Software Heritage : l'archive universelle des codes sources du logiciel

Par **Roberto DI COSMO**
Software Heritage

Le logiciel est devenu en quelques décennies le moteur de notre industrie, le carburant de l'innovation, l'instrument essentiel que nous utilisons pour communiquer, nous entretenir, réaliser tout type de transaction et opération, nous organiser en société et former nos opinions politiques. Il contrôle le système embarqué dans nos moyens de transport ou de communication, les échanges commerciaux et financiers. Il est au cœur des équipements et des dispositifs médicaux ; il assure le bon fonctionnement des réseaux de transport et de communication, des banques et des établissements financiers. Le logiciel est crucial dans le fonctionnement des organisations économiques, sociales et politiques, qu'elles soient publiques ou privées, que ce soit sur des terminaux mobiles ou sur le *cloud.* Il est aussi le médiateur indispensable qui permet d'accéder à toute l'information numérique, et il constitue, avec les articles et les données, un des piliers de la recherche moderne (Noorden *et al.,* 2014).

Le logiciel est donc en train d'incorporer *une partie importante* de notre patrimoine scientifique, technique et industriel, et il porte avec lui des enjeux stratégiques majeurs.

Si on y regarde de près, il est aisé de se rendre compte que la vraie connaissance qui est contenue dans les logiciels se trouve non pas dans les programmes exécutables, mais dans le « code source » qui, selon la définition utilisée dans la licence GPL, est « la forme préférée pour un développeur pour apporter une modification à un programme [(1)] ». Le code source est une forme de connaissance spéciale : il est fait pour être *compris par un être humain*, le développeur, et peut être mécaniquement traduit dans une forme pour être *exécuté* directement sur une machine. La terminologie même utilisée par la communauté informatique est parlante : on utilise des « langages de programmation » pour « écrire » des logiciels. Comme Harold Habelson l'écrivait déjà en 1985, « les programmes doivent être écrits d'abord pour que d'autres êtres humains puissent les lire » (Abelson et Sussman, 1985).

Le code source des logiciels est donc bien une *création humaine* au même titre que d'autres documents écrits, et les développeurs des logiciels méritent le même respect que d'autres créateurs.

Enfin, le code source des logiciels est de plus en plus complexe, et est modifié régulièrement par des groupes de développeurs qui collaborent pour le faire évoluer : il est devenu essentiel, pour le comprendre, d'avoir accès aussi à son historique de développement.

Le code source des logiciels constitue donc un patrimoine de grande valeur, comme déjà soutenu par Len Shustek dans un bel article de 2006 (Shustek, 2006) et par Donald Knuth (Knuth, 1984), et il est essentiel de s'atteler à sa préservation.

C'est bien là une des missions que s'est donné Software Heritage, une initiative lancée en 2015 avec le soutien de l'Inria [(2)], pour *récolter, organiser, préserver et rendre facilement accessible* l'ensemble

(1) GNU91, Gnu general public license, version 2, 1991, Retrieved September 2015.
(2) Créé en 1967, l'Inria est un établissement public à caractère scientifique et technologique spécialisé en mathématiques et informatique, placé sous la double tutelle du ministère de l'Enseignement supérieur, de la Recherche et de l'Innovation et du ministère de l'Économie et des Finances.

du *code source* disponible publiquement sur la planète, indépendamment d'où et comment il a été développé ou distribué. Le but est de construire une infrastructure commune qui permettra une multiplicité d'applications : bien sûr, préserver sur le long terme le code source contre les risques de destruction, mais aussi permettre des études à grande échelle sur le code et les processus de développement actuels, afin de les améliorer, et préparer ainsi un futur meilleur.

Une tâche complexe

Archiver tous les codes sources disponibles est une tâche complexe : comme détaillé dans l'article d'Abramatic (Abramatic *et al.,* 2018), on doit déployer des stratégies différentes selon qu'on cherche à collecter du code source ouvert ou propriétaire, et on ne traite pas de la même façon le code source facilement disponible en ligne que celui qui se trouve sur des supports physiques anciens. Pour le code source des logiciels anciens, on doit mettre en place un véritable processus d'archéologie informatique, et nous avons déjà commencé ce travail dans le cadre d'une collaboration avec l'Université de Pise et l'UNESCO, qui a abouti au processus SWHAP utilisé pour retrouver, documenter et archiver des logiciels marquants de l'histoire de l'informatique en Italie (voir https://www.softwareheritage.org/swhap). Pour le code source ouvert et facilement disponible en ligne, l'approche la plus appropriée est de construire un moissonneur qui collecte automatiquement les contenus d'une grande variété de plateformes de développement collaboratif, comme GitHub, GitLab.com ou BitBucket, ou de plateformes de diffusion de paquetages logiciel, comme Debian, NPM. CRAN ou Pypi.

Si, au premier abord, cette approche peut paraître similaire à celle mise en place pour l'archivage du Web, en y regardant de plus près, on s'aperçoit rapidement que la tâche est ici bien plus difficile. Tout d'abord, comme il n'y a pas de protocole standard, il faut construire un adaptateur spécifique pour chacune des plateformes de développement et de diffusion des logiciels, afin d'extraire la liste des projets logiciels qu'elles hébergent (on appelle justement *listers* ces adaptateurs).

Ensuite, on se trouve confronté à pléthore de formats et de modèles de données différents, utilisés pour garder trace de l'historique de développement, ce qui pose un gros problème si on veut s'assurer que cet historique sera lisible dans le futur, même quand les outils utilisés pour le construire – comme git, darcs, subversion ou mercurial – seront devenus obsolètes. Pour cela, nous construisons une deuxième famille d'adaptateurs, que nous appelons *loaders,* pour convertir dans une structure de données commune, simple et maintenable, les informations contenues dans les différents systèmes de contrôle de version et formats de paquetages.

Cette structure de données est une généralisation des arbres de Merkle, inventés il y a plus de quarante ans, et dont les principes sont aujourd'hui largement utilisés dans des applications aussi variées que les systèmes de contrôle distribués, les *blockchains* ou les systèmes de fichiers distribués (Merkle, 1987). Elle présente de nombreux avantages : chaque artefact archivé se voit attribuer un identifiant intrinsèque, qu'on peut vérifier indépendamment (Di Cosmo *et al.,* 2020), les contenus sont dédupliqués, ce qui permet de réduire considérablement la taille de l'archive, et le graphe résultant permet de suivre la façon dont les mêmes codes sources sont réutilisés entre différents projets. L'architecture résultante est décrite dans la Figure 1 page suivante.

Une mission universelle

Au-delà de la complexité technique, la mission universelle de Software Heritage pose des défis stratégiques considérables : comment s'assurer de sa viabilité à long terme ? Comment s'assurer qu'elle reste bien au service de tous, et ne soit pas privatisée ou mise sous contrôle par des intérêts particuliers ? Comment retrouver tous les codes sources écrits dans les dernières décennies ? Comment maximiser les chances que le précieux patrimoine ainsi collecté soit préservé sur le long terme ?

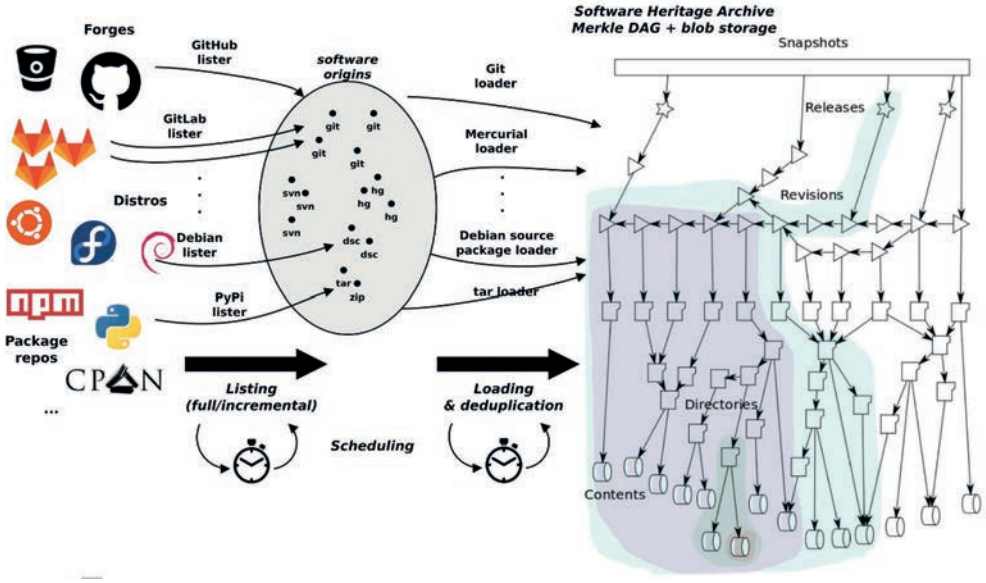

Figure 1 : Architecture du moissonneur de Software Heritage.

Ces questions ont été au cœur des réflexions qui ont mené à établir quelques principes fondateurs de Software Heritage (Abramatic *et al.*, 2018 ; Di Cosmo & Zacchiroli, 2017) : usage systématique de logiciels libres pour construire l'infrastructure de Software Heritage, afin de permettre de comprendre son fonctionnement, et de la répliquer si nécessaire ; construction d'un réseau mondial de miroirs indépendants de l'archive, parce qu'un grand nombre de copies est la meilleure protection contre les pertes et les attaques ; choix d'une structure sans but lucratif, internationale et multipartenaires, pour minimiser les risques d'avoir des points uniques de défaillance et pour s'assurer que Software Heritage sera bien au service de tous.

L'engagement dans cette initiative de personnalités qui ont une longue trajectoire au service du bien commun, comme Jean-François Abramatic et Stefano Zacchiroli, est un élément très important. Dans une telle mission, il faut bien sûr aussi une légitimité institutionnelle, et une véritable capacité d'ouverture pour fédérer un consensus large. L'accord-cadre signé entre Inria et UNESCO, le 3 avril 2017, est dans ce sens à la fois une reconnaissance de l'importance de la mission, et une grande opportunité d'établir des collaborations au niveau mondial pour l'accomplir.

Une importante étape dans cette direction a été la réunion du groupe d'experts internationaux organisée à l'UNESCO au mois de novembre 2018, qui a abouti à l'appel de Paris pour le code source des logiciels, disponible en ligne sur https://en.unesco.org/foss/paris-call-software-source-code. On y trouve une analyse détaillée des raisons pour lesquelles le code source des logiciels est devenu un enjeu majeur, et des recommandations pour des actions concrètes à mener afin de répondre aux défis qui sont posés. Parmi ces recommandations, figure le soutien à l'effort commencé avec Software Heritage pour construire une infrastructure internationale de préservation des codes sources des logiciels.

Passé, présent, futur : bien plus qu'une archive !

Software Heritage est aujourd'hui une infrastructure qui grandit jour après jour, et si le plus gros du contenu de l'archive résulte du moissonnage automatique, des perles commencent à y être introduites par un patient travail de récupération de logiciels historiques marquants, suivant un processus d'acquisition qui a été mis au point en collaboration avec l'Université de Pise et l'UNESCO [3].

(3) Voir SWHAP sur https://www.softwareheritage.org/swhap

Figure 2 : Nombre de projets, fichiers sources et versions archivées dans Sofware Heritage au mois de janvier 2020 (voir https://www.softwareheritate.org/archive).

L'exhaustivité est encore loin d'être atteinte, mais l'archive contient déjà le plus grand corpus de codes sources disponible sur la planète, avec plus de 90 millions d'origines archivées, pour plus de 6 milliards de fichiers sources uniques, chacun équipé d'un identifiant intrinsèque basé sur des *hash* cryptographiques (Di Cosmo *et al.*, 2018).

Cette infrastructure unique a de multiples missions : bien sûr, il s'agit de préserver pour les futures générations les codes sources du *passé* qui ont fait l'histoire de l'informatique et de la société de l'information, mais aussi, et surtout, on cherche ici à construire le *très grand télescope* qui permette d'explorer l'évolution *présente* de la *galaxie du développement logiciel*, afin de mieux la comprendre, et de l'améliorer, pour construire un *futur* technologique meilleur.

Un enjeu stratégique

L'archive de Software Heritage constitue déjà la collection de codes sources la plus importante de la planète, mais le chemin à parcourir est encore long : il est nécessaire de continuer à réunir les compétences scientifiques et techniques, ainsi que les ressources financières et humaines, afin de pouvoir construire la mémoire d'une partie importante de la technologie et de la science qui est au cœur de la transition numérique, à un moment où l'on peut encore espérer avoir accès à tout ce qui a été mis en œuvre dès le début de l'histoire, encore courte, de l'informatique.

Mais il y a bien plus que cela : à un moment où l'on voit clairement que le logiciel est devenu un composant essentiel de toute activité humaine, l'accès sans restriction aux codes sources des logiciels publiquement disponibles, ainsi qu'à l'information qualifiée sur leur évolution, devient un enjeu de souveraineté numérique pour toutes les nations.

L'infrastructure unique que construit Software Heritage, et son approche universelle sont un élément essentiel pour répondre à cet enjeu de souveraineté numérique, tout en préservant la dimension de bien commun propre à l'archive.

Il est donc de la plus haute importance que des acteurs institutionnels, industriels, académiques et de la société civile saisissent l'importance de ces enjeux, et que la France et l'Europe se positionnent rapidement, en fournissant les ressources nécessaires à faire grandir et pérenniser Software Heritage, en prenant leur place à côté des autres acteurs internationaux qui se sont déjà engagés, et en soutenant la création d'une institution internationale sans but lucratif qui porte cette mission sur le long terme.

Pour en savoir plus

On peut en savoir plus sur le projet en visitant www.softwareheritage.org, annex.softwareheritage.org et wiki.softwareheritage.org
Il est possible d'explorer aisément les codes sources contenus dans Software Heritage sur archive.softwareheritage.org

Licence

Bibliographie

ABELSON H. & SUSSMAN G. J. S. with J. (1985), *Structure and Interpretation of Computer Programs*, The MIT Press.

ABRAMATIC J.-F., DI COSMO R. & ZACCHIROLI S. (2018), "Building the Universal Archive of Source Code", *Commun. ACM*, *61*(10), 29-31. https://doi.org/10.1145/3183558

DI COSMO R., GRUENPETER M. & ZACCHIROLI S. (septembre 2018), "Identifiers for Digital Objects: The Case of Software Source Code Preservation", *Proceedings of the 15th International Conference on Digital Preservation*, iPRES 2018, Boston, USA. https://doi.org/10.17605/OSF.IO/KDE56

DI COSMO R., GRUENPETER M. & ZACCHIROLI S. (2020), "Referencing Source Code Artifacts: A Separate Concern in Software Citation", *Computing in Science & Engineering*. https://doi.org/10.1109/MCSE.2019.2963148

DI COSMO R. & ZACCHIROLI S. (septembre 2017), "Software Heritage: Why and How to Preserve Software Source Code", *Proceedings of the 14th International Conference on Digital Preservation*, iPRES 2017. https://hal.archives-ouvertes.fr/hal-01590958/

KNUTH D. E. (1984), "Literate Programming", *Comput. J.*, 27(2), 97-111. https://doi.org/10.1093/comjnl/27.2.97

MERKLE R. C. (1987), "A Digital Signature Based on a Conventional Encryption Function", *Advances in Cryptology - CRYPTO '87, A Conference on the Theory and Applications of Cryptographic Techniques,* Santa Barbara, California, USA, August 16-20, 1987, *Proceedings*, 369-378. https://doi.org/10.1007/3-540-48184-2_32

NOORDEN R. V., MAHER B. & NUZZO R. (2014), "The top 100 papers", *Nature*, 514(7524), 550-553. https://doi.org/10.1038/514550a

SHUSTEK L. J. (2006), "What Should We Collect to Preserve the History of Software?", *IEEE Annals of the History of Computing*, 28(4), 110-112. https://doi.org/10.1109/MAHC.2006.78