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 a **toujours existé en informatique**. Dans les années 60, les ordinateurs étaient livrés avec des logiciels dédiés et destinés, la plupart du temps, à des utilisateurs avisés qui n'acceptaient pas forcément de faire tourner des logiciels dont il ne pouvait **vérifier et comprendre le fonctionnement en détails**. Ainsi, les éditeurs fournissaient les codes source 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 aux clés 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 compréhensibles et manipulables, 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 accompagné 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 véhicule, 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 existent pour rendre ces investigations plus compliquées. De plus, si le code source n'est pas fourni, ces pratiques peuvent se révéler illégales au regard de la propriété intellectuelle. À noter que la rétro-ingénierie s'étend aussi à l'électronique et à l'industrie en général.

> 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 considérée 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 par 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 dus 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. Cet événement à révéler le **manque de moyens** de projets *open source* pourtant cruciaux et très utilisés 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-mêmes peuvent parfois confondre *open source* et **logiciel libre**.

Open Source ne veut pas dire gratuit

He 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 tort. 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 aient 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 respectent 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 définie 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 a pris de plus en plus d'importance, notamment depuis la pandémie du *COVID-19*. Une étude menée par ***Open UK*** sur 273 entreprises britanniques montre 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 période, 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, évaluait 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 incarnée par la ***open source Hardware Association*** (*OSHWA*), dans un secteur où la quasi-totalité des produits est propriétaire 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 dissimulée 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 intellectuelle** 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

La première étape pour un développeur *open source*, c'est **utiliser des logiciels *open source* et en promouvoir l'usage**. Comme on l'a vu, la communauté est la base de ce modèle et, ainsi, utiliser, c'est déjà un peu contribuer.

Privilégier les solutions open source

Vous ne vous en rendez peut-être pas compte, mais vous utilisez sûrement énormément de logiciel propriétaire. Sans reparler des services webs gratuits comme ***YouTube***, cette page est peut-être ouverte sur un navigateur ***Google Chrome*** connecté à votre messagerie ***GMail***, ou une fenêtre ***Microsoft Edge*** tournant sur ***Windows***. Personne n'est à blâmer puisque ces solutions sont souvent plus accessibles et « *user-friendly* » que leurs homologues *open source*. Mais vous serez étonné de découvrir des alternatives simples et gratuites comme ***PeerTube***, ***Mozilla Firefox*** ou une des centaines de distributions ***Linux*** disponibles.

Il peut être compliqué de choisir une alternative tant il peut en exister, mais l'avantage est que les retours d'expériences utilisateur sont nombreux. Le meilleur critère reste donc souvent l'**adoption d'un logiciel relativement à ses alternatives**. Attention cependant, en cas d'utilisation professionnelle, à ne pas négliger les **termes des licences** : il est alors primordial de vérifier ces termes, de préférence avant d'utiliser le logiciel.

Voici quelques sites webs répertoriant des alternatives *open source* ou libres à différents logiciels et services informatiques propriétaires :

- ***chatons.org***
- ***opensource.com***

- ***opensource.builders***
- ***www.opensourcealternative.to***

Bien choisir ses fournisseurs de services

Lorsqu'on choisit d'utiliser un service *open source*, comme ***Elastic Search***, on peut soit l'**auto-héberger** sur sa machine, ce qui n'est pas toujours optimal, soit le **déployer dans le « *cloud* »**. Vient alors la question du choix du fournisseur de services.

Le plus grand dans ce domaine est évidemment ***Amazon*** avec ***Amazon Web Services*** (*AWS*) qui comptent parmi ses clients ***Netflix***, la ***NASA*** ou encore la ***CIA***. Cette interface met notamment à disposition des développeurs des microservices principalement basée sur des technologies *open source*. Or, il est à noter qu'***Amazon*** est connu pour être un **utilisateur passif** d'*open source*, contribuant relativement peu par rapport aux autres ***GAFAM***.

Ainsi, afin d'éviter ces intermédiaires, beaucoup de projets *open source*, dont ***Elastic Search***, proposent des solutions d'hébergement. En tant qu'utilisateur, préférer souscrire directement auprès de l'éditeur peut être un moyen de soutenir l'*open source*.

Promouvoir l'open source

S'il y a un point sur lequel les projets *open source* ont des lacunes, c'est sur le ***marketing***. En effet, la plupart des contributeurs étant des développeurs, ingénieurs et autres universitaires, peu sont ceux à s'attarder sur le **design**, la **communication** ou l'**expérience utilisateur**, ce qui peut expliquer le manque d'adoption de ces logiciels par le grand public.

De fait, la publicité de beaucoup de projets *open source* passent par le bouche à oreille, et c'est pourquoi il est indispensable d'en parler autour de soi. Faire la promotion d'un projet *open source*, c'est plus que jamais lui permettre d'exister. Aussi, amener des utilisateurs à utiliser le logiciel, c'est renforcer sa communauté et donc sa qualité. D'autant plus si ces nouveaux utilisateurs sont différents des anciens, car ils pourraient amener des comportements inattendus et dévoiler des bogues, ou bien contribuer à des aspects moins développés.

Être un utilisateur actif

L'*open source* offre souvent des moyens insoupçonnés de **contribuer en tant qu'utilisateur**, sans développer.

Créer des retours

La correction d'un bogue dans un programme ne peut avoir lieu que si on a découvert le bogue. Rien d'extraordinaire jusque-là. Or pour qu'on le découvre, il faut d'abord que des personnes utilisent le logiciel (*cf. Promouvoir l'open source*), puis que ces utilisateurs rapportent les problèmes rencontrés aux développeurs, si possible de manière (*très*) détaillée. Ce comportement est relativement peu commun aujourd'hui pour un utilisateur lambda, car celui-ci se sent assez peu légitime à déposer une réclamation adressée à des développeurs expérimentés. C'est pourquoi beaucoup de projets tentent de **faciliter ces retours d'expériences** primordiaux dans le modèle *open source* pour les utilisateurs néophytes.

Assister à des événements

Le moyen le plus simple d'intégrer la communauté *open source* est de participer à l'un des **nombreux événements** organisées par différentes associations autour de ce thème. Il existe d'abord des sommets sponsorisés, comme le ***FOSDEM***, le ***Red Hat Summit*** de la société ***Red Hat*** ou encore l'***open source Experience*** organisée par les entreprises françaises ***Systematic*** et ***InfoPro Digital***, et qui peuvent notamment permettre de rencontrer les acteurs économiques de ce secteur. Ensuite, différentes organisations internationales de l'*open source* organisent des événements à travers le monde (*liste non-exhaustive de conférences*) et y invitent plusieurs chercheurs. Enfin, il existe des événements plus associatifs, comme le ***Capitole du Libre*** à Toulouse, qui rassemble cependant beaucoup d'entreprises de l'*open source* souvent disposés à embaucher.

S'impliquer dans une association

Pour s'engager davantage, il est aussi possible d'intégrer une **association engagée pour l'*open source***. Les variantes sont nombreuses, parfois orientées sur une technologie précise comme ***AFPY*** (*Association Francophone Python*) ou ***Debian France***, ou bien plus généraliste comme l'***AFUL*** (*Association Francophone des Utilisateurs de Logiciel Libre*) ou encore Framasoft**. *Vous pouvez trouver une liste non exhaustive d'association gravitant autour de l'*open source *sur le wiki de l'*_April___ (*Association de Promotion et de Défense du Logiciel Libre*).

Contribuer

Le meilleur moyen d'**allier mon projet professionnel et les valeurs de l'*open source*** est sans doute de contribuer à l'un des nombreux projets. Mais comment et à quels projets contribuer ? Et surtout comment intégrer cet **effort de contribution non rémunéré** à une activité professionnelle ?

Développer ses compétences

Si plusieurs projets souffrent de **manque de contribution**, les obligeant à faire des appels, et même à rémunérer des contributeurs, de nombreux développeurs ne se pensent pas assez compétents pour proposer des modifications. Or, s'il est vrai que des logiciels *open source* complexes et éprouvés ne seront jamais à la portée du premier venu, notamment s'ils sont affiliés à des organisations, on oublie souvent les **projets plus petits**, qui reposent souvent sur un unique mainteneur, et avec qui un dialogue direct peu s'installer. Même si vous pensez manquer de compétences, un retour extérieur sur ces projets peu regardés mènera toujours à des améliorations.

De plus, s'engager sur un projet communautaire peut permettre de **développer de nouvelles compétences** techniques, mais également d'**organisation** et de **communication**. En effet, cela revient à collaborer avec des développeurs à distance, le plus souvent en anglais, et en relative autonomie. Ainsi, cela peut être un bon moyen, pour des étudiants, de travailler sans risques sur des projets informatiques concrets.

On l'a vu, la communauté *open source* est principalement composé de développeurs idéalistes, et manque de fait de personnes compétentes dans des domaines tiers comme la **communication** ou le **graphisme**. Ces contributions peuvent faire la différence lors du développement d'un projet, et sont l'occasion pour un développeur d'étendre son domaine de compétences. Enfin, les organisations *open source* comme ***Tournesol*** sollicitent également les utilisateurs à contribuer à l'**administration de l'association**, en rédigeant des demandes de financement par exemple.

En faire une activité professionnelle

L'objectif annoncé étant de **concilier vie professionnelle et *open source***, j'ai été ravi de découvrir que bien des développeurs s'étaient déjà posé cette question, et qu'ils y avaient trouvé des réponses ! Il existe plusieurs situations possibles, les plus courantes étant la contribution en tant qu'entreprise et le travail dans une fondation *open source*.

Contribuer dans une entreprise

D'abord, comme vu précédemment, les entreprises constituent aujourd'hui le **pôle de contribution le plus actif** et de plus en plus de salariés sont **directement débauchés pour contribuer** à des projets *open source*. C'est notamment très courant dans les grandes entreprises, qui peuvent posséder des équipes dédiées et maintiennent parfois des projets entiers. C'est aussi répandu dans les **ESN** (*Entreprises de Services Numériques*) et autres **PME**, comme les Français ***Bootlin***, spécialisé dans l'ingénierie noyau ***Linux*** et embarqué, et dont les employés contribuent activement au noyau ***Linux***. Dans ce cas précis, on voit

bien l'intérêt de l'entreprise qui **améliore en continue un outil** au cœur de son activité, tout en confirmant son **status d'expert** sur cette technologie, puisque ***Bootlin*** est aujourd'hui co-mainteneur du ***Linux kernel***.

Devenir salarié d'un projet

Enfin, la situation idéale pour contribuer à temps plein à l'*open source* serait de **travailler directement pour un projet**. Et les offres ne semblent pas manquer puisqu'en 2023, la ***Linux Foundation*** recherche 12 salariés à temps pleins aux États-Unis. La fondation étant l'une des plus importantes de l'écosystème, ces offres vont de l'**événementiel** à la **comptabilité**, en passant par le **marketing**. En effet, comme dit précédemment, ce sont principalement ces compétences tierces qui manquent à l'*open source* et donc pour lesquels les projets engagent le plus. Ainsi, devenir un développeur *open source* à plein temps reste compliqué car ces places sont rares.

Cependant, il est aussi possible, et certains ont réussi, de se faire financer directement par la communauté, comme nous le verrons ensuite. On peut par exemple évoquer le cas exceptionnel de l'ancien ingénieur ***Google*** **Evan You**, créateur de ***Vue.js*** (*utilitaire de création de site web*), qui vit aujourd'hui entièrement, et confortablement (*$16k par mois*), grâce aux dons de la communauté. Accéder à cette situation requiert cependant d'avoir développé une communauté prête à contribuer financièrement, ce qui n'est pas toujours le cas, surtout si le code source est disponible gratuitement.

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. Mais comme on l'a vu, bien des entreprises vivent aujourd'hui de l'*open source*. Alors comment font-elles ?

Les modèles de financement

Avec le nombre croissant d'entreprises *open source*, de nombreux **modes de financement alternatifs** ont émergés pour allier les principes de liberté et le « *business* ».

Le dilemme de la publicité

Un des moyens les plus courant aujourd'hui pour financer un produit *open source* est le **financement par la publicité**. Il est très adopté chez les grandes

entreprises comme ***Google***, ***Mozilla*** ou ***Canonical*** (*co-mainteneur d'Ubuntu*), mais peu apprécié dans la communauté *open source*. Ce mode de financement soulève plusieurs **questions éthiques**, notamment quant à la complaisance envers les annonceurs.

Le cas d'***Adblock Plus***, une extension *open source* de navigateur bloquant les publicités, en est révélateur. Pour se financer, l'entreprise lié au projet à passer un partenariat avec ***Google*** pour **placer des publicités comme « *acceptables* »**, qui ne seraient donc plus bloquées par le logiciel...

Enfin, ce mode de financement créé un **sentiment de gratuité** chez les utilisateurs, qui efface la volonté des créateurs de l'*open source* de faire prendre conscience du coût des logiciels libres. Vous connaissez sûrement la maxime : « *Si c'est gratuit, c'est que c'est vous le produit.* »

Redistribution payante

Comme mentionné dans l'***OSD***, la redistribution payante d'un logiciel *open source* est possible. De plus, les **licences permissives** permettent à n'importe quelle entreprise de distribuer, sous licence propriétaire, un logiciel construit à partir d'outils *open source*.

Des entreprises vendent donc des logiciels basé sur des **composants *open source*** auxquels elles **contribuent activement ou financièrement**, ou qu'elles ont même créer.

Double licence Un modèle efficace à appliquer à un projet est celui de la **double licence**. L'idée est de distribuer, en plus d'une version *open source*, une version « *entreprise* » payante et dont la licence inclus des conditions propriétaires. Ce système permet notamment aux clients de **tester le logiciel** en version libre, puis de migrer vers une version plus professionnelle, à laquelle peut se greffer des **offres de support**. Il a fait ses preuves à travers de grands projets comme ***MySQL*** d'***Oracle*** ou le système d'exploitation ***Red Hat Entreprise Linux***.

Composants propriétaires

Une autre méthode, souvent nommée *open core*, consiste en la distribution d'un logiciel « *cœur* » sous licence *open source* d'une part, et la vente de contenu additionnel de l'autre.

Par exemple, dans les jeux vidéo, il est courant que le moteur d'un jeu soit disponible en *open source*, mais que les fichiers (*graphiques, audios, etc.*) soient propriétaires.

Extensions propriétaires Une variante très utilisée de cette méthode se concentre sur le **développement d'extensions propriétaires** pour un logiciel *open source*. L'écosystème ***Wordpress*** (*création de site web*) s'est notamment

construit sur ce modèle, avec un cœur *open source* et de nombreuses **extensions communautaires payantes**.

Système de mise à jour propriétaire Une autre variante peut être de réaliser différentes versions *open source* d'un logiciel, mais de rendre propriétaires les **utilitaires de migration** d'une version à l'autre. Ce système fonctionne particulièrement pour les systèmes centralisant des données, comme les bases de données. À noter que ce modèle n'est pas considéré par tous comme relevant de l'*open source*. Richard Stallman, initiateur du projet ***GNU***, dit à ce sujet :

> Toute version du programme de base de données peut être utilisée sans logiciel non-libre ni SaaSS. Le problème se pose lorsque vous essayez de continuer à **utiliser le programme sur le long terme**, ce qui implique de le mettre à jour de temps en temps ; vous ne pouvez pas l'utiliser de cette manière sans un logiciel non-libre ou équivalent. Ce programme de base de données est piégé dans le temps - on pourrait dire qu'il est "**piégé *diachroniquement***".

Vendre des services

La **vente de services professionnels** est sûrement le moyen le plus répandu pour faire fructifier un produit *open source*. Les services varient selon les entreprises et les logiciels : création et l'animation de **formation**, **support** technique, **conseil**, **certification**...

De grandes entreprises ont réussi à perdurer grâce à ce modèle. C'est le cas par exemple de ***Red Hat*** ou encore d'***IBM***. En France, le marché des services numériques représentait **61.7 milliards d'euros** en 2016, et repose, comme partout ailleurs, sur de nombreuses technologies *open source*.

Software as a Service Comme on l'a vu précédemment, les entreprises peuvent souvent avoir besoin de délocaliser leurs solutions dans le *cloud*. C'est pourquoi plusieurs entreprises, en plus d'éditer un logiciel *open source*, proposent **service *cloud* associé** : on appelle ça le ***SaaS*** (*Software as a Service*). L'avantage pour les fournisseurs est la garantie d'un **revenu relativement stable**, ce genre d'offre se faisant principalement par souscriptions.

Cependant, comme on l'a aussi vu, **la concurrence est rude** chez les fournisseurs de services, notamment face à des géants comme ***Amazon Web Services***, ***Microsoft Azure*** et ***Google App Engine***. D'autant plus que les **moyens nécessaires** pour construire une solution totalement auto-hébergée sont considérables. C'est pourquoi ce mode de financement est avant tout répandu dans les *start-ups* soutenues par de solides investisseurs.

Vendre son image

L'atout d'un projet *open source* est certainement sa communauté, et la **reconnaissance publique** que celle-ci lui accorde. Ce point est particulièrement

pertinent dans un contexte de « ***mode de l'open source*** ».

Usage de marque Si un projet à du succès, il peut être possible de monnayer l'**usage de la marque** associée à travers des partenariats commerciaux. Ce modèle a été initié par le système de gestion de l'apprentissage ***Moodle***, qui autorisent des **partenaires certifiés** à utiliser le nom et le logo ***Moodle*** dans leurs logiciels dérivés contre une part de leurs revenus.

Merchandising Des projets ayant mobilisé une grande communauté ont aussi réussi à se financer en partie grâce à la **vente de produits dérivés** à l'image de leurs logiciels, comme des vêtements ou des tasses. Cette offre ne permet sûrement pas de générer des revenus suffisants, mais elle est très appréciée de la communauté. Certaines fondations brillent particulièrement dans le domaine, comme ***Mozilla*** et ***Wikimedia***.

Compter sur la communauté

La communauté *open source* s'étant construite sur des notions de partage, elle se montre particulièrement généreuse lorsqu'elle croit en un projet.

Dons Les **dons volontaires** de particuliers, d'organisations ou d'entreprises permettent aujourd'hui de financer de nombreux projets *open source*. C'est par exemple le cas de l'association française d'éducation populaire ***Framasoft***, entièrement financée par des dons et qui compte une dizaine d'employés. Ce succès est notamment dû à une campagne de développement de services alternatifs à ***Google***, ainsi, pour espérer avoir des donations, il faut d'abord avoir un **bon projet**.

Ces dernières années, ce modèle à profiter du développement de **services de micro-paiements** comme ***PayPal***. Cependant, cela peut mener à une **dépendance vis à vie d'un fournisseur**, qui s'est révélé problématique récemment avec la fermeture de la plateforme française de financement ***uTip***. Avec la croissance des **cryptomonnaies**, les créateurs peuvent espérer des améliorations de ce côté.

Financement participatif Un nouveau modèle à émerger avec le développement de **financement participatif**, qui permet aux futurs utilisateurs d'un projet d'investir directement à sa réalisation. Généralement à durée limitée et avec des paliers à atteindre, ces campagnes de financement peuvent s'accompagner d'offres de précommandes afin de rétribuer les donateurs.

Organismes de financement

Enfin, il est possible de financer un projet *open source* auprès de différents organismes : gouvernements, universités, organisations, et même des entreprises qui proposent parfois des **subventions** ou des bourses.

Le danger des investisseurs De manière générale, la communauté *open source* se méfie des investisseurs extérieurs, qui **dévieraient inévitablement les projets des valeurs de libertés**. Cela à pu se vérifier lors de plusieurs investissements massifs dans des entreprises *open source*, comme ***Elastic Search*** qui a été accusé de se « ***refermer*** ». Il faut cependant comprendre la tentation d'un créateur de logiciel de se voir offrir des moyens qu'il n'espérait plus, quitte à rogner sur les bords.

Les clés du succès

Toutes les entreprises s'étant lancées dans l'*open source* n'ont pas réussi : soit parce qu'elles se sont trop ouvertes, soit parce qu'elles se sont trop fermées, souvent parce qu'elles n'ont pas **trouvé le bon modèle**. Il est a peu près certain qu'un mauvais logiciel ne réussira pas, mais tous les bons logiciels ne réussissent pas. Alors quelles sont les **clés du succès** ?

Développer ce qui n'existe pas

L'*open source* constitue aujourd'hui une base de connaissances libres sans égale, il peut donc paraître difficile de voir où l'on pourrait se faire une place. Or, s'il existe des entreprises de services numériques, c'est qu'il y a du travail à faire : l'idée est juste d'y intégrer l'*open source* ! Si cette méthode semble être un désavantage compétitif, elle **favorise davantage les projets innovants**, qui innoveront d'autant plus grâce à l'**amélioration continue** par la communauté.

Innover ne veut pas dire créer. En effet, certains logiciels libres ne respectent pas forcément toutes les prérogatives de l'*open source* ou les différents standards. Certaines entreprises, comme ***Bootlin***, s'attellent à la standardisation pour le noyau ***Linux*** de codes sources à la demande des auteurs, ce qui peut offrir à ces derniers des arguments commerciaux.

Se spécialiser pour gagner en efficacité

Lorsqu'on observe l'écosystème *open source*, on remarque que la plupart des entreprises sont **spécialisées** dans des compétences particulières, comme le noyau ***Linux*** pour ***Bootlin***. Selon le fondateur de la société, ce positionnement permet de faire valoir une expertise précise afin de proposer des solutions optimales.

Intégration de projets *open source* Dans la même veine que la standardisation de codes sources, l'**intégration** de logiciels *open source* est une compétence recherchée. L'idée est d'**intégrer des solutions libres dans des projets d'entreprises** souhaitant s'ouvrir davantage. Cela requiert cependant une certaine expertise sur l'écosystème des logiciels *open source*.

Contribuer pour se faire connaître

Il est primordial pour une entreprise basée sur l'*open source* de continuer à **contribuer au maximum**. Bien que cela ressemble à une activité bénévole, la contribution rapporte souvent de la notoriété dans la communauté d'un logiciel dont l'entreprise peut profiter. Elle permet également de renforcer l'image d'expertise.

Identifier sa valeur ajoutée Au-delà de partager son temps, il peut être compliqué de définir quels produits partagés en tant qu'entreprise. De fait, il est important d'évaluer les conséquences, négatives comme bénéfiques, d'une mise en *open source* d'un produit. Par exemple, la société ***Bootlin***, a décidé de distribuer l'ensemble de ses supports de formations librement. En effet, elle considère que la véritable **valeur ajoutée** de son offre de formation est la prestation du formateur. Et ce choix lui réussi puisqu'il a largement contribué à la faire connaître internationalement, si bien qu'elle reçoit des demandes jusqu'en Nouvelle-Zélande !

Conclusion

Pour répondre à ma problématique, je dirais qu'il existe de nombreuses manières d'**allier mon projet professionnel et les valeurs de l'*Open Source***, comme beaucoup de développeurs le font déjà. De plus, au vu de la démocratisation de cette méthode, m'orienter dans ce domaine pourrait représenter une grande **opportunité professionnelle**.

Une possibilité

Nous l'avons vu, il existe un véritable **écosystème *open source*** composé de nombreuses entreprises disposées à embaucher à différents postes. J'aimerais personnellement m'orienter sur le **développement de systèmes embarqués**, qui repose beaucoup sur le système d'exploitation *open source* ***Linux***. De plus, je pense que l'**animation de formation** et la **production de contenu éducatif** sont indispensables à l'avenir de l'*open source*, c'est pourquoi j'aimerais les intégrer à mon projet. Une entreprise comme ***Bootlin***, dans laquelle les développeurs sont à la fois **ingénieur, formateur et contributeur** au noyau ***Linux***, correspond parfaitement à ce que je pourrai rechercher !

Une opportunité

L'*open source* étant en pleine expansion, me spécialiser dans cette méthode pourrait se révéler judicieux. La France est en effet le **leader européen de l'*open source*** et les administrations nationales et locales s'investissent de plus en plus dans ces technologies, ce qui permet l'**ouverture de nouveaux**

services. Ainsi, l'ouverture des **données bancaires** à par exemple permis la création de services tiers de gestion bancaire, qui s'étendent aujourd'hui à la **sphère financière**. Le **secteur médical** est également très demandeur d'*open source* et d'*open data*.

Enfin, il est de d'autant plus probable que le monde de demain soit *open source* avec l'apparition des problématiques liées à l'**environnement**. En effet, les valeurs de **collaboration** portées par l'*open source* me semblent indispensables à la conception et dans l'exécution de la transition écologique.

Ressources

Pages web

- « *The Open Source Definition 1.9* » - **Open Source Initiative** (*consulté le 10/05/2023, opensource.org*)
- « *Careers* » - **Linux Foundation** (*consulté le 11/05/2023, www.linuxfoundation.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* » - 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*)
- « *How Vue.js Creator Evan You Earns $16k/mo on Patreon* » - Olivia Seitz ; **Patreon Blog** (*21/08/2018, blog.patreon.com*)
- « *When Free Software Depends on Nonfree* » - Richard M. Stallman ; **GNU** (*consulté le 11/05/2023, www.gnu.org*)

Wikipédia

- « *Open source* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *Openwashing* » - (*consulté le 10/05/2023, fr.wikipedia.org*)
- « *Open core* » - (*consulté le 11/05/2023, 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*)
- « *Modèles économiques des logiciels open source* » - (*consulté le 11/05/2023, fr.wikipedia.org*)
- « *Entreprise de services numériques* » - (*consulté le 11/05/2023, fr.wikipedia.org*)

Vidéos

- « *#AuPoste avec Framasoft* » - **Blast** (*14/01/2023, www.youtube.com*)

Conférences

- « *The physical future of open source* » - Alicia Gibb ; **Red Hat Summit** (*09/05/2017, www.youtube.com*)
- « *Financer son logiciel libre, c'est possible !* » - Ludovic Dubost ; **Capitole du Libre** (*20/01/2023, www.youtube.com*)
- « *Business model open-source: l'exemple de Bootlin dans le monde de l'embarqué* » - Thomas Petazzoni ; **Capitole du Libre** (*20/01/2023, www.youtube.com*)
- « *Le logiciel libre, un enjeu politique et social – Comment agir avec l'April* » - Étienne Gonnu, Isabella Vanni ; **Capitole du Libre** (*20/01/2023, www.youtube.com*)
- « *Les business models des éditeurs open source* » - Sylvain Wallez ; **Capitole du Libre** (*19/11/2018, www.youtube.com*)
- « *Donner vie au RER toulousain avec des logiciels libres* » - Benoît Lanusse ; **Capitole du Libre** (*04/12/2018, www.youtube.com*)