à tous ceux à qui il est redistribué, sans besoin de souscrire à une autre licence.

8. **Licence non-spécifique :** *Les droits attachés au logiciel ne peuvent pas dépendre de sa distribution (**ex :** si on distribue un logiciel dans un package, les droits attachés au logiciel doivent être le même que lorsqu'il est distribué hors du package).*
9. **Licence non-restrictive :** *Le logiciel ne peut pas "imposer ses droits" aux autres logiciels d'une distribution (**ex :** imposer que tous les logiciels de la distribution soient Open Source).*
10. **Neutralité technologique :** *La licence ne doit pas discriminer quelconque technologie ou style d'interface.*

Ces principes se posent en **opposition aux logiciels propriétaires**, ou autrement dit à **code source fermé**. La compagnie ***Red Hat***, l'une des plus grandes entreprises dédiées aux logiciels *Open Source*, les définis comme suit :

> Un logiciel fermé est **strictement protégé**. Seuls les propriétaires du code source sont en droit d'accéder à ce code. Un code source fermé **ne peut pas être modifié ni copié** de manière légale, et l'utilisateur paie uniquement pour pouvoir utiliser le logiciel conformément à l'usage prévu par le propriétaire. Il ne peut ni le modifier pour d'autres usages, ni le partager avec ses communautés.

Cette définition regroupe par exemple les produits ***Microsoft*** (*Windows, Office*) ou ***Apple*** (*iOS, macOS*). À noter qu'**un logiciel propriétaire n'est pas forcément payant** (***ex :*** *Facebook, Google Search*), tout comme **un logiciel *Open Source* n'est pas forcément gratuit**.

Code machine VS Code source

Si vous n'avez jamais fait de développement informatique, vous vous demandez peut-être **qu'est-ce que le code source** ? Et quelle différence y a-t-il entre le logiciel que vous utilisez et son code source ? Voici quelques explications :

Le code machine Lorsque que vous installez un logiciel (*sur ordinateur, smartphone ou même télévision*), vous téléchargez sur votre machine une version du code écrite dans un **langage adapté à votre machine**. Celui-ci sera lu par votre processeur et **peut varier selon le modèle** de ce dernier. Pour maximiser les performances, ce code est **écrit en binaire** (*des 0 et des 1*) et est donc difficilement compréhensible et manipulable par un humain.

> Pour faire une comparaison, si le logiciel était une voiture, le code machine équivaudrait à **toutes les pièces et au clé du véhicule** : vous pouvez démarrer, mais, en cas d'incident, pas sûr que vous puissiez soulever le capot puis comprendre où est le problème.

Le code source Vous imaginez bien que les développeurs de logiciels ne vont pas s'amuser à écrire une version de leurs logiciels pour chacun des processeurs

existants, c'est pourquoi les informaticiens ont rapidement inventé des **langages de programmation** adaptés aux humains. Il en existe des milliers, possédant chacun leurs spécificités et répondant à différents besoins, certains pouvant être **traduit directement en code machine**. Ces langages sont donc beaucoup plus compréhensible et manipulable, bien qu'il nécessite tout de même une maîtrise évidente de la programmation.

> Pour reprendre l'analogie de la voiture, avoir accès au code source d'un logiciel, c'est comme avoir accès aux **plans de fabrication** du véhicule : certes, pour la plupart des automobilistes, l'usage serait limité, mais pour les plus bricoleurs cela permettrait d'entretenir, réparer, ou améliorer le véhicule librement, et même d'en faire une activité professionnelle.

La documentation Il est important de préciser que la simple consultation du code source d'un logiciel plus ou moins complexe, même pour les développeurs les plus expérimentés, ne permet pas forcément de tout comprendre au fonctionnement du dit-logiciel. C'est pourquoi le code source est souvent accompagnés d'une **documentation à destination des développeurs** expliquant les fonctionnalités des différentes parties du code (***ex:*** *pour ouvrir un fichier, utiliser telle fonction*), à ne pas confondre avec les **documentations à destination des utilisateurs** qui détaillent les fonctionnalités dans le logiciel (***ex:*** *pour ouvrir un fichier, aller dans tel menu*).

> Toujours pour reprendre l'exemple de la voiture, on parlerait ici des **notices** du véhicules, qui peuvent se révéler plus utile que les plans de fabrication pour la plupart des utilisateurs.

L'ingénierie inversée Il me paraît important de préciser que, bien qu'il soit destiné aux machines, le code machine peut-être transformer en texte via des **méthodes d'ingénierie inversée**, dont certains développeurs ont fait leur spécialité. Ces techniques sont notamment très utilisées en **cyber-sécurité** pour comprendre les modes d'actions des pirates. Cependant, elles demandent des connaissances poussées, notamment en algorithmie, et plusieurs techniques existes pour rendre ces investigations plus compliquée. De plus, si le code source n'est pas fourni, ces pratiques peuvent se révéler illégales au regard de la propriété intellectuelles. À noter que la rétro-ingénierie s'étend aussi à l'électronique et à l'industrie en générale.

> Pour finir sur la voiture, il est évident que, si vous souhaitez comprendre comment fonctionne votre véhicule, vous pouvez le **démonter pièce par pièce**. Cependant, on peut attendre des constructeurs qu'ils vous fournissent des outils de compréhensions (*documentation, plans techniques*) pour éviter d'en arriver à ces extrémités. C'est là que l'***Open Source*** entre en jeu !

D'accord, mais pourquoi faire ?

L'*Open Source* est souvent considéré comme « *un problème de geek* » et, de fait, cette problématique à du mal à s'étendre au grand public. Cependant, s'y intéresser, à titre personnel ou professionnel, peut présenter plusieurs avantages, même si certains dangers persistent de part la nature même de l'*Open Source*.

Avantages L'*Open Source* présente de nombreux atouts reconnus vis-à-vis des logiciels propriétaires. Au-delà des **coûts d'utilisation inférieurs**, puisque le code est gratuit, les logiciels respectant ces principes offrent plusieurs garanties, notamment en termes de **qualité**, d'**indépendance** et de **sécurité**.

Qualité D'abord, l'écosystème *Open Source* repose beaucoup sur le système d'**examen par les pairs**, que l'on retrouve dans la recherche scientifique. Ainsi, le code source d'un logiciel peut-être constamment vérifié et amélioré. De plus, un logiciel propriétaire est entièrement dépendant de l'avenir de son entreprise, là où un logiciel *Open Source* peut largement **survivre à son auteur d'origine**, tant qu'il existe une communauté assez active pour le maintenir.

Indépendance Ensuite, l'*Open Source* permet de **limiter la dépendance** à des tiers puisqu'il permet à tout le monde de gérer soit même son utilisation des logiciels. Ces solutions sont aussi très **flexibles** puisque chacun peut les adapter selon ses besoins, et, la plupart du temps, une solution a déjà été partagées par la communauté. Ce pont explique notamment l'intérêt particulier des **pays émergents** (*Chine, Inde, Brésil...*) pour l'*Open Source*, dans lequel ils voient un moyen de garantir leur indépendance technologique.

Sécurité Enfin, la **sécurité** des produits est souvent supérieure : une faille sur un logiciel *Open Source*, maintenu par des centaines de développeurs, à potentiellement plus de chance d'être corrigé rapidement que sur un logiciel propriétaire, qui dépend d'une équipe limitée. La **collaboration ouverte** prônée par l'*Open Source* permet également à tous les utilisateurs d'accéder facilement à de l'aide ou des ressources en ligne, ce qui permet de partager les bonnes pratiques. Aussi, la **transparence** inhérente à ce mode de production garantit une **sécurité des données** qui peut être vérifiée par les utilisateurs eux-mêmes.

Dangers Ce mode de production particuliers présente évidemment plusieurs défauts, principalement dû au fait qu'il est **entièrement basé sur la communauté**, qui n'est pas toujours disposée à contribuer suffisamment.

L'exemple *Heartbleed* En avril 2014, un faille critique dans la bibliothèque de chiffrement ***OpenSSL***, utilisée sur presque l'ensemble des machines formant l'Internet, est découverte par l'équipe de sécurité de ***Google*** et la société ***Codenomicon***. Introduite par erreur en 2012 par un développeur bénévole de la

communauté, on découvrira qu'un demi-million de serveurs webs auraient été touchés au moment de sa découverte. Cette événement à révéler le **manque de moyens** de projets *Open Source* pourtant cruciaux et très utilisé comme ***OpenSSL***. Cela a notamment mené à la création du ***Core Infrastructure Iniative*** (*aujourd'hui remplacé par l'**Open Source Security Foundation***) à l'initiative de la ***Linux Foundation***, en collaboration avec plusieurs grandes entreprises des technologies de l'information, et dont l'objectif est de financer ces **projets *Open Source* critiques**.

---

50 nuances d'Open Source

Si vous parlez avec des développeurs, il est possible que, malgré l'existence de définitions officielles comme l'***OSD***, leurs significations du terme *Open Source* divergent. Là où des néophytes pourraient penser que « *Open Source = gratuit* », les développeurs eux-même peuvent parfois confondre *Open Source* et **logiciel libre**.

Open Source ne veut pas dire gratuit

Et oui, même si cela peut paraître contre-intuitif, tous les projets ***Open Source*** ne sont pas forcément gratuit. Si on reprend les principes de l'*Open Source*, on voit bien que rien n'empêche la distribution payante d'un logiciel.

Vous devez sûrement vous dire que, si on donne le code source gratuitement, rien n'empêche aux clients de l'utiliser directement plutôt que de nous acheter la version distribuée, et vous n'avez pas tord. Mais il y a plusieurs raisons pour lesquelles les produits *Open Source* payants existent encore :

1. D'abord, selon le logiciel vendu, il n'est certain que tous les clients est les compétences pour utiliser votre logiciel à partir du code source ;
2. Ensuite, beaucoup d'entreprises souhaitent généralement s'assurer un support sur les logiciels qu'ils utilisent, même sur les logiciels *Open Source*, et c'est généralement ce qui est compris dans le prix (***ex :** Red Hat Entreprise Linux*).

Gratuit ne veut pas dire Open Source

De même, un logiciel gratuit n'est pas forcément *Open Source*, et c'est particulièrement vrai pour les sites et applications web. Les codes sources de ***Facebook***, ***Google Search***, ***Instagram*** ou ***YouTube*** ne sont certainement pas accessibles, pourtant vous n'en payez jamais l'entrée. La plupart de ces sites sont financés grâce à la vente d'espaces et/ou de données publicitaires.

Sources ouvertes mais pas Open Source

Il existe aussi des logiciels dont le code source est accessible gratuitement, mais dont les termes de la licence ne respecte pas tous les principes de l'***OSD***. C'est par exemple le cas du moteur de jeu ***Unreal Engine*** derrière ***Fortnite***, dont le code est consultable en ligne. L'utilisation du logiciel est gratuite mais la licence défini plusieurs conditions en cas d'utilisation commerciale.

Avec le succès de l'*Open Source*, ce terme est même devenu un argument commercial convaincant menant à plusieurs scandales. On peut par exemple citer la campagne de promotion de ***Windows 10***, durant laquelle on laissait supposer un passage à l'*Open Source* pour le système d'exploitation le plus utilisé au monde à l'occasion des 40 ans de ***Microsoft***, et ce afin d'élargir sa clientèle auprès des développeurs sensibles à l'*Open Source*. Un jour qui ne vint jamais...

Open Source + éthique = logiciel libre

L'*Open Source* a été créé pour lever l'ambiguïté sur le terme ***free software*** qui, en anglais, peu aussi bien vouloir dire **logiciel libre** que **logiciel gratuit**. L'idée était donc de **créer un cadre favorable à l'application des libertés d'accès au code source dans une activité d'entreprise**. Or, certains acteurs comme ***GNU*** se revendique du logiciel libre plutôt que de l'*Open Source*. Bien que la différence soit parfois assez fine, plusieurs structures, comme l'association française ***Framasoft***, préfère voir dans le logiciel libre une **combinaison des valeurs de l'*Open Source* et d'une éthique morale** : ***Open Source* + éthique = logiciel libre** (*ou plutôt* ***logiciel libre - éthique = Open Source***).

L'importance des licences

Le degrés d'« *ouverture* » d'un logiciel est donc avant tout défini par les termes de la licence sous laquelle il est distribué. Cet élément est indispensable pour les créateurs de logiciels qui souhaitent protéger leur code, c'est pourquoi il important de bien choisir sa licence logicielle avant de distribuer son code.

L'Open Source aujourd'hui

Les projets *Open Source* sont aujourd'hui **au cœur des technologies de l'information**, et notamment de l'Internet. Pour preuve, ce site est presque entièrement réalisé à l'aide de technologies *Open Source* ! La **méthode *Open Source***, quant à elle, s'exporte au delà des logiciels.

Une pratique plus que courante

L'*Open Source* en entreprise à pris de plus en plus d'importance, notamment depuis la pandémie du *COVID-19*. Une étude menée par ***Open UK*** sur 273 entreprises britanniques montrent que **97% d'entre elles utilisent au moins**

un logiciel *Open Source* et **65% contribuent à au moins un projet**. Cela s'explique notamment par la **réduction des coûts** évoquée plus haut.

De nombreux projets bénéficient aussi du support des **grandes entreprises technologiques**, à la fois très consommatrices et contributrices d'*Open Source*. En effet, ***Microsoft***, ***Google*** et ***Apple*** étaient dans le top 5 des contributeurs sur la plateforme ***GitHub*** (*propriété de* ***Microsoft***) en 2019. De plus, **41% des développeurs les plus actifs étaient affiliés à une entreprise** sur la même périodes, contre 19% d'organisations *Open Source* et autant de développeurs indépendants.

Un marché français en pleine croissance

Oui, il est possible de gagner de l'argent en France en faisant de l'**Open Source**. C'est en tout cas l'avis du cabinet ***Pierre Audoin Consultants*** qui, en 2011, évaluaient le marché français de l'*Open Source* à **2,5 milliards d'euros**, soit 6% du marché total des logiciels et services, avec une **croissance de 30% par an**.

Une prévision qui semble s'être confirmée en partie puisqu'en 2022, selon ***MARKESS***, l'*Open Source* en France représentait **6 milliards d'euros de revenu** et **60 000 emplois directs**, faisant du pays le leader de l'*Open Source* en Europe. Une situation favorable qui devrait perdurer avec une **croissance prévue de 8% par an** entre 2022 et 2027.

Au-delà des logiciels

Le concept d'*Open Source* s'étant aujourd'hui bien au-delà des logiciels informatiques. Des **composant électroniques** jusqu'à la **gouvernance publique**, en passant par la **finance**, la méthode *Open Source* semble définitivement en voie d'adoption.

Open Hardware La volonté d'étendre les principes de l'*Open Source* à l'électronique est notamment incarné par la ***Open Source Hardware Association*** (*OSHWA*), dans un secteur où la quasi-totalité des produits sont propriétaires malgré des standards ouverts. Cette initiative fait écho, plus généralement, au mouvement du **matériel libre**.

Les enjeux sont de tailles, notamment lorsqu'on sait, suite aux révélations d'Edward Snowden et d'autres, que certains composants d'ordinateurs commerciaux cachent des **portes dérobées** (*faille de sécurité volontairement dissimulé pour être exploitée*).

Cependant, selon Alicia Gibb, co-fondatrice de l'***OSHWA***, avec le physique, la recette ne peut-être exactement la même que dans le virtuel pour deux raisons principales :

1. Pour les logiciels, le code source donnera toujours le résultat escompté (*pour peu d'avoir la bonne configuration*) sur tous les ordinateurs. Or,

pour les objets physiques, **le résultat produit ne dépend pas que des plans de fabrication**, mais également de la matière première, des outils, des contraintes physiques...

2. Les **législations entourant la propriété intellectuelles** pour les logiciels et les composants électroniques sont fondamentalement différentes : pour les premiers, c'est le **droit d'auteur** qui s'applique presque automatiquement ; pour les seconds, ce sont les **brevets** qui prévalent et qui représentent un certain coût.

Open Government La forme finale de la pensée *Open Source* est à trouver dans l'idée de **gouvernement ouvert** dont l'objectif est, notamment à l'aide de logiciels libres, de promouvoir la **transparence** et la **collaboration citoyenne** dans les modes de gouvernance publique. De plus en plus de projets voient le jour partout dans le monde, avec par exemple la plateforme ouverte des données publiques ***Etalab*** en France, ou encore le projet ***g0v*** à Taïwan.

Utiliser

Privilégier les solutions OpenSource

Partager de bonnes pratiques

Être un utilisateur actif

Retour, rapport de bug, etc.

Contribuer

...selon ses compétences

...en tant qu'amateur

. . . en tant que professionnel

Entreprendre

Si on voit clairement les avantages qu'apporte l'*Open Source* aux utilisateurs, il est plus compliqué de comprendre pourquoi certaines entreprises s'y risqueraient, à part par idéologie. En effet, on pense souvent, à raison, que mettre en place ce système dans une entreprise, c'est à dire **mettre à disposition ses « *schémas de production* » et autoriser leur redistribution**, reviendrait à se tirer une balle de pied.

Les modèles de financement

Les erreurs à éviter

Conclusion

Ressources

Pages web

- « *The Open Source Definition 1.9* » - **Open Source Initiative** (*consulté le 10/05/2023, opensource.org*)

Articles

- « *L'Open Source, qu'est-ce que c'est ?* » - **Red Hat** (*24/10/2019, www.redhat.com*)
- « *De l'open banking à l'open finance* » - Denis Beau ; **Banque de France** (*24/03/2022, www.banque-france.fr*)
- « *De l'open banking à l'open finance* » - Julien Maldonato, Ghislain Boulnois ; **Deloitte** (*consulté le 10/05/2023, www2.deloitte.com*)
- « *De l'open banking à l'open finance* » - Julien Maldonato, Ghislain Boulnois ; **Deloitte** (*consulté le 10/05/2023, www2.deloitte.com*)
- « *Un dev open source aurait volontairement corrompu des bibliothèques largement utilisées, affectant des tonnes de projets.* » - Stéphane le calme ; **Developpez.com** (*10/01/2022, www.developpez.com*)

- « *Open source : les Gafam en tête des plus gros contributeurs sur GitHub* » - Antoine Crochet-Damais ; **Journal du Net** (*12/12/2019, www.journaldunet.com*)
- « *La croissance des logiciels open source viendra des entreprises* » - Ridha Loukil ; **L'Usine nouvelle** (*02/10/2012, www.usinenouvelle.com*)
- « *Etude 2022 : Le marché de l'open source en France et Europe* » - **CNLL** (*08/11/2022, cnll.fr*)

Wikipédia

- « *Open source* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *Openwashing* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *Open Source Definition* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *Rétro-ingénierie* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *Red Hat* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *Heartbleed* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *OpenSSL* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *Open Core Initiative* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *Matériel libre* » - (*consulté le 11/05/2023, fr.wikipedia.org*)
- « *Gouvernement ouvert* » - (*consulté le 11/05/2023, fr.wikipedia.org*)
- « *g0v* » - (*consulté le 11/05/2023, fr.wikipedia.org*)

Vidéos

Conférences

- « *The physical future of open source* » - Alicia Gibb ; **Red Hat Summit** (*09/05/2017, www.youtube.com*)