Tests contre Bogues : une guerre sans fin (Libres conseils 13/42)

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

Traduction Framalang : Floxy, ga3lig, goofy, Astalaseven, Slystone, okram, KoS, Lycoris, 4nti7rust, peupleLa, Luc Didry, + Julius22

Même en multipliant les regards, les bogues ne sautent pas aux yeux.

Ara Pulido

Ara Pulido est ingénieure d'essais pour Canonical, d'abord comme membre de l'équipe assurance qualité d'Ubuntu (QA team), et maintenant dans le cadre de l'équipe de certification du matériel. Même si elle a commencé sa carrière en tant que développeuse, elle a vite découvert que ce qu'elle aimait vraiment, c'était tester les logiciels. Elle est très intéressée par les nouvelles techniques d'analyse et tente d'utiliser son savoir-faire pour améliorer Ubuntu.

Les tests maison ne suffisent pas

Je me suis impliquée dans le logiciel libre dès le début de mes études à l'Université de Grenade. Là-bas, avec des amis, nous avons fondé un groupe local d'utilisateurs de Linux et organisé plusieurs actions pour promouvoir le logiciel libre. Mais, depuis que j'ai quitté l'université, et jusqu'à ce que je commence à travailler chez Canonical, ma carrière professionnelle s'est déroulée dans l'industrie du logiciel propriétaire, d'abord comme développeuse puis comme testeuse.

Lorsque l'on travaille au sein d'un projet de logiciel propriétaire, les ressources

pour tester sont très limitées. Une petite équipe reprend le travail initié par les développeurs avec les tests unitaires, utilisant leur expérience pour trouver autant de bogues que possible afin de mettre à disposition de l'utilisateur final un produit aussi abouti que possible. Dans le monde du logiciel libre, en revanche, tout est différent.

Lors de mon embauche chez Canonical, hormis la réalisation de mon rêve d'avoir un travail rémunéré au sein d'un projet de logiciel libre, j'ai été émerveillée par les possibilités des activités de test dans le cadre d'un tel projet. Le développement du produit s'effectue de manière ouverte, et les utilisateurs ont accès au logiciel dès son commencement, ils le testent et font des rapports de bogues dès que c'est nécessaire. C'est un nouveau monde rempli de beaucoup de possibilités pour une personne passionnée par les tests. Je voulais en profiter au maximum.

Comme beaucoup de personnes, je pensais que les tests « maison », c'est-à-dire l'utilisation par soi-même du logiciel que l'on envisage de mettre à disposition, était l'activité de test la plus importante qu'on puisse mener dans le logiciel libre. Mais si, selon la formule de Raymond dans La cathédrale et le bazar « avec suffisamment d'observateurs, tous les bogues sautent aux yeux », alors comment se fait-il qu'avec ses millions d'utilisateurs Ubuntu comporte encore des bogues sérieux à chaque nouvelle version ?

La première chose dont je me suis aperçue quand j'ai commencé à travailler chez Canonical c'est que les activités de test organisées étaient rares ou inexistantes. Les seules sessions de test qui étaient d'une certaine façon organisées se présentaient sous la forme de messages électroniques envoyés à une liste de diffusion, manière de battre le rappel pour tester un paquetage dans la version de développement d'Ubuntu. Je ne pense pas que cela puisse être considéré comme une vraie procédure de test, mais simplement comme une autre forme de « test maison ». Cette sorte de test génère beaucoup de doublons, car un bogue facile à débusquer sera documenté par des centaines de personnes. Malheureusement le bogue potentiellement critique, vraiment difficile à trouver, a de bonnes chances de passer inaperçu en raison du bruit créé par les autres bogues, et ce même si quelqu'un l'a documenté.

En progrès

La situation s'améliore-t-elle ? Sommes-nous devenus plus efficaces pour les tests au sein des projets de développement libre ? Oui, j'en suis convaincue.

Pendant les derniers cycles de développement d'Ubuntu, nous avons commencé bon nombre de sessions de test. La gamme des objectifs pour ces sessions est large, elle comprend des domaines comme de nouvelles fonctionnalités de bureau, des tests de régression, des tests de pilotes X.org ou des tests de matériel d'ordinateur portable. Les résultats sont toujours suivis et ils s'avèrent vraiment utiles pour les développeurs, car ils leur permettent de savoir si les nouveautés fonctionnent correctement, au lieu de supposer qu'elles fonctionnent correctement à cause de l'absence de bogues.

En ce qui concerne les outils d'assistance aux tests, beaucoup d'améliorations ont été apportées :

- Apport(1) a contribué à augmenter le niveau de détail des bogues signalés concernant Ubuntu : les rapports de plantage incluent toutes les informations de débogage et leurs doublons sont débusqués puis marqués comme tels ; les utilisateurs peuvent signaler des bogues sur base de symptômes, etc.
- Le Launchpad(2), avec ses connexions en amont, a permis d'avoir une vue complète des bogues – sachant que les bogues qui se produisent dans Ubuntu se situent généralement dans les projets en amont, et permet aux développeurs de savoir si les bogues sont en cours de résolution.
- Firefox, grâce à son programme et à son extension Test Pilot, mène des tests sans qu'on ait à quitter le navigateur(3). C'est, à mon sens, une bien meilleure façon de rallier des testeurs qu'une liste de diffusion ou un canal IRC.
- L'équipe Assurance Qualité d'Ubuntu teste le bureau en mode automatique et rend compte des résultats toutes les semaines(4), ce qui permet aux développeurs de vérifier très rapidement qu'il n'y a pas eu de régression majeure pendant le développement.

Cependant, malgré l'amélioration des tests dans les projets de logiciel libre il reste encore beaucoup à faire.

Pour aller plus loin

Les tests nécessitent une grande expertise, mais sont encore considérés au sein de la communauté du le logiciel libre comme une tâche ne demandant pas beaucoup d'efforts. L'une des raisons pourrait être que la manière dont on les réalise est vraiment dépassée et ne rend pas compte de la complexité croissante du monde du logiciel libre durant la dernière décennie. Comment est-il possible que, malgré la quantité d'innovations amenées par les communautés du logiciel libre, les tests soient encore réalisés comme dans les années 80 ? Il faut nous rendre à l'évidence, les scénarios de tests sont ennuyeux et vite obsolètes. Comment faire grandir une communauté de testeurs supposée trouver des bogues avérés si sa tâche principale est de mettre à jour les scénarios de test ?

Mais comment améliorer la procédure de test ? Bien sûr, nous ne pouvons pas nous débarrasser des scénarios de test, mais nous devons changer la façon dont nous les créons et les mettons à jour. Nos testeurs et nos utilisateurs sont intelligents, alors pourquoi créer des scripts pas-à-pas ? Ils pourraient aisément être remplacés par une procédure de test automatique. Définissons plutôt une liste de tâches que l'on réalise avec l'application, et certaines caractéristiques qu'elle devrait posséder. Par exemple, « l'ordre des raccourcis dans le lanceur doit pouvoir être modifié », ou « le démarrage de LibreOffice est rapide ». Les testeurs trouveront un moyen de le faire, et créeront des scénarios de test en parallèle des leurs.

Mais ce n'est pas suffisant, nous avons besoin de meilleurs outils pour aider les testeurs à savoir ce qu'ils testent, où et comment. Pourquoi ne pas avoir des API (interfaces de programmation) qui permettent aux développeurs d'envoyer des messages aux testeurs à propos des nouvelles fonctionnalités ou des mises à jour qui doivent être testées ? Pourquoi pas une application qui nous indique quelle partie du système doit être testée ? en fonction des tests en cours ? Dans le cas d'Ubuntu, nous avons les informations dans le Launchpad (il nous faudrait aussi des données sur les tests, mais au moins nous avons des données sur les bogues). Si je veux démarrer une session de test d'un composant en particulier j'apprécierais vraiment de savoir quelles zones n'ont pas encore été testées ainsi qu'une liste des cinq bogues comptant le plus de doublons pour cette version en particulier afin d'éviter de les documenter une fois de plus. J'aimerais avoir toutes ces informations sans avoir à quitter le bureau que je suis en train de tester. C'est

quelque chose que Firefox a initié avec Test Pilot, bien qu'actuellement l'équipe rassemble principalement les données sur l'activité du navigateur.

La communication entre l'amont et l'aval et vice-versa doit aussi être améliorée. Pendant le développement d'une distribution, un bon nombre des versions amont sont également en cours de développement, et ont déjà une liste des bogues connus. Si je suis un testeur de Firefox sous Ubuntu, j'aimerais avoir une liste des bogues connus aussitôt que le nouveau paquet est poussé dans le dépôt. Cela pourrait se faire à l'aide d'une syntaxe reconnue pour les notes de versions, syntaxe qui pourrait ensuite être facilement analysée. Les rapports de bogue seraient automatiquement remplis et reliés aux bogues amont. Encore une fois, le testeur devrait avoir facilement accès à ces informations, sans quitter son environnement de travail habituel.

Les tests, s'ils étaient réalisés de cette manière, permettraient au testeur de se concentrer sur les choses qui comptent vraiment et font de la procédure de test une activité qualifiée ; se concentrer sur les bogues cachés qui n'ont pas encore été découverts, sur les configurations et environnements spéciaux, sur la création de nouvelles manières de casser le logiciel. Et, in fine, s'amuser en testant.

Récapitulons

Pour ce que j'en ai vu ces trois dernières années, les tests ont beaucoup progressé au sein d'Ubuntu et des autres projets de logiciels libres dans lesquels je suis plus ou moins impliquée, mais ce n'est pas suffisant. Si nous voulons vraiment améliorer la qualité du logiciel libre, nous devons commencer à investir dans les tests et innover dans la manière de les conduire, de la même façon que nous investissons dans le développement. Nous ne pouvons pas tester le logiciel du XXIe siècle avec les techniques du XXe siècle. Nous devons réagir. Qu'il soit open source ne suffit plus à prouver qu'un logiciel libre est de bonne qualité. Le logiciel libre sera bon parce qu'il est open source et de la meilleure qualité que nous puissions offrir.

1 http://wiki.ubuntu.com/Apport

2 http://launchpad.net

3 http://testpilot.mozillalabs.com

4 http://reports.qa.ubuntu.com/reports/desktop-testing/natty