Chouchoutez vos contributeurs et contributrices !

Le groupe Framalang a traduit l'article de Julien, qui a listé tous les moyens de se tirer une balle dans le pied quand on coordonne un projet libre.

Apprenez à les éviter !

Halte à la stratégie de l'échec !

Conduite de projets Open Source : 10 erreurs à éviter

Article original : https://julien.danjou.info/blog/2016/foss-projects-management-bad-practice

Auteur : Julien Danjou(CC-BY-SA)
Traduction : lyn., KoS, AlienSpoon, David 5.1, Simon, goofy, Slimane Aguercif, Fred, galadas, terhemis, gégé

Il y a quelques semaines, lors de l'*OpenStack Summit* (réunion annuelle des contributeurs à *OpenStack*, NDT), j'ai eu l'occasion de discuter de mon expérience de la conduite de projets *Open Source*. Je me suis rendu compte qu'après avoir fait partie de plusieurs communautés et beaucoup contribué pendant des années, je pourrais faire profiter les nouveaux venus de conseils et d'avis expérimentés.

Il existe une foule de ressources disponibles expliquant comment conduire un projet *Open Source*. Mais aujourd'hui, je voudrais aborder ce sujet sous un angle différent, en insistant sur ce qu'il convient de ne pas faire avec les personnes qui y participent. Je tire cette expérience des nombreux projets *Open Source* auxquels j'ai participé ces dernières années. Je vais décrire, sans ordre particulier, quelques mauvaises pratiques que j'ai constatées, en les illustrant par des exemples concrets.

1. Considérer les contributeurs comme une nuisance

Quand les informaticiens sont au travail, qu'ils soient chargés du développement ou de la maintenance d'un logiciel, il est une chose dont ils n'ont pas besoin : du travail supplémentaire. Pour la plupart d'entre eux, la première réaction à toute contribution externe est : « Zut, du travail en plus ». Et c'est effectivement le cas.

Par conséquent, certains développeurs essayent d'éviter ce travail supplémentaire : ils déclarent ne pas vouloir de contributions, ou font en sorte que les contributeurs se sentent indésirables. Cela peut prendre de nombreuses formes, comme les ignorer ou être désagréable avec eux. Ainsi, les développeurs évitent d'avoir à traiter le surplus de travail qu'on leur met sur le dos.

C'est une des plus grandes erreurs que l'on peut commettre, et une vision fausse de l'*Open Source*. Si des gens vous apportent du travail supplémentaire, faites tout ce que vous pouvez pour bien les accueillir afin qu'ils continuent à travailler avec vous. Il s'agit peut-être de ceux qui prendront votre relève d'ici peu. Préparez votre future retraite.

Prenons le cas de mon ami Gordon, que j'ai vu démarrer en tant que contributeur sur *Ceilometer* en 2013. Il examinait très bien le code, si bien qu'il me donnait en fait davantage de travail : non seulement en détectant les anomalies dans mes contributions, mais aussi en me faisant vérifier les siennes. Au lieu de m'en prendre à lui pour qu'il arrête de me faire retravailler mon code et vérifier ses correctifs, j'ai proposé qu'on lui fasse davantage confiance en l'ajoutant officiellement à l'équipe des correcteurs sur un des projets.

Et s’ils ne font pas cette première contribution, ils ne feront pas non plus la seconde. En fait, ils n’en feront aucune ; et ces projets perdraient alors leurs futurs correcteurs.

2. Ne confier aux autres que le sale boulot

Lorsque de nouveaux contributeurs se présentent pour travailler sur un certain projet, leurs motivations peuvent être très diverses. Certains sont des utilisateurs, d’autres veulent simplement voir ce que c’est de contribuer. Ressentir le frisson de la participation, comme un simple exercice ou avec la volonté d’apprendre afin de contribuer en retour à l’écosystème qu’ils utilisent.

En général, les développeurs confient le sale boulot à ces personnes, c’est à dire des tâches sans intérêt, à faible valeur ajoutée, et qui n’auront sans doute aucun impact direct sur le projet.

Pour certains, ce n’est pas grave, pour d’autres, ça l’est. Certains seront vexés de se voir confier du travail sans grand intérêt, alors que d’autres seront heureux de le faire pourvu qu’on leur témoigne de la reconnaissance. Soyez sensible à cela, et félicitez ces personnes. C’est la seule manière de les garder dans le projet.

3. Mépriser les petites contributions

Quand le premier patch d’un nouveau contributeur est une correction d’orthographe, qu’en pensent les développeurs ? Qu’ils s’en fichent, que vous gâchez de leur temps précieux avec votre petite contribution. Et personne ne s’intéresse à la perfection grammaticale d’une documentation, n’est-ce pas ?

C’est faux. Voyez mes premières contributions à *home-assistant* et *Postmodern* : j’ai corrigé des fautes d’orthographe dans la

documentation.

J’ai contribué au projet Org-mode pendant quelques années. Mon premier patch corrigeait simplement une chaîne de caractères du code. Ensuite, j’ai envoyé 56 patchs, corrigeant des bogues et ajoutant de nouvelles fonctionnalités élégantes, et j’ai aussi codé quelques extensions. À ce jour, je suis toujours seizième dans la liste des plus gros contributeurs d’*Org-mode*, qui en contient 390. Certainement pas ce qu’on appellerait un petit contributeur, donc. Je suis sûr que la communauté est bien contente de n’avoir pas méprisé ma première correction dans la documentation.

4. Mettre la barre trop haut pour les nouveaux arrivants.

Quand de nouveaux contributeurs arrivent, leurs connaissances du projet, de son contexte, et des technologies sont très variables. L’une des erreurs que les gens font le plus souvent consiste à demander aux nouveaux contributeurs des choses trop compliquées, dont ils ne pourront pas venir à bout. Cela leur fait peur (surtout les timides ou les introvertis) et ils risquent de disparaître, se croyant trop stupides pour aider.

Avant de faire le moindre commentaire, vous ne devriez avoir strictement aucun *a priori* sur leur niveau. Cela permet d'éviter ce genre de situations. Il faut également mettre beaucoup de délicatesse quand vous estimez les compétences des nouveaux, car certains pourraient se vexer si vous les sous-estimez trop.

Quand son niveau est bien estimé (un petit nombre d'échanges devrait suffire), vous devez former votre contributeur en ne le guidant ni trop ni trop peu, afin qu'il puisse s'épanouir. Il faut du temps et de l'expérience pour y parvenir, et vous allez probablement perdre certains contributeurs avant de bien maîtriser ce processus, mais c'est un chemin que tous ceux qui gèrent des projets doivent suivre.

Façonner ainsi les nouveaux arrivants est au cœur de la gestion de vos contributeurs, et ce quel que soit votre

projet. Et je suis quasiment sûr que cela s'applique à tout projet, même en dehors du logiciel libre.

5. Exiger des gens qu'ils fassent des sacrifices sur le plan personnel

C'est un point qui dépend beaucoup du projet et du contexte, mais il est très important. Dans le logiciel libre, où la plupart des gens vont contribuer par bonne volonté et parfois sur leur temps libre, vous ne devez surtout pas exiger d'eux qu'ils fassent des sacrifices personnels importants. Ça ne passera pas.

L'une des pires manières de faire cette erreur est de fixer un rendez-vous à l'autre bout du monde pour discuter du projet. Cela place les contributeurs qui vivent loin dans une situation injuste, car ils ne peuvent pas forcément quitter leur famille pour la semaine, prendre l'avion ou un autre moyen de transport, louer une chambre d'hôtel… Ce n'est pas bon : tout devrait être mis en œuvre pour que les contributeurs se sentent partie prenante du projet et intégrés à la communauté, sans que l'on exige cela de leur part. Entendons-nous bien, cela ne veut pas dire qu'il faut s'interdire toute rencontre ou activité sociale, au contraire. Pensez simplement à n'exclure personne lorsque vous discutez d'un projet.

Il en va de même pour les moyens de communication susceptibles de compliquer la vie des participants : se retrouver sur IRC (il est difficile pour certaines personnes de réserver une heure, surtout lorsqu'il faut tenir compte des différents fuseaux horaires), faire une visioconférence (en particulier quand on n'utilise pas de logiciel libre et gratuit), etc.

En gros, tout ce qui impose de se synchroniser en temps réel avec l'avancement du projet est contraignant pour certains contributeurs.

C’est pourquoi le meilleur moyen de communiquer reste le courriel, mais tous les outils de communication asynchrones (pisteurs de bogues, etc.) feront l’affaire dans la mesure où ils permettent à chacun de travailler à son rythme et à l’heure qui lui convient.

6. Ne pas avoir de code de conduite

À une époque où de plus en plus de communautés s’ouvrent à un public plus large que celui auquel elles étaient habituées (ce qui est fantastique), les codes de conduite semblent être un sujet à la mode, mais aussi délicat.

En réalité, toutes les communautés ont un code de conduite, qu’il soit inscrit noir sur blanc ou suivi inconsciemment par chacun. Sa forme dépend de la taille et de la culture de la communauté.

Cependant, en fonction de la taille de votre communauté et de la façon dont ce code s’applique, vous auriez peut-être intérêt à l’écrire dans un document, comme l’a par exemple fait Debian.

Ce n’est pas parce que vous aurez rédigé un code de conduite que tous les membres de votre communauté vont soudain se transformer en adorables Bisounours le suivant à la lettre, mais il fournit des règles que vous pouvez citer en cas de besoin. Il peut être utile de le transmettre à certains pour leur faire comprendre que leur comportement ne convient pas, et cela peut aider si une exclusion devient nécessaire, même s’il ne sert que rarement à cela, puisqu’en général personne ne veut aller aussi loin.

Je pense qu’on peut très bien se passer d’un tel document sur de petits projets. Mais vous devez garder à l’esprit qu’un code de conduite implicite découlera de votre comportement. La manière dont vos membres les plus influents communiqueront avec les autres installera l’ambiance à travers tout le

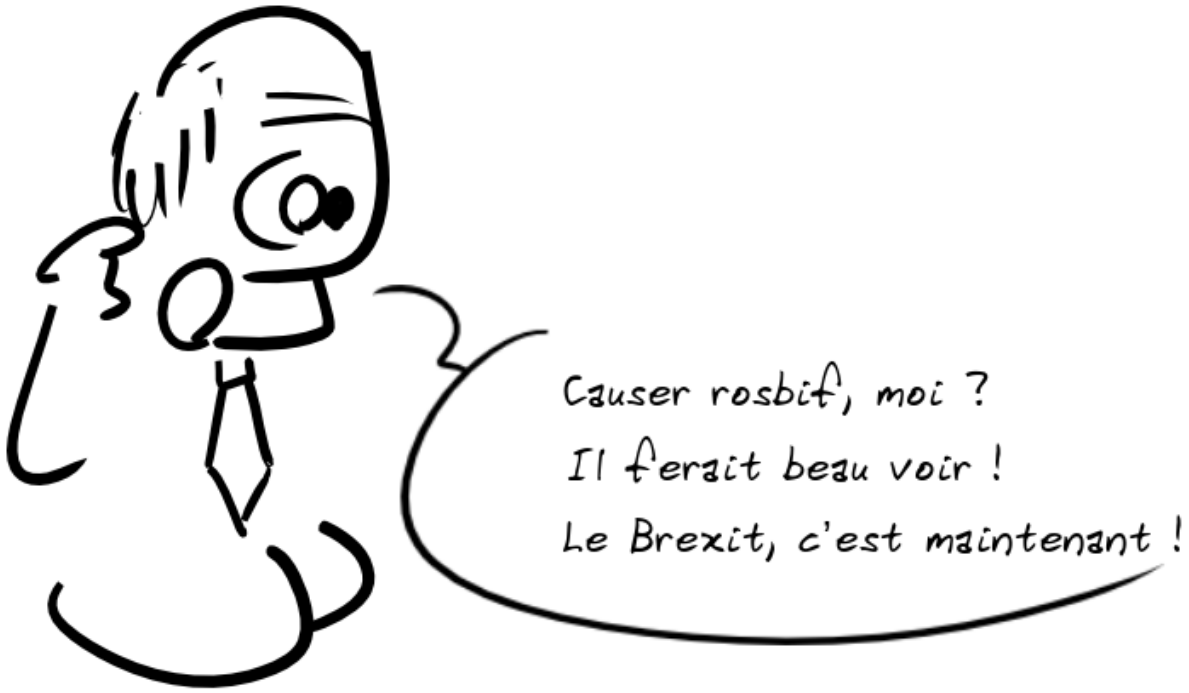
réseau. Ne sous-estimez pas cela.

Quand nous avons commencé le projet *Ceilometer*, nous avons suivi le code de conduite du projet *OpenStack* avant même qu'il ait été écrit, et probablement avons-nous même placé la barre un peu plus haut. En étant agréables, accueillants et ouverts d'esprit, nous avons réussi à obtenir une certaine mixité, avec plus de 25% de femmes dans notre équipe permanente – bien au dessus de la moyenne actuelle d'*OpenStack* et de la plupart des projets de logiciel libre !

7. Mettre les non-anglophones à l'écart

Il est important d'être conscient que la grande majorité des projets de logiciel libre utilisent l'anglais en tant que langue de communication principale. C'est logique. C'est une langue répandue, et qui semble remplir ce rôle correctement.

Mais une grande partie des codeurs n'ont pas l'anglais pour langue maternelle. Beaucoup ne le parlent pas couramment. Cela signifie que le rythme auquel ils peuvent converser peut-être très lent, ce qui peut en frustrer certains, notamment ceux qui sont nés en terre anglophone.



On peut observer ce phénomène dans les rencontres de codeurs, lors de conférences par exemple. Quand les gens débattent, il peut être très difficile pour certains d'expliquer leurs pensées en anglais et de communiquer à un rythme correct, ce qui ralentit la conversation et la transmission des idées. La pire chose qu'un anglophone puisse faire dans ce cas est de leur couper la parole, ou de les ignorer, pour la seule raison qu'ils ne parlent pas assez vite. Je comprends que cela puisse être frustrant, mais le problème n'est pas la façon de parler des non-anglophones mais les outils de communication utilisés qui ne mettent pas tout le monde au même niveau en privilégiant des conversations orales.

La même chose s'applique, à un moindre degré, aux rencontres sur IRC, qui sont relativement synchrones. Les médias complètement asynchrones ne pâtissent pas de ce défaut, et c'est pourquoi ils faudrait, à mon avis, les privilégier.

8. Pas de vision, aucune délégation des tâches

C’est une autre grosse erreur de gestion si le responsable ne parvient pas à gérer la croissance du projet alors que des gens sont disponibles et prêts à aider.

Évidemment, lorsque le flux des contributeurs commence à grossir, ajoutant de nouvelles fonctionnalités, demandant des retours et des instructions à suivre, certains responsables se retrouvent la tête sous l’eau et ne savent pas comment répondre. Ce qui a pour conséquences de frustrer les contributeurs, qui vont simplement partir.

Il est important d’avoir une vision de votre projet et de la communiquer. Dites clairement à vos contributeurs ce que vous voulez, et ne voulez pas, dans votre projet. Exprimer ces informations de manière claire (et non-agressive !) permet de minimiser les frictions entre vos contributeurs. Ils vont vite savoir s’ils veulent rejoindre ou non votre navire et quoi en attendre. Donc, soyez un bon capitaine.

S’ils choisissent de travailler avec vous et de contribuer, vous devez rapidement commencer à croire en eux et à leur déléguer certaines de vos responsabilités. Cela peut être n’importe quelle tâche que vous faites d’habitude vous-même : vérifier les patchs de certains sous-systèmes, traquer les bogues, écrire la documentation… Laissez les gens s’approprier entièrement une partie du projet car ils se sentiront responsables et ils y mettront autant de soin que vous. Faire le contraire en voulant tout contrôler vous-même est la meilleure façon de vous retrouver seul avec votre logiciel *open source*.

Aucun projet ne va gagner en taille et en popularité de cette manière.

En 2009, quand Uli Schlachter a envoyé son premier patch à

awesome, cela m'a donné plus de travail. J'ai du vérifier son patch, et j'étais déjà bien occupé pour sortir la nouvelle version d'*awesome* sur mon temps libre en dehors de mon travail ! Le travail d'Uli n'était pas parfait, et j'ai eu à gérer les bogues moi-même. Plus de travail. Alors qu'ai-je fait ? Quelques minutes plus tard, je lui ai répondu en lui envoyant un plan de ce qu'il devait faire et de ce que je pensais de son travail.

En retour, Uli envoya d'autres patchs et améliora le projet. Savez-vous ce que fait Uli aujourd'hui ? Il est responsable du gestionnaire des fenêtres *awesome* à ma place depuis 2010. J'ai réussi à transmettre ma vision, déléguer, puis à quitter le projet en le laissant dans de bonnes mains !

9. Ignorer certains types de contributions

Les gens contribuent de différentes manières, et pas toujours en codant. Il y a beaucoup de choses autour d'un projet de logiciel libre : la documentation, le tri de bogues, le support, la gestion de l'expérience utilisateur, la communication, les traductions…

Par exemple, il a fallu du temps pour que *Debian* songe à donner le statut de Développeur Debian à leurs traducteurs. *OpenStack* prend la même direction en essayant de reconnaître les contributions autres que techniques.

Dès lors que votre projet commence à récompenser certaines personnes et à créer différents statuts dans la communauté, vous devez faire très attention à n'oublier personne, car c'est le meilleur moyen de perdre des contributeurs en chemin.

10. Oublier d'être reconnaissant

Cette liste est le fruit de nombreuses années de bidouillages *open source* et de contributions à des logiciels libres.

L’expérience et le ressenti de chacun sont différents, et les mauvaises pratiques peuvent prendre différentes formes : si vous connaissez ou si vous avez vous-même rencontré d’autres obstacles dans vos contributions à des projets open-source, n’hésitez pas à compléter la liste dans les commentaires. C’est une forme de contribution.

Des ponts entre les hommes (Libres conseils 42/42)

Voici traduit en français le dernier épisode du recueil de conseils OpenAdvice, Notre expérience collaborative sur le pads de traduction s’est avérée un marathon (presque 42 kilomètres aussi) parcouru quelquefois au grand galop par une quasi-centaine de relayeurs. Merci à tous ceux qui sont passés parfois pour risquer un mot ou une phrase, parfois pour dévorer des paragraphes entiers, voire pour devenir des habitués indispensables d’une séance à l’autre.

L’aventure toutefois n’est pas terminée, des membres de Framalang vont maintenant réviser l’ensemble des traductions réunies sur la plateforme Booktype, et préparer un .PDF qui sera lui-même révisé avant le Framabook qui devrait paraître… quand il sera prêt.

Il est un peu tôt pour annoncer les dates avec certitude, mais Paris en mai et Bruxelles en juillet auront la primeur de la publication. Restez tunés !

* * * * *

Traduction Framalang : Ouve, Julius22, Sphinx, AC!63, jcr83, goofy, peupleLà, merlin8282, Sinma, tcit, SaSha_01, lamessen, Sky, jpierre03

Lancer des ponts

Shane Coughlan

Shane Coughlan est un expert en méthodes de communication et en développement d'affaires. Il est surtout connu pour créer des passerelles entre les différentes parties, commerciales et non commerciales, dans le secteur des technologies. Au plan professionnel il a œuvré avec succès pour la création d'un département juridique pour la Free Software Foundation Europe (FSFE), la principale organisation non gouvernementale pour la promotion du logiciel libre en Europe. Il a également contribué à l'établissement d'un réseau professionnel de plus de 270 experts juridiques et techniques à travers 27 pays, à la fondation d'un projet d'outil de conformité de code binaire. Il a travaillé à rapprocher intérêts privés et communautaires pour le lancement du premier examen critique d'une loi dédiée au logiciel libre et open source*. Shane a une connaissance étendue des technologies de l'Internet, des bonnes pratiques de gestion, de la construction de communautés et du logiciel libre et* open source*.*

Lorsque j'ai commencé à travailler dans le logiciel libre, j'ai été frappé par ce qui était perçu comme une différence entre les parties prenantes « communautaires » et « commerciales » dans ce domaine. Selon l'opinion largement admise à l'époque, les développeurs s'intéressaient au bidouillage du code et les commerciaux utilisaient leur travail de manière contestable s'ils n'étaient pas surveillés de près. C'était une supposition relativement infondée et presque entièrement défendue par des gens qui s'identifiaient

à la communauté plutôt qu'à ceux qui se préoccupaient davantage des intérêts commerciaux. Mais elle était très répandue.

Bien que je sois principalement associé à la partie communautaire des choses, j'ai résisté à l'idée selon laquelle il y aurait deux camps intrinsèquement ennemis se faisant face à propos de l'avenir du logiciel libre. Cela semblait trop simple pour décrire les dynamiques de contribution, d'utilisation et de support comme une interaction entre les nobles créateurs et les sournois téléchargeurs de gratuit. Cela semblait en effet plus être une situation dans laquelle la complexité, les changements et l'incertitude avaient conduit à la création de récits simplistes destinés à rassurer un peu ceux qui sortaient de leur position confortable. Je pouvais ressentir la tension ambiante, entendre les querelles autour des stands lors des rencontres et les commentaires acerbes ou bien les montées en pression dans les conférences. Mais que signifiait tout cela ?

Que nous parlions de contribution à des projets de logiciels libres, de gestion de projet ou de respect des licences, les relations entre les parties prenantes s'accompagnaient souvent de préjugés, d'un manque de communication et d'émotions négatives. Ceci conduisait ensuite à une plus grande complexité, à une augmentation proportionnelle de la difficulté à prendre des décisions unifiées ainsi qu'à résoudre les problèmes. Je savais que l'un des plus grands défis était de jeter des ponts entre les individus, les projets et les entreprises. C'est une étape nécessaire pour assurer une compréhension commune et une communication croisée des règles, des normes et des raisons qui sous-tendent les licences et autres mesures officielles qui régissent ce domaine. Mais le savoir ne suffit pas à trouver le moyen de s'attaquer efficacement au problème.

C'était la période charnière où la GPLv3 était en cours de rédaction. Les technologies fondées sur Linux commençaient à

apparaître dans toutes sortes de produits électroniques grand public et le logiciel libre était sur le point de se démocratiser. Il y avait du changement dans l'air et les investissements commerciaux autour des principaux projets de logiciel libre atteignaient des sommets. Tout à coup, on voyait des employés de grandes entreprises s'atteler à des tâches complexes, on avait des fonds importants pour les événements. De nombreux logiciels cessaient d'être une simple question de plaisir et commençaient à connaître les jalons, les livrables, l'assurance qualité et l'ergonomie.

Ce fut probablement un sérieux bouleversement pour ceux qui réalisaient du logiciel libre depuis longtemps : la majeure partie de l'évolution du logiciel libre ne concernait pas seulement l'exploration et la perfection technique, mais aussi l'interaction sociale. C'était un bon moyen pour des gens intelligents bien que parfois bizarres de partager un intérêt commun, de se lancer des défis et de coopérer au sein de projets bien définis et prévisibles. Tout comme collectionner des timbres, compter les trains ou connaître (par coeur) l'univers de Star Trek, c'était quelque chose qui fédérait des gens avec des centres d'intérêt bien particuliers en leur offrant de surcroît le bénéfice d'une agréable socialisation. Les premiers contributeurs ne s'attendaient sûrement pas à rencontrer là des cadres moyens et un développement orienté vers le volume de production. Pas étonnant que certains s'y soient cassé le nez.

Et pourtant… Tout s'est bien passé. Le logiciel libre est partout et sa position paraît être inexpugnable en tant que composant majeur de l'industrie des technologies de l'information. Des projets comme le noyau Linux ou le serveur Web Apache ont continué à se développer, à innover et à attirer de nouvelles parties prenantes, tant commerciales que non commerciales. L'équilibre des pouvoirs entre les individus, les projets et les entreprises a changé, parfois avec des conflits et des perturbations, mais jamais au

détriment d'une coopération sur le long terme, ni en portant préjudice aux valeurs fondamentales du logiciel libre.

De mon point de vue, dans le domaine juridique – qui n'est après tout qu'un langage formel qui fournit un contexte pour l'interaction au travers de règles mutuellement comprises et applicables – la tension dans le logiciel libre ne s'est pas formée avec l'introduction d'une activité commerciale accrue, ni avec la participation grandissante d'employés dans des projets, ni avec le changement lui-même. Le vrai problème réside dans l'écart entre une élite précédente en perte de vitesse et de nouveaux venus parfois très différents.

Le défi était de créer un terrain de jeu équitable où les différents intérêts pourraient coexister dans un respect mutuel. Le logiciel libre devait devenir un point de rencontre où n'importe qui pouvait à tout moment obtenir des informations comme les compétences appropriées, les obligations d'une licence ou les pré-requis pour la soumission de code à un projet. La subjectivité et l'imprécision devaient être mises de côté pour permettre l'émergence de transactions plus formelles, ce qui agit alors comme le précurseur essentiel d'une forte activité économique, en particulier dans le contexte d'une communauté internationale voire mondiale.

Ce qui a fonctionné dans les premiers jours – qu'il s'agisse de la confiance de quelques-uns ou de l'entente mutuelle d'un groupe homogène ayant des intérêts communs – ne pouvait plus agir en tant que facteur social ou économique pour l'avenir du domaine. Parfois, il semblait que c'était une barrière insurmontable et que les tensions entre les contributeurs de longue date au logiciel libre et les nouveaux acteurs devaient conduire à un effondrement de la coopération et, peut-être, à celui des progrès accomplis. Mais un résultat aussi sombre supposait des conditions qui n'existaient tout simplement pas.

Le logiciel libre a apporté beaucoup de choses à des tas de personnes et d'organisations, en se fondant sur quelques

concepts très simples comme la liberté d'utiliser, de modifier, d'améliorer et de partager la technologie. Ces concepts ont permis beaucoup de flexibilité. Et tant que les gens reconnaissaient leur valeur et continuaient à les respecter, les conflits portants sur des points secondaires comme la gouvernance ou les zones d'ombre des licences étaient plutôt sans importance – à long terme. Le reste est principalement du bruit. Les communications habituelles sont dérangées par tous ces pièges dramatiques qui arrivent inévitablement lorsqu'un groupe social est rejoint par un autre. Cela s'applique à toutes les discussions, que l'on parle d'un coin de pêche, d'un pays accueillant (ou non) les immigrants ou de deux entreprises fusionnant.

Les changements au sein du logiciel libre semblaient tous un peu confus à l'époque, mais ils se décomposent principalement en trois leçons utiles qui seront familières aux étudiants d'histoire ou de sciences politiques. Premièrement, chaque fois qu'il existe une élite, elle cherchera à préserver son statut et parlera du conflit perçu comme une évolution négative dans le but de l'ébranler. Deuxièmement, malgré la tendance inhérente de chaque base de pouvoir à être conservatrice, la participation statique dans un domaine en évolution aura pour seul effet de déplacer la possibilité d'une amélioration des parties existantes vers de tierces parties. Enfin, si quelque chose a de la valeur, alors les difficultés rencontrées en matière de gouvernance sont peu susceptibles de porter atteinte à cette valeur, mais fourniront au contraire une méthode pour redéfinir à la fois les mécanismes de gouvernance et les personnes en mesure de les appliquer.

Le développement du logiciel libre en tant que technologie démocratisée a connu une professionnalisation croissante parmi les développeurs comme dans la gestion de projets. Nous avons aussi vu s'accroître le respect envers les licences de la part des individus, des projets et des entreprises. Ce ne fut pas

une mauvaise chose, et malgré quelques moments houleux le long du chemin – que l'on peut choisir parmi les querelles inter-communautaires, les entreprises qui ne tiennent pas compte des termes des licences ou l'agacement causé par un éloignement de l'esprit décontracté habituel – notre position en est consolidée, plus cohérente et de plus grande valeur.

Le juriste et le logiciel libre (Libres conseils 41/42)

Traduction : Framalang d'après une première traduction effectuée par **Liu qihao** (aka Eastwind) qui remercie François van der Mensbrugghe, Catherine Philippe et Ciarán O'Riordan.

Quelques considérations sur le rôle du juriste dans le domaine du logiciel libre et *open source*

Till Jaeger

Le docteur Till Jaeger est collaborateur du cabinet d'avocats JBB Rechtsanwaelte depuis 2001. Avocat diplômé et spécialisé dans le domaine du droit d'auteur et du droit des médias, il conseille autant les grandes et moyennes entreprises du secteur des technologies de l'information que les institutions gouvernementales et les développeurs de logiciels sur des sujets impliquant contrats, licences et usage en ligne. Son travail est orienté vers les contentieux relatifs aux logiciels libres et open source*. Il est co-fondateur de l'institut pour l'étude juridique du logiciel libre et* open source *(ifrOSS). En outre, il aide développeurs et éditeurs de*

logiciels dans le processus de mise en compatibilité et en conformité de leurs licences libres. Till a représenté le projet gpl-violations.org dans plusieurs procès (NdT : devant les juridictions allemandes) dont l'objet portait sur le respect de la GPL. Il a également publié divers articles et livres à propos de questions juridiques sur les logiciels libres et open source*. Il a été membre du comité C lors de l'élaboration de la GPLv3.*

Pour commencer, clarifions une chose : je ne suis pas un geek. Je ne l'ai jamais été, et je n'ai aucune intention de le devenir. En revanche, je suis juriste. La plupart des lecteurs de ce livre auront probablement tendance à éprouver une plus grande sympathie à l'égard des geeks qu'envers les juristes. Cependant, je ne souhaite pas éluder ce fait : la communauté du logiciel libre et *open source* n'est pas nécessairement passionnée par les juristes, elle est trop occupée à développer du code. Cela, je le savais déjà au début de l'année 1999, lorsque nos chemins se sont croisés pour la première fois. Néanmoins, d'autres éléments m'étaient, à ce moment-là, encore inconnus. En 1999, tandis que je terminais ma thèse de doctorat portant sur le droit d'auteur classique, j'évaluais l'étendue des droits moraux. Dans ce contexte, j'ai passé un certain temps à réfléchir à la question suivante : comment les droits moraux des développeurs sont-ils protégés par la licence GPL, étant donné que celle-ci confère aux utilisateurs un droit de modification sur leurs logiciels ? C'est ainsi que je suis entré pour la première fois en contact avec le logiciel libre et *open source*.

À cette époque, les qualificatifs « libre » et « ouvert » avaient évidemment des significations différentes. Mais dans le monde dans lequel je vivais, cette distinction ne méritait pas d'être débattue. Cependant, vu que j'étais libre d'étudier ce qui m'intéressait et ouvert à l'exploration de nouvelles questions sur le droit d'auteur, j'ai rapidement découvert que les deux termes ont quelque chose en commun : bien qu'ils

soient effectivement différents, ils sont bien mieux utilisés lorsqu'ils sont ensemble…

Voici trois choses que j'aurais souhaité savoir à l'époque :

Tout d'abord, que mes connaissances techniques, en particulier dans le domaine du logiciel, étaient insuffisantes. Ensuite, que je ne comprenais pas véritablement la communauté et que j'ignorais ce qui importait aux yeux de ses membres. Et cerise sur le gâteau, que je ne connaissais pas grand-chose aux juridictions étrangères, à l'époque. Ces notions m'auraient été précieuses si j'avais pu les aborder dès le départ.

Depuis, j'ai appris suffisamment et, à l'instar de la communauté qui se réjouit de partager ses réalisations, je suis heureux de partager mes leçons (1).

Connaissances techniques

Comment est formée une architecture logicielle ? À quoi ressemble la structure technique d'un logiciel ? Quelles sont les licences compatibles ou incompatibles entre elles ? Comment et pourquoi ? Quelle est la structure du noyau Linux ? Pour citer un exemple, la question essentielle des éléments constitutifs d'une « œuvre dérivée » selon la GPL détermine la manière dont le logiciel pourra être licencié. Tout élément rentrant dans le champ d'une œuvre dérivée d'un logiciel originaire sous licence GPL doit être redistribué selon les termes de cette dernière. Pour évaluer si un programme constitue une « œuvre dérivée » ou pas, il est nécessaire d'avoir au préalable une compréhension technique approfondie. Ainsi, l'interaction des modules de programmes, des liaisons, des IPC (Communications inter-processus), des greffons, des infrastructures technologiques, des fichiers d'en-tête, etc. détermine au niveau formel (parmi d'autres critères) le degré de connexité d'un logiciel par rapport à un autre, ce qui aide à le qualifier ou non d'œuvre dérivée.

Connaissance de l’industrie et de la communauté

Au-delà de ces questions fonctionnelles, l’étendue de mes connaissances des principes régissant le libre était limitée, tant au regard de la motivation des développeurs que des entreprises utilisant du logiciel libre. En outre, je ne connaissais pas son arrière-plan philosophique, et n’étais pas plus familier avec les modalités pratiques d’interactions sociologiques de la communauté. Ainsi, les questions : « Qui est mainteneur ? » ou « Quel est le fonctionnement d’un système de contrôle de version ? » ne trouvaient pas d’écho à mes oreilles. Or, pour servir du mieux possible vos clients, ces questions sont toutes aussi importantes que la maitrise des aspects d’ordre purement technique. Par exemple, nos clients nous demandent de nous occuper du côté juridique des modèles économiques construits sur une double licence de type « *open core* ». Ceci inclut la gestion des contrats de supports, de services, de développements ainsi que les conventions applicables aux codes sources venant des contributions. Ce faisant, nous guidons entreprises et institutions dans la grande réserve du logiciel libre lors de la mise en place de ces modèles.

D’autre part, nous conseillons aussi les développeurs sur la manière de régler les litiges nés des violations de leur droit d’auteur, notamment via l’élaboration et la négociation de contrats en leur nom et pour leur compte. Ceci étant, pour répondre à tous ces besoins de manière complète, il est fondamental de s’être familiarisé avec cette multiplicité de points de vue.

Connaissances en droit comparé

La troisième chose dont un juriste libriste a besoin, c’est de connaissances à propos des juridictions étrangères, au moins

quelques-unes et, plus il en acquiert, mieux il se porte. Pour pouvoir interpréter les différentes licences correctement, il est essentiel de comprendre l'état d'esprit dans lequel s'inscrivaient les personnes qui les ont écrites.

Dans la plupart des cas, le système juridique américain est d'une importance capitale. Par exemple, lors de l'élaboration de la GPL, celle-ci a été écrite avec, à l'esprit, des notions issues de la *common law* étasunienne. Aux États-Unis le terme « distribution » inclut la distribution en ligne. Or, le système de droit d'auteur allemand établit un distinguo entre la distribution en ligne et hors-ligne. Dès lors, les licences qui ont été rédigées par des juristes de la *common law* étasunienne peuvent être interprétées comme incluant la distribution en ligne. Au cours d'un procès, cet argument peut devenir particulièrement pertinent (2).

Un apprentissage permanent

Au final, toutes ces connaissances sont d'une grande utilité. Aussi, j'espère qu'à l'image du processus d'évolution d'un logiciel, qui apporte son lot de solutions aux besoins de tous les jours, mon esprit continuera à répondre aux défis que la vibrante communauté du logiciel libre et *open source* pose constamment à l'attention d'un juriste.

(1) L'*Institut für Rechtsfragen der Freien und Open Source Software* (institut des questions de droit sur les logiciels libres et *open source*) propose, entre autres, une collection d'ouvrages et de jurisprudences en lien avec les logiciels libres et *open source* ; pour plus de détails, voir sur le site http://www.ifross.org/.

(2) Voir : http://www.ifross.org/Fremdartikel/LGMuenchenUrteil.pdf, Cf. Welte v.Skype, 2007

Les réalités économiques du logiciel libre (Libres conseils 40/42)

*** Aujourd'hui 28 mars, dernière séance de traduction collaborative sur ce projet avec l'épisode n°42 !**

Traduction Framalang : tcit, Julius22, Sphinx, goofy, peupleLà, merlin8282, lamessen, BAud, Jej, Alpha

Modèles économiques basés sur le libre et l'*open source*

Carlo Daffara

*Carlo Daffara est chercheur dans le domaine des modèles économiques basés sur l'*open source*, le développement collaboratif d'objets numériques et l'utilisation de logiciels* open source *dans les entreprises. Il fait partie du comité éditorial de relecture du journal international des logiciels et processus* open source *(International Journal of Open Source Software & Processes : IJOSSP), est membre du comité technique de deux centres régionaux de compétences* open source *et est également membre du réseau juridique européen FSFE (fondation européenne pour le logiciel libre). Il a pris part aux comités SC34 et JTC1 pour la branche italienne de l'ISO, UNINFO et a travaillé au sein du groupe de travail de la société Internet du logiciel public (Internet Society Public Software) ainsi que pour beaucoup d'autres initiatives liées à la normalisation.*

*Auparavant, Carlo Daffara était le représentant italien dans le groupe de travail européen sur le logiciel libre, la première initiative de l'Union européenne afin de soutenir l'*open source *et le logiciel libre. Il a présidé le groupe de travail SME du groupe d'étude de l'UE sur la compétitivité et le groupe de travail IEEE des intergiciels open source du comité technique sur le calcul évolutif. Il a travaillé en tant qu'examinateur du projet pour la commission Européenne dans le domaine de la coopération internationale, l'ingénierie logicielle, l'*open source *et les systèmes distribués et a été directeur de recherche dans plusieurs projets de recherche de l'Union européenne.*

Introduction

« Comment gagner de l'argent avec le logiciel libre ? » était une question très courante, il y a encore seulement quelques années. Désormais, cette question s'est transformée en « Quelles sont les stratégies commerciales pouvant être mises en œuvre en se basant sur le logiciel libre et *open source* ? ». Cette question n'est pas aussi gratuite qu'elle peut paraître, puisque de nombreux chercheurs universitaires écrivent encore ce genre de textes : « le logiciel *open source* est délibérément développé hors de tout mécanisme de marché… il échoue à contribuer à la création de valeur aux développements, contrairement au marché du logiciel commercial… il ne génère pas de profit, de revenus, d'emplois ou de taxes…

Les licences *open source* sur les logiciels visent à supprimer les droits d'auteurs sur le logiciel et empêchent d'établir un prix pour le logiciel. Au final, les logiciels développés ne peuvent être utilisés pour générer des profits. » [Koot 03] ou [Eng 10] indiquent que « des économistes ont montré que les collaborations *open source* dans le monde réel s'appuient sur plusieurs incitations différentes telles qu'enseigner, se démarquer et se créer une réputation » (sans parler des

incitations économiques). Cette vue purement « sociale » du logiciel libre et *open source* est partiale et fausse. Et nous démontrerons qu'il y a des raisons économiques liées au succès des métiers du libre et de l'*open source* qui vont au-delà des collaborations purement bénévoles.

Le logiciel libre et *open source* face aux réalités économiques

Dans la plupart des domaines, l'utilisation d'un logiciel libre et *open source* apporte un avantage économique substantiel, grâce aux développements partagés et aux coûts de maintenance, déjà décrits par des chercheurs comme Gosh, qui a estimé une réduction de coût de 36 % en R&D (« Recherche et Développement », NdT). La vaste part de marché des déploiements « internes » de logiciels libres et *open source* explique pourquoi certains des bénéfices économiques ne sont pas directement visibles sur le marché des services commerciaux.

L'étude FLOSSIMPACT a montré, en 2006, que les entreprises qui contribuent au code de projets de logiciels libres et *open source* ont, au total, au moins 570 000 employés et un chiffre d'affaires annuel de 263 milliards d'euros [Gosh 06], faisant ainsi du logiciel libre et *open source* l'un des phénomènes les plus importants des NTIC. Il est important aussi de reconnaître qu'un pourcentage non négligeable de cette valeur économique n'est pas directement perceptible du marché, vu que la majorité du logiciel n'est pas développée dans l'intention de le vendre (le soi-disant logiciel « prêt à l'emploi ») mais uniquement à usage interne. Comme le réseau thématique FISTERA EU l'a identifié, en réalité, la majorité du logiciel est développée seulement pour un usage interne.

Région	Licences de logiciels propriétaires	Services logiciel (développement personnalisation)	Développement interne
Union européenne	19 %	52 %	29 %
États-Unis	16 %	41 %	43 %
Japon	N/A	N/A	32 %

Il est clair que ce qui est appelé « le marché logiciel » est en réalité bien plus réduit que le vrai marché du logiciel et des services et que 80 % restent invisibles. Nous verrons que le FLOSS tient une place économique importante de ce marché, directement grâce à ce modèle de développement interne.

Modèles économiques et proposition de valorisation

L'idée de base d'un modèle économique est assez simple : j'ai quelque chose ou je peux faire quelque chose (la « proposition de valeur ») et c'est plus rentable de me payer ou d'obtenir ce quelque chose plutôt que de le faire soi-même (il est même parfois impossible de trouver des alternatives, comme dans le cas de monopoles naturels ou créés par l'homme, et l'idée même de le produire par soi-même n'est pas envisageable). Il y a deux sources possibles de valeur : une propriété (quelque chose qui peut être échangé) et l'efficacité (quelque chose propre à ce que fait une entreprise et la manière dont elle le fait).

Avec l'*open source*, la « propriété » est généralement non exclusive (à l'exception de ce qui est nommé « cœur ouvert », où une partie du code n'est pas libre du tout et cela sera abordé plus loin dans cet article). D'autres exemples de propriété concernent le droit des marques, les brevets, les licences… tout ce qui peut être transféré à une autre entité par contrat ou par une transaction légale. L'efficacité est la

capacité à effectuer une action avec un coût moindre (qu'il soit tangible ou intangible) et cela correspond à la spécialisation dans un domaine d'application ou apparaît grâce à une nouvelle technologie.

Pour le premier cas, les exemples sont simplement la réduction du temps nécessaire pour réaliser une action quand vous augmentez votre expertise concernant ce sujet. La première fois que vous installez un système complexe, cela peut demander beaucoup d'efforts et cet effort diminue d'autant plus que vous connaissez les tâches nécessaires pour réaliser l'installation elle-même. Pour le second, cela peut être l'apparition d'outils qui simplifient le processus (par exemple, avec le clonage d'images) et introduisent une importante rupture, un « saut » dans la courbe efficacité-temps.

Ces deux aspects sont la base de tout modèle économique que nous avons analysé par le passé ; il est possible de montrer que tout ceux-ci échouent afin de garantir une continuité entre les propriétés et l'efficacité.

Parmi les résultats de notre précédent projet de recherche, nous avons trouvé que les projets basés sur un modèle propriétaire ont tendance à obtenir moins de contributions extérieures car cela nécessite une opération juridique pour faire partie des propriétés de l'entreprise, pensez par exemple aux licences doubles : afin que son code fasse partie du code du produit, un contributeur extérieur doit signer l'abandon des droits sur son code afin que l'entreprise puisse vendre la version commerciale ainsi que la version *open source*.

D'un autre côté, les modèles totalement orientés sur l'efficacité ont tendance à avoir plus de contributions et de visibilité mais des résultats financiers plus faibles. Je l'ai écrit plusieurs fois : il n'y a pas de modèle économique idéal mais un éventail de modèles possibles et les entreprises

devraient s’adapter elles-mêmes pour changer les conditions du marché et aussi adapter leur modèle. Certaines entreprises débutent par des modèles entièrement axés sur l’efficacité puis construisent, avec le temps, une propriété en interne, d’autres ont commencé avec un modèle orienté vers la propriété et ont évolué différemment pour augmenter les contributions et réduire les efforts d’ingénierie (ou développer la base d’utilisateurs afin de créer d’autres moyens d’avoir un retour financier grâce aux utilisateurs).

Une typologie des modèles économiques

L’étude EU FLOSSMETRICS des modèles économiques basés sur le logiciel libre a identifié, après analyse de plus de 200 entreprises, une taxonomie des principaux modèles économiques utilisés par les entreprises *open source* ; les principaux modèles identifiés sur le marché sont :

- la double licence : le même code source logiciel distribué sous GPL et sous une licence propriétaire. Ce modèle est principalement utilisé par les producteurs de logiciel et outils pour développeurs et fonctionne grâce à une forte association de la GPL, qui requiert que les travaux dérivés et logiciels liés directement soient distribués sous la même licence. Les entreprises ne souhaitant pas distribuer leur propre logiciel sous GPL peuvent obtenir une licence propriétaire leur octroyant une exemption des conditions de la GPL, ce qui semble souhaitable à certains. L’inconvénient de cette licence double est que les contributeurs externes doivent accepter des conditions similaires et cela a révélé des réductions de contributions externes, se limitant à des corrections de bogues et des ajouts mineurs ;
- le modèle « cœur ouvert » (précédemment appelé « valeur ajoutée propriétaire » ou « séparation entre libre et

propriétaire » * ? *\) : ce modèle se distingue entre un logiciel libre basique et une version propriétaire, basée sur la version libre mais avec l'ajout de greffons propriétaires. La plupart des entreprises qui suivent un tel modèle adoptent la Mozilla Public Licence, car elle permet explicitement cette forme de mélange et permet une plus grande participation des contributions externes sans les mêmes contraintes de consolidation du droit d'auteur comme dans l'usage de doubles licences. Ce modèle a l'inconvénient intrinsèque que le logiciel libre doit être de grande valeur pour être attractif pour les utilisateurs, i.e. il ne doit pas être réduit à une version aux possibilités limitées, tout comme, dans le même temps, il ne doit pas « cannibaliser » le produit propriétaire. Cet équilibre est difficile à atteindre et à maintenir dans la durée ; en outre, si le logiciel est de grand intérêt, les développeurs peuvent essayer d'apporter les fonctionnalités manquantes dans le logiciel libre, réduisant ainsi l'intérêt de la version propriétaire et donnant potentiellement naissance à un logiciel concurrent entièrement libre qui ne souffrira pas des mêmes limitations ;

- les experts produits : des entreprises qui ont créé ou maintiennent un projet logiciel spécifique et utilisent une licence libre pour le distribuer. Les principaux revenus viennent du service, comme la formation ou l'expertise, et suivent la classification EUWG d'origine « le meilleur code vient d'ici » et « les meilleures compétences sont ici » [DB 00]. Cela conforte l'impression, courante, que les experts les plus compétents sur un logiciel sont ceux qui l'ont développé et qu'ils peuvent ainsi fournir des services au prix d'un démarchage minimal, s'appuyant sur la fourniture gratuite du code. L'inconvénient de ce modèle est que le coût d'entrée pour des concurrents potentiels est faible, vu que le seul investissement nécessaire est l'acquisition des compétences sur le logiciel lui-même ;

- les fournisseurs de plateforme :des entreprises qui apportent un ensemble de services, avec support et intégration de certains projets, constituant une plateforme cohérente et testée. En ce sens, même les distributions GNU/Linux sont classées en tant que plateforme ; une observation intéressante est que ces distributions sont distribuées en grande partie sous licence libre pour maximiser les contributions externes et s'appuyer sur la protection du droit d'auteur pour empêcher la copie sauvage sans empêcher les « déclinaisons » (suppression des particularités soumises à droit d'auteur comme les logos ou droit des marques, pour créer un nouveau produit). Des exemples de clones de Red Hat sont CentOS et Oracle Linux. La valeur ajoutée provient d'une qualité garantie, de la stabilité et de la fiabilité ainsi que d'une garantie de support pour les applications métier critiques ;
- les entreprises de conseil et de recrutement : les entreprises de cette catégorie ne font pas vraiment de développement mais fournissent des conseils de sélection et des services d'évaluation pour un large éventail de projets, d'une manière qui est proche du rôle de l'analyste. Ces entreprises ont tendance à avoir un impact très limité sur les communautés car les résultats de l'évaluation et du processus d'évaluation sont généralement des données propriétaires ;
- les fournisseurs de support global : des entreprises qui proposent un support centralisé pour un ensemble de produits de logiciel libre, généralement en employant directement les développeurs ou en remontant les demandes de support ;
- la validation juridique et l'expertise : ces entreprises n'apportent pas de développements de code source mais fournissent une aide à la vérification de conformité aux licences, parfois en apportant une garantie et une assurance contre les attaques juridiques ; certaines entreprises utilisent des outils pour assurer que le

code n’est pas réutilisé ;

- la formation et la documentation : des entreprises qui proposent de la formation, en ligne et en présentiel, des documentations et des manuels supplémentaires. Cela est généralement fourni dans le cadre d’un contrat de support, mais, récemment, quelques réseaux de centres de formation ont lancé des cours orientés spécifiquement vers le logiciel libre ;
- le partage des coûts de R&D : une entreprise ou une société peut avoir besoin d’une nouvelle version ou d’une amélioration d’un paquet logiciel et financer un consultant ou un développeur pour réaliser le travail. Plus tard, le logiciel développé est redistribué en *open source* pour bénéficier de l’ensemble des développeurs expérimentés pouvant le déboguer et l’améliorer. Un bon exemple est la plateforme Maemo, utilisée par Nokia pour ses smartphones (comme le N810) ; au sein de Maemo, seul 7,5 % du code est propriétaire, apportant une réduction des coûts estimée à 228 millions de dollars (et une réduction du temps de mise sur le marché d’un an). Un autre exemple est l’écosystème Eclipse, un environnement de développement intégré (EDI) distribué à l’origine par IBM comme logiciel libre puis ensuite géré par la fondation Eclipse. De nombreuses entreprises ont choisi Eclipse comme socle pour leur produit et ont ainsi réduit le coût global pour la création d’un logiciel fournissant une fonctionnalité pour les développeurs. Il y a un grand nombre d’entreprises, d’universités et de personnes qui participent à l’écosystème Eclipse. Comme récemment constaté, IBM contribue aux alentours de 46 % au projet, les contributeurs à titre personnel représentant 25 % et un grand nombre d’entreprises comme Oracle, Borland, Actuate et de nombreuses autres ayant des participations allant de 1 à 7 %. Ceci est semblable aux résultats obtenus grâce à l’analyse du noyau Linux et qui montre que, lorsqu’il y a un écosystème sain et de grande taille, le partage des tâches réduit de

manière significative les coûts de maintenance, dans [Gosh 06], on estime qu'il est possible de faire des économies de l'ordre de 36 % dans la recherche et la conception logicielle grâce à l'utilisation du logiciel libre, ces économies constituent en elles-mêmes le plus gros « marché » réel pour le logiciel libre, ce qui est démontré par le fait qu'au moins une partie du code des développeurs est basé sur du logiciel libre (56,2 % comme mentionné dans [ED 05]). Un autre excellent exemple de « coopétition » inter-entreprises est le projet WebKit, le moteur de rendu HTML à la base du navigateur Google Chrome ainsi que d'Apple Safari et qui est utilisé dans la majorité des appareils mobiles. Dans ce projet, après un délai initial d'un an, le nombre de contributions externes a commencé à devenir significatif et, après un an et demi, il surpasse largement les contributions d'Apple – réduisant de fait les coûts de maintenance et d'ingénierie grâce à la répartition des tâches entre les co-développeurs ;

- les revenus indirects : une entreprise peut choisir de financer des projets de logiciel libre si ces projets peuvent créer une source de revenus importante pour des produits dérivés, non liés directement au code source ou au logiciel. L'un des cas les plus courants correspond à l'écriture de logiciel nécessaire au fonctionnement de matériel, par exemple, les pilotes d'un système d'exploitation pour un matériel spécifique. En fait, de nombreux fabricants de matériel distribuent déjà gratuitement leurs pilotes logiciels. Certains d'entre eux distribuent déjà certains de leurs pilotes (surtout ceux pour le noyau Linux) sous une licence libre. Le modèle du produit d'appel est une stratégie commerciale traditionnelle, répandue même à l'extérieur du monde du logiciel : dans ce modèle, les efforts sont consacrés à un projet de logiciel libre et *open source* afin de créer ou d'étendre un autre marché dans des conditions différentes. Par exemple, les fournisseurs de composants

matériels investissent dans le développement de pilotes logiciels pour des systèmes d'exploitation *open source* (comme GNU/Linux) pour s'étendre sur le marché spécifique des composants. D'autres modèles de revenus auxiliaires sont ceux, par exemple, de la fondation Mozilla qui réunit une somme d'argent non négligeable grâce à un partenariat avec Google sur le moteur de recherche (estimé à 72 millions de dollars en 2006), tandis que SourceForge/OSTG est financé en majorité par les recettes des ventes en ligne du site partenaire ThinkGeek.

Certaines entreprises ont plus d'un modèle principal et sont, par conséquent, comptées en double ; notamment, la plupart des entreprises pratiquant une licence double vendent aussi du service de support. En outre, les experts d'un produit ne sont comptés que s'ils ont une partie visible de leur entreprise qui contribue au projet en tant que « commiter principal ». Autrement, le nombre d'experts serait bien plus élevé, du fait que certains projets sont au cœur du support commercial de nombreuses entreprises (de bons exemples sont OpenBravo et Zope).

Il faut aussi tenir compte du fait que les fournisseurs de plateforme, même s'ils sont limités en nombre, tendent à avoir des taux de facturation plus élevés que les experts ou que les entreprises à cœur ouvert. De nombreux chercheurs essaient d'identifier s'il y a un modèle plus « efficace » parmi ceux pris en compte ; ce que nous avons trouvé est que le futur le plus probable sera l'évolution d'un modèle à l'autre, avec une consolidation sur le long terme des consortiums de développement (comme les fondations Eclipse et Apache) qui fournissent une forte infrastructure légale et des avantages de développement ainsi que des spécialistes apportant des offres verticales pour des marchés spécifiques.

Conclusion

Le logiciel libre et *open source* permet non seulement une présence pérenne, et même très large, sur le marché (Red Hat est déjà proche du milliard de dollars de revenus annuels), mais aussi plusieurs modèles différents qui sont totalement impossibles avec le logiciel propriétaire. Le fait que le logiciel libre et *open source* est un bien non concurrent facilite aussi la coopération entre entreprises, tant pour accroître sa présence mondiale et pour signer des contrats à grande échelle pouvant demander des compétences multiples que sur le plan géographique (même produit ou service, région géographique différente) ; « verticalité » (entre produits) ou « horizontalité » (des domaines d'application). Cet adjuvant à créer de nouveaux écosystèmes est l'une des raisons expliquant que le logiciel libre et open source fait partie intégrante de la plupart des infrastructures informatiques dans le monde, enrichissant et aidant les entreprises et administrations publiques à réduire leurs coûts et à collaborer pour de meilleurs logiciels.

Bibliographie

- *[DB00] Daffara, C.Barahona, J.B. Free Software/Open Source: Information Society Opportunities for Europe working paper,http://eu.conecta.it paper, OSSEMP workshop, Third international conference on* open source*. Limerick 2007*
- *[ED05] Evans Data, Open Source Vision report, 2005*
- *[Eng10] Engelhardt S. Maurer S. The New (Commercial) Open Source: Does it Really Improve Social Welfare Goldman School of Public Policy Working Paper No.GSPP10-001, 2010*
- *[Gar06] Gartner Group, Open source going mainstream. Gartner report, 2006*
- *[Gosh06] Gosh, et al. Economic impact of FLOSS on innovation and competitiveness of the EU ICT sector.*

http://bit.ly/cNwUzû

- *[Koot03] Kooths, S.Langenfurth, M.Kaiwey,N.Open-Source Software: An Economic Assessment Technical report, Muenster Institute for Computational Economics (MICE), University of Muenster*

La délicate question du modèle économique (Libres conseils 39/42)

Bientôt la dernière séance… rejoignez-nous jeudi prochain sur le framapad de traduction

Traduction Framalang : Ouve, tcit, Julius22, goofy, merlin8282, lamessen, Jej, Alpha

Sous-estimer la valeur d'un modèle économique pour le logiciel libre

Frank Karlitschek

Frank Karlitschek est né en 1973 à Reutlingen, en Allemagne, et a commencé à écrire des logiciels à l'âge de 11 ans. Il a étudié l'informatique à l'université de Tübingen et s'est impliqué dans le logiciel libre et les technologies de l'internet dans le milieu des années 1990. En 2001, il a commencé à contribuer à KDE en lançant KDE-Look.org, un site communautaire d'œuvres qui deviendrait plus tard le réseau

*openDesktop.org. Frank a initié plusieurs projets et initiatives open source comme Social Desktop, Open Collaboration Services, Open-PC et ownCloud. En 2007, il a fondé une société appelée hive01 qui offrait des services et des produits autour de l'*open source *et des technologies de l'internet. Aujourd'hui, Frank est membre du conseil et vice-président de KDE e.V. et c'est un intervenant habitué des conférences internationales.*

Introduction

Il y a dix ans, j'ai sous-estimé la valeur d'un modèle économique. Logiciel libre et modèle économique ? Deux concepts incompatibles. Du moins, c'est ce que je pensais lorsque j'ai commencé à contribuer à KDE en 2001. Le logiciel libre, c'est pour le plaisir et pas pour l'argent. N'est-ce pas ? Les libristes veulent un monde où chacun peut écrire du logiciel et où les grandes entreprises, telles que Microsoft ou Google, sont superflues. Tout logiciel devrait être libre et tous ceux qui souhaitent développer du logiciel devraient en avoir la possibilité – même les développeurs du dimanche. Donc, gagner de l'argent importe peu. N'est-ce pas ? Aujourd'hui, j'ai une opinion différente. Les développeurs devraient parfois être rémunérés pour leurs efforts.

Les raisons d'être du logiciel libre

La plupart des développeurs de logiciels libres ont deux principales motivations pour travailler sur le logiciel libre. La première motivation est le facteur plaisir. C'est une expérience fantastique de travailler avec d'autres personnes très talentueuses du monde entier et de créer des technologies exceptionnelles. KDE, par exemple, est une des communautés les plus accueillantes que je connaisse. C'est tellement amusant de travailler avec des milliers de contributeurs du monde

entier pour créer des logiciels qui seront utilisés par des millions de personnes. Pour faire simple, tout le monde est expert dans un ou plusieurs domaines et nous collaborons pour créer une vision partagée. Pour moi, c'est toujours génial de rencontrer d'autres contributeurs de KDE, d'échanger des idées ou de travailler sur nos logiciels ensemble, que nous nous rencontrions en ligne ou dans la vie réelle à une des nombreuses conférences ou événements. Et il s'agit aussi d'amitié. Au fil des années, je me suis fait beaucoup de bons amis au sein de KDE.

Mais les contributeurs de KDE ne sont pas uniquement motivés par le plaisir de rejoindre KDE. Il y a aussi l'idée que chacun de nous peut rendre le monde meilleur par nos contributions. Le logiciel libre est essentiel si vous vous souciez de l'accès à la technologie et à l'informatique pour les pays en voie de développement. Cela permet aux personnes pauvres d'avoir leur place dans l'ère de l'information sans acheter des licences coûteuses pour des logiciels propriétaires. Il est essentiel pour les personnes qui se soucient de la confidentialité et de la sécurité, parce que le logiciel libre est le seul et unique moyen de savoir exactement ce que votre ordinateur fait avec vos données privées. Le logiciel libre est important pour un écosystème informatique sain, parce qu'il permet à tout le monde de bâtir à partir du travail des autres et de vraiment innover. Sans le logiciel libre, il n'aurait pas été possible à Google ou Facebook de lancer leurs entreprises. Il n'est pas possible d'innover et de créer la nouvelle technologie marquante si vous dépendez de logiciels propriétaires et que vous n'avez pas accès à toutes les parties du logiciel.

Le logiciel libre est aussi indispensable pour l'éducation, parce que tout le monde peut voir les entrailles du logiciel et étudier son fonctionnement. C'est comme cela que le logiciel libre contribue à faire du monde un endroit meilleur et c'est pourquoi je participe à des projets de logiciel libre

comme KDE.

La nécessité d’un écosystème

Voilà les principales raisons pour lesquelles je veux que le logiciel libre et particulièrement le bureau libre soient largement répandus. Pour y parvenir, il nous faut bien plus de contributeurs qu’aujourd’hui. Par contributeurs, j’entends des gens qui écrivent les infrastructures centrales, le bureau et les grandes applications. Nous avons besoin de gens qui travaillent sur l’utilisabilité, sur les illustrations, sur la promotion et sur bien d’autres aspects importants. KDE est déjà une grande communauté avec des milliers de membres. Mais nous avons besoin de davantage de gens pour aider à rivaliser de manière sérieuse avec le logiciel propriétaire.

La communauté du logiciel libre est minuscule comparée au monde du logiciel propriétaire. D’un côté, ce n’est pas un problème car le modèle de développement logiciel distribué du monde du logiciel libre est bien plus performant que la façon d’écrire du logiciel à sources fermées. Un grand avantage est, par exemple, la possibilité de mieux réutiliser du code. Mais même avec ces avantages, nous avons besoin de bien plus de contributeurs qu’aujourd’hui si nous voulons réellement conquérir le marché de l’ordinateur de bureau et celui du mobile.

Nous avons aussi besoin d’entreprises pour nous aider à apporter notre travail sur le marché de masse. Bref, nous avons besoin d’un grand écosystème en forme permettant de vivre en travaillant sur le logiciel libre.

La situation actuelle

J’ai commencé à contribuer à KDE il y a plus de 10 ans et, depuis, j’ai vu d’innombrables volontaires très motivés et talentueux rejoindre KDE. C’est vraiment génial. Le problème,

c’est que j’ai aussi vu beaucoup de contributeurs expérimentés abandonner KDE. C’est vraiment triste. Parfois, c’est simplement la marche normale du monde : les priorités changent et les gens se concentrent sur autre chose. Le problème, c’est que beaucoup abandonnent aussi à cause de l’argent. Il arrive un moment où les gens décrochent leur diplôme et veulent bouger de leur chambre d’étudiant.

Plus tard, ils veulent se marier et avoir des enfants. À partir de là, ils doivent trouver du travail. Il y a quelques entreprises dans l’écosystème de KDE qui proposent des postes liés à KDE. Mais cela ne représente qu’une petite part des emplois disponibles dans le secteur informatique. Du coup, beaucoup de membres chevronnés de KDE doivent travailler dans des entreprises où ils doivent utiliser des logiciels propriétaires qui n’ont rien à voir avec KDE ou le logiciel libre. Tôt ou tard, la plupart de ces développeurs abandonnent KDE. J’ai sous-estimé cette tendance il y a 10 ans, mais je pense que c’est un problème pour KDE sur le long terme, parce que nous perdons nos membres les plus expérimentés au profit des entreprises de logiciel propriétaire.

Le monde de mes rêves

Dans le monde idéal que j’imagine, les gens peuvent payer leur loyer en travaillant sur les logiciels libres et ils peuvent le faire de telle sorte que ça n’entre pas en conflit avec nos valeurs. Ceux qui contribuent à KDE devraient avoir tout le temps qu’ils veulent pour contribuer à KDE et au monde libre en général. Ils devraient gagner de l’argent en aidant KDE. Leur passe-temps deviendrait leur travail. Cela permettrait à KDE de se développer de manière spectaculaire, parce que ce serait super de contribuer et de fournir en même temps de bonnes perspectives d’emploi stables et à long terme.

Quelles possibilités avons-nous ?

Du coup, quelles sont les solutions possibles ? Que pouvons-nous faire pour que ça arrive ? Y a-t-il des moyens pour que les développeurs paient leur loyer tout en travaillant sur du logiciel libre ? Je voudrais exposer ici quelques idées que j'ai rassemblées au cours de plusieurs discussions avec des contributeurs au logiciel libre. Certaines d'entre elles sont probablement polémiques, parce qu'elles introduisent des idées complètement neuves au sein du monde du logiciel libre. Mais je pense qu'il est essentiel pour nous de voir au-delà de notre monde actuel si nous voulons mener à bien notre mission.

Du développement sponsorisé

Aujourd'hui, de plus en plus d'entreprises apprécient l'importance du logiciel libre et contribuent à des projets de logiciels libres, ou sortent même leurs propres projets de logiciel libre. C'est une chance pour les développeurs de logiciels libres. Nous devrions parler à davantage d'entreprises et les convaincre de s'associer au monde du logiciel libre.

Des dons de la part des utilisateurs

Il devrait y avoir une manière facile pour les utilisateurs de donner de l'argent directement aux développeurs. Si un utilisateur d'une application populaire veut soutenir le développeur et promouvoir ses développements à venir pour cette application, donner de l'argent devrait ne tenir qu'à un clic de souris. Le système de dons peut être construit au sein même de l'application pour rendre le don d'argent aussi facile que possible.

Des primes

L'idée derrière les primes est qu'un ou plusieurs utilisateurs d'une application peuvent payer pour le développement d'une

fonctionnalité particulière. Un utilisateur peut soumettre la liste de ses demandes de nouvelles fonctionnalités sur un site web et annoncer combien il est prêt à payer pour cela. D'autres utilisateurs qui apprécient ces propositions pourraient ajouter de l'argent à la demande de fonctionnalité. Au bout d'un moment, le développeur commence à mettre au point la fonctionnalité et récupère l'argent des utilisateurs. Cette possibilité de primes n'est pas facile à introduire dans le processus. Des gens ont déjà essayé de mettre en place quelque chose de similaire, sans succès. Mais je pense que ça peut marcher si on s'y prend bien.

Du support

L'idée est que le développeur d'une application vende directement du support aux utilisateurs de l'application. Par exemple, les utilisateurs d'une application achètent du support pour, supposons, 5 € par mois et obtiennent le droit d'appeler directement le développeur à des plages horaires spécifiques de la journée, ils peuvent poser des questions à une adresse de courriel spécifique, ou le développeur peut même aider les utilisateurs par le biais d'un bureau à distance. J'ai bien conscience que beaucoup de développeurs n'aimeront pas l'idée que les utilisateurs puissent les appeler et leur poser des questions bizarres. Mais si cela signifie qu'ils gagnent suffisamment avec le système de support pour travailler à plein temps sur leurs applications, alors c'est certainement une bonne chose.

Des soutiens

L'idée c'est que les utilisateurs finaux puissent devenir les soutiens d'une application. Le bouton « Soutenez ce projet » pourrait être intégré directement dans l'application. L'utilisateur devient alors un soutien par un paiement mensuel de, par exemple, 5 € qui vont directement au développeur. Tous les soutiens sont listés dans la fenêtre « À propos de

l’application » avec leurs photos et leurs noms réels. Une fois par an, tous les soutiens sont aussi invités à une fête spéciale avec les développeurs. Il est possible qu’un développeur puisse devenir capable de travailler à plein temps sur une application, si assez d’utilisateurs deviennent des soutiens.

Des programmes de fidélité

Certaines applications ont intégré des services web, et certains de ces services Web exécutent des programmes affiliés. Par exemple, un lecteur multimédia peut être intégré à la boutique en ligne de MP3 Amazon ou un lecteur PDF peut être intégré à une boutique en ligne de livres numériques. À chaque fois qu’un utilisateur achète du contenu via cette application, le développeur obtient un peu d’argent.

Des magasins d’applications sous forme de binaires

Beaucoup de gens ne savent pas qu’il est possible de vendre des binaires de logiciels libres. La licence GPL exige simplement de fournir également le code source. Il est donc parfaitement légal de vendre des binaires bien empaquetés de notre logiciel. En réalité, les sociétés comme Red Hat et Novell vendent déjà notre logiciel dans leurs distributions commerciales. Mais les développeurs n’en bénéficient pas directement. Tous les revenus vont aux sociétés et rien ne va aux développeurs. On devrait donc permettre aux développeurs de logiciels libres de vendre à l’utilisateur final des applications bien empaquetées, optimisées et testées. Cela pourrait particulièrement bien fonctionner pour Mac ou Windows. Je suis sûr qu’un tas de gens seraient prêts à payer quelque chose pour des binaires d’Amarok pour Windows ou de digiKam pour Mac, si tout l’argent allait directement au développeur.

Conclusion

La plupart de ces idées ne sont pas faciles à mettre en œuvre. Cela nécessite de modifier notre logiciel, nos méthodes de travail et même nos utilisateurs, qu'il faut encourager à montrer qu'ils apprécient le logiciel que nous créons, en nous aidant à financer son développement.

Cependant, les bénéfices potentiels sont énormes. Si nous pouvons assurer des sources de revenus pour notre logiciel, nous pouvons conserver nos meilleurs contributeurs et peut-être en attirer de nouveaux. Nos utilisateurs auront une meilleure expérience avec un développement logiciel plus rapide, la possibilité d'influencer directement le développement par le biais de primes et un meilleur support.

Le logiciel libre n'est plus seulement un loisir sur votre temps libre. Il est temps d'en faire un business.

Apporter le Libre dans le secteur public (Libres Conseils 38/42)

Bientôt la dernière séance de traduction ! Jeudi à 21h, rendez-vous sur le framapad de traduction, le travail collaboratif sera ensuite publié ici même.

Traduction framalang : tcit, Julius22, Sky, lamessen, goofy, peupleLà, merlin8282, lamessen, Jej, Alpha

Le Logiciel Libre dans le secteur public

Till Adam

Issu du milieu de la musique et des sciences humaines, Till Adam a passé pas loin des dix dernières années dans le monde de la programmation. Il travaille au sein de KDAB où il dirige plusieurs services, dont celui qui est en charge des logiciels libres. Till officie aussi au sein du conseil d'administration de Kolab Systems AG, une entreprise dont le modèle économique repose entièrement sur les logiciels libres. Il vit avec sa femme et sa fille à Berlin.

Introduction

J'imagine que comme de nombreux autres auteurs de cette compilation d'articles, j'ai commencé à contribuer au logiciel libre lorsque j'étais étudiant. J'avais décidé relativement tard dans ma vie de poursuivre un cursus en informatique (ayant échoué à devenir riche et célèbre en tant que musicien). Je m'attendais donc à être légèrement plus âgé que mes pairs en obtenant mon diplôme. J'ai donc pensé qu'il serait bénéfique d'apprendre par moi-même la programmation, qui ne m'était pas trop enseignée à l'école, afin de d'avoir plus d'atouts aux yeux de futurs employeurs, en dépit de mon âge. Après quelques incursions dans diverses petites communautés, j'ai finalement trouvé ma voie dans le projet KDE et j'ai commencé à travailler sur l'application de courriel.

Grâce aux personnes extrêmement serviables et douées techniquement que j'y ai rencontrées, j'ai pu apprendre rapidement et contribuer de façon significative au code, ce qui m'a entraîné de plus en plus dans leur réseau social, mais aussi vers le domaine fascinant des problèmes techniques liés à la gestion de données personnelles.

Lorsque KDAB, une entreprise remplie de gens qui utilisaient KDE, m'a demandé si, dans le cadre d'un stage étudiant, je voulais les aider sur la partie commerciale d'un projet en cours, j'ai bien sûr été ravi de pouvoir gagner ma vie et bidouiller le logiciel KDE en même temps. Au fil des ans, j'ai été témoin de l'adoption et de l'utilisation des architectures de gestion des données personnelles de KDE par le secteur public, particulièrement en Allemagne, où j'ai pu y assister personnellement à la croissance économique de KDAB dans ce secteur géographique. Alors que j'évoluais vers des postes plus orientés sur le management, vendre et livrer des services issus du logiciel libre comprenant des produits de KDE à de grandes organisations, en particulier dans le secteur public, a finalement fait partie de mon travail.

Il faut noter que la plupart du travail sur le projet qui a inspiré ce texte était généralement fait en collaboration avec d'autres entreprises du logiciel libre, à savoir glOcode, un spécialiste de la cryptographie qui se charge du maintien de GNUPG, et Intevation, une entreprise de conseil qui se concentre exclusivement sur le logiciel libre ainsi que ses défis stratégiques et opportunités. Mention spéciale à Bernhard Reiter, l'un des fondateurs d'Intevation, qui a joué un rôle clé lors de la vente et de la conduite de bon nombre de ces projets. Les quelques fragments de sagesse contenus dans ce texte sont probablement issus de son analyse et des nombreuses conversations que j'ai pu avoir avec lui au fil des ans.

Donc, si Bernhard et moi pouvions revenir dans le temps, quelles pourraient donc bien être les idées que nous partagerions avec nos « nous » plus jeunes et plus naïfs ? Eh bien, il s'avère qu'elles commencent toutes par la lettre **P**.

Personnes

Telles que sont les choses aujourd'hui, il est toujours plus

difficile pour les gens de terrain des technologies de l'information et pour les décideurs d'utiliser du logiciel libre que ça ne l'est d'utiliser les alternatives propriétaires. Même en Allemagne, où le logiciel libre a un soutien politique relativement fort, il est plus facile et plus sûr de suggérer l'utilisation de quelque chose qui est perçu comme un « standard de l'industrie » ou comme « ce que tous les autres font » ; en d'autres termes, des solutions propriétaires.

Celui qui propose une solution en logiciel libre fera probablement face à de l'opposition de la part de collègues moins aventureux (ou ayant moins de vision), à un examen minutieux des supérieurs, à de plus grandes attentes par rapport aux résultats et à une pression budgétaire irréaliste. Il faut donc un type particulier de personnes souhaitant prendre des risques personnels, potentiellement compromettre l'avancée de leur carrière et combattre dans une bataille presque perdue d'avance. Ceci est bien sûr vrai dans n'importe quelle organisation. Mais, dans une administration publique, une ténacité particulière est requise car les choses bougent généralement plus lentement. Et une hiérarchie organisationnelle inflexible ajoutée à des options de carrière limitées amplifient le problème.

Sans allié à l'intérieur, faire envisager de façon sérieuse les options du logiciel libre peut s'avérer quasiment impossible. Si de telles personnes existent, il est important de les soutenir autant que possible dans leurs luttes internes. Ceci signifie leur fournir des informations opportunes, fiables et vérifiables sur ce qui se passe dans la communauté avec laquelle l'organisation entend interagir. Ces informations doivent contenir suffisamment de détails pour fournir une image complète tout en atténuant la complexité de la communication et du chaos de planification faisant parfois partie de la façon de travailler dans le monde du logiciel libre, de façon à ce que ça devienne plus gérable et moins

effrayant. L’honnêteté et le sérieux aident à construire des relations fortes avec ces personnes-clés, qui sont la base du succès à plus long terme. Elles se reposent sur vous, en tant qu’intermédiaire avec le monde merveilleux et quelque peu effrayant des communautés du logiciel libre, pour trouver des chemins qui les mèneront elles et leurs organisations à leurs objectifs. Elles prennent également des décisions en se basant largement sur la confiance personnelle. Cette confiance doit être acquise et conservée.

Afin d’y parvenir, il est important de ne pas se concentrer uniquement sur les résultats techniques des projets, mais de garder en tête les objectifs plus larges, personnels et organisationnels que l’on doit atteindre lorsqu’on travaille sur ces projets. Le succès ou l’échec du projet en cours ne dépendra peut-être pas du fait qu’un responsable de projet au sein d’une agence soit capable de ne vanter que des fonctionnalités auxiliaires à ses supérieurs à des moments plus ou moins aléatoires du calendrier. Mais cela impactera peut-être le fait que le projet suivant se fasse ou ne se fasse pas. Lorsque vous avez peu d’amis, les aider à réussir est un bon investissement.

Priorités

En tant que technophiles, les gens du logiciel libre ont tendance à se concentrer sur ce qui est nouveau, excitant et qui paraît important au niveau technologique. En conséquence de quoi, nous mettons moins l’accent sur les choses qui sont plus importantes dans le contexte d’une administration publique (souvent vaste). Mais considérez quelqu’un désireux de changer tout un ensemble de technologies dans une structure qui a plutôt tendance à rester sur les mêmes technologies pendant une longue durée. Étant donné qu’un changement brusque est difficile et coûteux, il est de loin bien plus important d’avoir de la documentation sur les choses qui ne fonctionneront pas, de façon à pouvoir les éviter ou les

contourner, que de savoir qu'une version à venir fonctionnera beaucoup mieux. Il est peu probable que cette nouvelle version soit jamais disponible pour les utilisateurs dpnt nous parlons ici. Et il est bien plus simple d'avoir affaire à des problèmes connus et anticipés plutôt que d'être forcé de faire face à des surprises. Le bogue documenté d'aujourd'hui est paradoxalement préférable à sa résolution de demain avec ses effets de bord imprévisibles.

Dans une grande organisation qui utilise des logiciels pendant une longue durée, le coût d'acquisition du logiciel, que ce soit par le biais de licences ou dans le cadre de développement sur mesure de logiciels libres par contrat, a peu d'importance en comparaison du coût de maintenance et de support. Cela mène à penser que moins de fonctionnalités, plus stables, ce qui induit une moindre charge pour le l'organisme de support, auxquelles on peut faire plus confiance et qui ont moins besoin de maintenance intensive sont meilleures que de séduisantes nouveautés, complexes et sans doute moins matures.

Alors que ces deux sentiments vont à l'encontre des instincts des développeurs de logiciels libres, ce sont ces mêmes aspects qui rendent attractif pour le secteur public le fait de payer pour le développement de logiciels libres, plutôt que de dépenser de l'argent pour des licences de produits pris au hasard. En partant d'une large palette de logiciels gratuitement disponibles, l'organisation peut investir son budget dans le perfectionnement des parties précises qui sont pertinentes pour ses propres opérations. Elle n'a ainsi pas à payer (via les coûts de licences) pour le développement de fonctionnalités clinquantes et guidées par le marché dont elle n'a pas besoin. En soumettant tout ce travail à la communauté en retour, la maintenance à long terme de ces améliorations et du logiciel de base est partagée par un grand nombre de personnes. De plus, grâce au fait que ces améliorations deviennent publiques, d'autres organisations aux besoins similaires peuvent bénéficier de celles-ci sans coût

supplémentaire. Cela maximise donc l'utilité de l'argent du contribuable, ce que toute administration publique souhaite (ou devrait souhaiter).

Politique d'approvisionnement

Si les budgets informatiques des agences gouvernementales sont clairement mieux utilisés dans l'amélioration du logiciel libre et dans son adaptation à leurs besoins, pourquoi est-ce si rarement ce que l'on fait ? L'équivalence des fonctionnalités pour les types de logiciels les plus utilisés a depuis longtemps été atteinte, la convivialité est la même, la robustesse et le coût total de possession aussi. La notoriété et la connaissance sont bien sûr toujours des problèmes, mais le véritable obstacle à l'acquisition de services en logiciel libre réside dans les conditions légales et administratives sous lesquelles cela doit se produire. Changer ces conditions nécessite du travail, au niveau de la politique et du lobbying. C'est rarement possible dans le contexte d'un projet individuel. Heureusement, des organisations telles que la Free Software Foundation Europe et sa sœur aux États-Unis font du lobbying en notre nom et font lentement changer les choses. Jetons un coup d'œil à deux problèmes centraux d'ordre structurel.

Des licences, pas des services

Beaucoup de budgets informatiques sont structurés de telle façon qu'une partie de l'argent est mise de côté pour l'achat d'un nouveau logiciel ou pour le paiement continu de l'utilisation d'un logiciel sous forme de licences. Comme il était inimaginable pour ceux qui ont construit ces budgets qu'un logiciel puisse être autre chose qu'un bien achetable, représenté par une licence propriétaire, il est souvent difficile ou impossible pour les décideurs informatiques de dépenser cette même somme d'argent pour des services. La comptabilité de gestion n'en entendra simplement pas parler.

Cela peut mener à la situation malheureuse où une organisation a la volonté et l'argent pour améliorer un morceau de logiciel libre afin qu'il convienne parfaitement à ses besoins, pour le déployer et pour le faire tourner pendant des années et envoyer ses contributions à la communauté, en retour, mais où cela ne peut se faire tant que toute l'affaire n'est pas enveloppée dans une vente et un achat artificiels et non nécessaires d'un produit imaginaire basé sur une licence libre.

Pièges légaux

Les cadres légaux pour les fournisseurs de logiciels supposent souvent que quiconque signant la production d'un logiciel exerce le plein contrôle des copyrights, marques déposées et brevets afférents. L'organisation cliente attend une garantie contre des risques variés de la part du fournisseur. Dans le cas où une société ou une personne produit une solution ou un service basé sur du logiciel libre, cela est souvent impossible car il y a d'autres titulaires de droits qui ne peuvent pas être raisonnablement impliqués dans l'arrangement contractuel. Ce problème apparaît plus ostensiblement dans le contexte des brevets logiciels. Il est pratiquement impossible pour un fournisseur de services de s'assurer contre les risques de contentieux de brevets, ce qui rend très risqué pour lui d'endosser la pleine responsabilité.

Prix

Historiquement, l'argument le plus vendeur du logiciel libre donné au grand public a été son potentiel pour d'économie d'argent. Le logiciel libre a en effet permis des économies à grande échelle pour beaucoup d'organisations depuis de nombreuses années. Le système d'exploitation GNU/Linux a été le fer de lance de ce développement. Ceci en raison de sa libre disponibilité au téléchargement qui a été perçue en opposition frappante avec les licences onéreuses de son

principal concurrent, Microsoft Windows.

Pour quelque chose d'aussi utilisé et utile qu'un système d'exploitation, il est indéniable que le bénéfice des coûts structurels vient des coûts de développement qui sont répartis sur de nombreuses parties. Malheureusement, l'espoir que ceci reste vrai pour tous les logiciels libres a mené à la pensée irréaliste que les coûts seront toujours réduits, largement et immédiatement. D'après notre expérience, ce n'est pas vrai. Comme nous l'avons vu dans les sections précédentes de cet ouvrage, il est très logique de tirer le meilleur parti de l'argent dépensé dans l'utilisation de logiciels libres et il est probable qu'au fil du temps et pour de nombreuses organisations de l'argent puisse être économisé. Mais pour une agence isolée qui cherche seulement à déployer un logiciel libre, il devra y avoir un investissement initial et un coût nécessaire associé pour obtenir le niveau de maturité et de robustesse nécessaire.

Alors que cela semble largement raisonnable aux professionnels des opérations informatiques, il est souvent plus difficile de convaincre de cette vérité leurs supérieurs avec le bilan financier. Surtout lorsque la potentielle économie de moyens financiers a initialement été utilisée comme un argument pour faire entrer le logiciel libre, il peut s'avérer très difficile de gérer efficacement les attentes futures. Plus vite les décideurs sauront exactement de façon claire combien et dans quoi ils investissent, mieux ils accepteront de le faire sur le long terme

Le meilleur rapport qualité-prix est toujours attirant et un fournisseur de services informatiques qui cessera d'être disponible à cause de la pression des prix et n'obtiendra pas suffisamment de réussite économique est aussi peu attractif dans le logiciel libre que dans les modèles économiques basés sur des licences propriétaires. Il est donc aussi dans l'intérêt des clients que les estimations de coûts soient réalistes et que les conditions économiques dans lesquelles le

travail est effectué soient durables.

Conclusion

Notre expérience montre qu'il est possible de convaincre des organismes du secteur public de dépenser de l'argent dans des services basés sur des logiciels libres. C'est une proposition intéressante qui offre une plus-value et qui a un sens politique. Malheureusement, il existe encore des barrières structurelles. Mais avec l'aide de pionniers dans le secteur public, elles peuvent être contournées. Avec un soutien suffisant de notre part à tous, ceux qui travaillent pour le logiciel libre au niveau politique finiront par les surmonter. Une communication claire et honnête sur les réalités économiques et techniques peut favoriser des partenariats efficaces qui amènent des bénéfices à la communauté du logiciel libre, aux administrations publiques utilisant ces logiciels et à ceux qui les fournissent avec les services nécessaires dans un cadre viable et durable.

Trouver des sous ! (Libres conseils 37/42)

Chaque jeudi à 21h, rendez-vous sur le framapad de traduction, le travail collaboratif sera ensuite publié ici même.

Traduction Framalang : Ouve, Julius22, Sphinx, Garburst, goofy, peupleLà, audionuma, lamessen

Comment demander de l’argent

Selena Deckelmann

Selena Deckelmann est une importante contributrice de PostgreSQL. Elle donne des conférences dans le monde entier sur les logiciels libres, les communautés de développeurs et du trollage. Elle s’intéresse à l’ouverture des données publiques de la ville de Portland, aux poulets d’appartement et à la recherche de solutions pour permettre aux bases de données de fonctionner plus vite.

Elle a fondé Postgres Open, une conférence dédiée aux activités économiques autour de PostgreSQL et au bouleversement du secteur des bases de données. Elle a fondé et co-présidé Open Source Bridge, une conférence de développeurs pour les citoyens open source. Elle a fondé la Conférence PostgreSQL, une brillante série de conférences sur la côte Est et la côte Ouest des États-Unis pour PostgreSQL. Elle fait actuellement partie du comité de programme de PgCon, de la conférence des utilisateurs MySQL et de OSCON Data. Elle est l’une des contributrices au manuel du mentor des Google Summer of Code, et du Guide des Étudiants. Elle est conseillère pour l’initiative Ada et membre du conseil de la société Technocation.

Si je retrace mon parcours depuis la première fois où j’ai démarré un PC sous Linux en 1994, une chose ressort clairement de mon expérience avec l’open source : j’aurais aimé savoir comment demander de l’argent. Demander de l’argent est difficile. J’ai écrit des demandes de subventions, demandé des augmentations, négocié des salaires et des tarifs horaires de consultante et levé des fonds pour des conférences à but non lucratif. Après de nombreuses tentatives et échecs, j’ai développé une méthode qui fonctionne ! Ce qui suit est un condensé des trucs et astuces que j’ai utilisés durant ces cinq dernières années pour augmenter les fonds pour des non-

conférences, des sprints de code d'une journée et des conférences de plusieurs jours à propos de la culture et des logiciels open source.

La méthode pour obtenir des fonds pour une conférence comporte six étapes :

1. identifier un besoin ;
2. en parler à quelqu'un ;
3. demander de l'argent ;
4. récupérer l'argent ;
5. dépenser l'argent ;
6. Remercier.

Identifiez un besoin

Votre première mission en tant qu'organisateur de conférences consiste à expliquer pourquoi vous mettez en place une conférence de plus, en quoi elle sera utile à ceux qui y assisteront et quel intérêt un sponsor aurait à vous financer. On appelle ça « écrire un dossier de présentation ». Les éléments principaux d'un tel dossier sont les suivants :

- l'objectif : en un paragraphe, expliquez pourquoi vous faites la conférence. Qu'est-ce qui vous a poussé à rassembler des gens ? Et qui seront les participants ? De quoi parleront-ils une fois là-bas ? Si vous avez un sujet ou un but particulier en tête, mentionnez-le. Expliquez également pourquoi vous avez choisi tel endroit pour l'événement. Y a-t-il un lien avec le sujet de la conférence ? Est-ce que les personnes intéressantes s'y trouvent ? Est-ce qu'il y a un sponsor ? Enfin, mettez à disposition les chiffres intéressants à propos des événements précédents, comme le nombre de participants et des informations pertinentes sur les intervenants ou des détails sur le lieux choisi ;
- les possibilités de mécénat et les bénéfices escomptés :

cette partie du dossier va mettre en relief ce que les sponsors peuvent attendre de votre conférence. En règle générale, on y expose les retours sont évalués en termes financiers, mais on peut également y décrire des avantages comme des travaux en nature ou du bénévolat. Commencez simplement. Traditionnellement, les parrainages financiers des événements sont assurés par des services des ressources humaines qui cherchent à embaucher ou par des départements commercial-marketing qui cherchent à faire connaître leurs produits ou services. Voici, entre autres, le genre d'avantages que les sponsors en attendent : la mention du sponsor sur un site Web, dans les messages ou tweets pour les participants, l'accès à la liste des adresses électroniques ou aux informations sur les profils des participants, la présence des logos et des étiquettes sur les pochettes, tours de cou et autres gadgets distribués lors de la conférence, de même au moment des pauses cafés, des repas et casse-croûtes. Il leur faut aussi un stand sur la zone de la conférence et de l'espace publicitaire sur le programme de la conférence. Pensez aussi aux choses originales qui permettront de vous démarquer, à travers le déroulement et le lieu de la conférence. Par exemple, à Portland, il y a une boutique de beignets très populaire, avec un service de livraison. Nous avons trouvé un sponsor, et nous avons obtenu la permission d'amener le camion de livraison juste à l'endroit où nous étions et nous avons servi des beignets gratuitement pour le petit-déjeuner. Vous trouverez ci-dessous des liens pour des exemples de dossiers. Ils correspondent tous à de grosses conférences, donc vous n'obtiendrez peut-être pas le même résultat. J'ai déjà fait un dossier, avec une seule possibilité de parrainage, et l'accord était qu'en échange de la présence d'un de ses employés à la conférence, les organisateurs mentionnaient clairement l'entreprise et la remerciaient pour son soutien.

Quelques entreprises : OSCON, Open Source Bridge, MeeGo San Francisco.

- le contrat : toujours inclure un contrat avec votre dossier. Cela établit les attentes et les engagements ainsi que le calendrier et peut éviter beaucoup de problèmes en chemin. Je ne suis pas un avocat, donc ce qui suit relève plus de mon expérience que des conseils juridiques. Pour les événements plus mineurs, j'écris un contrat très simple qui expose mes attentes : les sponsors promettent de payer à une certaine date et je promets de tenir l'événement à une certaine date. Copier un contrat existant est quelque chose de délicat car les lois changent suivant les différents états et pays. J'ai consulté un avocat qu'un gestionnaire chevronné d'une communauté de l'open source m'avait recommandé. Le cabinet d'avocats a été assez agréable pour gracieusement créer des contrats et réviser des contrats entre nous et les hôtels. Le Software Freedom Law Center peut vous indiquer un avocat approprié si vous n'en avez pas.

Maintenant que vous avez créé le dossier de présentation, vous avez besoin de parler à quelques personnes.

Parlez-en

L'étape la plus difficile pour moi, c'est de faire passer le mot au sujet de mes événements ! Entraînez-vous à présenter votre événement en une ou deux phrases. Transmettez ce qui vous emballe et ce qui devrait emballer les autres.

Au fil des ans, j'ai appris qu'il fallait que je commence à parler SANS DÉLAI à mes connaissances plutôt que de m'inquiéter de savoir exactement quelles étaient les bonnes personnes à qui parler. Faites une liste des personnes à qui parler et que vous connaissez déjà et commencez à cocher cette liste.

La meilleure manière parler de votre projet est de le faire en personne ou au téléphone. Ainsi, vous ne spammez pas les les gens, vous captez leur attention et vous pouvez avoir un retour immédiat sur votre argumentaire. Les gens sont-ils enthousiastes ? Posent-ils des questions ? Ou bien trouvent-ils que c'est rasoir ? À qui d'autre pensent-ils que vous devriez en parler ? Demandez-leur ce qu'ils en pensent et comment vous pourriez rendre votre argumentaire plus attractif, plus intéressant, de sorte qu'ils en aient pour leur argent !

Une fois que vous aurez trouvé les mots-clés de votre argumentaire, écrivez-le et envoyez quelques courriels. Demandez des retours sur votre courriel et terminez toujours par un appel à agir avec une échéance pour la réponse. Gardez la trace des personnes qui répondent, de leurs réponses, et du moment favorable pour une relance de chacune d'elles.

Demandez de l'argent

Armé de votre dossier et de votre argumentaire réglé aux petits oignons, commencez à approcher des sociétés pour financer votre événement. À chaque fois que je lance une nouvelle conférence, je fais une liste de questions à son propos et je réponds à chacune avec une liste de personnes et de sociétés :

- Parmi les personnes que je connais, qui va trouver que c'est une idée géniale et faire la promo de mon événement (supporters) ;
- Quelles sont les personnes dont la présence à la conférence serait vraiment sympa (experts reconnus) ;
- Quelles sociétés ont des produits qu'elles voudraient promouvoir à mon événement (marketing) ;
- Qui voudrait embaucher les personnes qui participent (recruteurs) ;
- Quels projets libres et open source voudraient recruter

des développeurs (recruteurs open source)

En utilisant ces listes, envoyez votre brochure à travers le monde ! Voici un aperçu de la façon dont j'organise le processus de demande : je commence par envoyer les dossiers de présentation à mes supporters. J'en glisse aussi une copie aux experts, et je les invite à assister à la conférence ou à y intervenir. Je contacte ensuite les agences de publicité, les recruteurs et les recruteurs open source (parfois ça se recoupe !). En parallèle, j'ai généralement ouvert les inscriptions à la conférence et annoncé quelques allocutions ou événements spéciaux. Je croise les doigts pour que ça pousse à quelques inscriptions, que ça aide les sponsors à sentir que cette conférence va certainement avoir lieu et que tout va bien se passer.

Récupérez l'argent

Si tout se passe comme prévu, des sociétés et des individus vont commencer à vous proposer de l'argent. Lorsque cela se produit, vous aurez besoin de deux choses très importantes :

- un modèle de factures ou de devis ;
- un compte en banque pour recueillir les fonds.

Les modèles de factures sont simples à réaliser. J'utilise une feuille de calcul Google que j'actualise pour chaque facture. Vous pourriez facilement utiliser OpenOffice.org ou même TeX (si quelqu'un peut m'envoyer un modèle de facture LaTeX, merci d'avance !). On peut trouver des exemples de factures à l'adresse http://www.freetemplatesdepot.com.

Les éléments les plus importants d'une facture sont : le mot FACTURE, un numéro unique de facture, le nom et les informations de contact du sponsor, le montant que le sponsor est censé verser, les termes de l'accord (à quelle date le sponsor est censé payer, quelles sont les pénalités en cas de non-paiement) et le montant total dû. Il faut ensuite envoyer

une copie de la facture à la société. Gardez une copie pour vous !

Certaines sociétés peuvent exiger que vous remplissiez des formulaires plus ou moins complexes pour vous reconnaître, vous ou votre organisation, comme un fournisseur. De la paperasserie. Beurk ! Les délais de paiement pour de grandes entreprises peuvent atteindre deux mois. Les exercices budgétaires des sociétés sont en général annuels. Regardez si une société a un budget disponible pour votre événement et si vous pouvez être inclus dans les prévisions budgétaires de l'année suivante, si vous avez manqué l'occasion pour l'année en cours.

Le compte en banque peut être votre compte personnel, mais c'est risqué pour vous. Pour un événement à plusieurs milliers d'euros, vous préférerez peut-être trouver une ONG ou une association loi de 1901 qui peut détenir et dépenser les fonds en votre nom. Si votre conférence est à but lucratif, vous devriez consulter un comptable sur la meilleure manière de gérer ces fonds. Trouver une organisation sans but lucratif avec laquelle travailler peut se résumer à contacter une fondation qui gère un projet de logiciel libre.

Maintenant, passons à ce qui justifie tout ce processus : dépenser les dons durement acquis !

Dépensez l'argent

Maintenant que vos sponsors ont payé, vous pouvez dépenser l'argent.

Créez un budget qui détaille vos postes de dépenses et quand vous aurez besoin de les dépenser. Je conseille d'obtenir trois devis pour les produits et services qui ne vous sont pas familiers, simplement afin de vous faire une idée sur ce qu'est un prix correct. Faites comprendre aux fournisseurs que vous contactez que vous faites jouer la concurrence.

Une fois que j'ai établi une relation avec une entreprise, j'ai tendance à faire des affaires avec eux d'un an sur l'autre. J'aime avoir de bonnes relations avec les fournisseurs et je trouve que même si je paie un peu plus que si je faisais jouer la concurrence chaque année, je finis par gagner du temps et par obtenir un meilleur service de la part d'un vendeur qui me connaît bien.

Pour les petits événements, vous pouvez garder une trace de vos dépenses dans un tableur assez simple. Pour les projets plus grands, demander à un comptable ou utiliser des logiciels de comptabilité peut être utile. Voici une liste des alternatives libres à Quicken (à différents niveaux et avec différents aspects !).

Le plus important est de garder une trace de toutes vos dépenses et de ne pas dépenser de l'argent que vous n'avez pas ! Si vous travaillez avec une organisation à but non lucratif pour gérer le budget de l'événement, demandez-lui de l'aide et des conseils avant de commencer.

Remerciez

Il existe de nombreuses manières de remercier les gens et les entreprises qui ont apporté leur soutien à votre manifestation. Encore plus important, suivez toutes les promesses que vous avez faites dans le dossier. Communiquez à chaque fois qu'un engagement est tenu !

Durant la manifestation, trouvez des moyens d'entrer en contact avec les sponsors, en désignant un bénévole pour les inscrire et de les inscrire eux-mêmes auprès de vous.

Après la manifestation, assurez-vous de remercier individuellement chaque sponsor et chaque bénévole pour sa contribution. Une association avec laquelle je travaille envoie des remerciements écrits à chaque sponsor en début d'année.

D’une manière générale, la communication est le terreau fertile de la levée de fonds. Porter attention aux sponsors et construire des relations authentiques avec eux aide à trouver plus de sponsors, et à construire votre réputation de bon organisateur de manifestations.

Leçons apprises

Après avoir créé et animé des dizaines de manifestations, les deux aspects les plus importants que j’en tire ont été de trouver des mentors et d’apprendre à bien communiquer.

Les mentors m’ont aidée à transformer des coups de gueule en essais littéraires, du fouillis en dossiers et des conversations difficiles en perspectives. J’ai trouvé des mentors dans des entreprises qui parrainaient mes conférences, et me faisaient des retours détaillés, parfois pénibles. Et j’ai trouvé des mentors parmi les bénévoles qui passaient des centaines d’heures à écrire du logiciel pour mes manifestations, à recruter des orateurs, à documenter ce que nous étions en train de faire, et à poursuivre la conférence après moi.

Apprendre à bien communiquer prend du temps, et c’est l’occasion de faire de nombreuses erreurs. J’ai appris à mes dépens que ne pas développer de relations avec les meilleurs sponsors signifie ne pas être financé l’année suivante ! J’ai aussi appris que les gens sont capables d’une formidable indulgence envers les erreurs, dès lors que vous communiquez tôt et souvent.

Bonne chance dans votre recherche de fonds, et merci de me dire si ce qui précède vous a aidé.

Les tribulations d'un organisateur de conférences (Libres conseils 36/42)

Chaque jeudi à 21h, rendez-vous sur le framapad de traduction, le travail collaboratif sera ensuite publié ici même.

Traduction Framalang : Ouve, Julius22, Sphinx, CoudCoud, grosfar, lum', goofy, peupleLà, lamessen

Nous ne sommes pas fous… Nous organisons des conférences !

Gareth J. Greenaway

Gareth J. Greenaway s'est impliqué activement dans la communauté du logiciel libre et open source depuis 1997 après avoir découvert Linux. Sa contribution majeure a consisté à regrouper des gens ayant la même opinion pour apprendre et expérimenter de nouveaux éléments de logiciel libre et open source. Cette implication a débuté avec un petit groupe d'utilisateurs de Linux (un « GUL ») et s'est développée avec l'organisation de la « Southern California Linux Expo », aussi connue sous le nom de SCALE. En tant que membre fondateur de cet évènement, Gareth remplit actuellement deux fonctions importantes au sein de l'organisation. La première est la gestion des conférences et la seconde concerne les relations avec la communauté.

J'ai commencé à écrire cette section avec ce que je pensais être les besoins et les étapes pour organiser une conférence sur le libre et l'open source. Cependant, une grande partie de ce que je trouvais à dire avait déjà été abordé par Dave Neary, expert en gestion de communautés. Donc, pour éviter de répéter et de recouper ce que Dave voulait expliquer, j'ai

décidé de partager différentes histoires de l'organisation de SCALE et les leçons que j'en ai tiré pendant ces années.

Trop d'énergie !

SCALE a commencé il y a maintenant neuf ans avec des membres de trois groupes locaux d'utilisateurs de Linux, ce n'était à l'origine qu'un modeste évènement régional organisé par l'un de ces groupes. La première expérience fut vraiment enrichissante. Beaucoup de leçons en ont été tirées. On courait un peu partout et l'évènement semblait se dérouler à une vitesse folle. Étant donné qu'aucun de nous n'avait encore organisé d'évènement où il fallait se soucier des risques de surtension ou de consommation électrique, nous n'y avons pas pensé, et, du coup, nous avons dû ré-enclencher les disjoncteurs de la salle plusieurs fois pendant l'évènement.

Ça va marcher… c'est sans fil !

Le deuxième SCALE a pris en compte un grand nombre de leçons apprises l'année précédente mais un nouveau lieu de rencontre allait donner de nouvelles leçons. Le centre de conférences de Los Angeles est le lieu où s'est tenu SCALE 2, il fournissait un espace bien plus grand pour installer l'évènement. Le nouveau lieu nous a aussi permis d'apprendre notre première leçon sur les contrats avec un grand organisme pour gérer les choses telles que l'équipement audio et vidéo, l'accès à Internet et le matériel d'exposition.

Compte tenu de la situation de l'évènement à l'intérieur du centre de conférences, nous avons dû positionner les comptoirs d'enregistrement dans une zone qui, tout en étant visible des participants qui arrivaient, se trouvait à une certaine distance du reste du salon. Nos possibilités pour fournir un accès réseau à la zone d'enregistrement étaient limitées car les règles de protection anti-incendie proscrivaient l'utilisation de câbles ; le sans-fil était donc l'unique

option.

Tout a été mis en place très tôt le jour du salon et fonctionnait parfaitement bien, jusqu'à ce que cela cesse mystérieusement de marcher. La connexion sans fil, fournissant l'accès au réseau indispensable au comptoir d'enregistrement, a tout simplement disparu. Nous avons alors vécu beaucoup de tentatives de dépannages, beaucoup de déplacements d'équipements et d'antennes et beaucoup de frustration. « Ça devrait fonctionner. » était la seule conclusion à laquelle tout le monde pouvait parvenir, sans trop savoir pourquoi cela ne fonctionnait pas.

Soudain, un des membres de l'équipe, qui s'était tenu à l'écart de la séance de dépannage, a appelé tout le monde à venir à l'endroit où il se trouvait. En face d'une grande fenêtre qui surplombait un grand hall de réunion, nous avons tout à coup tous vu ce qu'il désirait nous faire voir. En-dessous de nous il y avait des dizaines de lumières clignotantes, tournantes et pulsantes qui nous regardaient. Des centaines d'appareils électroniques avec des lumières clignotantes, des sirènes et des panneaux à LED (diodes électro-luminescentes), interférant narquoisement avec les signaux sans fil de nos pauvres points d'accès. Nous avons soudain réalisé que nos heures de travail à tenter de résoudre ce problème de sans-fil avaient été vaines. Finalement, nous avons déroulé un câble Ethernet, l'avons scotché de la manière la plus sécurisée que nous avons pu, et nous avons dit une petite prière pour que le chef des pompiers ne fasse pas d'inspection surprise.

Soirées de gala, tireurs d'élite et disparition mystérieuse de mallette IBM

L'une des anecdotes les plus célèbres dans l'histoire de SCALE

est sans doute celle des incidents et péripéties qui sont survenus pendant SCALE 3. Ces aventures sont bien connues, et si vous assistiez à SCALE cette année-là, vous n'avez pas pu y échapper.

Le troisième SCALE devait se dérouler encore une fois au centre de conférences de Los Angeles au L.A. Convention Center. Tout le travail de planification et d'organisation avait été mené en amont pendant de nombreux mois et tout s'annonçait bien. Trois semaines environ avant l'évènement, nous avons reçu des informations à propos de plusieurs routes qui seraient fermées autour du centre de conférences à cause d'une soirée de gala qui devait avoir lieu. À cause de ces fermetures de routes, il n'y avait plus qu'une voie d'accès pour accéder au centre et en repartir, ce qui est loin d'être idéal. Fort heureusement, nous avons eu le temps d'avertir tous ceux qui venaient à l'évènement et de leur indiquer les routes fermées à la circulation et les itinéraires alternatifs. Cette année-là, c'était aussi la première fois que SCALE devait se dérouler sur deux jours, dans l'espoir de répartir un peu les choses pour ne pas être autant dans la précipitation et la frénésie.

L'un des plus anciens sponsors et exposants de SCALE est IBM. Ils ont toujours été une plus-value appréciée, mais, malheureusement, leur participation s'est généralement accompagnée de quelques difficultés. La veille de l'évènement, comme d'habitude, avait été réservée à la mise en place pour permettre à l'équipe de SCALE de tout installer et aux exposants de préparer leurs stands. C'est également le jour de réception de tous les paquets envoyés par les exposants. IBM avait prévu de présenter une nouvelle ligne de serveurs sur le salon et avait fait expédier un de ces serveurs au centre de conférences ; malheureusement, il n'avait pas été livré sur leur stand et personne dans le centre de conférences ne savait où pouvait bien se trouver le colis. Malgré de nombreuses heures à chercher dans tous les endroits possibles à

l’intérieur du centre de conférences, nous n’avions pas la moindre piste.

Il se trouve que le gala qui devait avoir lieu quelques jours plus tard avait loué un certain nombre de pièces pour en faire des bureaux et des espaces de stockage. Dans un éclair de génie, le coordinateur de l’évènement qui aidait à la recherche suggéra que nous pourrions chercher dans un de leurs espaces de stockage en espérant que la mallette d’IBM ait été livrée là par accident. La pièce en question était un petit placard de rangement dans lequel nous nous sommes trouvés face à des montagnes de boîtes, du sol jusqu’au plafond, remplies de tickets pour la soirée de gala à venir. Derrière ces boîtes, dans un coin, il y avait une grande mallette bleue avec le logo IBM bien visible. Crise évitée !

Le reste de l’évènement se déroula sans heurts et pratiquement sans incidents. Alors que l’évènement se finissait, une petite foule commença à se former près de grandes fenêtres donnant sur la rue. Alors que je passais à cet endroit, je pris conscience de ce que tout le monde était en train de regarder. Plusieurs silhouettes, toutes vêtues de noir, se déplaçaient sur les toits des bâtiments le long de la rue. Toutes ces silhouettes portaient des fusils de précision et étaient des membres de l’équipe du SWAT de la police de Los Angeles qui se préparaient pour la soirée de gala qui allait avoir lieu plus tard. Ce fut sans conteste une sortie mémorable du centre de conférences.

Pas de chambre à l’hôtel

Le quatrième SCALE a occasionné un nouveau changement de lieu. Cette fois-ci, nous sommes passés à un hôtel à la place d’un centre de conférences. Comme avec les années de plus en plus de personnes voyageaient pour assister à SCALE et séjournaient dans des hôtels proches, nous avons décidé d’étudier la possibilité que SCALE ait lieu dans un hôtel. Nous avons

parcouru la région et avons fini par consulter un organisateur d'évènements pour trouver le bon endroit pour le nôtre. En nous installant dans un hôtel proche de l'aéroport de Los Angeles, la planification commença. Tenir l'évènement dans un hôtel nous a rapidement confrontés à de nouvelles problématiques propres aux hôtels. Une des plus importantes leçons que nous avons alors apprises a été de s'assurer que tous les contrats comportaient une clause convenue d'annulation.

À peu près cinq semaines avant l'évènement, nous avons reçu un appel des responsables du lieu qui nous informaient que leur entreprise annulait notre évènement pour attribuer le lieu à une autre manifestation. Cela a évidemment été un choc pour nous et nous a plongé dans la plus grande confusion. Le contrat avec l'hôtel ne comprenait pas une ligne sur les cas de résiliation pour changement de lieu, mais précisait seulement qu'ils pouvaient annuler la manifestation sans aucun motif.

Après un grand nombre de coups de téléphone et de tractations avec les responsables du lieu initial, ils ont fini par accepter de nous indemniser pour nous aider à migrer vers un lieu de remplacement. Lequel nous a consenti les mêmes conditions pour tout ce qui concernait l'électricité, l'accès à Internet et l'équipement audio et vidéo. Tout s'est bien passé et l'équipe de SCALE en a tiré une précieuse leçon sur la façon de négocier ses futurs contrats.

Rappel !

Tout compte fait, organiser une conférence est une entreprise gratifiante et un excellent moyen de rendre à la communauté ce qu'elle nous a apporté. Les conférences constituent un moment privilégié car elles permettent des échanges en personne dans un monde qui repose couramment sur des moyens de communication virtuels.

Voici les conseils que je donnerais à des organisateurs de conférences :

- commencez modestement, ne vous lancez pas dans un gigantesque évènement dès la première année ;
- saisissez les occasions, faites des erreurs, n’ayez pas peur de l’échec ;
- tout est dans la communication !