Préparer un logiciel à sa diffusion (Libres conseils 29/42)

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

Traduction Framalang : merlin8282, Sphinx, Julius22, goofy, lerouge, lamessen, Asta, peupleLà, okram

Packager : la voie royale du logiciel libre

Thorn May

Thom May est développeur Debian et membre émérite de la fondation Apache. Il a été parmi les premiers à être engagés chez Canonical, l'entreprise mère d'Ubuntu. Il vit aujourd'hui à Londres et dirige l'unité de développement chez MacMillan Digital Science.

Introduction

Ça fait plus de dix ans que j'ai débuté dans le monde du logiciel libre. J'avais utilisé Debian pendant quelques années à l'université et décidé que je voulais donner quelque chose en retour. J'ai donc commencé un long voyage à travers les étapes permettant de devenir un « nouveau responsable Debian » sans avoir jamais vraiment contribué au logiciel libre auparavant et inquiet qu'un manque d'expérience en C pourrait se révéler être un gros problème.

Il s'avéra que cette inquiétude était largement infondée. En commençant à travailler avec des paquets que j'utilisais

régulièrement, j'ai pu contribuer efficacement. En même temps que mon expérience avec la myriade d'outils et de systèmes que Debian fournit à ses mainteneurs s'accroissait, je devenais plus efficace pour gérer mon temps et j'étais capable de travailler sur un ensemble de paquets plus étendu.

M'occuper de davantage de paquets m'amena à travailler avec un ensemble plus important de systèmes de compilation, de langages de programmation et de boîtes à outils. Cela m'aida également à m'intégrer à la communauté Debian. Aussi rugueuse et dogmatique soit-elle, le fait que la communauté Debian repose sur des mainteneurs doués et expérimentés est l'une des raisons principales pour lesquelles Debian a maintenu son excellence technique sur une si longue période.

À peu près à ce moment, le projet Apache httpd approchait enfin des premières versions bêta de httpd 2.0 qui était restée des années en chantier et était sur le point de subir une mise à jour majeure. L'équipe Apache de Debian avait été plutôt inactive depuis quelques temps – les paquets de la version 1.3 étaient stables et changeaient peu – et n'avait pas prévu d'empaqueter la version 2.0. J'avais un intérêt majeur à garantir que les paquets httpd étaient bien maintenus. Je travaillais comme administrateur système en charge de nombreux serveurs web Apache, il tombait donc sous le sens que je devais relever le défi de la production de paquets pour la nouvelle version.

Avec un ami, nous avons commencé le travail sur les paquets et nous avons rapidement découvert que, alors que le code approchait le niveau de qualité d'une bêta toute fraîche, l'outillage autour de la compilation et de la personnalisation de httpd était hélas manquants, ce qui est assez représentatif de bien des projets logiciels complexes.

Au cours de la majeure partie de l'année – alors que les développeurs en amont stabilisaient leur code et qu'un nombre croissant d'utilisateurs précoces commençaient à tester et à

déployer la nouvelle version – nous avons travaillé dur pour garantir que le système de compilation soit suffisamment flexible et robuste pour satisfaire aux rigoureux prérequis de la politique de Debian. Nous devions non seulement garantir que nos paquets étaient techniquement corrects, mais également nous assurer que notre relation avec les développeurs en amont nous permettait de leur faire remonter des correctifs dès que possible, de les avertir dès que des problèmes de sécurité faisaient surface et de leur transmettre les tests préliminaires des versions candidates.

Mes interactions avec Apache pendant l'empaquetage et la maintenance de httpd 2.0 m'ont amené à m'engager en amont du projet, ce qui signifiait que je pouvais directement contribuer au code. C'est, en général, la dernière étape avant de passer de l'empaquetage d'un logiciel à son développement actif à destination d'un public plus large que celui de votre distribution. À titre personnel, cette reconnaissance m'a donné la confiance pour contribuer à bien plus de projets libres puisque je savais que mon code était de qualité suffisante pour être bien accueilli.

L'évolution, du *packager* au développeur

Comment est-ce arrivé ? La création de paquets, dans sa forme la plus simple, permet d'assurer qu'un projet logiciel donné se conforme à la politique de la distribution ; dans mon cas, Debian. De manière générale, cela signifie configurer le logiciel au moment de la compilation afin qu'il soit installé dans les répertoires idoines spécifiés par le Filesystem Hierarchy Standard, ou FHS (NdT : Norme de la hiérarchie des systèmes de fichiers), que les dépendances aux autres paquets soient correctement spécifiées et que les logiciels fonctionnent normalement sur la distribution.

La création de paquets complexes peut nécessiter la division

d’un projet en amont en de multiples paquets. Par exemple, les bibliothèques et les fichiers d’en-tête permettant à l’utilisateur de compiler le logiciel avec leur bibliothèque sont fournis dans des paquets distincts, et des fichiers dépendant de la plate-forme peuvent être fournis séparément de ceux qui en sont indépendants. S’assurer que le logiciel en amont se déploie correctement dans ces situations nécessitera souvent des changements dans le code. Ces changements sont la première étape vers un travail actif sur un projet, plutôt que le travail parfois passif de création de paquet.

Une fois que votre paquet est disponible dans la distribution, il est exposé à des millions d’utilisateurs potentiels. Ces utilisateurs vont sans aucun doute exécuter votre logiciel selon des pratiques que ni vous, en tant que *packager*, ni vos développeurs en amont n’aviez prévues. Sans surprise, avoir de nombreux utilisateurs implique l’arrivée de nombreux rapports de bogue. Debian, comme la plupart des distributions, encourage ses utilisateurs à lui soumettre directement les rapports de bogue plutôt qu’à chacun des projets individuels en amont. Ceci permet aux mainteneurs de trier les rapports de bogue et d’assurer que les modifications effectuées lors du processus de création de paquet ne sont pas la cause du problème rapporté. Souvent, il peut y avoir un grand nombre d’interactions entre le rapporteur du problème et le mainteneur du paquet avant que les développeurs en amont ne soient impliqués.

Au fur et à mesure que le mainteneur du paquet accroît sa connaissance du projet, il sera en mesure de résoudre directement la plupart des problèmes. Le mainteneur publiera souvent des correctifs de bogue directement dans Debian tout en les faisant remonter en amont, permettant ainsi à la fois une résolution rapide des problèmes et de nombreux tests de correctifs. Une fois qu’un correctif est validé, le mainteneur travaillera alors avec le projet amont pour s’assurer que les changements requis y ont bien lieu, de manière à ce qu’ils

soient disponibles aux autres utilisateurs du logiciel.

Fournir des correctifs de bogue réussis pour des distributions telles que Debian relève souvent d'une forme complexe d'art. Debian fonctionne sur de nombreuses plates-formes, allant des gros serveurs IBM aux smartphones, et la gamme ainsi que la largeur de ces plates-formes révèlent rapidement les approximations dans le code. L'empaqueteur a, la plupart du temps, un accès plus aisé à une gamme de plates-formes plus étendue que les développeurs en amont et constitue, de ce fait, le premier point d'appel quand un problème épineux de portage apparaît. On apprend rapidement à reconnaître les symptômes d'une approximation de la taille d'un pointeur, les problèmes avec les *endianness* et bien d'autres problèmes ésotériques ; cette expérience permet de devenir un programmeur à la fois plus polyvalent et plus prudent.

Lorsqu'un paquet reçoit des corrections de bogues et des améliorations, il est essentiel que ces changements remontent en amont. Trop souvent, la différence entre un paquet et le logiciel définitif en amont peut s'accroître énormément, avec pour conséquence que les deux bases de code deviennent presque entièrement distinctes. Non seulement, cela alourdit la tâche de la maintenance des deux côtés, mais cela peut aussi créer une immense frustration et faire perdre beaucoup de temps en amont dans le cas où un utilisateur de votre paquet rapporte un bogue lié à l'un des changements dans la version empaquetée en amont. Il est en conséquence vital que s'établissent une relation de travail étroite avec la branche amont et une compréhension de la meilleure façon de collaborer entre les deux parties.

La collaboration entre les développeurs et le *packager* peut prendre bien des formes. Que ce soit trouver la bonne voie pour communiquer les rapports de bogue, s'assurer de l'utilisation du bon style de codage, du même usage du système de contrôle de version ou des interactions qui provoquent le moins de frictions possible. Tout ceci amène une bien

meilleure relation avec l'amont et accroît grandement la probabilité que ceux qui y travaillent prendront le temps de vous aider quand vous en aurez besoin.

Une fois que la relation de travail entre l'amont et vous est établie, il devient facile de contribuer plus directement en amont. Ceci peut également se faire de bien des façons différentes. Les premières étapes, simples, peuvent impliquer la synchronisation de n'importe quel rapport de bogue en amont avec ceux de votre distribution, afin d'être sûr que ce double effort impacte la cause primaire et résolve des bogues. Une implication plus directe consiste à développer des fonctionnalités et à changer plus largement que ce qui serait acceptable dans le cadre d'une version empaquetée.

Conclusion

Je pense que les deux choses essentielles que j'aurais aimé connaître lorsque j'ai commencé sont le sens de la communauté que le logiciel libre fait naître et la merveilleuse voie que le *packaging* de logiciel libre ouvre vers le plus vaste monde du logiciel libre

La communauté est essentielle au succès du logiciel libre. Elle se présente sous différentes formes, de la multitude d'utilisateurs souhaitant investir du temps dans l'amélioration de votre logiciel, jusqu'aux pairs d'une distribution ou d'un projet logiciel, qui investissent leur temps et leur énergie à affûter vos compétences et à s'assurer que vos contributions sont aussi bonnes que possible.

La voie qui part du *packaging* pour aller vers le développement est l'une des plus empruntées. Elle présente une courbe d'apprentissage moins raide qu'aborder directement le développement et permet d'acquérir des compétences à un rythme moins soutenu qu'en suivant d'autres chemins.