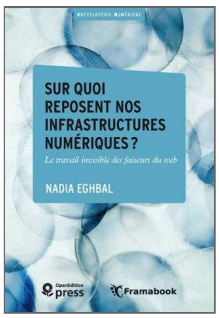

Nadia Eghbal

Sur quoi reposent nos infrastructures numériques ?
Le travail invisible des faiseurs du web

OpenEdition Press

Pourquoi les problèmes de support des infrastructures numériques sont de plus en plus pressants

DOI : 10.4000/books.oep.1817
Éditeur : OpenEdition Press, Framabook
Lieu d'édition : OpenEdition Press, Framabook
Année d'édition : 2017
Collection : Encyclopédie numérique
ISBN électronique : 9782821894938

POURQUOI LES PROBLÈMES DE SUPPORT DES INFRASTRUCTURES NUMÉRIQUES SONT DE PLUS EN PLUS PRESSANTS

L'*open source*, grâce à ses points forts cités plus haut dans cet ouvrage[1], est rapidement en train de devenir un standard pour les projets d'infrastructure numérique et dans le développement logiciel en général. Black Duck, une entreprise qui aide ses clients à gérer des programmes *open source*, dirige une enquête annuelle qui interroge les entreprises sur leur utilisation de l'*open source*. (Cette enquête est l'un des rares projets de banque de données qui existe sur le sujet.) Dans leur étude de 2015[2], 78 % des 1 300 entreprises interrogées déclarent que les logiciels qu'elles ont créés pour leurs clients sont construits grâce à l'*open source*, soit presque le double du chiffre de 2010.

L'*open source* a vu sa popularité s'accroître de manière impressionnante ces cinq dernières années, pas seulement grâce à ses avantages évidents pour les développeurs et les consommateurs, mais également grâce à de nouveaux outils qui rendent la collaboration plus facile. Pour comprendre pourquoi les infrastructures numériques rencontrent des problèmes de support grandissants, nous devons saisir comment le développement de logiciels *open source* prolifère.

1. Voir partie 1, chapitre 2 et chapitre 3.
2. Voir *The Ninth Annual Future of Open Source Survey* sur le site BlackDuck. Les résultats 2016 de cette même étude sont également disponibles.

GitHub, un espace standardisé pour collaborer sur du code

On n'insistera jamais trop sur le rôle clé de GitHub dans la diffusion de l'*open source* auprès du grand public. L'*open source* a beau exister depuis près de trente ans, jusqu'en 2008, contribuer à des projets *open source* n'était pas si facile. Le développeur motivé devait d'abord découvrir qui était le mainteneur du projet, trouver une manière de le contacter, puis proposer ses changements en utilisant le format choisi par le mainteneur (par exemple une liste courriel ou un forum). GitHub a standardisé ces méthodes de communication : les mainteneurs sont listés de façon transparente sur la page du projet, et les discussions sur les changements proposés ont lieu sur la plateforme GitHub.

GitHub a aussi créé un vocabulaire qui est désormais standard parmi les contributeurs à l'*open source*, tel que la *pull request* (où un développeur soumet à l'examen de ses pairs une modification à un projet), et changé le sens du terme *fork* (historiquement, créer une copie d'un projet et le modifier pour le transformer en un nouveau projet ; littéralement, « *fork* » signifie « bifurcation »). Avant GitHub, *forker* un projet revenait à dire qu'il y avait un différend irréconciliable au sujet de la direction qu'un projet devrait prendre. *Forker* était considéré comme une action grave : si un groupe de développeurs décidait de *forker* un projet, cela signifiait qu'il se scindait en deux factions idéologiques. *Forker* était aussi utilisé pour développer un nouveau projet qui pouvait avoir une utilisation radicalement différente du projet initial.

Ce type de « *fork* de projet » existe toujours, mais GitHub a décidé d'utiliser le terme *fork* pour encourager à davantage d'activité sur sa plateforme. Un *fork* GitHub, contrairement à un *fork* de projet, est une copie temporaire d'un projet sur laquelle on effectue des modifications et qui est généralement refusionnée avec le projet. Le *fork* en tant que pratique quotidienne sur GitHub a ajouté une connotation positive, légère au terme : à savoir prendre l'idée de quelqu'un et l'améliorer.

GitHub a aussi aidé à standardiser l'utilisation d'un système de contrôle de version appelé Git. Les systèmes de contrôle de versions sont un outil qui permet de garder une trace de

chaque contribution apportée sur un morceau de code précis. Par exemple, si le développeur 1 et le développeur 2 corrigent simultanément différentes parties du même code, enregistrer chaque changement dans un système de contrôle de version permet de faire en sorte que leurs changements n'entrent pas en conflit.

Il existe plusieurs systèmes de contrôle de versions, par exemple Apache Subversion et Concurrent Versions System (CVS). Avant GitHub, Git était un système de contrôle de version assez méconnu. En 2010, Subversion était utilisé dans 60 % des projets logiciels, contre 11 % pour Git[3].

C'est Linus Torvalds, le créateur de Linux, qui a conçu Git en 2005. Son intention était de mettre à disposition un outil à la fois plus efficace et plus rapide, qui permette de gérer de multiples contributions apportées par de nombreux participants. Git était vraiment différent des systèmes de contrôle de version précédents, et donc pas forcément facile à adopter, mais son *workflow*[4] décentralisé a résolu un vrai problème pour les développeurs.

GitHub a fourni une interface utilisateur intuitive pour les projets *open source* qui recourent à Git, ce qui rend l'apprentissage plus facile pour les développeurs. Plus les développeurs utilisent GitHub, plus cela les incite à continuer d'utiliser Git. En 2016, Git est utilisé par 38 % des projets de logiciels, tandis que la part de Subversion est tombée à 47 %[5]. Bien que Subversion soit encore le système de contrôle de version le plus populaire, son usage décline. L'adoption généralisée de Git rend plus facile pour un développeur la démarche de se joindre à un projet sur GitHub, car la méthode pour faire des modifications et pour les communiquer est la même pour tous les projets. Apprendre à contribuer à un seul des projets vous permet d'acquérir les compétences pour contribuer à des centaines d'autres. Ce n'était pas le cas avant GitHub, où des systèmes de contrôle de versions différents étaient utilisés pour chaque projet.

3. Voir Stephen O'Grady, « DVCS and Git Usage in 2013 », *RedMonk*, 19/12/2013.
4. Le *workflow* est un moyen de représenter un flux de travail. Pour en savoir plus, voir l'article « Workflow » sur Wikipédia.
5. Voir les données sur BlackDuck.com. Les données du site sont actualisées régulièrement. Les chiffres mentionnés ont été consultés le 06/01/2016. Au 16/01/2017, la part de Git est passée à 40 % et celle de Subversion à 46 %.

Enfin, GitHub a créé un espace de sociabilité qui permet de discuter et de tisser des liens au-delà de la stricte collaboration sur du code. La plateforme est devenue *de facto* une sorte de communauté pour les développeurs, qui l'utilisent pour communiquer ensemble et exposer leur travail. Ils peuvent y démontrer leur influence et présenter un portfolio de leur travail comme jamais auparavant.

Les usages de GitHub sont un reflet de son ascension vertigineuse. En 2011[6], il n'y avait que 2 millions de dépôts (*repository*). Aujourd'hui, GitHub a 14 millions d'utilisateurs et plus de 35 millions de dépôts[7] (ce qui inclut aussi les dépôts forkés, le compte des dépôts uniques s'élève plutôt à 17 millions). Brian Doll, de chez GitHub, a noté qu'il a fallu quatre ans pour atteindre le million de dépôts, mais que passer de 9 millions à 10 millions n'a pris que quarante-huit jours[8].

En comparaison, SourceForge, la plateforme qui était la plus populaire pour héberger du code *open source* avant l'apparition de GitHub, avait 150 000 projets en 2008. Environ 18 000 d'entre eux étaient actifs[9].

Stack Overflow, un espace standard pour s'entraider sur du code

L'une des autres plateformes importantes de l'*open source* est Stack Overflow, un site de questions/réponses populaire parmi les développeurs, créé en 2008 par Jeff Atwood (développeur déjà mentionné précédemment[10]) et par le blogueur Joel Spolsky. En avril 2014, Stack Overflow avait plus de 4 millions d'utilisateurs enregistrés et plus de 11 millions de questions résolues[11] (à noter qu'il n'est pas nécessaire de s'enregistrer pour voir les questions ou leurs réponses).

6. Voir GitHub, « Those are some big numbers », sur le blog de GitHub, 20/04/2011.
7. Chiffres issus de l'article « GitHub » sur Wikipédia.
8. Voir GitHub, « 10 Million Repositories » sur le blog de Github, 23/12/2013.
9. Amit Deshpande et Dirk Riehle, « The Total Growth of Open Source », *Software Research and the Industry*, Proceedings of the Fourth Conference on Open Source Systems, 2008.
10. Voir partie 3, chapitre 1 et partie 1, chapitre 2.
11. Chiffres issus de l'article « Stack Overflow » sur Wikipédia.

Stack Overflow est devenu *de facto* une plateforme d'entraide pour les développeurs, qui peuvent poser des questions de programmation, trouver des réponses à des problèmes de code spécifiques ou juste échanger des conseils sur la meilleure façon de créer un aspect précis d'un logiciel. On pourrait définir la plateforme comme un « support client » participatif pour les développeurs à travers le monde. Même si Stack Overflow n'est pas un endroit où l'on écrit directement du code, c'est un outil de collaboration essentiel pour les développeurs individuels, qui facilite grandement la résolution de problèmes et permet de coder plus efficacement. Cela signifie qu'un développeur individuel est capable de produire plus, en moins de temps, ce qui améliore le rendement global. Stack Overflow a également permis à certains utilisateurs d'apprendre de nouveaux concepts de développement (ou même de s'initier au code tout court) et a rendu le codage plus facile et plus accessible à tous.

Tendances macro dans un paysage en mutation constante

La popularité hors normes de l'*open source* a amené à des changements significatifs dans la manière dont les développeurs d'aujourd'hui parlent, pensent et collaborent pour des logiciels.

Premièrement, les attentes et exigences en termes de licence ont changé, reflétant un monde qui considère désormais l'*open source* comme une norme, et pas l'exception : un triomphe sur l'univers propriétaire des années 1980. Les politiques de GitHub et de Stack Overflow reflètent toutes deux cette réalité.

Dès le départ, Stack Overflow a choisi d'utiliser une licence Creative Commons de type CC-BY-SA[12] pour tous les contenus postés sur son site. La licence était cependant limitante, car elle requérait des utilisateurs qu'ils mentionnent l'auteur de chaque morceau de code utilisé, et qu'ils placent leurs propres contributions sous la même licence.

Beaucoup d'utilisateurs choisissaient d'ignorer la licence ou n'étaient même pas au courant de ses restrictions, mais pour

12. Voir le Contenu de la licence CC-BY-SA sur creativecommons.org.

les développeurs travaillant avec des contraintes plus strictes (par exemple dans le cadre d'une entreprise), elle rendait Stack Overflow compliqué à utiliser. S'ils postaient une question demandant de l'aide pour leur code, et qu'une personne extérieure réglait le problème, alors légalement, ils devaient attribuer le code à cette personne.

En conséquence, les dirigeants de Stack Overflow ont annoncé leur volonté de déplacer toutes les nouvelles contributions de code sous la licence MIT[13], qui est une licence *open source* comportant moins de restrictions[14]. En avril 2016, ils débattent encore activement et sollicitent des retours de leur communauté pour déterminer le meilleur moyen de mettre en œuvre un système plus permissif. Cette démarche est un encouragement à la fois pour la popularité de Stack Overflow et pour la prolifération de l'*open source* en général. Qu'un développeur travaillant dans une grosse entreprise de logiciel puisse légalement inclure le code d'une personne complètement extérieure dans un produit pour lequel il est rémunéré est en effet un accomplissement pour l'*open source*.

À l'inverse, GitHub fit initialement le choix de ne pas attribuer de licence par défaut aux projets postés sur sa plateforme, peut-être par crainte que cela ne freine son adoption par les utilisateurs et sa croissance[15]. Ainsi, les projets postés sur GitHub accordent le droit de consulter et de *forker* le projet, mais sont à part ça sous copyright, sauf si le développeur spécifie qu'il s'agit d'une licence *open source*.

En 2013, GitHub décida enfin de prendre davantage position sur la question des licences, avec notamment la création et la promotion d'un microsite, choosealicense.com[16], pour aider les utilisateurs à choisir une licence pour leur projet. Ils encouragent aussi désormais leurs utilisateurs à choisir une licence parmi une liste d'options au moment de créer un nouveau dépôt[17].

13. Voir The MIT License – Clarity on Using Code on Stack Overflow and Stack Exchange sur meta.stackexchange.com, 14/01/2016.
14. « A New Code License: The MIT, this time with Attribution Required », sur meta.stackexchange.com, 15/01/2016.
15. Simon Philipps, « GitHub needs to take open source seriously », *Infoworld*, 30/11/2012.
16. Voir le site Choosealicense.com.
17. *Idem*.

Ce qui est intéressant, cependant, c'est que la plupart des développeurs ne se préoccupaient pas de la question des licences : soit ils ignoraient que leurs projets *open source* n'étaient pas légalement protégés, soit ils n'en tenaient pas compte. Une étude informelle réalisée en 2013 par le Software Freedom Law Center (Centre du droit de la liberté des logiciels) sur un échantillon de 1,6 million de dépôts GitHub révéla que seuls 15 % d'entre eux avaient spécifié une licence[18]. Aussi, les entretiens avec des développeurs réalisés pour cette étude suggèrent que beaucoup se fichent de spécifier une licence, ou se disent que si elle leur est exigée, ils pourront toujours en ajouter une ultérieurement.

Ce manque d'intérêt pour les licences a amené James Governor, cofondateur de la firme d'analyse de développeurs Red Monk, à constater en 2012 que « les jeunes dévs aujourd'hui font du POSS – Post open source software[19] –, envoient chier les licences et la gestion, contribuent juste à GitHub ». En d'autres termes, faire de l'information ouverte par défaut est devenu une telle évidence culturelle aujourd'hui que les développeurs ne s'imaginent plus faire les choses autrement – un contexte bien différent de celui des rebelles politisés du logiciel libre des années 1980. Ce retournement des valeurs, quoique inspirant au niveau global, peut cependant amener à des complications légales pour les individus quand leurs projets gagnent en popularité ou sont utilisés à des fins commerciales.

Mais, en rendant le travail collaboratif sur le code aussi facile et standardisé, l'*open source* se retrouve aux prises avec une série d'externalités perverses.

L'*open source* a rendu le codage plus facile et plus accessible au monde. Cette accessibilité accrue, à son tour, a engendré une nouvelle catégorie de développeurs, moins expérimentés, mais qui savent comment utiliser les composants préfabriqués par d'autres pour construire ce dont ils ont besoin.

En 2012, Jeff Atwood, cofondateur de Stack Overflow, rédigea un article de blog intitulé ironiquement « Pitié, n'apprenez

18. Voir Neil McAllister, « Study: Most projects on GitHub not open source licensed », *The Register*, 18/04/2013.
19. Déclaration issue d'un tweet du 17/09/2012. *Dévs* est une abréviation de développeur. PPOS pour Post Open Source Software soit Post logiciel *open source*.

pas à coder », où il se plaint de la mode des stages et des écoles de code. Tout en se félicitant du désir des personnes non techniciennes de comprendre le code d'un point de vue conceptuel, Atwood émet des réserves[20] et se demande si « introduire parmi la main-d'œuvre ces codeurs naïfs, novices, voire même-pas-vraiment-sûrs-d'aimer-ce-truc-de-programmeur, a vraiment des effets positifs pour le monde ».

Dans ces circonstances, le modèle de développement de l'*open source* change de visage. Avant l'ascension de GitHub, il y avait moins de projets *open source*. Les développeurs formaient donc un groupe plus restreint, mais en moyenne plus expérimenté : ceux qui utilisaient du code partagé par d'autres étaient susceptibles d'être également ceux qui contribuent en retour.

Aujourd'hui, l'intense développement de l'éducation au code implique que de nombreux développeurs inexpérimentés inondent le marché. Cette dernière génération de développeurs novices emprunte du code libre pour écrire ce dont elle a besoin, mais elle est rarement capable, en retour, d'apporter des contributions substantielles aux projets. Beaucoup sont également habitués à se considérer comme des « utilisateurs » de projets *open source*, davantage que comme les membres d'une communauté. Les outils *open source* étant désormais plus standardisés et faciles à utiliser, il est bien plus simple aujourd'hui pour un néophyte de débarquer sur un forum GitHub et d'y faire un commentaire désobligeant ou une requête exigeante – ce qui épuise et exaspère les mainteneurs.

Cette évolution démographique a aussi conduit à un réseau de logiciels bien plus fragmenté, avec de nombreux développeurs qui publient de nouveaux projets et qui créent un réseau embrouillé d'interdépendances. Se qualifiant lui-même de « développeur-pie en rémission »[21], Drew Hamlett a écrit en janvier 2016 un post de blog devenu très populaire intitulé « Le triste état du développement web »[22]. L'article traite de

20. « Please Don't Learn to Code », *Coding Horror*, 15/05/2012.
21. Voir partie 2, chapitre 1 et chapitre 3. « Pie » est un surnom pour les développeurs opportunistes, d'après le nom de l'oiseau, la pie réputée voleuse (NdT).
22. Drew Hamlett, « The Sad State of Web Development », *Medium.com*, 10/01/2016.

l'évolution du développement web, se référant spécifiquement à l'écosystème Node.js :

> Les individus qui sont restés dans la communauté Node ont sans aucun doute créé l'écosystème le plus techniquement compliqué qui ait jamais existé. Personne n'arrive à y créer une bibliothèque qui fasse quoi que ce soit. Chaque projet qui émerge est encore plus ambitieux que le précédent… mais personne ne construit rien qui fonctionne concrètement. Je ne comprends vraiment pas. La seule explication que j'ai trouvée, c'est que les développeurs sont juste continuellement en train d'écrire et de réécrire en boucle des applis Node.js.

Aujourd'hui, il y a tellement de projets qui sont élaborés et publiés qu'il est tout simplement impossible pour chacun d'eux de développer une communauté suffisamment importante et viable, avec des contributeurs réguliers qui discuteraient avec passion des modifications à apporter lors de débats approfondis sur des listes courriels. Au lieu de cela, beaucoup de projets sont maintenus par une ou deux personnes seulement, alors même que la demande des utilisateurs pour ces projets peut excéder le travail nécessaire à leur simple maintenance.

GitHub a rendu simples la création et la contribution à de nouveaux projets. Cela a été une bénédiction pour l'écosystème *open source*, car les projets se développent plus rapidement. Mais cela peut aussi parfois tourner à la malédiction pour les mainteneurs de projets, car davantage de personnes peuvent facilement signaler des problèmes ou réclamer de nouvelles fonctionnalités, sans pour autant contribuer elles-mêmes en retour. Ces interactions superficielles ne font qu'alourdir la charge de travail des mainteneurs, dont on attend qu'ils répondent à une quantité croissante de requêtes.

Il ne serait pas déraisonnable d'affirmer qu'un monde « post-*open source* » implique une réflexion non seulement autour des licences, ainsi que James Governor l'exprimait dans son commentaire originel, mais aussi autour du processus de développement lui-même.

Noah Kantrowitz, développeur Python de longue date et membre de la Python Software Foundation, a résumé ce changement dans un post de blog[23] souvent cité :

> Dans les débuts du mouvement *open source*, il y avait assez peu de projets, et en général, la plupart des gens qui utilisaient un projet y contribuaient en retour d'une façon ou d'une autre. Ces deux choses ont changé à un point difficilement mesurable.
>
> […] Alors même que nous allons de plus en plus vers des outils de niche, il devient compliqué de justifier l'investissement en temps requis pour devenir contributeur. « Combler son propre besoin » est toujours une excellente motivation, mais il est difficile de construire un écosystème là-dessus.
>
> L'autre problème est le déséquilibre de plus en plus important entre producteurs et consommateurs. Avant, cela s'équilibrait à peu près. Tout le monde investissait du temps et des efforts dans les Communs et tout le monde en récoltait les bénéfices. Ces temps-ci, très peu de personnes font cet effort et la grande majorité ne fait que bénéficier du travail de ceux qui s'impliquent.
>
> Ce déséquilibre s'est tellement enraciné qu'il est presque impensable pour une entreprise de rendre (en temps ou en argent) ne serait-ce qu'une petite fraction de la valeur qu'elle tire des Communs.

Cela ne veut pas dire qu'il n'existe plus de grands projets *open source* avec des communautés de contributeurs fortes (Node.js, dont on parlera plus tard, est un exemple de projet qui est parvenu à ce statut). Cela signifie qu'à côté de ces réussites, il y a une nouvelle catégorie de projets qui est défavorisée par les normes et les attentes actuelles de l'*open source* et que

23. Noah Kantrowitz, « Funding FOSS », coderanger.net, 01/07/2015.

le comportement qui dérive de ces nouvelles normes affecte même des projets plus importants et plus anciens.

Hynek Schlawack, *fellow*[24] de la Python Software Foundation et contributeur à des projets d'infrastructure Python, exprime[25] ses craintes au sujet d'un futur où il y aurait une demande plus forte, mais seulement une poignée de contributeurs solides :

> Ce qui me frustre le plus, c'est que nous n'avons jamais eu autant de développeurs Python et aussi peu de contributions de haute qualité. Dès que des développeurs clés comme Armin Ronacher ralentissent leur travail, la communauté tout entière le ressent aussitôt. Le jour où Paul Kehrer arrêtera de travailler sur PyCA, on sera très mal. Si Hawkowl interrompt son travail de portage, Twisted ne sera jamais sur Python 3 et Git.
>
> La communauté est en train de se faire saigner par des personnes qui créent plus de travail qu'elles n'en fournissent. […] En ce moment, tout le monde bénéficie de ce qui a été construit, mais la situation se détériore à cause du manque de financements et de contributions. Ça m'inquiète, parce Python est peut-être très populaire aujourd'hui, mais une fois que les conséquences se feront sentir, les opportunistes partiront aussi vite qu'ils étaient arrivés.

Pour la plupart des développeurs, il n'y a guère que cinq ans peut-être que l'*open source* est devenue populaire. La large communauté des concepteurs de logiciel débat rarement de la pérennité à long terme de l'*open source*, et n'a parfois même pas conscience du problème. Avec l'explosion du nombre de nouveaux développeurs qui utilisent du code partagé sans contribuer en retour, nous construisons des palaces sur une infrastructure en ruines.

24. Le terme *fellow* est intraduisible sans longue périphrase, pour en savoir plus, voir l'article « Fellow » sur Wikipédia.
25. Source : entretien par mail avec l'auteur.