Entree : Code Source
Auteur : Roberto Di Cosmo

« *Les programmes doivent être écrits pour que les gens puissent les lire, et seulement accessoirement pour que les machines les exécutent* »

Harold Abelson, Structure and Interpretation of Computer Programs, 1984

Dans un système informatique, on distingue generalement deux parties : le *matériel (hardware* en anglais*)*, qui constitue la partie physique du système (processeur, mémoire, disques, écran, carte réseau, carte son, clavier, souris, etc.), et le *logiciel* (*software* en anglais), qui désigne l'ensemble des instructions qui peuvent être mémorisées et executées par la machine. C'est bien le logiciel qui donne vie à un système informatique.

Les instructions directement exécutables par une machine (on appelle cela le « langage machine ») sont souvent de très bas niveau, représentées par des simples suites de bits, et difficiles à comprendre par un être humain. C'est pour cela que les logiciels ne sont presque jamais produits directement en langage machine, mais « écrits » par des dévéloppeurs en utilisant un « langage de programmation », qui peut être ensuite traduit automatiquement dans le langage machine[1].

A titre d'exemple, voici un extrait du programme executable qui imprime un simple message « Hello World »

```
4004e6: 55
4004e7: 48 89 e5
4004ea: bf 84 05 40 00
4004ef: b8 00 00 00 00
4004f4: e8 c7 fe ff ff
4004f9: 90
4004fa: 5d
4004fb: c3
```

et voici le programme écrit dans langage de programmation C à partir duquel le programme éxécutable dont on a extrait le fragment présenté plus haut a été produit

```
/* Hello World program */

#include<stdio.h>

void main()
```

1 Voir l'entrée « langage » du dictionnaire

```
{
    printf("Hello World");
}
```

On entend en général par « code source » d'un logiciel le programme qui a été écrit par le développeur et à partir duquel on obtien le code exécutable sur une machine, et on pourrait être temptés de considérer comme « code source » tout programme écrit dans un langage de programmation. Avec l'évolution de la technologie, la situation s'est faite en réalité plus complexe : les développeurs disposent d'outils sophistiqués qui peuvent « produire » des programmes dans un langage de programmation (comme C) à partir de programmes écrits dans des langages de programmation de niveau plus élévé, au point qu'il ne suffit pas de regarder le langage dans lequel est écrit un programme pour savoir si ce programme a bien été écrit par un développeur, ou engendré automatiquement à partir d'un programme de plus haut niveau.

C'est pour cela que la définition de « code source » d'un logiciel que l'on retrouve dans la licence GPL[2] est « la forme préférée pour un développeur pour apporter une modification à un programme ».

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 *exécutée* 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 Donald Knuth, un des fondateurs de l'Informatique, l'a écrit, « la programmation est l'art d'expliquer à un autre être humain ce que l'on veut faire faire à un ordinateur ».

Le code source des logiciels est donc bien une *création humaine* au même titre que d'autres documents écrits, et c'est pour cela qu'il rentre dans le domaine d'application du droit d'auteur. Les développeurs des logiciels méritent ainsi le même respet que les autres créateurs, et il est essentiel de s'assurer que toute modification du droit d'auteur prenne en compte l'impact potentiel sur le développement des logiciels. Cela n'a pas été le cas lors de la redaction de la Directive Europeenne 2019/790, dont la première mouture a mis sérieusement en danger le développement collaboratif des logiciels libres, et qui a nécéssité un effort significatif pour apporter les indispensables corrections[3].

De plus en plus complexe, le code source des logiciels 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 est donc en train d'incorporer *une partie importante* de notre patrimoine scientifique, technique et industriel, et constitue donc un patrimoine de grande valeur, comme déjà soutenu par Len Shustek dans un bel article de 2006[4.]

2 Gnu general public license, version 2, 1991. Une des premières licences de logiciel libre, voir l'entrée « logiciel libre ».

3 Voir un résumé sur https://www.dicosmo.org/MyOpinions/index.php?post/2019/04/17/Saving-software-development-from-the-European-copyright-reform

4 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

C’est bien celle-là une des missions qui s’est donné Software Heritage[5], une initiative lancée en 2015 avec le soutien de l’Inria[6], en partenariat avec l’UNESCO, pour *récolter, organiser, préserver et rendre facilement accessible* l’ensemble 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.

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 enjeux de souveraineté numérique pour toutes les nations. L’infrastructure unique que Software Heritage construit, et son approche universel, est un élément essentiel pour repondre à cet enjeu de souveraineté numérique tout en préservant la dimension de bien commun propre à l’archive.

5 Voir https://www.softwareheritage.org/

6 Créé en 1967, 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.