Du bon usage des mentors (Libres conseils 5/42)

Vous finirez par savoir tout ce qu'ils ont oublié

Leslie Hawthorn

Gestionnaire de communautés internationalement reconnue, conférencière et auteur, Leslie Hawthorn a plus de 10 ans d'expérience dans la gestion de projets high tech, le marketing et les relations publiques. Elle a récemment rejoint AppFog[(1)] en tant que responsable de la communauté, où elle est chargée du recrutement de développeurs. Auparavant, elle a travaillé comme responsable de communication au laboratoire open source de l'Université de l'état de l'Oregon et

comme responsable de programme au sein de l'équipe open source de Google, où elle a géré le Google Summer of Code[2], créé le concours que l'on connaît maintenant sous le nom Google Code-in et lancé le blog de développement open source de la société.

> *« La documentation la plus importante pour les nouveaux utilisateurs concerne les bases : comment mettre rapidement le logiciel en route, une vue d'ensemble de son fonctionnement et peut-être quelques guides pour les tâches courantes. Or, c'est exactement tout ce que les auteurs de la documentation connaissent parfaitement. Si parfaitement, qu'il peut être difficile pour eux de voir les choses du point de vue du lecteur, et d'énumérer laborieusement les étapes qui (aux yeux des auteurs) semblent évidentes ou inutiles à mentionner. » Karl Fogel, Produire du logiciel libre[3]*

Quand pour la première fois vous commencez à travailler sur un projet de logiciel libre et *open source*, la courbe d'apprentissage est raide et le chemin difficile. Vous risquez de vous retrouver abonné à des listes de diffusion ou dans des salons de discussion avec toutes sortes de gens renommés, comme le créateur de votre langage de programmation favori ou le responsable de votre logiciel préféré, et vous vous demanderez si vous serez un jour suffisamment qualifié pour contribuer efficacement. Ce dont vous n'aurez pas forcément conscience, c'est à quel point ces gens sages ont oublié le long chemin qui les a menés au succès.

Prenons une analogie simple : dans un projet *open source*, le processus d'apprentissage, comme utilisateur ou comme développeur, c'est un peu comme apprendre à faire du vélo. Pour les cyclistes expérimentés, « c'est aussi facile que de monter à vélo ». Vous avez probablement fait du vélo quelques fois et vous comprenez son architecture : une selle, des roues, des pédales et un guidon. Pourtant, vous montez en selle, concentré sur votre avancée et soudainement vous découvrez que ce n'est pas aussi simple que ce que vous pensiez : à quelle hauteur faut-il régler votre selle ? Quel équipement vous faut-il quand vous grimpez une colline ? Quand vous en descendez une ? D'ailleurs, avez-vous vraiment besoin de ce casque ? (un conseil : oui, absolument).

Lorsque vous vous mettez au vélo, vous ne savez même pas quelles questions poser et vous ne les trouverez que dans vos genoux endoloris, des points de côté et des courbatures dans le dos. Même dans ce cas, vos questions ne

correspondront pas toujours aux réponses dont vous avez besoin ; quelqu'un pourrait s'aviser de vous dire d'abaisser la selle quand vous lui dites que vos genoux font mal, mais d'autres peuvent aussi bien supposer que tout ça est nouveau pour vous et que vous finirez bien par le découvrir par vous-même. Ils ont oublié qu'il faut se battre avec les changements de vitesse, se rendre compte qu'on n'a pas les bons éclairages ni les réflecteurs adéquats, comment tourner à gauche en levant la bonne main, parce qu'ils font du vélo depuis si longtemps que tous ces gestes sont pour eux comme une seconde nature.

Le scénario reste le même lorsque vous débutez dans le monde des logiciels libres et *open source*. Lorsque vous compilez un paquet pour la première fois, vous allez inévitablement arriver à un obscur message d'erreur ou un autre genre d'échec. Et lorsque vous demanderez de l'aide, une bonne âme vous dira sans doute : « c'est facile, il suffit de faire make -toto -titi -tata ». Sauf que pour vous, ce n'est pas facile. Il n'y aura probablement pas de documentation pour toto, titi ne fera pas ce qu'il est supposé faire et qu'est ce que ce truc tata avec ses huit homonymes sur Wikipédia ? Évidemment, vous ne voulez pas être un boulet, mais vous allez avoir besoin d'aide pour réussir vraiment à faire quelque chose.

Vous allez peut-être persister à reprendre les mêmes étapes, rencontrer les mêmes échecs, et la frustration ira grandissant. Peut-être que vous allez vous lever pour prendre un café en pensant que vous reviendrez sur le problème plus tard. Ce qu'aucun de nous dans le monde des logiciels libres et *open source* ne voudrait voir se produire, c'est précisément ce qui se passe pour beaucoup : boire cette tasse de café est infiniment meilleur que de se sentir ignorant et intimidé, et vous n'allez pas plus avant dans votre découverte du Libre.

Prenez conscience dès maintenant que vous finirez par connaître ces choses que les experts autour de vous ont oubliées ou qu'ils ne communiquent pas car ces étapes sont évidentes pour eux. Toute personne plus expérimentée que vous est passée par les mêmes affres que vous en ce moment pour apprendre à faire ce que vous vous efforcez de faire. Voici quelques conseils pour rendre votre parcours plus facile :

N'attendez pas trop longtemps avant de demander de l'aide. Personne ne veut être un boulet et personne n'aime avoir l'air perdu. Cela dit, si vous n'arrivez pas à résoudre votre problème après avoir essayé pendant 15 minutes, il est temps de demander de l'aide. Vérifiez dans la documentation sur le site web du

projet que vous utilisez le bon canal IRC, le forum ou la liste de diffusion pour demander de l'aide. De nombreux projets ont des canaux d'aide en ligne spécialement pour les débutants, gardez donc un œil sur des mots tels que mentor, débutant et mise en route.

Parlez de votre processus (de réflexion). Il ne s'agit pas seulement de poser des questions, mais de savoir quelles sont les bonnes questions à poser. Au début, vous ne saurez pas forcément quelles sont ces bonnes questions. Donc quand vous demanderez de l'aide, détaillez ce que vous essayez de faire, les étapes par lesquelles vous êtes passé, et les problèmes que vous avez rencontrés. Signalez aux futurs mentors du canal IRC ou de la liste de diffusion que vous avez lu le manuel en incluant des liens vers la documentation que vous avez lue sur le sujet. Si vous n'avez trouvé aucune documentation, le signaler poliment peut aider.

Apprenez à connaître votre propre valeur. En tant que nouveau contributeur dans un projet, vous êtes un atout précieux. Non pas pour vos connaissances, mais pour votre ignorance. Lorsque vous commencez à travailler sur des logiciels libres et *open source*, rien n'est assez évident à vos yeux et tout mérite donc d'être expliqué. Prenez des notes à propos des problèmes que vous avez rencontrés, et de la façon dont ils ont été résolus. Puis utilisez ces notes pour mettre à jour la documentation du projet, travailler avec la communauté à des démos vidéo ou autres documents de formation pour les cas les plus épineux. Quand vous rencontrez un problème vraiment frustrant, comprenez que vous êtes en position idéale pour faire en sorte que le prochain qui tombera dessus ne rencontrera pas les mêmes difficultés.

1. http://www.appfog.com/ ^
2. http://fr.wikipedia.org/wiki/Google_Summer_of_Code ^
3. http://framabook.org/8-produire-du-logiciel-libre ^