`OPENOFFICE.ORG`, POURQUOI PAS ?

BENJAMIN BAYART

Résumé. Le monde du logiciel libre s'ébahit tous les jours d'avantage d'avoir enfin réussi à copier le fleuron des logiciels Microsoft® : KDE vous offre un menu « démarrer », et `OpenOffice.org` remplace facilement la suite Ms-Office.

Dans un premier temps, après avoir ouvert quelques pistes sur la notion de logiciel libre en posant la notion de logiciel *libérateur*, cet article tâchera d'exposer quelques critères simples d'évaluation de la qualité d'un logiciel.

Ensuite, en se basant sur ces critères, on soulignera que certaines visions trop simplistes donnent de fausses joies, et on rappellera qu'`OpenOffice.org` est tout de même plus un échec qu'une victoire.

C'est volontairement que cet article souligne les défauts d'`OpenOffice.org`, et d'autres projets similaires, le propos étant de discuter ces défauts. Ces logiciels ont aussi de grandes qualités, mais c'est une autre histoire.

1. Introduction

Longtemps, comme tous les utilisateurs de Linux de l'époque, lorsque j'essayais d'expliquer à mes amis qu'ils devraient envisager d'utiliser Linux, la limitation venait de l'interface. Ils avaient l'habitude d'un système très pauvre, instable, lent, mais qui donnait l'illusion d'être facile à utiliser : trois clics de souris par-ci, deux copier-coller par-là, et tout donnait l'illusion de marcher tout seul.

J'avoue que sans trop y réfléchir, je me suis dit plus d'une fois qu'ils avaient raison, et que ce qui manquait le plus à Linux, c'était une interface graphique avancée, et une suite bureautique qui soit utilisable par des débutants. Aussi, dans mon for intérieur, je me félicitait de l'avènement de KDE, d'`OpenOffice.org` et autres Gnomeries. Je me laissais impressionner par les manipulations complexes à base de copier-coller, de drag'n'drop, et autres cloneries d'OLE, des conférenciers qui venaient nous démontrer combien tout cela était beau.

Pourtant, bien que très satisfait de tous ces produits qui mettaient à la portée de mes amis mes logiciels préférés[1], je ne m'en suis jamais servi.

La contradiction est évidente, mais j'ai mis très longtemps à la comprendre, puis à la réfléchir. C'est avec un peu de recul que je viens vous en faire part dans cet article.

2. Liberté

Souvent, on se contente d'une définition assez formelle du logiciel libre, dérivée de la définition de GNU. On dit qu'un logiciel est libre si le source de ce logiciel est livré en même temps que le logiciel lui-même, s'il est librement modifiable, et si les versions modifiées sont librement diffusables.

Date: 19 mars 2004.

[1] Pouvoir enfin initier mes amis et mes proches aux joies de Linux, de LaTeX dans un environnement normal.

Cette définition est intéressante, mais elle semble de nos jours nettement trop restreinte.

Elle ne garantit que la liberté des gens qui sont *capables* de modifier du source. Pour moi, qui suis ingénieur en informatique, et qui suis spécialisé dans le développement, fort de plus de 20 ans de pratique, c'est une bonne définition, elle garantit *ma* liberté.

Le logiciel libre assure la liberté des informaticiens, ou des gens qui ont les moyens de s'offrir leurs services, mes patrons successifs par exemple. C'est bien, mais c'est un peu restreint. Qu'en est-il pour les utilisateurs *normaux*, pour les juristes, les philosophes, les informaticiens non-développeurs, bref, les *utilisateurs* de programmes informatiques ?

L'aspect libre d'un logiciel est pour eux une vertu qui relève de l'auto-suggestion, ou de la croyance.

Alors quelle définition nouvelle ? On pourrait proposer une alternative, un nouveau mot, qui néglige totalement la notion de code source[2], mais qui se focalise sur la notion d'utilisateur et d'utilisation.

Ce qui est le plus intéressant, dans ce concept, ce n'est pas que le logiciel soit libre, mais que l'utilisateur soit libre. On appellera donc les logiciels qui aident à cela les « logiciels libérateurs ».

Une première proposition de définition donnerait :

> *Est dit libérateur un logiciel qui n'entrave pas sans raison ni acceptation de sa part la liberté de celui qui l'utilise.*

Cette définition est volontairement en creux, elle indique quel défaut des logiciels en général les logiciels libérateurs permettent d'éviter.

Tout d'un coup, la perspective change totalement par rapport au logiciel libre.

Bien entendu, cette définition est imparfaite. Elle est encore trop floue. Plusieurs points demandent à être étudiés un peu plus finement.

2.1. **Propriété intellectuelle.** Il est communément admis dans le monde du logiciel libre que la propriété intellectuelle est une entrave au bon développement de l'humanité en général et de la science en particulier.

Celle qui consiste à empêcher de réfléchir à partir des idées d'autrui, celle qui entrave la bonne circulation des idées et des informations, a certainement ces défauts.

La propriété intellectuelle que l'on souhaite évoquer ici n'est pas celle des brevets et autres licences, mais plutôt celle du droit moral accordé à un auteur, qui est incessible et inaliénable en droit français.

Qui est propriétaire des données ? Lorsqu'un texte est écrit, puis mis en forme avec un logiciel, de qui est-il la propriété ? Lorsqu'une donnée est entrée dans une application, qui en est propriétaire ?

Il semble que la réponse soit claire : l'auteur. À savoir l'utilisateur du logiciel. Si le simple fait d'entrer une donnée dans un logiciel entravera par la suite sa réutilisation, alors ce logiciel ne peut pas être dit libérateur au sens de la définition proposée ici.

[2]Cette notion n'est de toutes façons pas assez claire pour beaucoup de monde, et n'évoquera rien à un juriste ou à un grammairien.

On peut donc en retenir que :

> *Si un logiciel s'accapare les données de l'utilisateur et est incapable d'interopérer de manière normalisée à partir de ces données, il porte atteinte sans raison à la liberté de l'utilisateur.*

2.2. **Le savoir c'est le pouvoir.** De tous temps, le savoir et le pouvoir ont été intimement liés. Détenir l'information, c'est détenir à court ou moyen terme, les rennes d'un pouvoir sans limites.

C'est vrai dans bien des domaines, de la politique au monde des finances. Combien donnerait un politique pour décider des informations à diffuser ? Combien donnerait un financier pour savoir ce que font, et ce que pensent, ses concurrents ?

L'information qui est confiée à un logiciel doit rester la propriété de l'utilisateur, pas celle de logiciel ou de l'auteur du logiciel. L'utilisateur-auteur doit donc à tout moment disposer de ses données, *sans qu'il ne doive faire appel au logiciel pour la retrouver.*

En effet, de plus en plus, le monde moderne tend à louer ce qui se vendait auparavant. Si l'on confie un contenu à un logiciel x et qu'il faut impérativement faire usage du logiciel x pour ré-accéder à cette donnée, alors ce logiciel devient une forme de droit à verser par l'auteur pour accéder à sa création, alors même qu'il n'a pas vendu cette création. Un tel système ne peut pas être dit libérateur.

Ce qui nous amène à :

> *Si un logiciel stocke les données de l'utilisateur dans un format qu'il est seul à savoir lire, il ne peut pas être dit libérateur.*

2.3. **Entrave et mensonge.** Sous le prétexte trop simple de vouloir aider l'utilisateur débutant, bien des logiciels tendent à ne présenter qu'une version édulcorée des données qu'ils possèdent.

Par exemple le lecteur de mail qui néglige de montrer qu'il a reçu le mail en deux versions (html et texte) et n'en montre que l'une des deux en masquant totalement l'autre ; ou qui néglige de donner accès aux en-têtes du mail, entrave la liberté de son utilisateur, et lui ment.

Bien des logiciels, prenant les utilisateurs pour des sots[3], croient malin de ne pas tout dire, pour ne pas perturber, ou pour ne pas faire peur. Un peu comme un gouvernement qui préfère ne pas dire, croyant le peuple trop bête pour comprendre.

On peut en retenir :

> *Un logiciel qui tend à masquer tout ou partie de l'information ne peut pas être dit libérateur.*

2.4. **Pérennité.** Depuis plus de 30 ans que le logiciel libre existe, une des principales caractéristiques qui faisait son avantage sur bien des logiciels propriétaires était la pérennité.

D'une part la pérennité de l'outil : le logiciel lui-même, étant écrit et maintenu par des volontaires, se trouvait à l'abri de la faillite d'un éditeur, ou d'un krach boursier. Et à chaque apparition d'une version divergente, l'ancienne version continuait d'être diffusée, et le plus souvent les données dans le format de l'ancienne version pouvaient être reprises.

[3]C'est un point de vue qui se défend, le plus souvent, mais ce n'est pas une raison pour s'en satisfaire.

En matière d'automobile, on considère que la durée de vie d'un véhicule est de 10 ans, ainsi c'est pendant cette durée que le constructeur s'engage à fournir des pièces détachées. En matière d'aéronautique, les durées sont encore plus longues (parfois jusqu'à 30 ou 40 ans).

Combien de temps des données peuvent-elles vivre ? Virtuellement infiniment.

On pourrait donc considérer qu'un logiciel qui disparaît sans raison valable (certains cas de force majeur, par exemple), tend à brider la liberté de son utilisateur en voulant lui interdire d'utiliser autre chose que la dernière version à la mode.

Ce point peut sembler anecdotique, mais, dans la pratique, pour relire des données datant de plus de 10 ans, la disponibilité des outils de l'époque est une condition presque incontournable.

> *Un logiciel dont on n'assure pas la diffusion de toutes les versions ne peut pas être dit libérateur.*

Bien entendu, ce postulat est très intransigeant, et pourrait être adouci en un postulat moins rigide :

> *Un logiciel dont on n'assure pas qu'il soit capable de lire ses anciens documents, fût-ce en diffusant ses vieilles versions, ne peut pas être dit libérateur.*

2.5. **Un exemple.** Les notions de logiciel libre et de logiciel libérateurs sont relativement indépendantes. Certains logiciels libres sont libérateurs, ce sera pas exemple le cas de `gcc` ou de Linux (le noyau). Certains logiciels non-libres le sont également, un exemple frappant étant AcrobatReader.

3. Le génie

Tous les logiciels libres ne sont pas de bons logiciels. De même que tous les logiciels propriétaires ne sont pas de mauvais logiciels.

Si le fait d'être libre est une qualité pour un logiciel, ce n'est pas la seule qualité que l'on doive lui demander. D'autres qualités sont par exemple sa stabilité, sa souplesse, sa conception soignée et attentive, ou sa capacité à être facilement étendu.

Bref, tout ce que l'on désigne parfois sous le nom de « génie logiciel », ou plus exactement les objectifs de celui-ci.

Il ne suffit pas de diffuser le source du premier logiciel venu, mal pensé, mal écrit, rempli de bugs et de limitations aberrantes, pour que ce machin devienne une merveille.

Unix n'est pas, historiquement, un logiciel libre. Et pourtant il représente une avancée majeure de l'informatique moderne. Un concept génial, pensé et mûri, qui a fait progresser l'ensemble de l'informatique, y compris les systèmes d'exploitation concurrents : on retrouve dans les Windows modernes plus d'un concept hérité d'Unix (sans parler des derniers Mac OS X).

Quels critères simples peuvent servir, sur ce plan là, à différencier le niveau de qualité de deux logiciels ? On en proposera ici que quelques-uns.

3.1. **La stabilité.** C'est probablement le critère le plus complexe à mesurer. On trouvera au moins deux axes pour en juger : la fréquence des plantages, qui est plutôt mauvais signe pour l'utilisateur, et la fréquence des révisions.

Au premier abord, on est tenté de croire qu'un logiciel qui est mis à jour très souvent est un logiciel qui fait l'objet d'un suivi attentif et régulier. Mon opinion personnelle serait plutôt qu'un logiciel qui est mis à jour trop souvent n'est pas un produit stable, soit mal fini, soit truffé d'erreurs.

On a donc bien la stabilité à l'utilisation, et la stabilité dans le temps.

3.2. **L'ergonomie.** Là encore, un critère délicat à mesurer. Mais pourtant important pour l'utilisateur final. Pour le public visé, le logiciel est-il facile à utiliser, ou nécessite-t-il un apprentissage anormalement long ?

3.3. **La taille.** En effet, tous autres critères égaux par ailleurs, il vaut mieux avoir un logiciel qui prenne moins de place, que ce soit en mémoire ou sur disque. Ce n'est certes pas le critère le plus important, mais il n'est pas à négliger.

Ce critère est d'autant moins à négliger qu'il est bien connu que plus un système est complexe (plus il comporte de pièces), plus il a de chances d'être en panne. Logiciellement, cela se traduit par une probabilité de bugs qui croît *plus vite* que la taille du logiciel[4].

3.4. **La finition.** Il s'agit là de savoir si le logiciel dont on parle est abouti, s'il a été suffisamment réfléchi pour qu'on puisse considérer qu'il représente un état de l'art, une base de référence. Ou, au contraire, s'il représente juste une preuve de principe, un intéressant prototype.

3.5. **La conception.** Le logiciel que l'on cherche à évaluer a-t-il été conçu[5], et si oui, a-t-il été *bien* conçu ?

A-t-il été pensé comme un tout monobloc, ou comme un ensemble de parties interopérant de manière claire et bien étable, pouvant être facilement adaptées à de nouveaux usages, pouvant être réutilisées, y compris pour des applications inattendues ?

En général, les signes de bonne conception sont le nombre de reprises du logiciel : plus il est repris, en tout ou en partie, plus il montre qu'il avait un fort potentiel.

4. OpenOffice.org, pourquoi pas.

Le projet `OpenOffice.org` [6] a été choisi, parce qu'il est emblématique. D'autres projets du même ordre auraient pu être retenus, en développant le même argumentaire.

---

[4]On considère que la nombre de bugs augmente plus que linéairement par rapport à la taille d'un programme, c'est-à-dire que lorsque l'on multiplie par 2 la taille d'un programme, on multiple par *plus que 2* le nombre de bugs.

[5]Dans le monde du développement, la *conception* est la phase de reflexion où l'on définit quel programme on souhaite écrire, et surtout comment on souhaite l'écrire.

[6]Si j'ai choisi ce logiciel, c'est simplement parce qu'il rentre précisément dans le cadre de ma spécialité en informatique, à savoir la production de documents. En général, on me connaît plutôt pour mes travaux autour de LaTeX.

4.1. **Un vrai problème.** Ce projet répond à un vrai problème profond, et sérieux, que ses auteurs traitent avec tout le sérieux voulu.

Ce vrai problème est que la suite Office, et l'environnement logiciel qui lui est immanquablement associé, ne répondent à aucun, ou presque, des critères de qualité et de liberté qui ont été évoqués plus haut. En effet, une fois un document entré dans Word ou dans Excel, il est pratiquement impossible d'en extraire les données pour en faire quoi que ce soit si on ne possède pas le même logiciel pour faire l'extraction des données.

Ainsi, taper son journal intime sur Word, c'est attribuer à Microsoft un droit de regard sur qui pourra, ou ne pourra pas, à l'avenir, le lire.

Or les formats de fichiers produits par ces logiciels sont considérés par trop de gens, faute de conscience technique ou politique, comme une norme, et servent donc de base d'échange.

Le seul *vrai* problème posé par cette série d'outils, c'est le format des fichiers.

4.2. **Une mauvaise réponse.** La réponse proposée par le projet `OpenOffice.org`, et par les autres projets similaires (AbiWord, Koffice, etc.) est mauvaise sur plusieurs points.

D'abord, elle entérine que le format `.doc` est le seul format d'échange de fichiers issus d'un traitement de texte. Ensuite, elle prétend que pour soigner le mal produit par Ms-Office, il faut le copier.

Étudions les différents points sur la liberté, et sur la qualité logicielle, qui font que cette réponse n'est pas satisfaisante.

4.2.1. *Propriété intellectuelle.* Les données confiées à `OpenOffice.org`, ou à Koffice, deviennent dans la pratique inutilisables par un autre logiciel. Le format de stockage de ces données a beau être documenté (en 571 pages), dans la pratique il n'est pas simplement exploitable[7], et n'a pour le moment été repris par aucun autre logiciel. Il n'est pas véritablement assis sur un standard communément admis.

Certes, le stockage est fait en XML, qui est une norme. Mais c'est une norme qui permet de stocker n'importe quoi n'importe comment. XML est une norme permettant à chacun de stocker des informations à sa sauce. Chacune des suites de logiciels concurrents de Ms-Office a choisi son propre mode de stockage XML, bien entendu incompatibles l'un avec l'autre.

4.2.2. *Contre-productif.* Le but premier de ces projets, à bien y réfléchir, était d'essayer de créer une alternative, d'essayer de lutter contre la standardisation de fait des formats propriétaires et non-documentés de Microsoft.

Or c'est précisément le contraire qui est atteint : quelle que soit la suite Office que vous choisissiez, celle de Microsoft, ou une des alternatives pseudo-libres, le seul format d'échange commun sera le fameux `.doc` contre lequel on voulait lutter au départ.

[7]Même pour un bon développeur, qui voudrait user de la liberté que lui accorde ce logiciel, cela demande un investissement considérable de coder un programme capable de comprendre ce format de fichiers.

4.2.3. *Et le génie logiciel ?* Le principal argument qui est le plus souvent avancé contre les produits Microsoft, en dehors de leur aspect propriétaire, c'est leur manque de stabilité. Or, les informaticiens le savent bien, plus un produit est gros, moins il est stable.

En reprenant la même stratégie de développement, à savoir des applications monumentales qui mettent en branle plusieurs millions de ligne de code, on a délibérément choisi de reproduire la même instabilité. Objectivement, à l'usage, la suite Ms-Office est plus stable que ses concurrents. Et, tout aussi objectivement, Windows est plus stable que KDE ou Gnome.

Certes, Linux reste plus stable que Windows, mais le critère de comparaison est faussé : dans un cas on ne teste que le noyau, dans l'autre cas, on test le noyau et une interface graphique complète. Si l'on rééquilibre la comparaison en mettant d'un côté Linux et une interface graphique, et de l'autre côté Windows, ou Mac OS, alors le résultat de la comparaison est moins favorable.

On examinera à peine la question de la taille des applications ou de la quantité de ressources consommées, la balance dans ce domaine là penche sans l'ombre d'un doute *contre* les solutions libres.

4.2.4. *La stratégie de l'échec.* Jouer en défensive, copier l'adversaire, ne pas chercher à innover, c'est choisir délibérément la stratégie de l'échec.

Sur ce point, deux projets peuvent être comparés : Mozilla et OOo[8].

Aux origines du projet Mozilla, on trouve Netscape 6, une monstruosité, un échec retentissant. Le seul objectif était d'essayer de rattraper le retard sur Internet Explorer. Le résultat a été un produit particulièrement inefficace, instable, en retard. Quand le projet a mué, faisant sa crise de croissance, pour devenir Mozilla, des choix radicalement différents, voire opposés, ont été faits. Il n'était plus question de chercher à talonner le concurrent, mais de refaire, de repenser, de *concevoir* un navigateur de fond en comble.

L'aboutissement de ce mode de fonctionnement, c'est un navigateur qui est utilisé (ce qui ne prouve rien), mais surtout qui est repris, dont plusieurs éléments sont réutilisés par d'autres projets. Le moteur de rendu des pages HTML est repris par d'autres navigateurs. Jusqu'au système de gestion des bugs qui est devenu un produit a lui tout seul !

Cette piste ne semble pas être la piste privilégiée, pour le moment, des projets autour des suites Office.

4.3. **Un bon point.** Cependant, en creusant un peu autour des projets de suite Office pour Linux, quelques points positifs ont fait surface. Pas encore des promesses pour un avenir radieux, plutôt des portes intéressantes ouvertes pour l'avenir.

Le principal est la qualité du format de fichier retenu pour ces outils. Autant la qualité du produit lui-même laisse à désirer, et ne semble pas très abouti, autant le format de fichier a l'air prometteur. Il aurait encore besoin d'être réfléchi pour arriver à quelque chose de convaincant, mais il part de bases qui ont l'air bien saines.

Avec encore un peu de maturation, il pourrait probablement être proposé comme norme. Il resterait alors à espérer que cette norme soit reprise, au moins par l'ensemble de ces logiciels.

[8]C'est l'abréviation utilisée pour désigner `OpenOffice.org`, jusque sur leur site.

5. Conclusion

Cet article se voulait ouvertement polémique. Objectif, certes, mais polémique, cherchant à souligner les erreurs stratégiques commises ces dernières années par la communauté de développement du libre.

Le logiciel libre avait pour habitude de représenter la pointe dans un domaine, l'aboutissement des recherches personnelles des auteurs de logiciels. Cette tendance a disparu il y a plusieurs années, au profit d'un engagement plus politique, visant à libérer le logiciel, et l'informaticien.

Ce combat est un bon combat, mais `OpenOffice.org` ne représente pas encore une avancée dans ce domaine. Vu par l'utilisateur, il n'affranchit pas des formats de fichiers propriétaires, et il n'offre pas vraiment d'autre avantage que la gratuité.

On pourra en retenir que pour quelques-uns qui, comme moi, n'ont pas l'usage d'outils type Office, mais qui une fois de temps en temps reçoivent un fichier Word ou Excel dont ils ne savent pas quoi faire, c'est un bon ersatz.

Pour les gens qui n'en ont pas vraiment besoin, ça peut suffire. Pour un usage professionnel, malheureusement, il n'y a toujours pas d'alternative convenable au monopole actuel.