Notre gitlab évolue en Framagit. C'est très efficace !

> *Warning : cet article parle de forge logicielle qui sert à développer collaborativement du code. Il est donc un peu velu et technique, mais il fera plaisir aux plus « barbu-e-s » d'entre vous !*

Préviousselaid, chez Framasoft : nous avions besoin d'une forge logicielle comme outil interne à l'asso… parce que même si nous ne développons pas (ou exceptionnellement) de logiciel libre ; les mettre en avant, les améliorer (parfois), les promouvoir et ouvrir des services au monde, ben ça demande de créer, maintenir, échanger et améliorer du code !

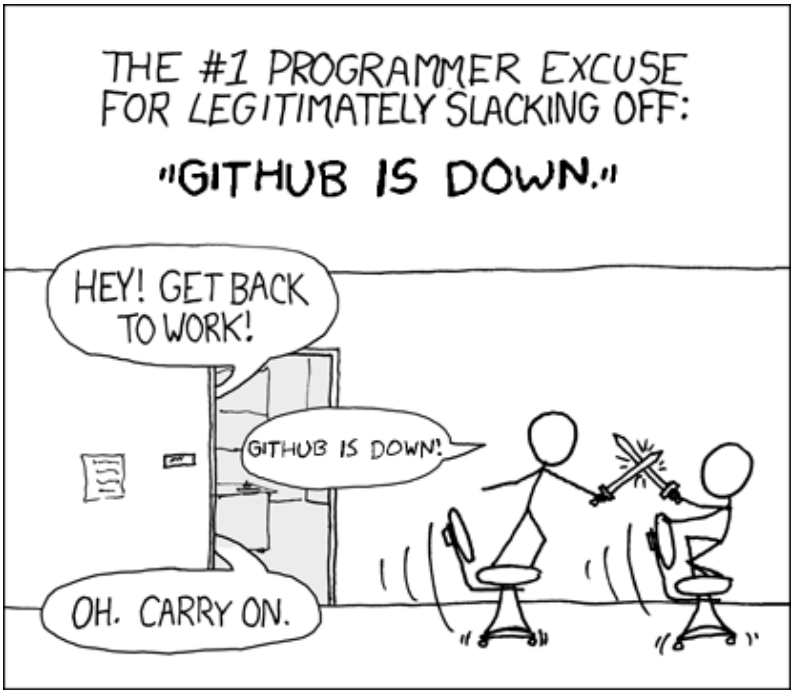
Nous nous étions donc installé Gitlab à la main, sur un coin de serveur, juste pour nous… Étant les seuls utilisateurs, on s'est dit que ce ne serait pas grave s'il n'était pas toujours à jour, à traquer la dernière version… (oui : nous sommes moins exigeants sur nos outils internes que pour les services que nous ouvrons au grand public ^^).

Franchement, merci Google !

Merci, parce qu'à chaque fois que vous prenez des décisions unilatérales aux dépens de vos utilisateurs-produits, vous nous offrez l'occasion de prouver que le Libre offre des alternatives bien plus respectueuses des personnes qui vous ont confié leur vie numérique (et leur code).

Le jour où nous avons appris que Google Code fermait ses portes, nous avons donc décidé d'ouvrir les nôtres. Cela nous a aussi permis de sensibiliser au fait que, dans le mode des

codeurs et développeuses, [GitHub est devenu un point central et monopolistique assez inquiétant](#).



L'excuse n°1 des programmeurs pour se lâcher sans scrupules :
« GitHub est en panne »
— Hé, au boulot les gars ! — Github est en panne !
— Ah bon, continuez alors.

Forcément, l'ouverture à tous de notre *git* et les nouvelles fonctionnalités des nouvelles versions de Gitlab (une nouvelle version tous les 22 du mois) nous ont incités à mettre à jour plus régulièrement, ce qui prend plusieurs heures à chaque fois… et plusieurs fois par mois, car des versions correctives sont régulièrement publiées.

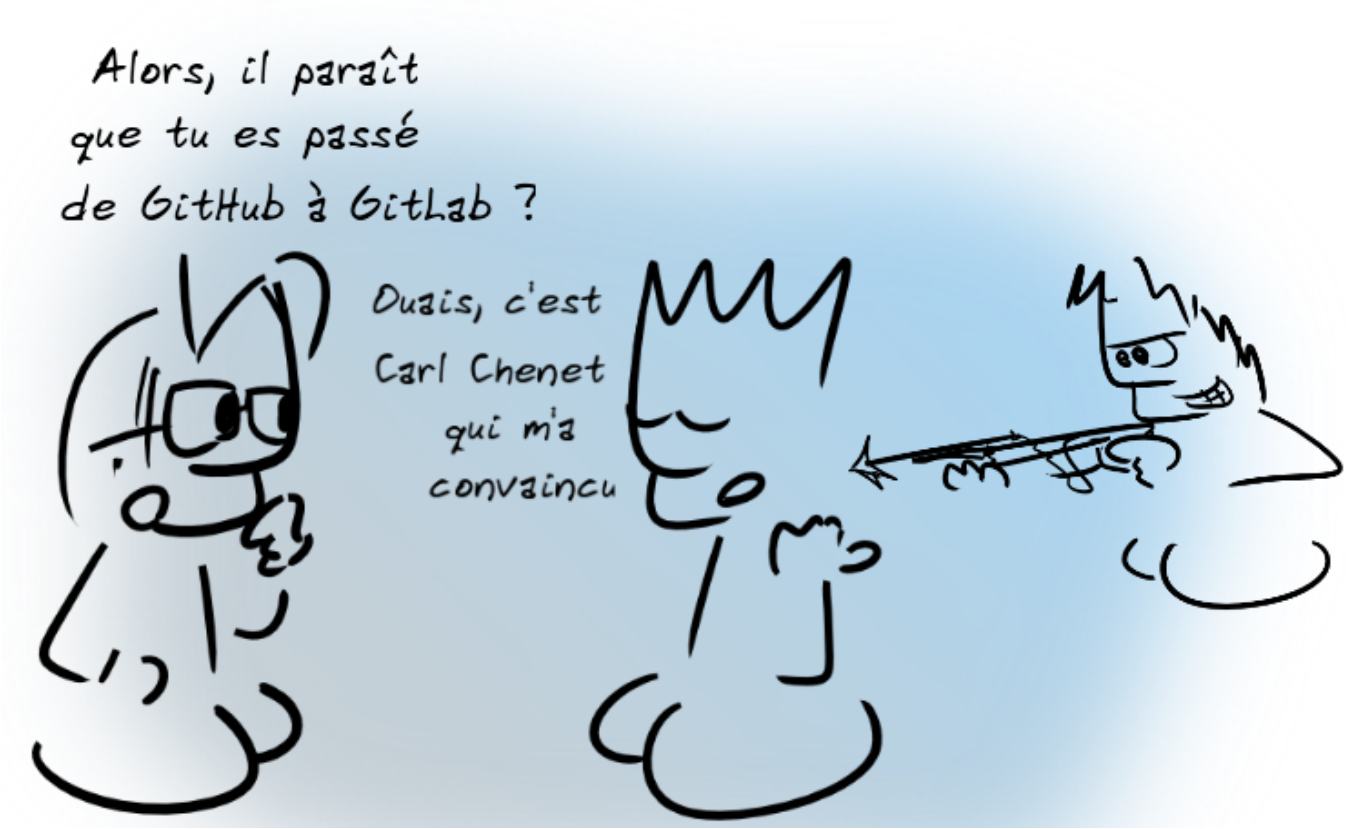
Améliorer le Framagit… une priorité

Ceci, ajouté à l'utilisation grandissante de notre forge qui allait bientôt poser des problèmes de taille de disques, nous a amenés à migrer (le 17 mars dernier) notre Gitlab vers une

machine avec plus de disque et surtout avec une installation utilisant les paquets dits « omnibus ».

Ces paquets omnibus nous ont permis d'installer Gitlab à l'aide d'un simple apt-get install gitlab-ce plutôt que de suivre la longue procédure d'installation manuelle. Non seulement l'installation est simplifiée, mais – et c'est surtout là la plus-value que nous en attendions – mettre à jour Gitlab devient tout aussi simple avec une seule commande apt-get dist-upgrade.

Résultat : notre Gitlab suit scrupuleusement la publication des nouvelles versions, avec leur lot de nouvelles fonctionnalités !



Pour fêter cela, nous avons étrenné un nouveau nom de domaine… inspiré par vous ! Avouons-le, «Git point Framasoft point orrrrrrrgueh », ça accroche un peu en bouche. De partout, nous avons entendu parler du « Framagit » : alors tant qu'à faire, autant l'appeler comme vous le faites déjà. Bien entendu, il n'est nul besoin de modifier vos URL, elles restent valides…

mais la nouvelle est à votre disposition !

Et si on ajoutait de l'intégration continue ?

Derrière ce terme barbare se cache une fonctionnalité très pratique : on crée une « recette » qui sera exécutée dans une machine virtuelle à chaque *push*. Cela peut par exemple permettre de lancer une suite de tests pour vérifier que l'on n'a rien cassé.

Pour utiliser cette fonctionnalité, il faut disposer de ce que l'on appelle un *runner*, c'est à dire un logiciel qui va récupérer la recette et l'exécuter. Il est possible d'installer un *runner* sur n'importe quel ordinateur, même votre ordinateur de bureau.

Pour ceux qui ne souhaitent pas gérer leur *runner* eux-mêmes, Framasoft met à disposition deux *runners* partagés entre tous les utilisateurs de Framagit, que vous pouvez utiliser comme bon vous semble. Notez toutefois que Gitlab indique que quiconque utilise un *runner* partagé peut accéder aux informations des projets utilisant ce *runner* : il vaut mieux monter votre propre *runner* pour vos projets sensibles.

De plus, en utilisant les *runners* partagés de Framasoft, il est possible que votre projet soit mis en file d'attente, en attendant que les recettes précédentes aient fini de s'exécuter… à vous de voir !

Pouhiou-le-moldu-du-code lisant cet article, allégorie.

Vous voulez des pages Gitlab ? Nous aussi !

Github permet à tout un chacun d'héberger un site statique. Gitlab propose une fonctionnalité similaire mais hélas, uniquement dans sa version entreprise… Nous utilisons pour notre part la version communautaire qui est la version libre

de Gitlab… donc *sans* les pages Gitlab.

Nous avons donc ouvert un ticket pour demander que cette fonctionnalité soit incluse dans la version communautaire. Si vous aussi vous aimeriez voir cela arriver, aidez-nous tout simplement en votant sur https://gitlab.com/gitlab-org/gitlab-ce/issues/14605.

En attendant, profitez d'une forge logicielle à jour et libre sur Framagit.org !

Mise à jour du 5/08/2016 :
Le tutoriel d'installation de Gitlab est -enfin- disponible sur le Framacloud.
Notez que cette installation est conjointe à celle de Mattermost (Framateam) puisque c'est ainsi que nous avons procédé

GitHub et les libristes : un danger et un défi !

Lorsqu'une personnalité notoire du Libre comme Carl Chenet s'attaque avec pertinence à la tendance massive du « tous sur GitHub » et égratigne la communauté du Libre pour son immobilisme (et même sa paresse !), Framasoft trouve que c'est une bonne occasion de lui donner un peu plus de voix encore.

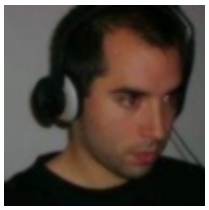
S'adressant principalement aux développeurs, il pointe les dangers d'un service centralisateur, privateur, qui uniformise les pratiques en étouffant les alternatives. Ça ne vous rappelle rien ? Oui, les mêmes écueils contre lesquels nous vous mettons en garde dans notre campagne degooglisons ! Ajoutons que nous avons déjà basculé sur GitLab, comme le

recommande Carl, dès 2014 et mis à la disposition de tous depuis le mois de mars 2015 notre GitLab qui héberge à ce jour 3017 projets, 2071 utilisateurs inscrits, 242 groupes.

Nous reprenons ici avec son autorisation le récent billet de Carl qui a déjà suscité d'intéressants commentaires et en provoquera probablement d'autres ici même.

Le danger GitHub

Un article de Carl Chenet d'abord publié sur son blog



Alors que le projet CPython (implémentation historique du projet Python) a annoncé son passage chez GitHub (avec quelques restrictions, nous reviendrons là-dessus), il est plus que jamais important de s'interroger sur les risques encourus d'utiliser un logiciel propriétaire dans notre chaîne de création du Logiciel Libre.

Des voix critiques s'élèvent régulièrement contre les risques encourus par l'utilisation de GitHub par les projets du Logiciel Libre. Et pourtant l'engouement autour de la forge collaborative de la startup Californienne à l'*octocat* continue de grandir.

L'octocat, mascotte de GitHub

Ressentis à tort ou à raison comme simples à utiliser, efficaces à l'utilisation quotidienne, proposant des fonctionnalités pertinentes pour le travail collaboratif en entreprise ou dans le cadre d'un projet de Logiciel Libre, s'interconnectant aujourd'hui à de très nombreux services d'intégration continue, les services offerts par GitHub ont pris une place considérable dans l'ingénierie logicielle ces dernières années.

Quelles sont ces critiques et sont-elles justifiées ? Nous proposons de les exposer dans un premier temps dans la suite de cet article avant de peser le pour ou contre de leur validité.

1. Points critiques

1.1 La centralisation

L'application GitHub appartient et est gérée par une entité unique, à savoir GitHub, inc, société américaine. On comprend donc rapidement qu'une seule société commerciale de droit américain gère l'accessibilité à la majorité des codes sources des applications du Logiciel Libre, ce qui représente un problème pour les groupes utilisant un code source qui devient indisponible, pour une raison politique ou technique.

De plus cette centralisation pose un problème supplémentaire : de par sa taille, ayant atteint une masse critique, elle s'auto-alimente. Les personnes n'utilisant pas GitHub, volontairement ou non, s'isolent de celles qui l'utilisent, repoussées peu à peu dans une minorité silencieuse. Avec l'effet de mode, on n'est pas « dans le coup » quand on n'utilise pas GitHub, phénomène que l'on rencontre également et même devenu typique des réseaux sociaux propriétaires (Facebook, Twitter, Instagram).

1.2 Un logiciel privateur

Lorsque vous interagissez avec GitHub, vous utilisez un

logiciel privateur, dont le code source n’est pas accessible et qui ne fonctionne peut-être pas comme vous le pensez. Cela peut apparaître gênant à plusieurs points de vue. Idéologique tout d’abord, mais peut-être et avant tout pratique. Dans le cas de GitHub on y pousse du code que nous contrôlons hors de leur interface. On y communique également des informations personnelles (profil, interactions avec GitHub). Et surtout un outil crucial propriétaire fourni par GitHub qui s’impose aux projets qui décident de passer chez la société américaine : le gestionnaire de suivi de bugs.

1.3 L’uniformisation

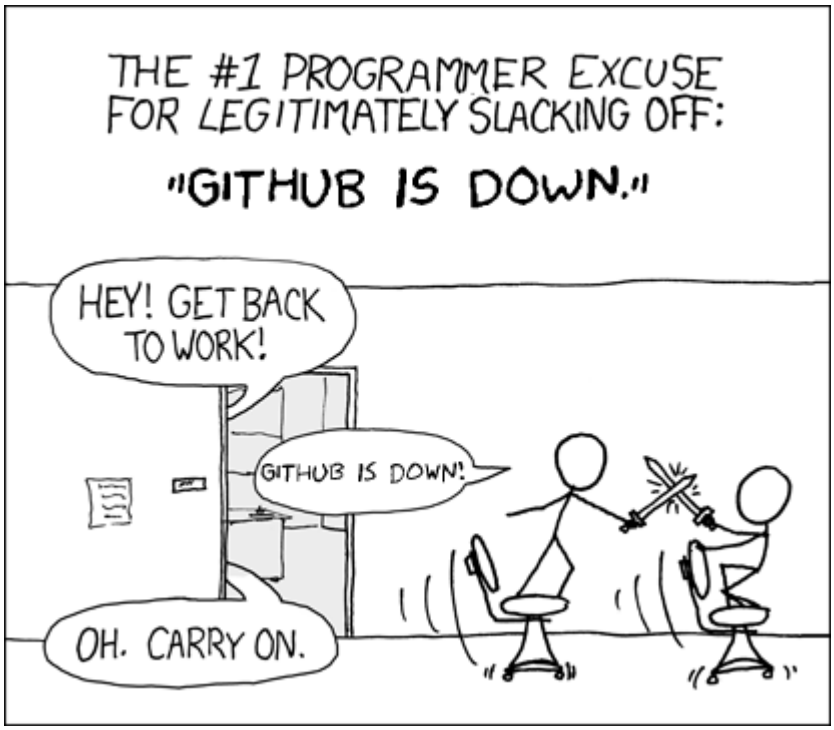
Travailler via l’interface GitHub est considéré par beaucoup comme simple et intuitif. De très nombreuses sociétés utilisent maintenant GitHub comme dépôt de sources et il est courant qu’un développeur quittant une société retrouve le cadre de travail des outils GitHub en travaillant pour une autre société. Cette fréquence de l’utilisation de GitHub dans l’activité de développeur du Libre aujourd’hui participe à l’uniformisation du cadre de travail dudit développeur.

L’uniforme évoque l’armée, ici l’armée des clones

2. Validité des points critiques

2.1 Les critiques de la centralisation

Comme dit précédemment, GitHub est aujourd'hui la plus grande concentration de code source du Logiciel Libre. Cela fait de lui une cible privilégiée. Des attaques massives par déni de service ont eu lieu en mars et août 2015. De même, une panne le 15 décembre 2015 a entraîné l'indisponibilité de 5% des dépôts. Idem le 15 novembre. Et il s'agit des incidents récents déclarés par les équipes de GitHub elles-mêmes. On peut imaginer un taux d'indisponibilité moyen des services bien supérieur.



L'excuse n°1 des programmeurs pour se lâcher sans scrupules : « GitHub est en panne »

– Hé, au boulot les gars ! – Github est en panne !

– Ah bon, continuez alors.

2.2 Les critiques relatives à l'usage d'un

logiciel privateur

Cette critique, avant tout idéologique, se heurte à la conception même que chacun des membres de la communauté se fait du Logiciel Libre, et en particulier d'un critère : contaminant ou non, qu'on résume en général par GPL versus MIT/BSD.

> *Framanote : MIT/BSD sont des licences permissives, laissant toutes les libertés, même celle de reprendre le code dans un logiciel privateur/propriétaire. Cela correspond à la CC-BY ou à la CC-0 dans les licences Creative Commons.*
>
> *GPL est une licence copyleft (ou contaminante). Le principe est que tout développement utilisant un code sous licence contaminante doit rester Libre, donc être diffusé sous la même licence. Cela correspond à la mention SA dans les licences Creative Commons.*

Les défenseurs du Logiciel Libre contaminant vont être gênés d'utiliser un logiciel propriétaire car ce dernier ne devrait pas exister. Il doit être assimilé, pour citer Star Trek, car il est une boîte noire communicante, qui met en danger la vie privée, détourne nos usages à des fins commerciales, gêne ou contraint la liberté de jouir entièrement de ce qu'on a acquis, etc.

Les tenants d'une totale liberté sont moins complexés dans leur utilisation des logiciels privateurs puisqu'ils acceptent l'existence desdits logiciels privateurs au nom d'une liberté sans restriction. Ils acceptent même que le code qu'ils développent aboutissent dans ces logiciels, ce qui arrive bien plus souvent qu'on ne le croit, voir à ce sujet la liste à couper le souffle des produits commerciaux reposant sur FreeBSD. On peut donc voir dans cette aile de la communauté du Logiciel Libre une totale sérénité à utiliser GitHub. Et ce qui est cohérent vis-à-vis de l'idéologie soutenue. Si vous êtes déjà allé au Fosdem, un coup d'œil dans l'amphithéâtre Janson permet de se rendre compte de la présence massive de portables Apple tournant sous MacOSX.

FreeBSD, principal projet des BSD sous licence MIT

Mais au-delà de cet aspect idéologique pur et pour recentrer sur l'infrastructure de GitHub elle-même, l'utilisation du gestionnaire de suivi de bugs de GitHub pose un problème incontournable. Les rapports de bugs sont la mémoire des projets du Logiciel Libre. Il constitue le point d'entrée des nouveaux contributeurs, des demandes de fonctionnalités, des rapports de bugs et donc la mémoire, l'histoire du projet qui ne peut se limiter au code seul. Il est courant de tomber sur des rapports de bugs lorsque vous copiez/collez votre message d'erreur dans un moteur de recherche. Mémoire précieuse non seulement pour le projet lui-même, mais aussi pour ses utilisateurs actuels et à venir.

GitHub propose d'extraire les rapports de bugs via son API, certes, mais combien de projets anticiperont une éventuelle

défaillance de GitHub ou un retournement de situation arrêtant brusquement le service ? Très peu à mon avis. Et comment migrer vers un nouveau système de suivi de bugs les données fournies par GitHub ?

L'exemple de l'utilitaire de gestion de listes de choses à faire (TODO list) Astrid, racheté par Yahoo! il y a quelques années reste un très bon exemple de service ayant grandi rapidement, largement utilisé et qui a fermé du jour au lendemain, proposant pendant quelques semaines seulement d'extraire ses données. Et il s'agissait là d'un simple gestionnaire de tâches à faire. Le même problème chez GitHub serait dramatiquement plus difficile à gérer pour de très nombreux projets, si on leur laisse la possibilité de le gérer. Certes le code reste disponible et pourra continuer de vivre ailleurs, mais la mémoire du projet sera perdue, alors qu'un projet comme Debian approche aujourd'hui les 800 000 rapports de bugs. Une vraie mine d'or d'informations sur les problèmes rencontrés, les demandes de fonctionnalités et le suivi de ces demandes. Les développeurs du projet CPython passant chez GitHub ont anticipé ce problème et ne vont pas utiliser le système de suivi de bugs de GitHub.

Debian, l'un des principaux projets du Logiciel Libre

avec autour de 1000 contributeurs officiels

2.3 L'uniformisation

La communauté du Logiciel Libre oscille sans cesse entre un besoin de normes afin de réduire le travail nécessaire pour l'interopérabilité et l'attrait de la nouveauté, caractérisée par l'intrinsèque besoin de différence vis-à-vis de

l’existant.

GitHub a popularisé l’utilisation de Git, magnifique outil qui aujourd’hui touche des métiers bien différents des programmeurs auxquels il était initialement lié. Peu à peu, tel un rouleau compresseur, Git a pris une place si centrale que considérer l’usage d’un autre gestionnaire de sources est quasiment impossible aujourd’hui, particulièrement en entreprise, malgré l’existence de belles alternatives qui n’ont malheureusement pas le vent en poupe, comme Mercurial.

Un projet de Logiciel Libre qui naît aujourd’hui, c’est un dépôt Git sur GitHub avec un README.md pour sommairement le décrire. Les autres voies sont totalement ostracisées. Et quelle est la punition pour celui qui désobéit ? Peu ou pas de contributeurs potentiels. Il semble très difficile de pousser aujourd’hui le contributeur potentiel à se lancer dans l’apprentissage d’un nouveau gestionnaire de sources ET une nouvelle forge pour chaque projet auquel on veut contribuer. Un effort que fournissait pourtant tout un chacun il y a quelques années.

Et c’est bien dommage car GitHub, en proposant une expérience unique et originale à ses utilisateurs, taille à grands coups de machette dans les champs des possibles. Alors oui, sûrement que Git est aujourd’hui le meilleur des système de gestion de versions. Mais ça n’est pas grâce à cette domination sans partage qu’un autre pourra émerger. Et cela permet à GitHub d’initier à Git les nouveaux arrivants dans le développement à un ensemble de fonctionnalités très restreint, sans commune mesure avec la puissance de l’outil Git lui-même.

Centralisation, uniformisation, logiciels privateurs et bientôt… fainéantise ?

Le combat contre la centralisation est une part importante de l'idéologie du Logiciel Libre car elle accroît le pouvoir de ceux qui sont chargés de cette centralisation et qui la contrôlent sur ceux qui la subissent. L'aversion à l'uniformisation née du combat contre les grandes firmes du logiciel souhaitant imposer leur vision fermée et commerciale du monde du logiciel a longtemps nourri la recherche réelle d'innovation et le développement d'alternatives brillantes. Comme nous l'avons décrit, une partie de la communauté du Libre s'est construit en opposition aux logiciels privateurs, les considérant comme dangereux. L'autre partie, sans vouloir leur disparition, a quand même choisi un modèle de développement à l'opposé de celui des logiciels privateurs, en tout cas à l'époque car les deux mondes sont devenus de plus en plus poreux au cours des dernières années.

L'effet GitHub est donc délétère au point de vue des effets qu'il entraîne : la centralisation, l'uniformisation, l'utilisation de logiciels privateurs comme leur système de gestion de version, au minimum. Mais la récente affaire de la lettre « Cher GitHub… » met en avant un dernier effet, totalement inattendu de mon point de vue : la fainéantise. Pour les personnes passées à côté de cette affaire, il s'agit d'une lettre de réclamations d'un nombre très important de représentants de différents projets du Logiciel Libre qui réclament à l'équipe de GitHub d'entendre leurs doléances, apparemment ignorées depuis des années, et d'implémenter de nouvelles fonctionnalités demandées.

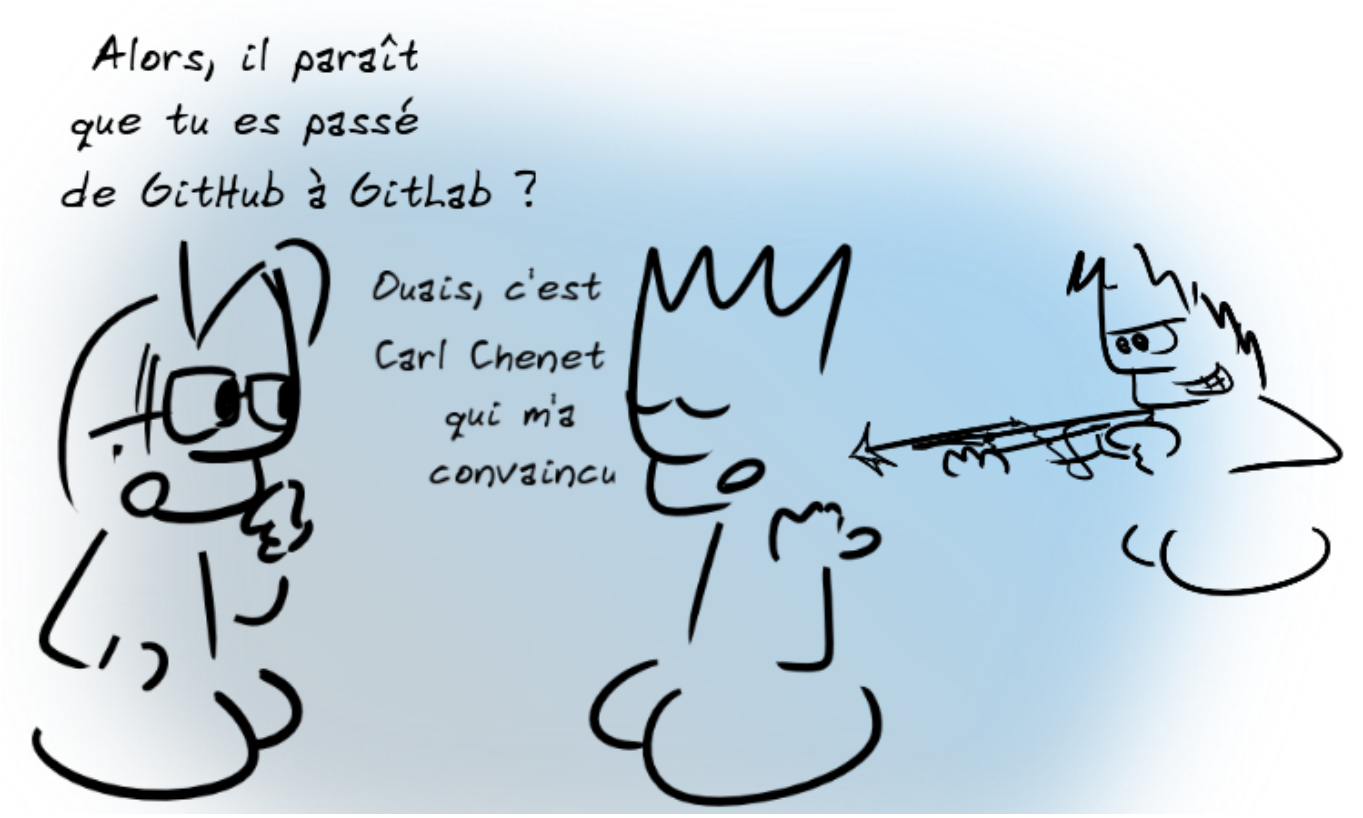
Mais depuis quand des projets du Logiciel Libre qui se heurtent depuis des années à un mur tentent-ils de faire pleurer le mur et n'implémentent pas la solution qui leur manquent ? Lorsque Torvald a subi l'affaire Bitkeeper et que l'équipe de développement du noyau Linux n'a plus eu

l’autorisation d’utiliser leur gestionnaire de versions, Linus a mis au point Git. Doit-on rappeler que l’impossibilité d’utiliser un outil ou le manque de fonctionnalités d’un programme est le moteur principal de la recherche d’alternatives et donc du Logiciel Libre ? Tous les membres de la communauté du Logiciel Libre capables de programmer devraient avoir ce réflexe. Vous n’aimez pas ce qu’offre GitHub ? Optez pour Gitlab. Vous n’aimez pas Gitlab ? Améliorez-le ou recodez-le.

Logo de Gitlab, une alternative possible à GitHub

en choisissant la version Communauté

Que l’on soit bien d’accord, je ne dis pas que tout programmeur du Libre qui fait face à un mur doit coder une alternative. En restant réaliste, nous avons tous nos priorités et certains de nous aiment dormir la nuit (moi le premier). Mais lorsqu’on voit 1340 signataires de cette lettre à GitHub et parmi lesquels des représentants de très grands projets du Logiciel Libre, il me paraît évident que les volontés et l’énergie pour coder une alternative existe. Peut-être d’ailleurs apparaîtra-t-elle suite à cette lettre, ce serait le meilleur dénouement possible à cette affaire.



Finalement, l’utilisation de GitHub suit cette tendance de massification de l’utilisation d’Internet. Comme aujourd’hui les utilisateurs d’Internet sont aspirés dans des réseaux sociaux massivement centralisés comme Facebook et Twitter, le monde des développeurs suit logiquement cette tendance avec GitHub. Même si une frange importante des développeurs a été sensibilisée aux dangers de ce type d’organisation privée et centralisée, la communauté entière a été absorbée dans un mouvement de centralisation et d’uniformisation. Le service offert est utile, gratuit ou à un coût correct selon les fonctionnalités désirées, confortable à utiliser et fonctionne la plupart du temps. Pourquoi chercherions-nous plus loin ? Peut-être parce que d’autres en profitent et profitent de nous pendant que nous sommes distraits et installés dans notre confort ? La communauté du Logiciel Libre semble pour le moment bien assoupie.

Le « lion » du Libre assoupi devant la cheminée (allégorie)

Liens :

- **Source de l'article : blog de Carl Chenet**
- **Notre GitLab ouvert à tous (pour sortir de Github)**

Google Code ferme ses portes ? Nous, on les ouvre.

C'est officiel : Google Code, qui permettait aux développeurs de déposer, partager, et collaborer sur du code logiciel (libre ou pas), va bientôt fermer ses portes.

Il va donc rejoindre le mémorial des projets sabordés par Google.

La raison la plus probable, c'est que GitHub (une plateforme concurrente) attire bien plus de développeurs, et donc de

code, que Google Code. Non seulement grâce à une interface plus intuitive, mais aussi par une facilité bien plus grande pour les développeurs à collaborer ensemble (plus on est de fous, plus il y a de code produit).

D'ailleurs, Google ne s'en cache pas et propose, dans le courrier annonçant la clôture prochaine du service, un outil permettant de transférer votre projet logiciel de Google Code à GitHub.

Quelles réflexions cela devrait-il nous inspirer ?

D'abord, que malgré sa puissance financière massive, Google n'est pas systématiquement le meilleur dans son domaine. Et qu'une « petite » entreprise (267 salariés, tout de même) comme GitHub, Inc, peut amener le géant de Mountain View à fermer un service qui hébergeait malgré tout plus de 250 000 projets logiciels.

Cela pourrait paraître pour une bonne nouvelle : la diversité et l'innovation resteraient possibles ! L'argent n'achèterait pas tout ! Skynet (pardon, Googleternet) n'aurait pas encore un pouvoir absolu !

Ensuite, que Google continue à être une entreprise qui ne s'entête pas. Si un projet fonctionne, tant mieux (et autant devenir le meilleur au monde dessus). Sinon, tant pis, c'est que le marché n'est pas mûr, que les technologies utilisées n'étaient pas les bonnes, que les équipes n'étaient pas les meilleures, ou que les utilisateurs n'étaient pas prêts. Google Plus étant *pour l'instant* l'exception à la règle.

Cependant, peut-on considérer cela comme un fait positif ?

Pas vraiment. Car cela concentre encore un peu plus les utilisateurs sur GitHub.

Alors certes, il est toujours possible de quitter GitHub, de reprendre son code et d'aller le déposer ailleurs. Mais si tous les développeurs sont sur GitHub, il y aura une forme de pression sociale à continuer d'utiliser cette plateforme.

Donc, cela soulève deux questions.

1. Les développeurs de logiciels libres ont-il intérêt à utiliser GitHub ?

La plateforme est extrêmement pratique, confortable et performante, il faut le reconnaître.

Mais **le code de GitHub n'est pas libre**.

Ce manque de transparence peut avoir des conséquences importantes.

D'abord, GitHub pourrait peu à peu se garnir de publicités, tel un sapin de Noël. Cela serait désagréable, mais pas bloquant.

Ensuite, GitHub pourrait modifier les données hébergées sans les accords des auteurs. Par exemple, intégrer des fichiers (publicitaires, malveillants, etc.) dans les .zip téléchargés par millions quotidiennement sur la plateforme. Ca serait peut-être se tirer une balle dans le pied pour la société, mais cela n'a pas empêché Sourceforge, alors plus importante forge logicielle mondiale, de le faire. Et rien que le fait que GitHub **puisse** le faire est inquiétant et devrait interroger tout développeur de logiciel libre.

Enfin, nous, utilisateurs, n'avons pas le pouvoir sur les choix technologiques ou ergonomiques de GitHub. Si, demain, GitHub décide de modifier l'interface de telle ou telle façon, les développeurs seront tels des consommateurs dans un supermarché qui changerait ses produits d'allées, ou qui supprimerait tel ou tel produit : pris au piège de la volonté d'un tiers.

2. Quel est le modèle économique de GitHub ?

Certes, GitHub est une boite « sympa » (comme l'était Google à ses débuts). L'entreprise est toujours en mode start-up : largement financée par des fonds levés auprès de sociétés de capital-risque. Sans cet argent, GitHub serait déficitaire. Or, si des entreprises comme Andreessen Horowitz (fondées par des anciens de