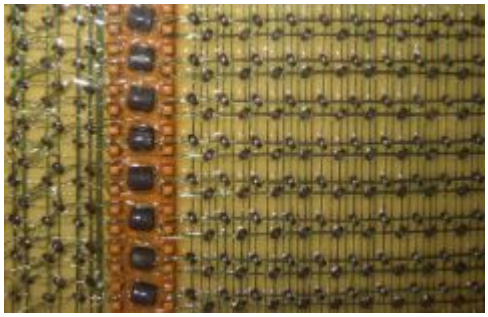
Développer en public



Voici, révélé sur son blog par Bradley M. Kuhn (cadre dirigeant à la FSF impliqué dans le projet GNU), l'un des meilleurs ingrédients pour la réussite d'un projet de logiciel libre : rendre son développement public, dès la conception. C'est du moins l'enseignement qu'il retire d'un échange qu'il a eu avec Loïc Dachary, pionnier du logiciel libre, fondateur de la FSF France, d'EUCD.info ou encore de la plateforme d'hébergement de projets libres Gna! [1].

Or, si l'idée peut sembler simple, sa réalisation est loin d'être anecdotique et porte à de nombreux débats, comme le synthétise ici Bradley. Mais surtout, ce qui se dessine au travers de cette réflexion, c'est un élément fondamental de la définition de ce qu'est un projet libre, un projet ouvert, un projet pérenne : c'est un projet dont on partage les idées et la conception, en plus des sources. Bradley Kuhn oppose, en filigrane, à cette vision, celle de grands projets (qui se clament bien souvent eux-même « OpenSource ») dont en effet seul le code est publié. Des projets opposant une résistance aux contributions extérieures, et qui peuvent alors presque paradoxalement sembler hermétiques, fermés…

Plusieurs exemples viennent aisément en tête car c'est entre autres l'un des principaux reproches fait à Chromium, la version « OpenSource » du navigateur Google Chrome. Mais LibreOffice, le récent fork d'OpenOffice, illustre parfaitement la conclusion de Bradley, quand la communauté d'un projet de logiciel libre finit par n'avoir d'autre choix que de partir sur un embranchement définitif des fichiers sources pour en ouvrir au public le développement, et plus le code seulement.

Où sont les octets ? [2]

Where Are The Bytes?

Bradley M. Kuhn – 11 juin 2010 – EBB.org
(Traduction Framalang : Barbidule, Loquemunaine, Goofy)

Il y a quelques années, j'avais envisagé de me lancer dans un projet de logiciel libre. Il n'a pas vu le jour, mais j'ai appris au passage des choses bonnes à savoir. Quand j'ai pensé à démarrer ce projet, j'ai fait comme à mon habitude : j'ai demandé à quelqu'un qui en savait plus que moi. J'ai donc téléphoné à Loïc Dachary, qui a initié de nombreux projets de logiciels libres, pour lui demander conseil.

Avant même que je puisse ne serait-ce qu'évoquer mon idée, Loïc m'a demandé : « Tu as une URL » ? J'étais abasourdi. Ben, je n'ai pas encore commencé, répondis-je. Bien sûr que si, reprit-il, puisque tu m'en parles c'est que tu as commencé. Le plus important, c'est que tu me dises où sont les octets.

Loïc m'expliqua ensuite que la plupart des projets échouent. Le plus difficile dans un projet de logiciel libre est de le pousser à un stade suffisamment avancé pour qu'il puisse survivre même si ses créateurs initiaux le quittent. Donc, selon la théorie de Loïc, la tâche la plus urgente à accomplir au démarrage d'un projet, c'est de générer ces octets, dans l'espoir qu'ils se fraieront un chemin jusqu'à une équipe de développeurs qui contribueront à maintenir le projet actif.

Mais qu'est-ce qu'il entend par « octets » au juste ? Il veut tout simplement dire que vous devez exposer vos réflexions, votre code, vos projets, vos idées, presque tout en fait sur une adresse publique où tout le monde pourra les voir. Montrez vos octets, montrez-les à chaque fois que vous en créez si peu que ce soit. C'est la seule chance de survie de votre projet de logiciel libre.

Le premier objectif d'un projet de logiciel libre est de

rassembler des développeurs. Aucun projet ne peut avoir de succès à long terme sans une base diversifiée de développeurs. Le problème c'est que le travail initial de développement et le planning du projet finissent trop souvent enfermés dans la tête d'un petit noyau de développeurs. C'est dans la nature humaine : comment puis-je passer mon temps à expliquer à chacun ce que je suis en train de faire ? Si je le fais, quand trouverai-je le temps de faire vraiment avancer les choses ? Ceux qui dirigent les projets de logiciels libres savent résister à ce désir naturel et font ce qui peut sembler contre-intuitif : ils exposent leurs octets publiquement, même si cela les ralentit un peu.

Ce processus est d'autant plus nécessaire à l'ère des réseaux. Si quelqu'un veut créer un programme qui remplisse sa mission, son premier outil est le moteur de recherche : il s'agit de savoir si quelqu'un d'autre a déjà fait le boulot. L'avenir de votre projet dépend entièrement du fait que chaque recherche de ce type aide des développeurs à découvrir vos octets.

Début 2001 j'ai demandé à Larry Wall quel était le projet le plus difficile parmi tous ceux sur lesquels il a travaillé. Sa réponse fut immédiate : "quand j'ai développé la première version de Perl5," m'a dit Larry, "j'avais l'impression que je devais coder tout seul et le faire tourner par mes propres moyens". Bien sûr, Larry est un gars tellement doué qu'il peut se permettre de créer à lui tout seul un programme que tout le monde voudra utiliser. Bien que je ne lui aie pas demandé ce qu'il ferait aujourd'hui dans une situation pareille, je devine – particulièrement quand on voit comment le développement de Perl6 est devenu public – qu'il utiliserait plutôt les nouveaux outils en ligne, tels que DVCS, pour montrer plus vite et plus souvent ses octets, et chercher à impliquer plus tôt davantage de développeurs[3].

Il est vrai que la priorité de la plupart des développeurs est de tout cacher. "On publiera quand ce sera prêt", ou bien –

pire encore – "le noyau dur de l'équipe travaille bien ensemble ; rendre le projet public maintenant ne ferait que nous ralentir". En vérité, c'est un mélange dangereux de peur et de narcissisme, exactement la même pulsion que celle qui pousse les développeurs de logiciels non-libres à les conserver propriétaires.

Les développeurs de logiciels libres ont la possibilité de dépasser la réalité fondamentale de tout développement logiciel : le code est mal fichu, et n'est généralement pas terminé. Malgré tout, il est essentiel que la communauté puisse voir ce qu'il se passe à chaque étape, dès le noyau initial du code et au-delà. Quand un projet est perçu comme actif, cela attire les développeurs et donne au projet une chance de succès.

Quand j'étais à la fac, une des équipes d'une classe de génie logiciel s'est complètement plantée. C'est arrivé alors même qu'un des membres de l'équipe avait passé près de la moitié du semestre à coder par lui-même, nuit et jour, sans se soucier des autres membres de l'équipe. Durant l'évaluation finale, le professeur lui fit remarquer : « un développeur de logiciel, ce n'est pas un pilote de chasse ». L'étudiant, ne voyant pas le rapport, plaisanta : « Ouais, je sais, au moins le pilote de chasse, il a un copilote ». En vérité, il ne suffira pas d'une personne ou deux, ou même d'une petite équipe, pour faire aboutir un projet de logiciel libre. Le projet ne marchera que s'il est soutenu par une communauté importante qui évitera tout point individuel de défaillance.

Il n'en reste pas moins que la plupart des projets de logiciels libres sont voués à l'échec. Cependant, il n'y a aucune honte à balancer quelques octets, pour inciter les gens à y jeter un oeil, quitte à laisser tomber si la mayonnaise ne prend pas. Toute la recherche scientifique fonctionne ainsi, et il n'y a pas de raison pour que l'informatique fasse exception. Garder un projet privé, c'est garantir son échec ; le seul intérêt, c'est que vous pouvez dissimuler le fait que

vous avez essayé. Comme le disait mon directeur de thèse lorsque je me faisais du souci quant à la réussite de ma recherche : un résultat négatif peut être aussi intéressant qu'un résultat positif. Ce qui est important, c'est d'être sûr que tous les résultats seront publiés et que le public pourra les examiner.

Quand j'ai commencé à parler de cette idée il y a quelques semaines, certains m'ont répondu que les premiers programmes GNU, les logiciels fondateurs de notre communauté ont d'abord été développé en privé. C'est vrai, mais le fait que les développeurs du projet GNU aient procédé de cette façon ne veut pas dire que c'est la bonne. Nous disposons désormais des outils pour faire facilement du développement en public, et nous devrions le faire. De mon point de vue, aujourd'hui nous ne sommes pas vraiment dans l'esprit du logiciel libre tant que le projet, y compris les discussions sur sa conception, les plans et les prototypes, ne sont pas développés publiquement. Le code (quelque soit sa licence) qui n'est que balancé à intervalles plus ou moins réguliers mérite d'être repris par une communauté qui rende son développement public.

Mise à jour (2010-06-12) : J'ai complètement oublié de parler de « The Risks of Distributed Version Control » par Ben Collins-Sussman, qui date de cinq ans maintenant mais qui est toujours d'actualité. Ben fait un constat similaire au mien, et remarque que certaines utilisations de DVCS peuvent avoir les effets que j'encourage les développeurs à éviter. Je pense que DVCS est comme n'importe quel autre outil : il peut être mal utilisé. Il faut éviter de s'en servir comme Ben le signale, et DVCS, lorsqu'il est utilisé correctement, aide dans le processus de développement public de logiciel.

Notes

[1] Fonctionnant tout comme GNU Savannah grâce au logiciel

Savane dont il est le principal développeur.

[2] Crédit photo : Steve Jurvetson (Creative Commons By). Cette photo, intitulée « mémoires primitives » est un gros plan sur une barrette de mémoire vive du milieu du siècle dernier. On y voit un quadrillage de « fins » fils de laiton, tricoté à la main avec de petits anneaux de ferrite à chaque intersection, le tout noyé dans de la colle. L'ensemble, qui pouvais occuper le volume d'un magazine papier, était capable de conserver plusieurs centaines de … bits de mémoire. Le circuit tricoté ici représente ainsi 38 octets de mémoire vive, et il est assez vertigineux de constater que 50 ans plus tard, on stocke environ 25 millions de fois plus d'information dans le volume de chaque anneau de ferrite.

[3] Notez que rendre son code public au milieu des années 1990 était plus difficile (d'un point de vue technologique) que maintenant. Ceux qui n'ont pas connu les archives shar ne s'en rendent pas compte.