Des routes et des ponts (12) – en quête de modèle économique

Dans notre projet de traduction de l'ouvrage de Nadia Eghbal Roads and Bridges (tous les épisodes déjà traduits), nous abordons aujourd'hui une section importante consacrée aux modes de financement de ce qu'elle appelle l'infrastructure numérique et qui est comme l'épine dorsale de du monde informatique.

Elle donne ici un aperçu avec quelques exemples significatifs des trois principales voies explorées avec des succès variables par les développeurs et les entreprises : l'incitation par des récompenses, la monétisation par des services et le recours à des licences open source hybrides, en partie payantes…

Des modèles économiques pour les infrastructures numériques

Traduction Framalang : Piup, xi, Penguin, Bidouille, Lumibd, Opsylac, goofy

Certains aspects des infrastructures numériques peuvent fonctionner dans un contexte concurrentiel. Les bases de données et les services d'hébergement, par exemple, sont souvent des affaires profitables, bien financées, parce qu'elles peuvent faire payer l'accès. Tout comme l'accès à l'eau ou à l'électricité, l'accès à un serveur ou à une base de données peut être mesuré, facturé, et fermé si les honoraires ne sont pas réglés.

Heroku (mentionné au début de ce rapport) et Amazon Web Services sont deux exemples notables de plateformes qui

vendent des services d'infrastructure numérique à des développeurs logiciels contre une redevance (à noter qu'aucun des deux n'est un projet *open source*). Des projets *open source* similaires, à ce niveau d'infrastructure, tels que OpenStack (une plate-forme concurrente d'Amazon Web Services) ou MySQL (une base de données), ont trouvé leurs assises dans des entreprises. OpenStack est financé par un consortium d'entreprises, et MySQL a été racheté par Oracle.

Une partie de ce qui rend ces services financièrement attractifs, c'est l'absence de « bruit ». Pour un seul logiciel, un développeur utilise parfois 20 bibliothèques différentes, avec chacune des fonctions différentes, mais il n'a besoin que d'une seule base de données. En conséquence, les projets à succès ont plus de chances d'obtenir l'attention et le soin dont ils ont besoin.

Il existe une autre façon utile de cerner les infrastructures que l'on peut facturer : s'il y a un risque immédiat de défaillance, alors il y a probablement un modèle économique. En d'autres termes, un serveur peut subir des interruptions de service inattendues, tout comme l'électricité peut sauter à l'improviste, mais un langage de programmation ne « casse » ni n'a des périodes d'indisponibilités de cette même façon, parce qu'il s'agit d'un système d'information.

Pour ce genre de projets *open source*, le modèle économique a tendance à se focaliser sur la recherche de services ou d'assistance facturables. Cela fonctionne pour les projets qui bénéficient d'un usage significatif par les entreprises, en particulier quand il s'agit d'un problème techniquement complexe, ou lorsqu'une entreprise a besoin qu'une fonction soit développée.

Récompenses

Logo de Bountysource

À petite échelle, des gens ou des entreprises promettent parfois des « récompenses » d'ordre pécuniaire pour l'atteinte de certains objectifs de développement.
Par exemple, IBM demande régulièrement de nouvelles fonctionnalités pour divers projets par le biais d'un site web appelé Bountysource, offrant jusqu'à 5 000 $ par tâche. Bountysource est une plateforme populaire pour trouver et proposer des récompenses ; elle compte plus de 26 000 membres. 120 récompenses aident à régler les problèmes précédemment mentionnés liés aux simples dons à un projet. Comme les récompenses sont clairement liées à un résultat, l'argent va être utilisé. En revanche, les récompenses peuvent avoir des effets pervers pour l'incitation à contribuer à un projet.
Les récompenses peuvent dicter quel travail sera ou ne sera pas effectué, et parfois ce travail n'est pas en phase avec les priorités d'un projet. Il peut aussi introduire du bruit dans le système : par exemple, une entreprise peut offrir une forte récompense pour une fonctionnalité que les propriétaires du projet ne considèrent pas comme importante.

Du côté des contributeurs, des personnes extérieures sans connaissances sur un projet peuvent y participer seulement pour obtenir la récompense, puis le quitter. Ou bien elles peuvent bâcler le travail requis, parce qu'elles essaient d'obtenir des récompenses. Enfin, les récompenses peuvent être une façon appropriée de financer de nouvelles fonctionnalités ou des problèmes importants, mais sont moins pratiques lorsqu'il s'agit de financer des opérations continues, comme le service client ou la maintenance.

Jeff Atwood, le créateur de Stack Overflow, a remarqué les problèmes suivants avec les programmes de récompenses, en particulier en ce qui concerne la sécurité :

> *L'un des effets pervers de cette tendance à attribuer des récompenses pour les rapports de bugs est que cela n'attire pas seulement de véritables programmeurs intéressés par la sécurité, mais aussi tous les gens intéressés par l'argent facile. Nous avons reçu trop de rapports de bugs de sécurité « sérieux » qui n'avaient qu'une importance très faible. Et nous devons les traiter, parce qu'ils sont « sérieux », n'est-ce pas ? Malheureusement, beaucoup d'entre eux ne représentent qu'un gaspillage de temps… Ce genre d'incitation me semble mauvais. Même si je sais que la sécurité est extrêmement importante, je vois ces interactions avec de plus en plus d'inquiétude parce qu'elles me créent beaucoup de travail et que le retour sur investissement est très faible.*

Services

Photo par Rich Bowen (CC BY 2.0)

À une plus vaste échelle, un des exemples bien connus et les plus souvent cités de modèle économique *open source*, c'est Red Hat, l'entreprise dont nous avons déjà parlé, qui propose une assistance, des sessions de formation et autres services à des entreprises qui utilisent Linux. Red Hat a été fondée en 1993, il s'agit d'une entreprise cotée en bourse avec un chiffre d'affaires déclaré de 2 milliards de dollars par an.

Bien que Red Hat ait connu un succès fantastique d'un point de vue financier, nombreux sont ceux qui soulignent qu'il s'agit d'une anomalie qui n'aura pas de lendemains. Red Hat a bénéficié de l'avantage du premier arrivé dans son domaine

technologique. Matt Asay, un journaliste spécialisé en *open source*, a remarqué que Red Hat utilise un ensemble unique de licences et brevets pour protéger ses parts de marché. Asay, qui auparavant était un fervent défenseur des entreprises *open source*, est maintenant persuadé que certaines licences propriétaires sont nécessaires pour faire sérieusement des affaires. Matthew Aslet du 451 Group, un organisme de recherche, a découvert lui aussi que la plupart des entreprises *open source* qui réussissent utilisent en fait un type ou un autre de licence commerciale.

Docker, déjà mentionné plus haut, est un projet *open source* qui aide les applications à fonctionner efficacement. C'est l'exemple le plus récent d'entreprise qui s'inspire de ce modèle. Docker a levé 180 millions de dollars en capital-risque auprès d'investisseurs, avec une valorisation d'un milliard de dollars de la part d'investisseurs privés. Comme sa part de marché s'est accrue, Docker a commencé à proposer des services d'assistance au niveau des entreprises. Mais sans solides revenus, Docker pourrait n'être qu'un exemple de plus de capital-risque qui fait un investissement dans une entreprise d'infrastructure leader sur son marché, mais qui réalise des pertes.

À petite échelle, beaucoup de développeurs proposent des services de consultants pour pouvoir financer leur travail. Hoodie est un *framework* poids plume qui repose sur Node et qui a réussi dans les services de consultants.

Hoodie lui-même est un projet *open source*. Plusieurs mainteneurs gagnent leur vie grâce à la boutique de l'entreprise, Neighbourhoodie, qui propose des services de développement logiciel. Bien que Neighbourhoodie se spécialise dans le *framework* de Hoodie, ce dernier est encore un projet plutôt jeune, de sorte que certaines parties de son travail proviennent de pojets qui ne sont pas lié à Hoodie. Dans le cas de Hoodie, le modèle de services choisi est censé payer le salaire de plusieurs mainteneurs, plutôt que de viser une

stratégie d'entreprise de l'échelle de Red Hat.

Le conseil est une option viable pour les développeurs indépendants, s'il y a suffisamment de gens qui utilisent le projet qui sont d'accord et ont assez d'argent pour payer de l'aide supplémentaire. Mais à petite échelle, cela peut aussi les empêcher d'améliorer le projet lui-même, puisque les deux personnes au plus qui le maintiennent passent désormais leur temps à développer leur affaire et à fournir des services qui peuvent ou non être en accord avec les besoins du projet en termes de maintenance.

Aspirer à une activité de consultant peut aussi entrer en contradiction avec l'objectif de rendre le produit facile à utiliser et à appréhender, ce qui est bien dans l'esprit de l'*open source*. Twisted, la bibliothèque Python déjà citée, a mentionné un témoignage plein d'humour de l'un de ses utilisateurs, une entreprise nommée Mailman : « Les gars, vous avez un gros problème, parce que c'était vraiment trop facile ! Comment vous comptez vous faire un paquet d'argent juste avec du conseil ? □ »

En fin de compte, le « modèle économique » pour un projet *open source* n'est pas très différent du simple travail indépendant.

Licences payantes

Certains développeurs ont l'impression que mettre les projets sous licence serait une solution au moins partielle aux problèmes de financement de l'*open source*. Si les projets *open source* sont fortement utilisés, pourquoi ne pas les facturer ? Ces « licences payantes » ne sont techniquement pas des licences *open source*, selon la définition de l'*open source* Initiative. Il s'agit plutôt d'initiatives qui tentent d'apporter un équilibre entre le besoin très concret de travail rémunéré et le désir de rendre le code accessible au public. Ce type de code peut être appelé « à source visible » ou « à source disponible ». Fair Source, par exemple, se

décrit lui-même comme « [offrant] certains des avantages de l'*open source* tout en préservant la possibilité de faire payer pour le logiciel. »

La licence Fair Source fut créée en novembre 2015 par une entreprise appelée Sourcegraph pour répondre au besoin de licence payante. Les termes de la licence ont été rédigés par Heather Meeker, un juriste qui a également travaillé dans l'équipe principale de la Mozilla Public License v2.0. Avec la licence Fair Source, on peut librement consulter, télécharger, exécuter et modifier le code, jusqu'à un certain nombre d'utilisateurs par organisation. Une fois cette limite dépassée, l'organisation doit payer un forfait de licence, dont le montant est déterminé par l'éditeur. En d'autres termes, le code Fair Source est gratuit pour un usage personnel et pour les PME, mais fournit une base légale pour facturer les cas de plus gros usages commerciaux.

L'annonce par Sourcegraph de la création de la licence Fair Source, qu'ils utilisent maintenant eux-mêmes, a provoqué un débat animé sur la monétisation de l'*open source*. (Il est à noter qu'un mouvement similaire autour du « *shareware* », logiciel propriétaire gratuit, avait émergé avec un certain succès populaire dans les années 1980).
Mike Perham, l'un des mainteneurs de Sidekiq, un outil populaire pour le développement en Ruby, a aussi récemment suggéré aux contributeurs et contributrices *open source* d'utiliser une « licence duale » pour monétiser leur travail, faisant payer les entreprises l'accès à une licence MIT permissive plutôt qu'une licence AGPL plus restrictive qui impose l'attribution. Sa théorie est qu'en faisant d'AGPL la licence par défaut, « les entreprises vont payer pour l'éviter. »
Pour justifier cette idée, Perham a rappelé à son public :

> *« Souvenez-vous : logiciel open source ne signifie pas logiciel gratuit. Ce n'est pas parce que l'on peut consulter la source sur GitHub que tout le monde peut l'utiliser et en*

> *faire n’importe quoi. Il n’y a aucune raison pour laquelle vous ne pourriez pas donner l’accès à votre code mais aussi faire payer pour son utilisation. Tant que vous possédez le code, vous avez le droit d’y attribuer la licence que vous voulez.*
> *…[La] réalité, c’est que la plupart des petits projets open source dépendent d’une seule personne qui fait 95 % du travail. Si c’est votre cas, soyez reconnaissants envers les gens qui vous aident gratuitement mais ne vous sentez pas coupable de garder 100 % du revenu. »*

Faire payer les entreprises offre une autre possibilité aux développeurs et développeuses qui souhaitent poursuivre leur travail, en particulier s’il n’y a qu’une ou deux personnes pour maintenir un projet actif. Cependant, tous les projets ne peuvent pas faire payer pour le travail fourni, en particulier les projets plus vieux, ou les projets d’infrastructure qui ressemblent plus à des biens publics qu’à des produits de consommation, comme les langages de programmation.

Même si les licences payantes peuvent fonctionner pour certains scénarios, ce modèle est aussi pour beaucoup en opposition avec l’énorme valeur sociale offerte par l’*open source*, qui suggère que lorsque le logiciel est libre, l’innovation suit.

L’objectif ne devrait pas être le retour à une société qui repose sur les logiciels fermés, où le progrès et la créativité sont limités, mais de soutenir de façon durable un écosystème public dans lequel le logiciel peut être créé et distribué librement.