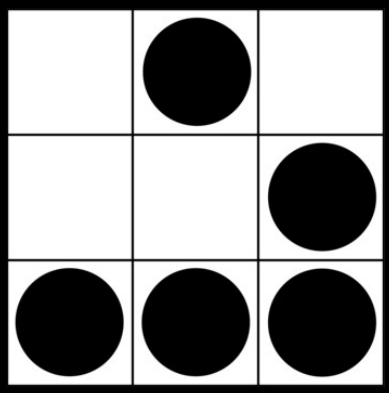
Pour la Liberté

Libre comme dans Liberté

Auteur: Sam Williams

Traduction: Collectif Wikisource

Version 9.3



Pour la Liberté...

Libre comme dans Liberté

Un hacker est quelqu'un qui apprécie l'intelligence espiègle. Je sais que pour beaucoup de personnes il représente un pirate informatique, mais puisqu'au sein de ma communauté nous nous appelons « hacker » je n'accepterai aucune autre signification. Si vous voulez parler de ces personnes qui cassent les codes de sécurité vous devriez parler de « cracker ». Le terme « hacker » ne se limite pas au domaine des ordinateurs. Au Massachusetts Institute of Technology il existe une ancienne tradition, les gens « hackent » les bâtiments et les lieux publics en y accrochant le fameux panneau de signalisation « Nerd Crossing » par exemple. Aucune sécurité n'est détournée et c'est espiègle et intelligent.

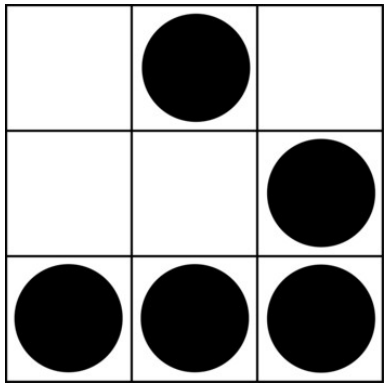
Richard M. Stallman

Table des matières

Libre comme dans Liberté

La Croisade de Richard Stallman pour le Logiciel Libre

Auteur: Sam Williams
Traduction: Collectif Wikisource



Cette version est tirée de FAIFzilla 1.0 (http://www.faifzilla.org/), dernière entrée en 2004 . Ce livre est licencié sous la GNU Free Documentation License. Il fut initialement publié en ligne et imprimé par O'Reilly and Associates en 2002.

Cette traduction peut être consultée en ligne à l'adresse http://fr.wikisource.org/wiki/Libre_comme_Liberté

Préface

L'œuvre de Richard M. Stallman parle d'elle-même. Du code source documenté aux publications et aux discours enregistrés, peu de gens ont montré tant de volonté à mettre en jeu leur pensée et leur travail.

Une telle ouverture — si l'on peut excuser momentanément un tel adjectif non-stallmanien — est rafraîchissante. Après tout, nous vivons dans une société qui considère l'information, notamment personnelle, comme une chose de valeur. On peut effectivement se demander pour quelle raison quelqu'un voudrait donner autant d'informations et sembler ne rien demander en retour!

Comme nous le verrons dans les chapitres suivants, Stallman ne partage pas ses mots ou son travail de façon altruiste. Chaque programme, discours et bon mot enregistré a un prix, mais, dans son genre, pas celui que nous avons l'habitude de payer.

Je n'avance pas cela comme un avertissement, mais comme un postulat. Ayant passé cette dernière année à déterrer des éléments de l'histoire personnelle de Stallman, il est plutôt intimidant d'aller à l'encontre de l'œuvre de Stallman. « Ne choisis pas comme adversaire un homme qui achète son encre par tonneau », dit le vieil adage de Mark Twain. Dans le cas de Stallman : ne t'attaque pas à la biographie définitive d'un homme qui confie sa pensée au domaine public.

Pour les lecteurs qui ont décidé de consacrer quelques heures de leur temps à explorer ce livre, je peux en confiance affirmer qu'il y a ici des faits et des citations qu'on ne trouve dans nulle histoire de *Slashdot* ou recherche de *Google*. Cependant, avoir accès à ces informations a un prix. Concernant la version papier, vous pouvez payer pour celle-ci de la façon traditionnelle, c'est-à-dire en achetant le livre. Dans le cas des versions électroniques, vous pouvez payer pour ces informations à la façon des logiciels libres. Grâce aux gens de *O'Reilly & Associates*, ce livre est distribué sous la Licence de Documentation Libre de GNU, ce qui signifie que vous pouvez aider à améliorer le travail ou créer une version personnalisée et la publier sous la même licence.

Si vous lisez une version électronique et préférez accepter cette dernière option de paiement, c'est-à-dire si vous voulez améliorer ou compléter ce livre pour les lecteurs futurs, votre contribution est la bienvenue. À partir de juin 2002, je vais publier une version HTML de ce livre sur le site web http://www.faifzilla.org. Mon but est de le mettre à jour régulièrement et de compléter le récit de "*Free as in freedom*" au fur et à mesure des évènements. Si vous choisissez cette dernière solution, consultez s'il vous plaît l'annexe C de ce livre. Elle contient une copie de vos droits sous la Licence de Documentation Libre de GNU.

Pour ceux dont l'intention est seulement de s'asseoir et de lire, en ligne ou autrement, je considère votre intérêt comme une forme de paiement ayant tout autant de valeur. Ne soyez pas surpris cependant, si, vous aussi, vous vous surprenez à chercher d'autres façons de récompenser la bonne volonté qui a rendu ce travail possible.

Une dernière remarque : ceci est un travail de journalisme, mais c'est également un travail de documentation technique. Au cours du travail d'écriture et d'édition de ce livre, les éditeurs et moi-même avons évalué les commentaires et les faits apportés par les différents participants de l'histoire, y compris Richard Stallman en personne. Nous avons pris conscience que de nombreux détails techniques de l'histoire pourraient bénéficier d'informations complémentaires ou affinées. Comme cet ouvrage est publié sous la LDLG[1], nous acceptons les patchs comme nous le ferions pour n'importe quel logiciel libre. Les modifications acceptées seront envoyées électroniquement et finalement incorporées dans les futures versions imprimées de cet ouvrage. Si vous souhaitez contribuer à l'amélioration future de ce livre, écrivez-moi à l'adresse sam@inow.com.

Remerciements

Remerciements particuliers à Henning Gutmann pour s'être accroché à ce livre. Remerciements particuliers à Aaron Oas pour avoir suggéré l'idée à Tracy en premier. Merci à Laurie Petrycki, Jeffrey Holcomb, et tous les autres de *O'Reilly & Associates*. Merci à Tim O'Reilly pour avoir soutenu ce livre. Merci à tous les relecteurs du premier essai : Bruce Perens, Eric Raymond, Eric Allman, Jon Orwant, Julie et Gerald Jay Sussman, Hal Abelson et Guy Steele. J'espère que vous appréciez cette version exempte de coquilles. Merci à Alice Lippman pour les interviews, les cookies, et les photographies. Merci à ma famille, Steve, Jane, Tish et Dave. Et enfin, le dernier mais pas le moindre : merci à Richard Stallman pour avoir eu les tripes et la persévérance de « nous montrer le code ».

Sam Williams

Notes

1. Il s'agit de la Licence pour la Documentation Libre GNU (GNU Free Documentation Licence -- GFDL). Pour plus d'information, voir http://www.gnu.org/licenses/fdl-1.2.html et http://www.gnu.org/licenses/licenses.fr.html -- NdT.

(Traduction [FR] de cette page : <Valéry Beaud>)

Chapitre I — Faute d’une imprimante

« Je crains les Grecs. Même lorsqu’ils apportent des cadeaux. »

— Virgile *(L’Énéide)*

La nouvelle imprimante était une nouvelle fois bloquée.

Pour Richard M. Stallman, vingt-sept ans, programmeur au laboratoire d'intelligence artificielle (AI Lab) du Massachussetts Institute of Technology (Institut Technologique du Massachussetts -- MIT), la découverte du dysfonctionnement fut rude. Une heure après l'envoi d'un fichier de cinquante pages à l’imprimante laser du bureau, il dût interrompre une séance productive de travail afin de récupérer ses documents. À l’arrivée, il ne trouva que quatre pages dans le bac. Plus frustrant, elles appartenaient à un autre utilisateur, signifiant que son travail d’impression, et celui inachevé d'une seconde personne, étaient coincés quelque part dans les conduits électriques du réseau informatique du labo.

Attendre les machines fait partie des risques du métier de développeur de logiciels. Stallman accusa donc le coup. Il y a, cependant, une différence notable entre attendre une machine et attendre devant une machine. Ce n’était pas la première fois qu’il se voyait contraint à rester devant l’imprimante regardant les pages sortir une à une. Passant la plupart de ses jours et nuits à améliorer les performances des machines et des logiciels les contrôlant, il ressentit tout naturellement l’envie de l'ouvrir, regarder dans ses entrailles, et rechercher l'origine du problème.

Malheureusement, ses talents de programmeur ne s’étendaient pas au monde de l’ingénierie mécanique. Alors que les documents fraîchement imprimés sortaient de la machine, il put réfléchir à d’autres manières de contourner le problème des bourrages de papier.

Combien de temps s’était-il écoulé depuis la réception, à bras ouverts, de la nouvelle imprimante par le personnel du AI Lab ? Stallman se le demandait. La machine était un don de la Xerox Corporation : un prototype de pointe, version modifiée du classique photocopieur Xerox. Ne se limitant pas à faire des copies, il transformait des données, acheminées par le réseau informatique, en documents d’aspect professionnel. Inventé par les ingénieurs du mondialement renommé Centre de Recherche de Xerox à Palo Alto, c’était, tout simplement, un avant-goût de la révolution touchant l’impression bureautique qui saisirait le reste de l’industrie informatique à la fin de la décennie.

Poussés par un besoin instinctif de jouer avec l’équipement dernier cri, les programmeurs du AI Lab avaient rapidement incorporé la nouvelle machine dans l’infrastructure informatique sophistiquée du labo. Les résultats avaient immédiatement plu. Comparé à la vieille imprimante laser, le nouvel appareil de Xerox était rapide. Les pages volaient au débit d’une par seconde : un travail d’impression de vingt minutes passait à deux minutes. Il était également plus précis. Les cercles ressemblaient à des cercles, non à des ovales. Les lignes droites étaient droites, pas des courbes sinusoïdales de faible amplitude.

C’était, à tout point de vue, un cadeau impossible à refuser.

Quelques semaines à peine après son arrivée, les imperfections de la machine firent surface. La plus sérieuse était sa propension aux bourrages de papier. Les programmeurs, ingénieurs dans l'âme, comprirent rapidement la raison de ce défaut : comme un photocopieur, la machine réclamait, généralement, une surveillance humaine. Pensant qu'un opérateur humain serait toujours disponible pour réparer ces incidents s’ils se produisaient, les ingénieurs de Xerox consacrèrent leur temps et leur énergie à résoudre d'autres problèmes empoisonnants. En termes technologiques, la diligence des utilisateurs avait été intégrée au système.

En transformant un photocopieur en imprimante, les ingénieurs de Xerox avaient changé de manière subtile et profonde le rapport entre l’utilisateur et la machine. Au lieu de la rendre dépendante d'un seul humain, elle devint tributaire d'un réseau entier d'opérateurs. Plutôt que de se tenir à proximité, un utilisateur, situé à une extrémité du réseau, envoyait sa commande d'impression à travers une chaine étendue de machines, espérant que le contenu souhaité arrivât à destination prévue sous une forme convenable. Ce

n’était qu’en vérifiant la production finale qu’il réalisait que, seule, une infime partie de son document correspondait au résultat désiré.

Stallman fut l’un des premiers à identifier le problème et suggérer un remède. Des années auparavant, alors que le laboratoire utilisait toujours sa vieille imprimante, il résolut un incident semblable en ouvrant le logiciel pilotant la machine sur le PDP-11. Stallman ne put éliminer les bourrages de papier, mais il réussit à insérer un morceau de code, lequel demandait au PDP-11 de vérifier l'appareil périodiquement et d'envoyer les rapports au PDP-10, l’ordinateur central du laboratoire. Afin de s'assurer que la négligence d'un utilisateur n’avait pas empêché l’impression de documents, Stallman inséra également un code demandant au PDP-10 d’informer du blocage de l’imprimante toute personne en attente d'un tirage. La notification était simple et se résumait à quelques mots comme « L’imprimante est bloquée, merci de vérifier ». Ce message étant envoyé aux plus pressés de voir leurs travaux imprimés, la probabilité était grande de voir le problème rapidement résolu.

La solution de Stallman, tenant du système D, était cependant élégante. Elle ne réglait pas le problème mécanique, mais permettait de créer un retour d’information entre l’utilisateur et la machine. Grâce à quelques lignes de code, les employés du AI Lab évitaient les dix minutes voire les quarts d’heure perdus chaque semaine en aller-retours pour vérifier l'imprimante. En termes de programmation, l’astuce de Stallman exploita l’intelligence amplifiée du réseau entier.

« Si vous receviez ce message, vous ne pouviez en déduire qu'un autre s’en chargeait », affirme logiquement Stallman. « Vous deviez aller à l’imprimante. Une ou deux minutes après que l’appareil tombait en panne, les deux ou trois personnes ayant reçu le message arrivaient pour réparer la machine. Sur ces deux ou trois, au moins une savait comment résoudre le problème ». De tels stratagèmes étaient une marque de fabrique du laboratoire et de sa population résidente de programmeurs. En fait, les meilleurs dédaignaient ce dernier terme, lui préférant celui, plus argotique, de hacker (bidouilleur). Ce titre couvrait une foule d’activités – tout, de la création amusante à l’amélioration des programmes et systèmes informatiques existants. Cette appellation sous-entend, toutefois, la notion démodée d’ingéniosité du Yankee. Pour être un hacker, il fallait accepter la philosophie selon laquelle écrire un logiciel n'était que le début. Améliorer un programme était le véritable test validant les compétences du hacker[1].

Cette philosophie motiva principalement des sociétés comme Xerox pour mener une politique de don de machines et logiciels aux endroits où les hackers se rassemblaient habituellement. En retour, si ces derniers amélioraient le logiciel, les sociétés pouvaient leur emprunter ces améliorations et ensuite les intégrer dans des mises à jour de versions commerciales. En termes économiques, la communauté des hackers créait, par effet de levier, un département auxiliaire de recherche et développement disponible à moindre coût.

À cause de cette culture de la concession mutuelle, Stallman ne s’est pas affolé lorsqu’il découvrit le problème de bourrage. Il chercha simplement à mettre à jour l’ancienne version ou plutôt, « hacker » une nouvelle. Cependant, lorsqu’il chercha le logiciel de l’imprimante Xerox, il fit une découverte troublante : il n’y avait aucun logiciel, du moins, rien de lisible par Stallman ou d’autres programmeurs. Jusqu’alors, par courtoisie, la plupart des sociétés publiaient le code source dans des fichiers lisibles, documentant chaque commande disant ce qu’une machine devait faire. Or, en l’occurrence, Xerox avait fourni les fichiers du logiciel sous une forme compilée (binaire). Les programmeurs étaient libres d’ouvrir ces derniers s’ils le voulaient, mais à moins d’être un expert dans le décryptage d’une suite infinie de zéro et de un, le texte qui en sortait était un pur charabia.

Quoique Stallman en sût long sur les ordinateurs, il ne pouvait traduire les fichiers binaires. En tant que hacker, cependant, il avait d’autres ressources à sa disposition. Le partage de l’information était une notion tellement fondamentale dans cette culture que Stallman savait que ce n’était qu'une question de temps avant qu’un autre hacker, quelque part dans un laboratoire universitaire ou une salle informatique d’entreprise, soit en mesure d’offrir une version du code source de l’imprimante avec les fichiers désirés.

Après les premiers bourrages d’imprimante, Stallman se réconforta au souvenir d’une situation similaire quelques années auparavant. Le laboratoire avait eu besoin d’un programme fonctionnant en réseau afin de permettre au PDP-11 de travailler plus efficacement que le PDP-10. Les hackers du laboratoire étaient alors vraiment surchargés de travail, mais Stallman, qui avait étudié à Harvard, se rappelait d’un programme

identique écrit par les programmeurs du département de science informatique de son ancienne université. Ce laboratoire informatique utilisait le même modèle d'ordinateur, le PDP-10, mais avec un système d'exploitation différent. Il avait aussi une politique stipulant que tous les programmes installés sur le PDP-10 devaient être fournis avec une version publiée des fichiers du code source.

Profitant de son accès au laboratoire d'informatique d'Harvard, Stallman s'y rendit, fit une copie du code source à travers le réseau, et le ramena au AI Lab. Là, il réécrivit le code source pour l'adapter au système d'exploitation du laboratoire. Ainsi, sans difficulté et avec un minimum d'efforts, le AI Lab combla une faille majeure dans son infrastructure logicielle. De plus, Stallman avait ajouté au programme quelques fonctions qui n'apparaissaient pas dans la version originale d'Harvard, ce qui augmenta d'autant plus son utilité. Il conclut : « Nous avons fini par l'utiliser quelques années ».

Du point de vue d'un programmeur des années 1970, cette opération sur logiciel équivalait à la visite d'un voisin s'arrêtant chez vous pour vous emprunter un appareil électro-ménager ou un peu de sucre. La seule différence était qu'en empruntant une copie du logiciel pour le AI Lab, Stallman n'avait rien fait qui ne privât les hackers de Harvard de l'utilisation de leur programme original. Bien au contraire, ils étaient gagnants car Stallman avait intégré ses propres fonctions supplémentaires, fonctions qu'ils étaient tout à fait libres d'emprunter à leur tour. Quoique personne à Harvard ne se manifestât pour emprunter la nouvelle version du programme, Stallman se souvient d'un programmeur de chez Bolt, Beranek & Newman — une société d'ingénierie privée — qui le fit, y ajoutant des fonctions supplémentaires que Stallman, à son tour, intégra dans l'archive du code source du AI Lab.

Selon Stallman, rappelant ainsi l'infrastructure logicielle du AI LAb, « un programme devrait se développer comme une ville. Des portions peuvent être remplacées et reconstruites. De nouveaux éléments peuvent être ajoutés. Mais vous pourriez toujours en regarder un bout et dire : 'Hum, d'après le style, je pense que cette partie a été écrite dans les années 1960 et cette autre au milieu des années 1970'. »

Par ce simple système d'addition intellectuelle, les hackers du AI Lab et d'ailleurs avaient créé des programmes robustes. Sur la Côte Ouest, les informaticiens de Berkeley (UC), coopérant avec quelques ingénieurs en langage de bas-niveau de chez AT&T, avaient ainsi construit un système d'exploitation complet. Baptisé Unix, un jeu de mot à partir de Multics (système d'exploitation plus ancien et plus respectable sur un plan académique), cet ensemble logiciel était disponible pour tout programmeur prêt à payer le prix d'une copie sur bande magnétique neuve et les frais d'expédition. Tous ceux qui prenaient part à cette culture ne se disaient pas hackers, mais la plupart partageaient les sentiments de Richard M. Stallman. Si un programme ou une correction logicielle sont assez bons pour résoudre vos problèmes, ils sont assez bons pour résoudre ceux d'autres personnes. Pourquoi ne pas les partager, sinon pour y gagner un bon karma ?

Au début, le fait que Xerox n'était pas disposé à partager les fichiers du code source ne semblait être qu'un petit désagrément. Stallman indique que, dans ses recherches de la copie du code source, il n'a même pas pris la peine de contacter Xerox. « Ils nous avaient déjà donné l'imprimante laser », dit-il, « pourquoi les aurais-je sollicité davantage ? »

Cependant, les fichiers désirés ne faisant toujours pas surface, Stallman commença à avoir des soupçons. L'année précédente, il avait connu une expérience houleuse avec un doctorant de l'Université Carnegie Mellon. L'étudiant, Brian Reid, était l'auteur d'un programme de formatage de texte appelé *Scribe*. Il s'agissait de l'un des premiers programmes permettant à l'utilisateur de définir les polices et les styles d'un document envoyé à travers un réseau informatique. Ce programme était un précurseur du HTML, la lingua franca du World Wide Web. En 1979, Reid prit la décision d'envoyer *Scribe* à une société informatique de la région de Pittsburgh appelée Unilogic. Ses hautes études terminées, Reid affirma qu'il cherchait simplement un moyen de se décharger du programme sur une équipe de développeurs prête à tout pour l'empêcher de tomber dans le domaine public. Pour assouplir la transaction, Reid accepta également d'intégrer un ensemble de fonctions-calendrier -- des « bombes à retardement », dans le langage des programmeurs -- désactivant les versions du programme copiées gratuitement après 90 jours de délai d'expiration. Afin d'empêcher la désactivation, les utilisateurs payaient la société informatique, laquelle fournissait ensuite un code désamorçant le dispositif à retardement.

Pour Reid, l'échange était gagnant-gagnant. *Scribe* ne tombait pas dans le domaine public et Unilogic

amortissait son investissement. Pour Stallman, cela trahissait purement et simplement l'éthique du programmeur. Au lieu d'appliquer l'idée de partage mutuel, Reid avait initié une méthode permettant aux entreprises de contraindre les programmeurs à payer l'accès à l'information.

Au fil des semaines, ses tentatives pour trouver le code source de l'imprimante Xerox se heurtant à un mur, Stallman subodora la mise en oeuvre d'un scénario identique. Cependant, avant qu'il ne puisse dire ou faire quoi que ce soit, de bonnes nouvelles se diffusèrent via le bouche-à-oreille des programmeurs. On disait qu'un chercheur du département de sciences informatiques de l'Université Carnegie Mellon avait récemment quitté un poste de travail au Xerox Palo Alto Research Center (PARC). Non seulement ce chercheur avait travaillé sur l'imprimante laser en question mais, d'après la rumeur, il y travaillait encore pour les besoins de ses recherches à Carnegie Mellon.

Mettant de côté ses soupçons, Stallman prit la ferme résolution d'aller trouver l'individu en question lors de sa prochaine visite sur ce campus.

Il n'eut pas à attendre longtemps. Carnegie Mellon disposait aussi d'un laboratoire spécialisé dans la recherche en intelligence artificielle, et, après quelques mois, Stallman profita d'une raison professionnelle pour effectuer sa visite. Au cours de celle-ci, il s'arrangea pour passer au département de sciences informatiques. Les employés le dirigèrent alors vers le bureau du chercheur dirigeant le projet Xerox. Arrivé au bureau, Stallman y trouva le professeur au travail.

Comme toute conversation entre ingénieurs, celle-ci fut cordiale mais directe. Après s'être brièvement présenté comme un visiteur du MIT, Stallman demanda une copie du code source du pilote de l'imprimante laser, afin de pouvoir le porter sur le PDP-11. À sa surprise, le professeur refusa d'accéder à sa requête.

« Il m'a confié qu'il avait promis de ne pas me donner de copie », dit Stallman.

La mémoire humaine est étrange. Vingt ans après les faits, il est notoire que la bande mentale des souvenirs de Stallman comporte quelques blancs. Non seulement il ne se souvient pas des raisons motivant ce voyage, ou de la période de l'année, mais aussi du professeur ou doctorant avec qui il parla. Selon Reid, la personne ayant probablement répondu aux demandes de Stallman est Robert Sproull, ancien chercheur au Xerox PARC et actuel directeur de Sun Laboratories, une division de recherche du conglomérat informatique Sun Microsystems. Dans les années soixante-dix, lors de son passage au Xerox PARC, Sproull fut le principal développeur du logiciel de l'imprimante laser en question. Vers 1980, il intégra le département de recherche de Carnegie Mellon où il poursuivit ses travaux sur les imprimantes laser, entre autres projets.

« Le code demandé par Stallman était le nec plus ultra. Un code pointu rédigé par Sproull une année environ avant d'aller à Carnegie Mellon », se rappelle Reid. « Je soupçonne que Sproull y était depuis moins d'un mois quand cette demande est arrivée. »

Interrogé directement sur cette demande, Sproull a un trou de mémoire. « Je ne peux commenter les faits », écrit-il dans un courriel. « Je n'ai aucun souvenir de cet incident. »

Les deux participants à ce bref entretien ayant du mal à se rappeler de détails cruciaux, y compris le lieu où il se produisit, il est difficile de juger de la fermeté de la réponse de Sproull, du moins d'après les dires de Stallman. En public, Stallman fit plusieurs fois référence à cet incident, pour faire remarquer que la réticence de Sproull à donner le code source venait d'un accord de non-divulgation, d'un contrat entre Sproull et la Xerox Corporation, lequel donnait accès au code source du logiciel à lui ou à tout autre signataire en échange de leur discrétion assurée. Maintenant habituelle dans l'industrie du logiciel, la clause de confidentialité en était à ses balbutiements. C'était le reflet à la fois de la valeur commerciale de l'imprimante laser pour Xerox, et des informations nécessaires à son bon fonctionnement. « Xerox, à l'époque, essayait de faire de l'imprimante laser un produit commercial », explique Reid. « Ils auraient été fous de faire cadeau du code source. »

Pour Stallman, la clause de non-divulgation était tout à fait autre chose. C'était le refus de Xerox et de Sproull, ou de quiconque ayant refusé d'accéder à sa demande de code source ce jour-là, de participer à un système qui, jusqu'à présent, avait encouragé les programmeurs à considérer leurs travaux comme des ressources communes. Tel le paysan qui voit le ruisseau irriguant ses champs depuis des siècles se tarir brutalement, Stallman avait suivi le ruisseau jusqu'à sa source, pour n'y trouver qu'un barrage

hydroélectrique flambant neuf, orné d'un beau logo Xerox.

Stallman mit quelques temps à digérer le fait que Xerox ait contraint un collègue programmeur à prendre part à ce système en vogue de secret forcé. Dans un premier temps, il ne put se concentrer que sur le caractère personnel de ce refus. Se sentant maladroit et décalé dans la plupart des face-à-face, sa tentative de visite imprévue à un collègue programmeur se voulait une marque de bon voisinage. Maintenant que sa requête était refusée, cela lui semblait être une grossière erreur. « J'étais tellement en colère que je n'arrivais pas à l'exprimer. J'ai juste fait demi-tour, et suis sorti sans un mot », se souvient Stallman. « J'ai peut-être claqué la porte. Qui sait ? Tout ce dont je me souviens, c'est que je voulais sortir de là. »

Vingt ans après les faits, la colère est toujours là, à tel point que Stallman a promu cet évènement au rang de tournant décisif. Les mois suivants, une série d'évènements impliquant Stallman et la communauté de hackers du AI Lab aurait dû, en comparaison, renvoyer au rang de simple détail ces trente secondes de tension dans un bureau obscur de Carnegie Mellon. Mais au moment de trier les évènements qui le transformèrent de hacker isolé, instinctivement suspicieux des autorités centralisées, en un activiste en croisade appliquant, au monde du logiciel libre, les notions traditionnelles de liberté, d'égalité, et de fraternité, Stallman choisit de donner un relief tout particulier à cette rencontre de Carnegie Mellon.

« Cela me rappela quelque chose qui m'avait déjà traversé l'esprit », raconte-t-il. « J'avais déjà l'idée que les logiciels devaient être partagés, mais je n'étais pas sûr de ce qu'il fallait en conclure. Mes pensées n'étaient pas claires et organisées au point de pouvoir les exprimer de manière concise au reste du monde. »

Bien que divers épisodes aient auparavant déclenché les foudres de Stallman, il dit ne pas s'être rendu compte, avant la rencontre de Carnegie Mellon, que ces évènements commençaient à faire intrusion dans une culture qu'il considérait depuis longtemps comme sacro-sainte. Appartenant à l'élite des programmeurs dans l'une des meilleures institutions du monde, Stallman était tout à fait prêt à ignorer les compromis et les marchandages de ses collègues, tant qu'ils n'interféraient pas avec son travail. Jusqu'à l'arrivée de l'imprimante laser Xerox, Stallman se contentait de prendre de haut ces machines et programmes que d'autres utilisateurs toléraient à reculons. Les rares occasions où un tel programme franchissait les portes du AI Lab -- par exemple le jour où le labo remplaça son vénérable système d'exploitation ITS (Incompatible Time Sharing) par une variante commerciale, le TOPS 20 -- Stallman et ses collègues hackers furent libres de réécrire, remettre en forme et renommer le logiciel selon leurs goûts personnels.

Pourtant, une fois que l'imprimante laser s'était insinuée dans le réseau du AI Lab, quelque chose avait changé. La machine fonctionnait très bien, en dépit de bourrages épisodiques, mais la possibilité de l'accommoder à son goût personnel avait disparu. Au niveau de l'industrie du logiciel toute entière, l'imprimante était un réveil brutal. Le logiciel était devenu un actif de valeur telle que les sociétés ne ressentaient plus la nécessité de publier le code source, surtout si sa publication revenait à donner la possibilité aux concurrents potentiels de reproduire à meilleur marché. Pour Stallman, l'imprimante était un cheval de Troie. Après dix ans d'échecs, le logiciel du domaine privé -- les futurs hackers utiliseront l'expression « logiciel propriétaire » -- avait établi une tête de pont à l'intérieur du AI Lab par la méthode la plus sournoise qui soit. Il s'était introduit déguisé en cadeau.

Il était tout aussi irritant que Xerox ait pu offrir des cadeaux supplémentaires à quelques programmeurs en échange du secret. Toutefois, Stallman reconnaît à contrecœur que dans sa jeunesse, il aurait sauté sur l'offre de Xerox, même si le marché proposé reposait sur un tel quiproquo. L'entretien malencontreux de Carnegie Mellon eut un effet certain sur le moral de Stallman. En conséquence, non seulement était-il assez en colère pour considérer toutes les futures démarches avec suspicion, mais il était obligé de se poser la délicate question : qu'arriverait-il si, un jour, un ami hacker surgissait dans son bureau pour lui demander le code source et que, du jour au lendemain, son travail consistait à le lui refuser ?

« C'était la première fois que je rencontrais le problème de la clause de confidentialité, et, sur le coup, j'ai tout de suite pensé qu'elle ferait des victimes », pensait sérieusement Stallman. « En l'occurrence, j'étais la victime. [Mon labo et moi] étions des victimes. »

Telle était la leçon que Stallman retint au cours des tumultueuses années 80, une décennie durant laquelle beaucoup de ses collègues du MIT partirent du AI Lab et signèrent de leur propre chef des clauses de confidentialité (*Nondisclosure Agreement* -- NDAs). Comme celles-ci contiennent en général des dates d'expiration, quelques-uns des hackers les ayant signées ressentirent le besoin d'une introspection

personnelle. Ils raisonnèrent ainsi : tôt ou tard, le logiciel tombera dans le domaine public. En même temps, pensaient-ils, la promesse de garder le secret du logiciel durant les étapes cruciales de son développement n'était, après tout, qu'une affaire de compromis qui leur permettait de travailler sur les meilleurs projets. Mais pour Stallman, c'était poser le pied sur un terrain glissant.

« Quand quelqu'un m'invita à trahir tous mes collègues de cette façon, je me souvins quelle fut ma colère lorsqu'on nous en fit payer le prix, le labo et moi », dit Stallman. « Alors j'ai répondu : "Je vous remercie beaucoup pour cette superbe collection de logiciels, mais je ne peux l'accepter aux conditions demandées, je vais donc m'en passer". »

Comme Stallman l'apprendrait très vite, refuser de telles offres impliquait davantage qu'un sacrifice personnel. Cela l'amena à s'isoler de ses camarades hackers qui, bien que partageant un dégoût semblable pour le secret, essayaient de l'exprimer d'une manière moralement plus flexible. Il ne fallut pas longtemps avant que Stallman, de plus en plus marginalisé au sein du AI Lab, se targua d'être « le dernier des vrais hackers », tout en s'éloignant d'autant du marché dominé par le logiciel propriétaire. Refuser à autrui le code source, jugeait Stallman, cela revenait non seulement à trahir la mission scientifique nourrissant le développement du logiciel depuis la fin de la Seconde Guerre mondiale, mais cela revenait aussi à violer la Règle d'Or, la maxime morale selon laquelle « ne fais pas à autrui ce que tu ne voudrais pas qu'il te fasse. »

Telle fut toute l'importance que prirent l'épisode de l'imprimante laser et l'entrevue qui s'ensuivit. Sans cela, nous dit Stallman, sa vie aurait peut-être suivi un chemin plus ordinaire, alliant la richesse d'un programmeur commercial et la frustration suprême d'une vie passée à écrire des codes invisibles. Il n'y aurait alors pas eu d'intérêt ni d'urgence à s'attacher à un problème qui ne dérangeait personne. En outre, le plus important est qu'il n'y aurait pas eu non plus cette colère légitime qui animait Stallman, cette émotion qui, comme nous le verrons bientôt, a propulsé sa carrière aussi sûrement que l'aurait fait une idéologie politique ou une conviction morale.

À la fois en référence à la pratique d'achat des libertés personnelles par le biais d'« arrangements » -- c'est ainsi que Stallman décrivait la transaction de la clause de confidentialité -- et à la culture générale qui encourageait un tel marchandage moralement suspect, Stallman affirma : « À partir de ce jour, j'ai décidé que c'était quelque chose à quoi je ne pourrais jamais participer [...] J'ai décidé de ne jamais faire d'autres victimes comme moi. »

Notes

1. Pour plus d'informations sur le terme de *hacker*, voir l'annexe B.

Chapitre II — 2001 : l'odyssée d'un hacker

Le département d'informatique de l'Université de New York est situé à l'intérieur du *Warren Weaver Hall*, forteresse s'élevant deux blocs à l'est du *Washington Square Park*. Le souffle généré par l'air conditionné crée un fossé à l'entrée du bâtiment avec un air chaud et moite, tendant à décourager la présence des vagabonds et l'assaut des avocats. Les visiteurs qui osent s'aventurer au-delà sont confrontés à une autre barrière : la sécurité qui campe juste derrière l'unique entrée de l'immeuble.

Passé le poste de sécurité, l'atmosphère se détend quelque peu. Cependant, de nombreuses affichettes disséminées dans tout le rez-de-chaussée proclament les dangers des portes non sécurisées et des sorties de secours. Ensemble, ces signes rappellent qu'à New York, même dans les temps relativement tranquilles de l'avant 11 septembre, on n'est jamais assez prudent ou soupçonneux.

Ces affichettes offrent un contrepoint thématique intéressant au nombre croissant de visiteurs qui se réunissent dans l'atrium intérieur du hall. Certains ressemblent aux étudiants de l'université de New-York. La plupart ressemblent à des habitués de concerts, velus-chevelus s'agitant devant un music-hall dans l'attente du spectacle principal. Pour un court matin, les masses ont échoué sur *Warren Weaver Hall*, ne laissant aux préposés à la sécurité rien de mieux à faire que de regarder Ricky Lake à la télé et de faire un signe de l'épaule vers la salle voisine chaque fois que les visiteurs demandent « la conférence ».

Une fois à l'intérieur de l'auditorium, le visiteur trouve la personne qui a provoqué l'arrêt temporaire des procédures de sécurité de l'immeuble. Cette personne est Richard M. Stallman, fondateur du Projet GNU, lauréat du prix MacArthur en 1990, du prix Grace Murray Hopper de l'association d'ingénierie informatique (en 1990 aussi), co-bénéficiaire du prix de la Takeda Foundation's en 2001, et hacker au AI Lab par le passé. Comme annoncé dans une multitude de sites internet destinés aux hackers, y compris celui du Projet GNU, gnu.org, Stallman est à Manhattan, son ancien domicile, pour prononcer un discours récusant la récente campagne de la société Microsoft contre la Licence Publique Générale GNU (General Public Licence — GPL).

Le discours porte sur l'histoire et l'avenir du mouvement du logiciel libre. L'endroit où il est prononcé est significatif. Moins d'un mois auparavant, depuis la Stern School of Business de l'Université de New York, toute proche, le directeur général de Microsoft, Craig Mundie, prononçait un discours critique envers la GPL, le dispositif légal conçu par Stallman seize ans auparavant. Pensée pour contrer la vague croissante du secret de conception (*software secrecy*) qui submergeait l'industrie informatique — vague remarquée pour la première fois par Stallman en 1980 lors de ses difficultés avec l'imprimante laser Xerox — la GPL est devenue un outil central de la communauté du logiciel libre. Très simplement, grâce à la puissance légale du copyright, la GPL verrouille les logiciels en une forme de propriété commune, ce que les juristes d'aujourd'hui nomment désormais le bien commun numérique (la *digital commons*). Une fois verrouillés, ces programmes demeurent immuables. Leurs versions dérivées doivent conserver le même copyright — même celles qui ne comportent qu'une infime bribe du code source d'origine. Pour cette raison, plusieurs membres de l'industrie du logiciel qualifièrent alors la GPL de licence « virale », car elle se propage par elle-même dans tous les logiciels qu'elle touche [1].

Dans une économie de l'information de plus en plus dépendante des logiciels et toujours plus liée aux standards logiciels, la GPL était devenue la proverbiale « main de fer ». Même les sociétés les ayant d'abord moqué de logiciels socialistes s'accordaient alors pour en reconnaître les bénéfices. Linux, le noyau Unix-like développé par l'étudiant finnois Linus Torvalds en 1991, est licencié par la GPL, comme un grand nombre des instruments de programmation les plus populaires du monde : GNU Emacs, le Debugger GNU, le compilateur C GNU, etc. Ensemble, ces outils forment les composants d'un système d'exploitation libre développé, nourri et possédé par la communauté mondiale des hackers. Au lieu de voir cette communauté comme une menace, des compagnies high-tech comme IBM, Hewlett Packard, et Sun Microsystems commencèrent à s'appuyer sur eux, vendant des applications logicielles et des services adaptés à l'infrastructure toujours grandissante du logiciel libre.

Ainsi ces compagnies commencèrent-elles à compter une arme stratégique parmi la communauté des hackers perpétuellement en guerre contre Microsoft, la compagnie de Redmond basée à Washington qui,

pour le meilleur ou pour le pire, dominait le marché des logiciels PC depuis la fin des années 80. En tant que propriétaire du célèbre système d'exploitation Windows, Microsoft est en position de perdre le plus dans une industrie totalement bouleversée par la GPL. Quasiment chaque ligne du code source du colosse Windows est protégée par des copyrights réaffirmant la nature privée de celui-ci ou, du moins, réaffirmant à ce titre la faculté légale de Microsoft de le traiter ainsi. Du point de vue de Microsoft, incorporer au colosse Windows des programmes protégés par la virale GPL équivaudrait, en ce qui concerne les logiciels, à voir Superman avaler une bouteille de pilules de Kryptonite. Les compagnies rivales pourraient tout à coup copier, modifier et envoyer des versions améliorées de Windows, rendant instantanément vulnérable l'irréductible Position n°1 du fournisseur de logiciels destinés aux consommateurs. D'où l'intérêt de la compagnie pour le taux d'adoption de la GPL. D'où le récent discours de Mundie s'attaquant à la GPL ainsi qu'à sa conception du développement et de la vente de logiciels. D'où, enfin, la décision de Stallman de réfuter aujourd'hui publiquement, sur ce même campus, les arguments de ce discours.

Vingt ans est une longue période pour l'industrie logicielle. Considérez ceci : en 1980, lorsque Richard Stallman maudissait l'imprimante Xerox du AI Lab, Microsoft, que les hackers considèrent comme la compagnie la plus puissante de l'industrie mondiale du logiciel, était toujours une compagnie privée au stade embryonnaire. IBM, elle, vue comme la plus grande force dans l'industrie du matériel informatique, devait encore introduire son premier ordinateur personnel et ainsi lancer le bal de l'actuel marché du PC abordable. De nombreuses technologies que nous considérons comme acquises — la grande toile mondiale (World Wide Web), la télévision par satellite, les consoles de jeux vidéo 32 bits — n'existaient pas encore. La même chose peut être dite des entreprises au sommet des échelons de la corporation telles AOL, Sun Microsystems, Amazon.com, Compaq, et Dell. La liste est longue.

Le fait que le marché de la haute technologie soit allé aussi loin en si peu de temps alimente les deux côtés du débat entourant la GPL. Les défenseurs de la GPL soulignent la courte vie de la plupart des composants matériels des plateformes informatiques. Face au risque d'acheter un produit obsolète, les consommateurs tendent à aller massivement vers les compagnies aux meilleures chances de survie à long terme. Résultat, le marché du logiciel est devenu une arène où « le gagnant prend tout ».[2] L'environnement actuel du logiciel privé, selon les défenseurs de la GPL, mène au monopole, à l'abus et à la stagnation. Les compagnies dominantes accaparent tout l'oxygène du marché au dépens des compagnies rivales et des start-up innovantes.

Les opposants à la GPL argumentent le contraire. Vendre un logiciel est tout aussi risqué, sinon plus, que l'achat, disent-ils. Sans les garanties légales que procurent les licenses privées, sans oublier la perspective économique d'une propriété privée sur un logiciel révolutionnaire (*killer app*, c'est-à-dire une nouvelle technologie se lançant sur un tout nouveau marché[3]), les compagnies perdent toute motivation pour participer. Une nouvelle fois, le marché stagne et l'innovation décline. Comme l'a noté personnellement Mundie dans son discours du 3 mai sur ce même campus, la nature « virale » de la GPL « présente une menace » pour toute entreprise s'en tenant à l'unicité de son logiciel comme atout compétitif. Mundie renchérit :

> Cela ébranle fondamentalement le secteur indépendant du logiciel commercial en rendant impossible la distribution de logiciels sur une base où les personnes paient pour le produit plutôt que pour le seul coût de distribution.[4]

Le succès mutuel de GNU/Linux, l'assemblage du système d'exploitation construit autour du noyau Linux protégé par la GPL, et Windows au cours des dix dernières années révèlent la sagesse de ces deux perspectives. Quoiqu'il en soit, la bataille pour la réactivité est importante dans l'industrie du logiciel. Même les grosses entreprises, tel Microsoft, se fient au support par les développeurs de logiciels tiers, qui, par leurs outils, logiciels et jeux, rendent plus attrayante une plateforme comme Windows pour le consommateur moyen. Citant l'évolution rapide du marché technologique depuis les vingt dernières années, sans oublier l'admirable parcours de sa propre entreprise, Mundie avise ses auditeurs de ne pas se laisser emporter par l'élan récent du logiciel libre :

> L'expérience de deux décennies a démontré qu'un modèle économique protégeant la propriété intellectuelle associé à un modèle d'affaire récupérant les coûts de recherche et développement, peuvent créer d'impressionnants bénéfices économiques et les redistribuer avec largesse.[5]

De telles critiques servent de fond au discours de Stallman aujourd'hui. Moins d'un mois après ces déclarations, Stallman se tient adossé à l'un des tableaux noirs au bout de la salle, quelque peu nerveux avant de commencer.

Si les deux dernières décennies ont amené d'impressionnants changements dans le marché du logiciel, elles conduisirent à de plus spectaculaires transformations chez Stallman lui-même. Il n'est plus ce hacker mince, rasé de près, qui passait ses journées entières à communier avec son PDP-10 bien-aimé. En lieu et place se trouve un homme d'âge moyen, bien portant, aux cheveux longs et à la barbe digne d'un rabbin, un homme qui passe le plus clair de son temps à écrire et répondre à des courriels, haranguant ses confrères programmeurs, et donnant des discours comme celui d'aujourd'hui. Habillé d'un t-shirt couleur eau et d'un pantalon de polyester brun, Stallman a l'allure d'un ermite du désert sortant d'un vestiaire de l'Armée du Salut.

La salle est remplie de visiteurs partageant ses goûts vestimentaires. Beaucoup viennent avec leur portable et modem cellulaire : quoi de mieux pour enregistrer et transmettre les paroles de Stallman à un auditoire Internet dans l'expectative. Le ratio des genres est d'environ 15 hommes pour 1 femme, et l'une des 7 ou 8 présentes tient un pingouin en peluche, la mascotte officielle de Linux, alors qu'une autre porte un ours en peluche.

[ILLUSTATION: photo de RMS] *Richard Stallman, en 2000. « J'ai décidé de développer un système d'exploitation gratuit ou de mourir en essayant... [mourir] de vieillesse bien entendu. »*

Agité, Stallman quitte son poste en bout de salle, et prend place sur une chaise du premier rang, tapant quelques commandes sur un portable déjà ouvert. Les dix minutes suivantes, Stallman reste inconscient du nombre croissant d'étudiants, professeurs et admirateurs qui passent devant lui au pied de la scène de l'auditorium.

Avant de commencer le discours, le rituel baroque des formalités académiques doit être observé. La présence de Stallman ne mérite pas une, mais deux introductions. Mike Uretsky, co-directeur de la *Stern School's Center for Advanced Technology* présente la première.

« Le rôle d'une université est de favoriser le débat et d'avoir des discussions intéressantes », dit Uretsky. « Cette présentation particulière, ce séminaire, suivent ce modèle. Je trouve singulièrement intéressante la discussion de l'*open source*. »

Avant qu'Uretsky ne puisse prononcer un autre mot, Stallman est debout agitant la main tel un automobiliste égaré.

« Je fais du logiciel libre », dit Stallman sous les rires croissants. « L'*open source* est un tout autre mouvement. »

Les rires cèdent la place aux applaudissements. La salle est pleine de partisans de Stallman, des gens connaissant sa réputation d'exactitude verbale, mais aussi son conflit très médiatisé en 1998 avec les défenseurs de l'*open source*. Beaucoup sont venus pour anticiper de tels éclats tout comme les amateurs de radio s'attendaient au classique « Mais arrête ça !» des émissions radio de Jack Benny.

Uretsky termine promptement son introduction et cède la scène à Edmond Schonberg, professeur au département des sciences informatiques de l'Université de New York. Programmeur et contributeur au Projet GNU, Schonberg sait quels pièges linguistiques éviter. Il résume adroitement la carrière de Stallman, celle d'un programmeur des temps modernes.

« Richard est le parfait exemple de quelqu'un qui, en agissant localement, a commencé à penser globalement les problèmes de pénurie de code source », dit Schonberg. « Il a développé une philosophie cohérente qui nous contraint tous à réexaminer nos idées sur la manière de produire un programme, sur la signification de la propriété intellectuelle, et sur ce que représente en réalité la communauté du logiciel. »

Schonberg invite Stallman sous les applaudissements redoublés. Ce dernier prend un moment pour éteindre son portable, se lève et monte sur scène.

Au départ, l'allocution de Stallman est plus proche d'un numéro comique de Catskills que d'un discours politique. « J'aimerais remercier Microsoft pour m'avoir donné l'opportunité d'être présent sur cette estrade », ironise Stallman. « Depuis les dernières semaines, je me suis senti comme un auteur dont les livres

ont été fortuitement interdits quelque part. »

Pour le néophyte, Stallman se lance, en échauffement, dans une rapide analogie du logiciel libre. Il compare un logiciel à une recette de cuisine. Les deux donnent d'utiles instructions, pas à pas, pour terminer une tâche souhaitée, et peuvent être aisément modifiés en fonction des désirs spécifiques de l'usager ou pour des circonstances particulières. « Vous n'avez pas à suivre une recette avec précision », note Stallman. « Vous pouvez laisser de côté certains ingrédients. Ajouter quelques champignons parce que vous aimez les champignons. Mettre moins de sel car votre docteur vous conseille d'en consommer moins -- peu importe. »

De surcroît, dit Stallman, logiciels et recettes sont faciles à partager. En donnant une recette à un invité, un cuisinier n'y perd que du temps et le coût du papier sur lequel est inscrite la recette. Les logiciels nécessitent encore moins, habituellement quelques clics de souris et un minimum d'électricité. Dans les deux cas, par contre, la personne donnant cette information y gagne deux choses : davantage d'amitié et la possibilité de partager des recettes intéressantes en retour.

« Imaginez si les recettes étaient emballées dans des boîtes noires », dit Stallman, renchérissant. « Vous ne pourriez connaître les ingrédients utilisés, encore moins les changer, et imaginez si vous faisiez une copie à un ami. Ils vous qualifieraient de pirate et essaieraient de vous faire emprisonner des années durant. Ce monde créerait un énorme scandale chez les gens ayant l'habitude de partager des recettes. Mais c'est exactement ce qu'est le monde du logiciel propriétaire. Un monde dans lequel la bienséance commune envers les autres est prohibée ou empêchée. »

Avec cette analogie introductrice peu commune, Stallman se lance une nouvelle fois dans le récit de l'épisode de l'imprimante laser Xerox. Comme l'analogie culinaire, l'histoire de l'imprimante est un outil de rhétorique fort utile. Avec sa structure de parabole, elle illustre comment les choses peuvent changer rapidement dans le monde du logiciel. Ramener l'auditoire à une ère antérieure à Amazon.com-achetez-en-un-clic, à Microsoft Windows et aux bases de données Oracle, cela demande au public d'examiner la notion de propriété logicielle sans ses logos corporatifs actuels.

Stallman livre son histoire avec tout le vernis et l'expérience d'un procureur menant son plaidoyer final. Arrivé au moment où le professeur de Carnegie Mellon lui refuse une copie du code source de l'imprimante, Stallman fait une pause.

« Il nous a trahis », dit-il. « Mais il ne l'a pas fait qu'à nous. Il y a des chances qu'il *vous* l'ait fait aussi. »

Sur le mot « vous », Stallman pointe son index accusateur vers un membre insouciant de l'auditoire . La cible sourcille à peine que les yeux de Stallman sont déjà ailleurs. Lentement et délibérément, Stallman montre un second auditeur gloussant nerveusement dans la foule. « Et je crois encore plus probable qu'il l'ait fait à vous aussi », dit-il désignant un spectateur trois rangées derrière le premier.

Au moment où Stallman a le troisième auditeur sous son index, la fébrilité cède la place au rire général. Le geste semble un peu mis en scène, et c'est le cas. Puis, quand vient le temps de terminer l'histoire de l'imprimante, Stallman le fait avec le brio d'un homme de scène. « Il l'a probablement fait à tous ici présents dans cette salle à l'exception, peut-être, de ceux qui n'étaient pas encore nés en 1980 », dit-il, provoquant de nouveaux rires. « [C'est] parce qu'il a promis de refuser de coopérer avec pratiquement toute la population de la planète Terre. »

Stallman laisse ce commentaire faire son effet un laps de temps. « Il avait signé une clause de confidentialité », ajoute-t-il.

Au cours des vingt dernières années, du chercheur académique déçu au leader politique, l'ascension de Richard Matthew Stallman raconte beaucoup de choses. Elle parle de sa nature bornée et de sa volonté prodigieuse. Elle parle de la conception éclairée et des valeurs du mouvement du logiciel libre qu'il aida à construire. Elle parle des logiciels de haute qualité qu'il a créés, des programmes informatiques qui ont cimenté sa réputation de programmeur légendaire. Elle parle de l'impulsion de la GPL, une innovation légale que beaucoup d'observateurs de Stallman voient comme son plus grand accomplissement. Enfin, plus important encore, elle parle de la nature du changement de pouvoir politique dans un monde de plus en plus soumis à la technologie informatique et aux logiciels qui en sont le moteur.

Peut-être est-ce pour cela que, même à une époque où la plupart des célébrités de la haute technologie sont en déclin, l'étoile de Stallman brille davantage. Depuis le lancement du Projet GNU en 1984[6], Stallman fut tour à tour ignoré, satirisé, vilipendé et attaqué — autant à l'extérieur qu'à l'intérieur du mouvement du logiciel libre. Passant outre, le Projet GNU a réussi à atteindre ses objectifs, malgré des retards notoires, et ainsi demeurer pertinent dans un marché hautement plus complexe que celui existant à son arrivée, il y a dix-huit ans. On peut dire la même chose pour l'idéologie du logiciel libre, une idéologie méticuleusement soignée par Stallman lui-même.

Pour comprendre les raisons de cette adéquation, il est utile d'examiner Richard Stallman au travers de ses propos et des paroles de ceux qui collaborèrent ou luttèrent avec lui tout au long de ce parcours. Le personnage de Richard Stallman n'est pas compliqué. Si quelqu'un personnifie l'adage « ce que vous voyez est ce que vous obtenez (*what you see is what you get*) », c'est bien Stallman.

« Je crois que si vous voulez comprendre l'être humain Richard Stallman, vous devez voir toutes ses facettes comme un ensemble cohérent », conseille Eben Moglen, juriste à la *Free Software Foundation* (Fondation pour le Logiciel Libre — FSF) et professeur de droit à la *Columbia University Law School*. « Toutes ces excentricités personnelles que beaucoup voient comme un obstacle à la compréhension de Stallman *sont* réellement Stallman : la forte sensation de frustration personnelle, son énorme sens de l'engagement éthique et son incapacité de compromission face aux problèmes jugés fondamentaux. Tout cela explique ses actes ». Il n'est pas évident d'expliquer comment une journée débutant avec une imprimante laser puisse finalement conduire à une confrontation verbale avec l'entreprise la plus riche du monde. Cela requiert une estimation approfondie des forces qui ont rendu la propriété des logiciels si importante dans la société d'aujourd'hui. Cela requiert également l'examen sérieux d'un homme qui, comme bien des leaders politiques avant lui, comprend la malléabilité de la mémoire humaine. Cela requiert encore une certaine habileté à interpréter les mythes et les mots politiquement chargés qui se sont accumulés avec le temps autour de Stallman. Enfin, il faut comprendre son génie en tant que programmeur, ainsi que ses échecs et ses succès à transposer ce génie vers d'autres quêtes.

Lorsqu'il en vient à résumer personnellement ce cheminement, Stallman reconnaît la fusion, observée par Moglen, entre personnalité et principes. « La ténacité est mon point fort », dit-il. « La plupart des gens qui essaient de faire quelque chose de très difficile se découragent et abandonnent. Je n'ai jamais abandonné. »

Il croit aussi à sa bonne étoile. Sans cette mésaventure avec l'imprimante laser, sans les conflits personnels et politiques compromettant sa carrière au MIT, sans une demi-douzaine d'autres facteurs opportuns, Stallman imagine très facilement sa vie prenant un chemin différent. Cela dit, Stallman remercie les forces et les circonstances qui lui ont permis de changer.

« Je n'avais que les bonnes compétences », dit-il, résumant à son auditoire sa décision de lancer le Projet GNU. « Je sentais qu'il n'y avait personne d'autre que moi, alors j'ai songé : 'je suis élu. Je dois travailler sur ce projet. Qui sinon moi ?'. »

Notes

1. Dans les faits, la GPL n'a pas tout à fait ce pouvoir. Selon la section 10 de la licence publique générale GNU, Version 2 (1991), la nature virale de la licence dépend fortement des volontés de la *Free Software Foundation* (FSF) de présenter un programme comme un travail dérivé, sans parler de la licence existante qui serait remplacée par la GPL.
   « Si vous désirez incorporer des éléments du programme dans d'autres programmes libres dont les conditions de distribution diffèrent, vous devez écrire à l'auteur pour lui en demander la permission. Pour ce qui est des programmes directement déposés par la FSF, écrivez-nous : une exception est toujours envisageable. Notre décision sera basée sur notre volonté de préserver la liberté de notre programme ou de ses dérivés et celle de promouvoir le partage et la réutilisation du logiciel en général. » (GPL v.2 section 10)
   « Comparer quelque chose à un virus est vraiment sévère », dit Stallman. « Une plante grimpante serait une comparaison plus exacte car elle s'en va en un autre lieu si vous vous emparez d'un outil tranchant. »

Pour plus d'informations sur la licence publique générale GNU, consultez http://www.gnu.org/copyleft/gpl.html

2. Shubha Ghosh, *Revealing the Microsoft Windows Source Code* [Révéler le code source de Microsoft Windows], Gigalaw.com (janvier 2000) http://www.gigalaw.com/articles/2000-all/ghosh-2000-01-all.html
3. Un logiciel révolutionnaire (*killer app*) n'a pas besoin d'être un logiciel propriétaire. Voyez, évidemment, le légendaire fureteur Mosaic, un logiciel dont les droits d'auteurs permettent des dérivés non commerciaux sous certaines restrictions. De plus, je crois que le lecteur comprend : le marché du logiciel est comme une loterie. Plus le potentiel en gains est grand, plus les gens veulent participer. Pour un bon résumé sur le phénomène du logiciel révolutionnaire, voyez *Whatever Happened to the 'Killer App'?*, e-Commerce News (7 décembre 2000). http://www.ecommercetimes.com/perl/story/5893.html
4. Craig Mundie (vice-président senior, Microsoft Corp.), *The Commercial Software Model*, Extrait d'une transcription en ligne du discours du 3 mai 2001 au New York University Stern School of Business. http://www.microsoft.com/presspass/exec/craig/05-03sharedsource.asp
5. idem.
6. L'acronyme GNU signifie « GNU's not Unix » (GNU n'est pas Unix). Dans une autre partie du discours de Stallman du 29 mai 2001 à la NYU, Stallman résume l'origine de cet acronyme :

   > Nous, hackers, cherchons toujours un nom trivial ou coquin pour un logiciel, parce que le nommer représente la moitié du plaisir que procure la programmation. Nous avons aussi cette tradition où les acronymes sont récursifs, pour signifier que tel programme que vous créez est similaire à un autre déjà existant... Je cherchais un acronyme récursif pour « quelque chose » n'est pas Unix. J'ai essayé les vingt-six lettres pour découvrir qu'aucune [combinaison] ne composait un mot. J'étais décidé à en faire une forme contractée. De cette manière, je pourrais avoir un acronyme de trois lettres pour « Quelque chose pas Unix ». J'ai essayé des lettres, et je suis tombé sur « GNU ». C'était enfin ça.
   >
   > Bien qu'amateur de jeux de mots, Stallman recommande la prononciation du « g » au début de l'acronyme (c'est-à-dire « gue-niou »). Non seulement cela aide-t-il à éviter la confusion d'avec le mot « gnu » (gnou), cet antilope africain, *connochaetes gnou*, et cela évite aussi la confusion avec l'adjectif « new » (nouveau). « Nous y travaillons depuis dix-sept ans maintenant, alors ce n'est plus réellement nouveau », dit Stallman.:

   Source : notes de l'auteur et transcriptions en ligne de *Free Software: Freedom and Cooperation*, discours de Richard Stallman du 29 mai 2001 à l'Université de New York. http://www.gnu.org/events/rms-nyu-2001-transcript.txt

Chapitre III — Portrait d'un jeune hacker

Alice Lippman, mère de Richard Stallman, se rappelle encore le moment où elle comprit que son fils avait un don particulier.

« Je pense que c'était quand il avait huit ans », se souvient-elle.

C'était en 1961 et Mme Lippman, récemment divorcée, passait un après-midi de week-end dans le petit studio familial de l'*Upper West Side de Manhattan*. Feuilletant le *Scientific American*, elle arriva à sa chronique préférée : les *Jeux Mathématiques* de Martin Gardner. Alice Lippman, alors professeur intérimaire d'arts plastiques, aimait les rubriques de Gardner pour les énigmes proposées. Son fils déjà plongé dans un livre sur un canapé voisin, elle décida d'essayer de résoudre le problème de la semaine.

« Je n'étais pas la meilleure pour résoudre les casse-tête », admet-elle, « mais en tant qu'artiste, je pense qu'ils m'aidaient réellement à dépasser les obstacles conceptuels. »

Mme Lippman se trouva vite face à un mur en tentant de résoudre le problème. Prête à jeter le magazine de dépit, elle fut surprise de sentir qu'on lui tirait doucement la manche.

« C'était Richard, se souvient-elle, il voulait savoir si j'avais besoin d'aide. »

Regardant tour à tour le problème et son fils, Mme Lippman assure qu'initialement elle considéra l'offre avec scepticisme. « J'ai demandé à Richard s'il avait lu le magazine », dit-elle. « Il me répondit que oui, il l'avait fait, et qu'en plus il avait déjà résolu le problème. Je me souviens qu'ensuite il m'expliqua comment résoudre l'énigme. »

En écoutant l'approche logique de son fils, son scepticisme se changea rapidement en incrédulité. « J'ai toujours su qu'il était un garçon brillant, dit-elle, mais c'était la première fois que je voyais quelque chose suggérant combien il était avancé ! »

Trente ans après, elle rit de ce souvenir. « Pour vous dire la vérité, je ne pense pas que j'aurais réussi à comprendre la résolution de ce problème. Tout ce dont je me souviens, c'est d'être surprise qu'il connaisse la réponse. »

Assise à la table de la salle à manger de son second appartement à Manhattan — un logement spacieux de trois chambres à coucher où elle emménagea avec son fils après son mariage en 1967 avec Maurice Lippman, maintenant décédé — Alice Lippman affiche un mélange de fierté et de perplexité de mère juive au souvenir des jeunes années de son fils. À côté, sur le buffet, se trouve une photo de 25x20 cm d'un Stallman sombre, barbu et vêtu de sa toge doctorale. L'image éclipse les photos des neveux et nièces de Mme Lippman, mais avant que le visiteur n'en déduise quelque chose, Mme Lippman compense son importance par une vanne ironique : « Richard a insisté pour que je l'aie après avoir reçu son doctorat *honoris causa* de l'Université de Glasgow. Il m'a dit : 'Tu sais quoi, maman, c'est la première remise de diplôme à laquelle j'ai assisté !'. »

De tels commentaires reflètent le sens de l'humour développé en élevant un enfant prodige. Mais détrompez-vous ! Pour chaque histoire que Mme Lippman entend ou lit sur l'obstination ou le comportement inhabituel de son fils, elle peut en raconter au moins une douzaine d'autres.

« Il était tellement conservateur », dit-elle en levant les bras au ciel en feignant l'exaspération. « Nous avions les pires disputes ici à cette table. J'appartenais au premier groupe de professeurs des écoles publiques de la ville qui fit grève pour créer un syndicat, et Richard était très fâché contre moi. Il voyait les syndicats corrompus. Il était aussi très opposé à la Sécurité Sociale. Il pensait que les gens pourraient gagner plus d'argent en investissant eux-mêmes. Qui pouvait savoir qu'en dix ans il deviendrait si idéaliste ? Je me souviens de sa demi-soeur me disant : 'Que deviendra-t-il en grandissant ? Un fasciste ?'. »

Mère célibataire pendant près d'une décennie — elle et le père de Richard, Daniel Stallman, se sont mariés en 1948, ont divorcé en 1958 et ensuite se sont partagés la garde de leur fils – Alice peut témoigner de l'aversion de son fils pour l'autorité. Elle peut aussi attester sa soif de connaissances. C'est au moment où ces deux passions s'entremêlaient que son fils et elle ont eu leurs plus grosses disputes.

« C'était comme s'il ne voulait jamais manger », dit-elle, se rappelant le comportement de son fils entre l'âge de huit ans et la fin du lycée en 1970. « Je l'appelais pour dîner et il ne m'entendait jamais. Je devais crier neuf ou dix fois pour attirer son attention. Il était complètement absorbé. »

Stallman, de son côté, se souvient des choses de la même manière mais avec une connotation politique supplémentaire.

« J'adorais lire », dit-il. « Si je voulais lire et que ma mère me disait d'aller manger à la cuisine ou d'aller dormir, je ne l'écoutais pas. Je ne voyais pas de raison à ne pas pouvoir lire, ni de raison pour laquelle elle pouvait me dire ce que je devais faire, point. Fondamentalement, ce que j'avais lu à propos des idées de démocratie et de liberté individuelle, je me l'appliquais. Je ne voyais aucune raison d'exclure les enfants de ces principes. »

Cette croyance en la prééminence de la liberté individuelle sur l'autorité arbitraire se manifestait aussi à l'école. À onze ans, deux ans en avance sur ses camarades de classe, Richard Stallman subissait toutes les frustrations habituelles d'un écolier doué. C'est peu après la péripétie de l'énigme mathématique que sa mère participa à la première d'une longue série de rencontres entre parents et professeurs.

« Il refusait absolument de faire les travaux écrits », dit-elle, se rappelant une ancienne controverse. « Je pense que le dernier travail qu'il ait écrit, avant sa dernière année de lycée, était une dissertation sur l'histoire du système numérique dans le monde occidental pour un professeur de quatrième année. »

Doué pour tout ce qui exigeait un raisonnement analytique, Richard Stallman était attiré par les mathématiques et les sciences au détriment des autres matières. Ce que certains professeurs voyaient comme de l'obstination, sa mère le considérait comme de l'intolérance. Les maths et les sciences offraient trop de possibilités d'apprendre, surtout quand on les comparait aux sujets et travaux pour lesquels son fils avait moins de dispositions. Quand, vers l'âge de dix ou onze ans, les garçons de sa classe ont commencé à jouer au football, elle se souvient que son fils est rentré en rage à la maison. « Il voulait vraiment jouer, mais il n'avait pas le talent requis et ça le rendait furieux. »

La colère a finalement amené son fils à se concentrer encore plus sur les maths et les sciences. Pourtant, même dans le domaine scientifique, son intolérance pouvait causer des problèmes. Plongé dans des livres de calcul depuis l'âge de sept ans, il ne voyait pas la nécessité d'expliquer son raisonnement aux adultes. Un jour, sa mère engagea un étudiant de l'Université de Columbia toute proche pour jouer au grand frère avec son fils. L'étudiant quitta l'appartement après la première séance et ne revint plus. « Je pense que ce dont parlait Richard lui passait par dessus la tête », suppose sa mère.

Une autre histoire favorite de la mère de Richard Stallman remonte au début des années soixante, peu après l'épisode du jeu mathématique. Vers l'âge de sept ans, deux ans après le divorce et le déménagement, Richard entreprit de lancer des modèles réduits de fusée dans le parc de Riverside Drive. Ce qui commença comme un amusement sans but prit bientôt un tour plus sérieux lorsqu'il nota les données de chaque lancement. Comme les jeux mathématiques, ce passe-temps n'attira guère l'attention jusqu'au jour où, juste avant un lancement important de la NASA, Mme Lippman demanda à son fils s'il voulait regarder.

« Il enrageait », dit-elle. « Tout ce qu'il put dire fut : 'Mais, je n'ai encore rien publié'. Apparemment, il avait quelque chose qu'il voulait réellement montrer à la NASA. »

De telles anecdotes sont les premiers témoignages du bouillonnement intellectuel qui deviendrait la principale marque de Stallman dans la vie. Quand les autres enfants venaient à table, Stallman restait dans sa chambre à lire. Quand les autres enfants jouaient à Johnny Unitas, Stallman jouait à Werner von Braun. « J'étais bizarre », dit Stallman résumant brièvement sa jeunesse lors d'une entrevue en 1999. « Après un certain âge, les seuls amis que j'avais étaient des enseignants.[1] »

Quoique cela signifiât un risque de nouvelles prises de bec à l'école, Mme Lippman décida de satisfaire la passion de son fils. À douze ans, Richard fréquentait des camps scientifiques l'été, et des écoles privées durant l'année scolaire. Lorsqu'un enseignant lui recommanda l'inscription de son fils au Columbia Science Honors Program, un programme post-Spoutnik créé pour les étudiants et collégiens doués de la ville de New-York, Stallman étendit ses activités parascolaires et fit régulièrement la navette les samedis jusqu'au campus de la Columbia University.

Dan Chess, un ancien camarade de classe de ce programme, se rappelle que Stallman semblait un peu étrange, même parmi ces étudiants qui partageaient pourtant une fascination similaire pour les sciences et les mathématiques. « Nous étions tous des *nerds* et des *geeks*, mais il était particulièrement mal adapté », se souvient Chess, maintenant professeur de mathématiques au Hunter College. « Il était terriblement intelligent. J'ai connu beaucoup de personnes intelligentes, mais je pense qu'il est la plus intelligente que j'ai rencontrée. »

Seth Breidbart, qui, lui aussi, suivait ce programme, offre un témoignage corroborant. Programmeur informatique toujours en contact avec Stallman grâce à une passion commune pour la science-fiction et les conventions afférentes, il se souvient du Stallman de quinze ans, la boule à zéro, comme « effrayant », surtout pour un compagnon de 15 ans.

« C'est difficile à décrire », dit Breidbart. « Non pas qu'il était inapprochable. Il était tout simplement très sensible. [Il était] très instruit mais aussi très têtu d'une certaine manière. »

De telles descriptions font naître des spéculations : des adjectifs comme « sensible » et « têtu » sont-ils simplement une manière de décrire des traits de personnalité qui, aujourd'hui, pourraient être vus comme des troubles du comportement juvéniles ? Un article de la revue Wired de décembre 2001, intitulé « The Geek Syndrome », dépeint le portait de plusieurs enfants doués pour les sciences et atteints d'autisme de haut niveau, ou syndrome d'Asperger. Sous de nombreux aspects, les souvenirs des parents cités dans cet article présentent une similitude étrange avec ceux de Mme Lippman. Même Stallman s'est livré au révisionisme psychiatrique de temps en temps. Lors d'une entrevue au *Toronto Star* en 2000, Stallman se décrivait à son interlocuteur « à la limite de l'autisme » [2], une description qui explique largement cette tendance à l'isolement social et émotionnel au cours de sa vie, et les efforts perpétuels pour la surmonter.

De telles spéculations sont favorisées, bien sûr, par la nature vague et changeante de la plupart des « troubles du comportement », comme on les nomme aujourd'hui. Comme l'observe Steve Silberman, auteur de l'article « The Geek Syndrome », les psychiatres américains n'ont que récemment accepté le syndrome d'Asperger comme terme générique englobant nombre de ces traits de comportement. Ces caractéristiques vont du manque d'adresse motrice et d'une socialisation limitée à une grande intelligence et une affinité presque obsessionnelle envers les chiffres, les ordinateurs et autres systèmes ordonnés [3]. Réfléchissant à la nature très large de cette définition, Stallman raconte qu'il est possible que, né quarante ans plus tard, il aurait probablement mérité un tel diagnostic. De nouveau, comme le seraient beaucoup de ses collègues du monde informatique.

« Il est possible que j'aie pu avoir quelque chose de semblable », dit-il. « D'un autre côté, un des aspects de ce syndrome est la difficulté à suivre le rythme. Je peux danser. En fait, j'aime suivre les rythmes les plus compliqués. Ce n'est pas assez précis pour vraiment savoir. »

Chess, pour sa part, rejette de telles tentatives de diagnostic rétrospectif. « Je ne l'ai jamais pensé atteint d'une telle chose », dit-il. « Il était juste très asocial, mais là, nous l'étions tous. »

Mme Lippman, pour sa part, envisage cette possibilité. Elle se rappelle de quelques histoires d'enfance de son fils qui donnent matière à spéculation. Un symptôme important de l'autisme est l'hypersensibilité aux bruits et aux couleurs. Mme Lippman se souvient de deux anecdotes marquantes à ce sujet. « Lorsque Richard était enfant, nous l'emmenions à la plage », dit-elle. « Il commençait à hurler deux ou trois pâtés de maison avant d'atteindre le rivage. Ce n'est que la troisième fois que nous comprîmes ce qui se passait : le son des vagues lui faisait mal aux oreilles ». Elle se souvient d'une réaction bruyante similaire et relative à la couleur: « Ma mère avait des cheveux rouges brillants, et chaque fois qu'elle se penchait pour le prendre, il lâchait un hurlement. »

Mme Lippman raconte qu'elle s'est mise à lire à propos d'autisme ces dernières années et croit que ces épisodes sont plus qu'une coïncidence. « Je sens bien que Richard possède certaines qualités d'un enfant autiste » dit-elle. « je regrette que l'on en connût si peu sur l'autisme à cette époque. »

Avec le temps, cependant, Mme Lippman confie que son fils apprit à s'adapter. À l'âge de sept ans, dit-elle, il adorait se tenir à la fenêtre de devant du métro, traçant et mémorisant le labyrinthe des pistes de chemins de fer sous la ville. C'était un passe-temps qui nécessitait la capacité à s'accoutumer aux bruits intenses qui accompagnaient chaque trajet en train. « Seul le bruit initial semblait l'incommoder », raconte

Mme Lippman. « C'était comme s'il était choqué par le son, mais ses nerfs apprirent à s'adapter. »

Plus généralement, Mme Lippman se rappelle que son fils présentait les signes d'excitation, d'énergie et de socialisation propre à tout garçon normal. Ce n'est qu'après une série d'événements bouleversant le foyer des Stallman, dit-elle, que son fils devint introverti et distant.

Le premier événement traumatisant fut le divorce d'Alice et Daniel Stallman, le père de Richard. Mme Lippman raconte que, bien qu'elle et son ex-mari aient tenté de préparer leur fils au choc, ce dernier fut dévastateur malgré tout. « Il semblait ne pas y prêter attention lorsque nous lui avons expliqué ce qui se passait », se souvient-elle. « Mais la réalité le rattrapa quand lui et moi avons emménagé dans un nouvel appartement. La première chose qu'il dit fut : 'Où sont les meubles de papa?' »

La décennie suivante, Stallman vivait en semaine chez sa mère à Manhattan, et le week-end au domicile de son père dans le quartier du Queens. Les aller-retours lui permirent d'observer deux styles différents d'éducation parentale qui, jusqu'à présent, le laissent fermement hostile à l'idée d'élever des enfants lui-même. Parlant de son père, vétéran de la Deuxième Guerre mondiale décédé début 2001, Stallman oscille entre respect et colère. D'un côté, il y a l'homme dont l'intégrité morale le poussa à apprendre le français afin d'être plus utile aux alliés lorsqu'ils arriveraient enfin. De l'autre côté, il y avait le parent ayant toujours su dénigrer habilement pour obtenir un effet cruel [4].

« Mon père avait un horrible tempérament », dit-il. « Il ne hurlait jamais, mais trouvait toujours une manière de critiquer froidement pour vous démolir. »

Concernant la vie chez sa mère, Stallman est moins équivoque. « C'était la guerre », décline-t-il. « Dans ma misère, je disais : 'Je veux aller à la maison;' exprimant un endroit inexistant que je n'aurais jamais. »

Les premières années suivant le divorce, Stallman trouvait calme et échappatoire chez ses grands-parents paternels. Mais aux alentours de dix ans, ses grands-parents décédèrent l'un après l'autre en peu de temps. La perte fut dévastatrice pour lui. « Je leur rendais visite et me sentais entouré d'amour et de gentillesse », se souvient-il. « C'était le seul endroit où je retrouvais cela, jusqu'à mon départ pour le collège. »

Mme Lippman considère le décès des grands-parents de Richard comme le deuxième événement traumatisant. « Il en était réellement bouleversé », dit-elle. Il était très proche de ses deux grands-parents. Avant qu'ils ne meurent, il était très extraverti, pratiquement leader de bande avec les autres enfants. Après leur mort, il devint beaucoup plus renfermé.

Du point de vue de Stallman, ce repli sur soi était une tentative de faire face à l'agonie de l'adolescence. Qualifiant ses années d'adolescent comme « une pure horreur », Stallman raconte qu'il se sentait tel un sourd parmi une foule jacassante d'amateurs de musique.

« J'avais souvent le sentiment de ne pas comprendre ce que les autres disaient », continue Stallman, en souvenir de la bulle émotionnelle l'isolant du reste du monde des adolescents et des adultes. « Je pouvais comprendre les mots, mais quelque-chose au-delà de la conversation se produisait que je ne comprenais pas. Je ne comprenais pas pourquoi les gens s'intéressaient à ce que d'autres racontaient. »

Malgré toute la souffrance engendrée, l'adolescence aurait un effet stimulant sur le sens de l'individualité de Stallman. À une période où ses collègues de classe laissaient leurs cheveux s'allonger, Stallman préférait les siens courts. À une époque où le monde des adolescents écoutait du rock and roll, Stallman préférait la musique classique. Fervent amateur de science-fiction, de la revue Mad et des émissions télé de fin de soirée, Stallman cultivait une personnalité vraiment hors norme qui suscitait l'incompréhension des parents et de ses pairs.

« Ah! les jeux de mots! », s'exclame Mme Lippman, exaspérée au souvenir de la personnalité adolescente de son fils. « Il n'y avait rien qu'on ne puisse dire à table sans qu'il ne vous le renvoie en calembour. »

Hors du domicile, Stallman réservait cet humour aux adultes tendant à l'indulgence envers ses dons. L'un des premiers était un moniteur de camp d'été qui lui remit un manuel imprimé de l'ordinateur IBM 7094 durant sa douzième année. Pour un pré-adolescent fasciné par les sciences et les chiffres, c'était un don du

ciel[5]. À la fin de cet été-là, Stallman écrivait des programmes sur papier selon les spécifications internes du 7094, anticipant fébrilement l'opportunité de les essayer sur une véritable machine.

À une décennie du premier ordinateur personnel sur le marché, Stallman fut obligé d'attendre quelques années avant d'avoir accès à son premier ordinateur. Cette chance se présenta enfin durant son année de première au lycée. Embauché à l'*IBM New York Scientific Center*, un centre de recherche maintenant fermé au centre-ville de Manhattan, Stallman passa l'été de son diplôme à composer son premier programme informatique, un pré-processeur pour le 7094 dans le langage de programmation PL/1. « Je l'ai tout d'abord écrit en PL/1, puis recommencé en langage assembleur lorsque le programme est devenu trop important pour tenir dans l'ordinateur », se souvient-il.

Après ce travail au centre IBM, Stallman obtint un poste d'assistant au département de biologie de l'université Rockefeller. Bien qu'il s'acheminât vers une carrière dans le domaine des mathématiques ou de la physique, l'esprit d'analyse de Stallman impressionna tant le directeur du labo que, quelques années après que Stallman eut quitté l'université, Mme Lippman reçut un appel inattendu. « C'était le professeur de Rockefeller », dit-elle. « Il voulait savoir comment Richard allait. Il fut surpris d'apprendre qu'il travaillait dans l'informatique. Il avait toujours pensé que Richard avait un grand avenir devant lui en tant que biologiste. »

Les compétences en analyse de Stallman impressionnèrent également les membres de la faculté de Columbia, même lorsqu'il devint la cible de leur colère. « Typiquement, deux ou trois fois par heure, Stallman relevait une erreur dans le cours », raconte Breidbart. « Et il ne se gênait pas de le faire savoir au professeur. Cela lui valut beaucoup de respect, mais peu de popularité. »

Entendre à nouveau l'anecdote de Breidbart provoque un sourire désabusé chez Stallman. « J'ai dû être un peu crétin parfois », admet-il. « Mais j'ai trouvé des âmes soeurs chez les professeurs car eux aussi aiment apprendre. Ce n'est pas la cas de la plupart des jeunes, du moins pas de la même façon. »

Traîner avec de jeunes doués le samedi a tout de même encouragé Stallman à réfléchir aux mérites d'une plus grande socialisation. L'entrée à l'université approchant à grand pas, Stallman, comme beaucoup au *Columbia Science Honors Program*, réduisit sa liste d'universités souhaitées à deux options : Harvard et MIT. Entendre que son fils désirait entrer dans une université prestigieuse de l'*Ivy League* inquiéta Mme Lippman. En tant que lycéen de quinze ans, Stallman avait toujours des difficultés avec les professeurs et les administrateurs. Rien que l'année précédente, il avait obtenu la note « A » dans ses cours d'histoire américaine, chimie, français et algèbre, mais un notable « F » en anglais, reflet de son boycott continu des travaux écrits. Une telle mésaventure pouvait susciter un sourire entendu au MIT, mais à Harvard, c'était un drapeau rouge.

Durant l'avant-dernière année de lycée de son fils, Mme Lippman prit rendez-vous avec un thérapeute. Ce dernier exprima immédiatement son inquiétude devant le refus de Stallman de rédiger ses travaux et au sujet des démêlés avec ses professeurs. Son fils avait certainement les aptitudes intellectuelles pour réussir à Harvard, mais avait-il la patience de suivre des cours universitaires exigeant la remise régulière de mémoires ? Le thérapeute suggéra un essai. Si Stallman réussissait une année entière à l'école publique de New York, incluant un cours d'anglais avec épreuve écrite obligatoire, il pourrait probablement le faire à Harvard. L'année achevée, Stallman s'inscrivit promptement l'été à l'école Louis D. Brandeis High School, une école située sur la 84ème rue, et se mit au rattrapage des cours d'art obligatoires évités auparavant dans son parcours lycéen.

À l'automne, Stallman était à nouveau dans la norme de la population estudiantine new-yorkaise. Il ne fut pas aisé de suivre des cours apparentés à du rattrapage en comparaison aux études du samedi à Columbia, mais Mme Lippman se souvient avec fierté de la capacité de son fils à rentrer dans le rang.

« Il a été obligé de courber l'échine jusqu'à un certain point, mais il l'a fait », dit-elle. « Je n'ai été convoquée qu'une seule fois, ce qui était un miracle. C'était le professeur de mathématiques qui se plaignait que Richard interrompait sa leçon. Je lui demandai comment. Il répondit que Richard accusait le professeur d'utiliser de fausses démonstrations. 'A-t-il raison ?' dis-je. L'enseignant répondit 'Ouais, mais je ne peux le dire à la classe. Ils ne comprendraient pas.' »

À la fin de son premier semestre à Brandeis, les choses se mirent en place. Une note de 96 en anglais

effaça la plupart des stigmates engendrés par celle de 60 obtenue deux ans auparavant. Pour faire bonne mesure, Stallman confirma avec d'excellentes notes en histoire américaine, calculs avancés, et microbiologie ; le couronnement fut une note parfaite de 100 en physique. Toujours exclu socialement, Stallman termina ses onze mois à Brandeis au quatrième rang d'une promotion de 789 élèves.

En dehors de la classe, Stallman poursuivait ses études avec bien plus d'assiduité, courant en semaine à l'université Rockefeller pour accomplir son devoir d'assistant au laboratoire, et évitant le samedi les protestants contre la guerre du Vietnam sur le chemin de l'école de Columbia. Ce fut là, alors que le restant des élèves du Columbia Science Honors Program s'asseyaient et discutaient de leurs choix d'universités, que Stallman prit enfin un moment pour participer à la séance de discussion avant la classe.

Breidbart se souvient : « La plupart des étudiants allaient vers Harvard ou le MIT, bien sûr, mais quelques uns se tournaient vers les autres universités de l'*Ivy League*. Alors que la conversation faisait le tour de la classe, il devint évident que Richard n'avait toujours rien dit. Je ne sais plus qui, mais quelqu'un eut le courage de lui demander ce qu'il pensait faire. »

Trente ans plus tard, Breidbart se souvient clairement de cet instant. Aussitôt que Stallman annonça qu'il irait aussi à l'université d'Harvard à l'automne, un silence embarrassant envahit la salle. Presque comme par hasard, les coins des lèvres de Stallman se relevèrent lentement pour y dessiner un sourire d'autosatisfaction.

Selon Breidbart, c'était sa manière silencieuse de dire : « En effet! vous n'êtes pas encore débarrassés de moi. »

Notes

1. Un des ouvrages importants pour la réalisation de ce chapitre est l'entrevue *Richard Stallman: High School Misfit, Symbol of Free Software, MacArthur-Certified Genius* de Michael Gross, auteur du livre *Talking About My Generation*, un recueil d'entrevues avec des personnalités notoires de la soi-disant génération « Baby Boom ». Bien que Stallman n'apparaisse pas dans l'ouvrage, Gross a publié l'entrevue sur le site Internet du livre en tant que matériel complémentaire. L'URL de cette entrevue a changé plusieurs fois depuis que je l'ai trouvée. Selon plusieurs lecteurs qui l'ont débusquée, vous pourrez la retrouver [en anglais] sur http://www.mgross.com/MoreThgsChng/interviews/stallman1.html
2. Judy Steed, *Toronto Star*, section affaire (9 octobre 2000):C03. Sa vision du logiciel libre et de la coopération sociale contraste beaucoup avec la nature isolée de sa vie privée : un excentrique comme le pianiste canadien Glenn Gould, aussi brillant, éloquent et solitaire. Stallman se considère affecté, jusqu'à un certain point, par l'autisme : une condition qui, dit-il, complique ses relations avec les gens.
3. Steve Silberman, *The Geek Syndrome*, Wired (décembre 2001).
4. Malheureusement, je n'ai pu interviewer Daniel Stallman pour cet ouvrage. Pendant les recherches préliminaires pour ce livre, Stallman m'avait informé que son père souffrait de la maladie d'Alzheimer. Lorsque j'ai repris l'enquête en 2001, j'ai appris avec regret que Daniel Stallman était décédé plus tôt cette année-là.
5. Athée, Stallman aurait probablement eu maille à partir avec cette description . Il suffirait de dire que c'était quelque chose que Stallman accueillit avec enthousiasme. Voir la note 1: « Aussitôt que j'ai entendu parler d'ordinateurs, je voulais en voir un et jouer avec. »

Chapitre IV — Destituer Dieu

Bien que leur relation fût tendue, Richard Stallman hérita de sa mère un caractère marquant : une passion pour la politique progressiste.

Toutefois, ce caractère prendrait plusieurs décennies à émerger. Au cours des premières années de sa vie, Stallman vécut dans ce qu'il admet maintenant être un « vide politique ».[1] Comme la plupart des américains durant les années Eisenhower, la famille Stallman passa les années 50 à essayer de retrouver une normalité perdue lors de la Deuxième Guerre mondiale.

« Le père de Richard et moi étions des démocrates, mais assez heureux pour en rester là », raconte Mme Lippman, se rappelant les années que la famille avait passé dans le Queens. « Nous n'étions pas beaucoup impliqués dans la politique locale ou nationale. »

Cependant, tout cela commença à changer à la fin des années 1950 quand Alice divorça de Daniel Stallman. Le retour à Manhattan représenta davantage qu'un changement d'adresse : ce fut une nouvelle identité, indépendante, mais aussi la fin d'une tranquille harmonie.

« Je pense que mon goût pour l'activisme politique m'est venu pour la première fois lorsque je me suis rendue dans le Queens à la bibliothèque municipale et que j'ai découvert qu'il n'y avait qu'un seul livre sur le divorce », se rappelle Mme Lippman. « Le divorce était très contrôlé par l'Église catholique, du moins à Elmhurst, où nous vivions. Je pense que c'est la première fois que j'eus conscience des forces qui contrôlent secrètement nos vies. »

De retour dans les quartiers de son enfance, au *Upper West Side* de Manhattan, Mme Lippman fut surprise des changements survenus depuis son départ vers *Hunter College* dix ans et demi plus tôt. Durant l'après-guerre, la demande croissante en logements transforma le quartier en un lieu d'affrontement politique. D'un bord se tenaient les politiciens municipaux pro-développement et les affairistes espérant reconstruire de nombreux immeubles du quartier pour faire face à l'accroissement du nombre d'employés de bureau arrivant en ville. De l'autre bord se tenaient les pauvres irlandais et les locataires porto-ricains qui s'étaient trouvés un refuge abordable dans le quartier.

Au début, Mme Lippman ne savait quel camp choisir. En tant que nouvelle arrivante, elle se sentit attirée par un logement neuf. Cependant, en tant que mère célibataire aux revenus minimaux, elle partageait les mêmes difficultés que les plus pauvres locataires alors qu'un nombre croissant de projets de développement n'était destiné qu'aux plus riches résidents. Indignée, Mme Lipmann commença à chercher les moyens de combattre cette machine politique qui essayait de transformer son quartier en clone d' *Upper East Side*.

Elle raconte que son premier contact avec le siège local du parti Démocrate eut lieu en 1958. À la recherche d'une garderie qui s'occuperait de son fils pendant qu'elle travaillait, elle avait été abasourdie par l'état de l'un des centres appartenant à la ville et destiné aux citoyens à faibles revenus. « Tout ce dont je me souviens, c'est l'odeur de lait caillé, les couloirs sombres, et la pénurie de provisions. J'avais déjà été enseignante en école maternelle privée. Le contraste était tellement immense. Nous n'avons jeté qu'un regard à la salle, et nous sommes partis. J'en étais bouleversée. »

Cependant, sa visite à l'antenne du parti s'avéra décevante. La décrivant comme « la proverbiale salle enfumée », Mme Lippman raconte qu'elle est alors devenue consciente, pour la première fois, que la corruption pouvait être à l'origine de l'hostilité à peine voilée de la ville envers les citoyens les moins nantis. Au lieu de retourner au siège, Mme Lippman décida de rejoindre l'un des nombreux clubs dont l'objectif était de réformer le parti Démocrate et renverser les derniers vestiges de la machine Tammany Hall. Mme Lippman et son club, dénommé le Woodrow Wilson/FDR Reform Democratic Club, commencèrent à se présenter aux réunions de planification et au conseil municipal, demandant à y participer.

« Notre but principal était de combattre Tammany Hall, Carmine DeSapio et son acolyte »[2] poursuit Mme Lippman. « J'étais la représentante au conseil municipal et impliquée dans la création d'un plan viable de renouveau urbain qui allait au-delà d'une simple addition de demeures luxueuses dans le quartier. »

Cette implication s'épanouit alors en une plus grande activité politique durant les années 1960. En 1965, Mme Lippman était devenue une sympatisante affirmée de candidats politiques tel William Fitts Ryan, un démocrate élu au congrès américain avec l'aide de clubs réformistes, et l'un des premiers représentants américains à se déclarer ouvertement contre la guerre du Vietnam.

Il fallut peu de temps à Mme Lippman pour qu'elle aussi s'opposât à l'intervention des États-Unis en Indochine. « J'étais contre la guerre du Vietnam depuis le moment où Kennedy y avait envoyé des troupes », dit-elle. « J'avais lu les reportages des journalistes qui rendaient compte des premiers instants du conflit. Je croyais leurs pronostics qui prédisaient que cela deviendrait un véritable bourbier. »

Cette opposition pénétra le ménage Stallman-Lippman. En 1967, Mme Lippman se remaria. Son nouvel époux, Maurice Lippman, un major de la Garde Nationale Aérienne, démissionna de ses fonctions en opposition à la guerre. Le fils de Maurice Lippman, Andrew Lippman, était au MIT et donc admissible au sursis étudiant. Cela dit, si ce sursis devait disparaître, comme ce fut le cas plus tard, la menace d'incorporation représentait plus que tout un risque d'escalade pour les États-Unis. Finalement, il y avait Richard qui, bien que plus jeune, faisait face à l'éventualité d'un choix entre le Canada et le Vietnam où la guerre allait perdurer jusqu'au années 1970.

« Le Vietnam était une question majeure dans notre foyer », raconte Mme Lippman. « Nous en parlions continuellement : que ferions-nous si la guerre continuait? que feraient Richard ou son demi-frère s'ils étaient affectés ? Nous étions tous opposés à la guerre et à l'incorporation. Nous pensions cela tout à fait amoral. »

Chez Stallman, la guerre du Vietnam éveillait un mélange complexe d'émotions : confusion, horreur, et en fin de compte, un sentiment profond d'impuissance politique. Enfant ayant du mal à tenir le coup dans l'univers modérément autoritaire de l'école privée, Stallman éprouvait un frisson à la simple idée du camp d'entraînement militaire.

« J'étais anéanti par la peur, mais je ne pouvais imaginer que faire et je n'avais pas le courage d'aller manifester », se souvient Stallman, à qui son dix-huitième anniversaire en mars valut un chiffre effroyablement bas à la loterie d'incorporation lorsque le gouvernement fédéral abolit le sursis collégial en 1971. « Je ne pouvais envisager de déménager au Canada ou en Suède. L'idée de me lever de mon propre chef et de partir quelque part... comment pouvais-je faire cela? Je ne savais pas vivre seul. Je n'étais pas de ceux qui se sentaient confiants devant ce genre de chose. »

Stallman disait qu'il était autant impressionné que honteux devant les membres de la famille qui s'exprimaient. Au souvenir d'un auto-collant sur la voiture de son père faisant un lien entre le massacre de My Lai et les atrocités commises par les nazis durant la seconde guerre mondiale, il se disait « excité » par les gestes d'outrage de son père. « Je l'admirais pour l'avoir fait », dit-il. « Mais je ne m'imaginais pas capable d'y faire quelque chose. J'avais peur que les forces aveugles de l'incorporation ne me détruisent. »

Quoique les détails de sa réticence à se prononcer portassent une teinte de regret nostalgique, Stallman rapporte qu'il était en fait désenchanté par le ton et la direction que prenait le mouvement anti-guerre. Tels les autres membres du Science Honors Program (SHP — cf. chapitre 3), il voyait les manifestations de fin de semaine à Columbia comme rien de plus qu'un spectacle distrayant[3]. Au bout du compte, poursuit-il, les forces irrationnelles qui menaient le mouvement anti-guerre devenaient indissociables des forces irrationnelles du reste de la culture des jeunes. Au lieu d'aduler les Beatles, les filles de l'âge de Stallman adorèrent soudainement des agitateurs comme Abbie Hoffman et Jerry Rubin. Pour un jeune comme Stallman, qui bataillait pour comprendre ses pairs adolescents, les slogans illusoires tels "faites l'amour, pas la guerre" suscitaient le sarcasme. Ce n'était pas seulement que Stallman, ce jeune décalé aux cheveux courts détestant le rock and roll et les drogues et ne participant pas aux manifestations du campus, ne comprenait rien politiquement : il n'y comprenait rien sexuellement non plus.

« Je n'aimais pas beaucoup la contre-culture », admet-il. « Je n'aimais pas cette musique. Je n'aimais pas la drogue. J'avais peur de la drogue. Je n'aimais surtout pas l'anti-intellectualisme, et n'aimais pas les préjugés contre la technologie. Après tout, j'aimais un ordinateur. Et je n'aimais pas l'anti-américanisme idiot que je rencontrais souvent. Il y avait des gens au mode de pensée si simpliste qui, n'approuvant pas la conduite des États-Unis dans la guerre du Vietnam, se devaient de supporter les vietnamiens du Nord. Ils ne pouvaient imaginer une position plus complexe je suppose. »

De tels commentaires atténuent les tendances à la timidité. Ils soulignent aussi un trait de caractère qui deviendrait la clé de la maturation politique de Stallman. Pour lui, la confiance politique était directement proportionnelle à la confiance personnelle. Avant 1970, Stallman devint confiant en peu de choses en dehors du domaine des mathématiques et des sciences. Néanmoins, son assurance en mathématiques lui donna une bonne base pour examiner de manière purement logique le mouvement anti-guerre. Ce faisant, Stallman y trouva ce qui faisait défaut. Bien qu'opposé à la guerre du Vietnam, il ne voyait aucune raison de désavouer la guerre comme moyen de défense de la liberté ou de correction des injustices. Plutôt que d'élargir le fossé entre lui et ses pairs, Stallman choisit de garder l'analyse pour lui.

En 1970, à son départ pour Harvard, Stallman laissa derrière lui les conversations nocturnes du dîner à propos de politique ou de la guerre du Vietnam. Rétrospectivement, Stallman décrit la transition entre l'appartement maternel de Manhattan et le dortoir de Cambridge comme une « évasion ». Cependant, les pairs qui l'observèrent virent peu de signes suggérant une expérience libératrice.

Dan Chess, un ancien collègue de classe du SHP également admis à l'université, se souvient : « Il semblait plutôt misérable les premiers temps à Harvard. On pouvait voir que l'interaction humaine lui posait une réelle difficulté, or, c'était inévitable ici. Harvard était un endroit intensément social. »

Pour faciliter la transition, Stallman s'appuyait sur ses forces : les mathématiques et la science. Comme les autres membres du SHP, Stallman réussit aisément l'examen de qualification à Math 55, le légendaire cours « camp d'entraînement » pour les étudiants des premières années « concentrées » mathématiques à Harvard. Dans cette classe, les membres du SHP formaient une unité solide. « Nous étions la mafia math », relate Chess en riant. « Harvard n'était rien, du moins comparé au SHP. »

Pour obtenir le droit de se vanter, Stallman, Chess et les autres devaient passer par Math 55. Promettant l'équivalent de quatre ans de maths en deux semestres, le cours favorisait les vrais fanatiques. « C'était un cours fascinant », raconte David Harbater, ancien membre de la « mafia math », maintenant professeur de mathématiques à l'université de Pennsylvanie. « On peut probablement dire sans crainte qu'il n'y a jamais eu de cours pour universitaire débutant aussi intense et avancé. La phrase que j'utilise pour bien en rendre compte est que, entre autres choses, au deuxième semestre, nous discutions de géométrie différentielle des espaces de Banach. C'est habituellement à ce moment que les yeux s'écarquillent, car la plupart ne commencent à parler des espaces de Banach qu'en deuxième année. »

De soixante-quinze étudiants, la classe s'est rapidement réduite à vingt vers la fin du second semestre. De ces vingt, raconte Harbater, « seulement dix savaient réellement ce qu'ils faisaient ». De ces dix, huit deviendraient professeurs de mathématiques, et un enseignerait la physique.

« L'autre », souligne Harbater, « était Richard Stallman. »

Seth Breidbart, vétéran du SHP et collègue de classe de Math 55, se souvient de Stallman se distinguant de ses collègues, même à ce niveau: « Il était tatillon d'une manière bien étrange », poursuit Breidbart. « Il y a une technique standard en mathématiques que tout le monde fait de travers. C'est un abus de notation où vous définissez une fonction pour quelque chose, et ce que vous faites, c'est définir une fonction et ensuite prouver qu'elle est bien définie. Sauf que la première fois qu'il l'a fait et présenté, il a défini une relation et prouvé que c'était une fonction. C'est exactement la même preuve, mais il a utilisé la bonne terminologie ce que personne d'autre n'a fait. C'était simplement sa manière d'être. »

Ce fut en Math 55 que Richard Stallman commença à cultiver sa réputation de génie. Breidbart en convint, mais Chess, à la fibre plus compétitive, se refusa à cette conclusion hâtive disant que le couronnement de Stallman en tant que meilleur mathématicien de la classe n'intervint que l'année suivante. Aujourd'hui professeur de maths au Hunter College, Chess affirme: « C'était durant la classe traitant de l'analyse réelle. Je me souviens effectivement que, dans une démonstration sur les mesures de nombres complexes, Richard proposa une idée qui était une métaphore de l'équation différentielle. C'était la première fois que je voyais quelqu'un résoudre un problème de manière brillante et originale. »

C'était pour Chess un moment troublant. Tel un oiseau heurtant une fenêtre transparente en plein vol, il lui faudrait quelque temps pour réaliser que certains niveaux d'intuition étaient tout simplement hors de portée.

« C'est ainsi avec les mathématiques », reprend Chess. « Vous n'avez pas besoin d'être mathématicien

de haut niveau pour reconnaître un grand talent en mathématiques. Je savais que je l'étais, mais je pouvais aussi voir que je n'occupais pas le premier rang. Si Richard avait choisi d'être mathématicien, il serait devenu un mathématicien hors pair. »

Les succès de Stallman en classe étaient contrebalancés d'autant par son manque de réussite dans l'arène sociale. Alors que les autres membres de la mafia math se rassemblaient pour s'attaquer aux problèmes de Math 55, il préférait travailler seul, même dans la vie quotidienne. Dans sa demande d'hébergement à Harvard, Stallman avait exposé ses préférences. « J'avais indiqué que je préférais un camarade de chambre invisible, inaudible et imperceptible », dit-il. Effort lucide plutôt rare de la part de la bureaucratie, les bureaux d'hébergement de Harvard acceptèrent sa demande en lui octroyant une chambre pour une personne au cours de sa première année.

Breidbart, le seul de la mafia math à partager un dortoir avec Stallman cette première année, raconte qu'il apprit lentement mais sûrement à interagir avec les autres étudiants. Il se souvient que les autres collègues du dortoir, impressionnés par la logique perspicace de Stallman, commençaient à apprécier ses interventions lorsqu'un débat intellectuel faisait rage dans la salle à manger ou les pièces communes.

« Nous avions ces séances habituelles où nous refaisions le monde ou bien imaginions ce que serait le résultat de telle chose », se rappelle Breidbart. « Supposons que quelqu'un découvre un sérum d'immortalité. Que faites-vous? Qu'en seront les résultats politiques? Si vous le donnez à tout le monde, la planète devient surpeuplée et l'humanité meurt. Si vous limitez le sérum, si vous décidez que seuls ceux vivant actuellement peuvent en avoir mais pas leurs enfants, alors vous vous retrouvez avec une sous-classe de personnes qui n'en ont pas. Richard était plus compétent que la majorité pour voir les conséquences insoupçonnées de toute décision. »

Stallman se rappelle les discussions d'une manière saisissante. « J'étais toujours en faveur de l'immortalité », dit-il. « J'étais choqué que la plupart des gens voyaient l'immortalité comme une mauvaise chose. De quelle autre manière pourrions-nous voir ce que sera le monde dans 200 ans? »

Quoique mathématicien et débatteur de premier ordre, Stallman se tenait à l'écart de challenges précis qui auraient pu confirmer sa réputation brillante. Vers la fin de la première année à Harvard, Breidbart se souvient comment Stallman évita subtilement l'examen Putnam, une prestigieuse épreuve ouverte aux étudiants en mathématiques américains et canadiens. En plus de donner une chance aux étudiants de mesurer leurs connaissances avec celles de leurs pairs, l'examen Putnam était un outil de premier ordre pour le recrutement dans les départements académiques de mathématiques . Selon une légende du campus, le meilleur score vous qualifiait automatiquement pour l'obtention d'une bourse universitaire à l'école de son choix, Harvard inclus.

Comme le cours de math 55, l'examen Putnam était un terrible test de mérite. Examen de six heures en deux volets, il semblait clairement conçu pour séparer le bon grain de l'ivraie. Breidbart le décrit comme étant de loin le plus difficile auquel il participa. « Pour vous donner une idée de la difficulté », commente Breidbart, « la note maximale était de 120, et ma note la première année était dans les 30. Cette note fut suffisamment bonne pour me classer 101e à l'échelle du pays. »

Étonné que Stallman, le meilleur étudiant de la classe, ait sauté ce test, Breidbart raconte que lui et un collègue de classe l'avaient coincé dans la salle à manger commune pour lui demander des explications. « Il disait qu'il craignait de ne pas bien réussir », se souvient Breidbart.

Breidbart et cet ami écrivirent rapidement de mémoire sur un papier quelques problèmes qu'ils donnèrent ensuite à Stallman. « Il les a tous résolus », rapporte Breidbart, « m'amenant à croire qu'en craignant de ne pas bien réussir, il voulait dire finir deuxième ou se tromper quelque part. »

Stallman se souvient de cet épisode un peu différemment. « Je me souviens qu'ils m'avaient apporté les questions, et il est possible que j'en aie résolu une, mais je suis certain de ne pas les avoir toutes résolues », dit-il. Néanmoins, Stallman est d'accord avec le souvenir de Breidbart: la peur était la première raison de ne pas passer le test. Malgré un empressement notoire à signaler les faiblesses intellectuelles de ses pairs et professeurs en classe, Stallman haïssait la notion de compétition directe.

« C'est pour la même raison que je n'ai jamais aimé les échecs », renchérit Stallman. « Lorsque je jouais, j'étais si absorbé par la crainte de faire la moindre erreur que j'en faisais des idiotes très tôt dans la

partie. La crainte devenait une prophétie s'auto-réalisant. »

Savoir si de telles peurs ont finalement éloigné Stallman d'une carrière en mathématiques est sans importance. À la fin de sa première année à Harvard, il avait d'autres intérêts qui l'éloignaient du domaine. La programmation informatique, une fascination latente durant ses années de collège, devenait une passion véritable. Alors que d'autres étudiants de mathématiques trouvaient un refuge occasionnel dans les cours d'art ou d'histoire, Stallman le trouvait dans le laboratoire de sciences de l'informatique.

Son premier contact concret avec la programmation informatique au centre scientifique d'IBM à New York éveilla son désir d'en apprendre plus. « Vers la fin de ma première année à Harvard, je commençais à avoir assez de courage pour aller visiter les labos informatiques et voir ce qu'ils avaient. Je leur demandais s'ils avaient des copies supplémentaires de manuels que je pourrais lire. »

Emportant ces manuels chez lui, Stallman examinait le cahier des charges des machines, et, comparant avec d'autres appareils qu'il connaissait déjà, il concoctait un programme d'essai, lequel était ensuite ramené au labo avec le manuel emprunté. Bien que certains labos se fussent dérobés à l'idée qu'un garçon étrange venant de la rue puisse travailler sur les machines, la plupart savaient reconnaître une compétence lorsqu'elle se présentait, et laissaient donc Stallman exécuter les programmes qu'il avait créés.

Un jour, vers la fin de sa première année universitaire, Stallman entendit parler d'un laboratoire spécialisé près du MIT. Celui-ci était situé au neuvième étage d'un édifice hors campus, au Tech Square : une nouvelle construction dédiée à la recherche avancée. Selon les rumeurs, le laboratoire se consacrait à l'intelligence artificielle, une science de pointe, et s'enorgueillait de son lot de programmes informatiques et de machines ultramodernes.

Intrigué, Stallman décida de s'y rendre.

Le voyage était court, environ deux miles à pied, dix minutes en train, mais comme allait le découvrir bientôt Stallman, le MIT et Harvard donnent l'impression d'être les pôles opposés d'une même planète. Avec ses connections labyrinthiques entre édifices, le campus de l'institut offrait une architecture ying comparée au yang du spacieux village colonial de Harvard. On pourrait en dire autant du corps étudiant : une collection de « geeks » et anciens lycéens inadaptés, plus connus pour leur prédilection aux canulars que pour leur influence politique.

La relation ying-yang s'étendait aussi au laboratoire d'intelligence artificielle (AI Lab). Contrairement aux laboratoires informatiques d'Harvard, il n'y avait pas de gardien, pas de liste d'attente pour l'accès aux terminaux, pas d'atmosphère formelle du genre : « regardez, mais ne touchez pas ». Au lieu de cela, Stallman trouva une collection de terminaux ouverts et de bras robotiques, vraisemblablement les artefacts de quelque expérience en intelligence artificielle.

Même si les rumeurs laissaient entendre que n'importe qui pouvait s'asseoir à un terminal, Stallman s'en tint au plan original. Lorsqu'il rencontrait un employé du laboratoire, il demandait s'il y avait des manuels supplémentaires à prêter à un étudiant curieux. « Il y en avait, mais beaucoup de choses n'étaient pas documentées », se souvient Stallman. « C'étaient des hackers après tout. »

Stallman repartit avec quelque chose de bien mieux qu'un manuel : un emploi. Alors qu'il ne se souvient pas du premier projet sur lequel il a travaillé, il se rappelle néanmoins être retourné au AI Lab la semaine suivante, s'être accaparé un terminal ouvert et avoir écrit du code informatique.

Rétrospectivement, Stallman ne note rien d'inhabituel à la bonne volonté du AI Lab d'accepter un novice au premier coup d'œil . « C'était comme ça à cette époque », dit-il. « C'est toujours comme ça aujourd'hui. J'embauche quelqu'un quand je le rencontre si je vois qu'il est bon. Pourquoi attendre? Les gens étouffants, qui insistent pour mettre de la bureaucratie partout, n'ont rien compris. Si une personne est compétente, elle ne devrait pas avoir à passer par un long et fastidieux processus d'embauche ; elle devrait être assise à un ordinateur en train d'écrire du code informatique. »

Pour un aperçu de bureaucratie étouffante, Stallman n'avait qu'à visiter les laboratoires informatiques de Harvard. Là, l'accès aux terminaux était attribué au compte-gouttes selon le rang académique. En tant que nouveau, Stallman devait s'inscrire ou attendre jusqu'à minuit, heure à laquelle la plupart des étudiants et professeurs finissaient leurs travaux quotidiens. Attendre n'était pas difficile mais frustrant. L'attente d'un

terminal public, tout en sachant qu'une demi-douzaine de machines étaient inutilisées dans les bureaux fermés à clé des professeurs, paraissait le comble de l'illogisme. Bien que Stallman visitât occasionnellement les labos informatiques de Harvard, il préférait la politique plus égalitaire du AI Lab au MIT. « C'était une bouffée d'air frais », dit-il. « Ici, les gens semblaient davantage préoccupés par le travail que par le statut. »

Stallman apprit rapidement que la politique du premier venu, premier servi du AI Lab était amplement due aux efforts de quelques personnes vigilantes. Beaucoup venaient du projet MAC, le programme de recherche subventionné par le Département de la Défense qui avait donné naissance au premier système d'exploitation à temps partagé. Quelques-uns étaient déjà des légendes dans le monde de l'informatique. Il y avait Richard Greenblatt, l'expert maison en langage Lisp et auteur de MacHack, le programme de jeu d'échecs ayant humilié Hubert Dreyfus, un détracteur de l'intelligence artificielle. Il y avait Gerald Sussman, l'auteur du programme robotique HACKER pour empiler des blocs. Et il y avait Bill Gosper, le génie de la maison en mathématiques, alors plongé en pleine programmation frénétique, qui dura 18 mois, motivée par les implications philosophiques du jeu LIFE[4].

Les membres de ce groupe soudé s'entr'appelait « hackers ». Avec le temps, ils étendirent aussi cet attribut à Stallman. Ce faisant, ils lui inculquèrent la déontologie traditionnelle de « l'éthique hacker ». Être hacker signifie davantage que d'écrire des programmes, apprit Stallman. Cela voulait dire écrire les meilleurs programmes possibles. Cela voulait dire s'asseoir à un terminal trente-six heures durant, sans interruption si nécessaire, pour écrire le meilleur programme possible. Plus important, cela signifiait avoir accès en tout temps aux meilleures machines et aux informations les plus utiles. Les hackers parlaient ouvertement de changer le monde par les logiciels, et Stallman apprit ce dédain instinctif du hacker pour tout obstacle empêchant la réalisation de cette noble cause. Les principales barrières étaient les logiciels de piètre qualité, la bureaucratie universitaire et le comportement égoïste.

Stallman se familiarisa également avec les habitudes et apprit les petites histoires sur la manière dont les hackers, face à un obstacle, le contournaient de manière créative. Stallman s'initia aussi au « hacking de verrou », l'art d'entrer dans les bureaux des professeurs pour « libérer » des terminaux séquestrés. Contrairement à leurs homologues gâtés de Harvard, les membres de la faculté du MIT avaient la sagesse de ne pas traiter un terminal du AI Lab en tant que propriété privée. Si un membre faisait l'erreur d'enfermer un terminal la nuit, les hackers étaient prompts à corriger cette aberration. Ils étaient aussi rapides à envoyer un message si cela se répétait. « On m'avait montré un petit chariot muni d'un lourd cylindre de métal qui avait servi à enfoncer la porte d'un des bureaux des professeurs »,[5] raconte Stallman.

Ces méthodes, quoique manquant de subtilité, avaient un but. Même si, au AI Lab, le nombre de professeurs et administrateurs doublait celui des hackers, l'éthique hacker prévalait. En effet, à l'arrivée de Stallman au laboratoire, les hackers et l'administration avaient évolué ensemble vers une relation proche de la symbiose. En échange de la réparation de machines et de la mise en fonction des logiciels, les hackers obtenaient le droit de travailler sur leurs projets favoris. Souvent ces études tournaient autour d'une amélioration plus poussée des machines et des logiciels. Comme ces jeunes fous de mécanique automobile (les *hot-rodders*), les hackers voyaient le bricolage de ces appareils comme un divertissement en soi.

Le système d'exploitation pilotant le mini-ordinateur central PDP-6 du labo reflète le mieux cette manie du bidouillage. Surnommé *ITS*, abrégé pour *Incompatible Time Sharing system* (Système Incompatible à Temps Partagé), le système opératoire intègre l'éthique hacker dans sa conception. Les hackers l'avaient conçu et nommé en réaction au système d'exploitation originel du projet MAC, le *Compatible Time Sharing System* (CTSS -- Système Compatible à Temps Partagé). À cette époque, ils trouvaient la conception du CTSS trop restrictive, limitant les possibilités du programmeur à modifier et à améliorer, si nécessaire, l'architecture interne du logiciel même. Selon une légende transmise par les hackers, la décision de compiler l'ITS avait aussi des couleurs politiques. Contrairement au CTSS, conçu pour l'IBM 7094, l'ITS était compilé spécifiquement pour le PDP-6. En laissant les hackers programmer eux-mêmes le système, l'administration du AI Lab garantissait que seuls les hackers pourraient l'utiliser aisément. Dans le monde féodal de la recherche universitaire, la manœuvre réussit. Bien que la propriété du PDP-6 fût partagée avec d'autres départements, les chercheurs en Intelligence Artificielle en ont rapidement eu le monopole.[6]

L'*ITS* possédait des caractéristiques que la plupart des logiciels commerciaux n'offriraient pas avant des années, tels le multi-tâches, le dépannage [*debugging*], et l'édition en mode plein-écran. En l'utilisant avec le PDP-6 comme support, le labo put déclarer son indépendance vis-à-vis du projet MAC peu avant

l'arrivée de Stallman.[7]

En tant qu'apprenti-hacker, Stallman s'enticha rapidement de l'*ITS*. Bien que rebutant pour la plupart des nouveaux venus, le programme contenait beaucoup de caractéristiques internes pouvant servir de leçons en développement informatique pour un néophyte comme lui.

« L'*ITS* avait un mécanisme interne très élégant permettant à un programme d'en examiner un autre », dit-il au souvenir du logiciel. « Vous pouviez analyser tous les statuts d'un autre programme d'une manière très propre et bien détaillée. »

Utilisant cette capacité, Stallman pouvait observer comment des programmes écrits par des hackers traitaient les instructions au moment de l'exécution. Une autre caractéristique appréciée permettait au programme de surveillance de geler, entre deux instructions, une des tâches du programme surveillé. Dans les autres systèmes d'exploitation, une telle commande aurait conduit à un galimatias informatique voire une interruption automatique du système [*system crash*]. L'*ITS* permettait de surveiller la performance étape par étape.

« Si vous disiez 'arrête le travail', [le programme] s'arrêtait toujours en mode utilisateur. Il s'arrêtait entre deux instructions sur ce mode, et tout le travail était ordonné jusqu'à ce point là », raconte Stallman. « Si vous disiez 'continue le travail', il continuait proprement. De plus, si vous changiez le statut de la tâche puis la modifiiez à nouveau, tout restait cohérent. Il n'y avait de statut caché nulle part. »

Vers la fin de l'année 1970, faire du hacking au AI Lab était devenu une activité régulière dans l'agenda hebdomadaire de Stallman. Du lundi au jeudi, il consacrait son temps à ses cours de Harvard. Cependant, dès le vendredi après-midi, il prenait le *T*, en route vers le MIT pour le restant de la semaine. Stallman faisait habituellement coïncider son arrivée avec la course rituelle à la nourriture. Rejoignant cinq ou six autres hackers dans leur quête nocturne de cuisine chinoise, il sautait à bord d'une vieille bagnole pour passer le pont de Harvard en direction de Boston près de là. Pendant les deux heures suivantes, lui et ses collègues discutaient de tout, passant de l'ITS à la logique interne de la langue chinoise et son système pictographique. Après le dîner, le groupe s'en retournait au MIT et programmait jusqu'à l'aurore.

Pour l'exclu excentrique s'associant peu à ses pairs étudiants, c'était une expérience grisante que de soudainement flâner avec des gens partageant la même prédilection pour les ordinateurs, la science-fiction et la cuisine chinoise. Quinze ans après les faits, lors d'un discours à l'Institut Technique Royal de Suède, Stallman se remémore avec nostalgie : « Je me souviens de nombreux levers de soleil vus en voiture au retour de Chinatown. C'était alors magnifique de voir un lever de soleil, car c'est un moment si calme de la journée. C'est un instant merveilleux pour se préparer à aller dormir. C'est si beau de rentrer chez soi alors que se lève la lumière du jour et que les oiseaux commencent à chanter. Vous pouvez avoir une réelle sensation de douce satisfaction, de tranquilité à propos du travail accompli cette nuit-là. »[8]

Plus Stallman fréquentait les hackers, plus il adoptait leur vision du monde. Étant déjà engagé dans la notion de liberté personnelle, il commença à insuffler à ses actions un sens de la responsabilité collective. Lorsque les autres violaient le code de la communauté, Stallman haussait le ton rapidement. Moins d'un an après sa première visite, il faisait déjà partie de ceux qui forçaient les bureaux fermés pour tenter de récupérer les terminaux enfermés appartenant à l'ensemble de la communauté du laboratoire . Tel un véritable hacker, Stallman ajouta sa touche personnelle à l'art du « hacking de verrou ». L'une des manières les plus artistiques d'ouvrir les portes, communément attribuée à Greenblatt, consistait à passer une tige de fil électrique dans une canne avec, attachée à l'extrémité, une boucle faite de bande magnétique. Glissant la tige sous la porte, un hacker pouvait ainsi la tordre et la tourner de manière à ce que la boucle atteignît la poignée. Si l'adhésif ajouté sur la bande tenait bon, le hacker pouvait ouvrir la porte en quelques mouvements brusques.

Lorsque Stallman essaya, il trouva cela bien mais améliorable. Accrocher la bande au bâton n'était pas toujours facile, de même pour courber la tige de manière à faire tourner la poignée. Stallman se souvient que le plafond des couloirs avaient des plaques pouvant être retirées en les glissant. Quelques hackers, en fait, avaient utilisé le faux plafond pour venir à bout des portes verrouillées, une approche qui recouvrait généralement le hacker de fibre de verre mais qui réussissait.

Stallman étudia une autre alternative. Et si, au lieu de passer une tige sous la porte, un hacker faisait

glisser un des panneaux et se tenait au-dessus du montant de la porte?

Stallman prit sur lui d'essayer. Au lieu d'utiliser une tige, Stallman laissa pendre un long ruban magnétique en forme de « u » avec une boucle de ruban adhésif à la base du « u ». Par-dessus le montant de la porte, il laissa pendre le ruban jusqu'à ceinturer la poignée. Remontant le ruban jusqu'à ce que l'adhésif prenne, il le tira vers la gauche, tournant la poignée dans le sens contraire des aiguilles d'une montre. Comme on pouvait s'y attendre, la porte s'ouvrit. Stallman ajouta ainsi un tour personnel à l'art du « hacking de verrou ».

« Parfois vous deviez frapper la porte avec le pied après avoir tourné la poignée de porte », ajoute-t-il, se souvenant des ratés de la nouvelle méthode. « Il fallait un peu d'équilibre pour la tourner. »

Ces activités reflètent la volonté grandissante de la part de Stallman à parler et agir pour défendre ses convictions politiques . L'esprit d'action directe du AI Lab fut une motivation suffisante pour que Stallman sorte de la timide impuissance de son adolescence. Forcer l'entrée d'un bureau pour libérer un terminal n'était pas la même chose que de participer à une marche de protestation, mais c'était bien plus efficace d'une certaine manière : cela résolvait les problèmes de l'instant.

Pendant ses dernières années à Harvard, Stallman commença à y appliquer à l'école les leçons fantasques et irrévérencieuses du AI Lab.

« Vous a-t-il raconté l'histoire du serpent? », demande sa mère pendant une entrevue. Lui et ses compères de dortoir avaient soumis la candidature d'un serpent pour les élections d'étudiants. Apparemment, le serpent a obtenu un nombre considérable de votes. »

Stallman confirme la candidature du serpent avec quelques mises en garde. Le serpent était un candidat à l'élection au sein de la Currier House, le dortoir de Stallman, et pas au conseil des étudiants du campus. Il se souvient que le serpent avait attiré un nombre significatif de votes principalement grâce au nom de famille qu'il partageait avec son propriétaire. « Les gens peuvent avoir voté pour lui parce qu'ils croyaient voter pour son propriétaire », dit Stallman. « Les affiches de la campagne disaient que le serpent 'rampait vers' son siège. Nous disions aussi que c'était un candidat 'itinérant' puisqu'il avait grimpé dans le mur par une unité de ventilation quelques semaines auparavant, et que personne ne savait où il se trouvait. »

Présenter un serpent au conseil du dortoir n'était qu'une des farces liées aux élections. Pour un vote ultérieur, Stallman et ses compères de dortoir nominèrent le fils du maître de maison. « Sa plate-forme électorale demandait la retraite obligatoire à l'âge de sept ans », se souvient Stallman. Cependant, les canulars de Harvard étaient moins éclatants que les faux candidats du campus du MIT. L'une des plus belles réussites fut un chat dénommé Woodstock qui dépassa en votes tous les candidats humains au cours d'un scrutin à l'échelle du campus. « Ils n'ont jamais annoncé combien de votes Woodstock avait obtenu, et les ont traités comme nuls », se souvient Stallman. « Mais le grande nombre de votes nuls suggérait que Woodstock avait réellement gagné. Deux ou trois années plus tard, une voiture a écrasé Woodstock de manière suspecte. Personne ne sait si le conducteur travaillait pour l'administration du MIT ». Stallman rapporte n'avoir rien eu à faire avec la candidature de Woodstock, « mais je l'ai admiré », ajoute-t-il.[9]

Au AI Lab, les activités politiques de Stallman avaient une couleur plus tranchée. Durant les années 1970, les hackers faisaient face au défi constant des membres de la faculté et de ses administrateurs qui voulaient mettre fin à l'*ITS* et sa conception engageante envers les hackers. Une des premières tentatives eut lieu au milieu des années 1970, alors que de plus en plus de membres de la faculté demandaient un système de fichiers sécuritaire pour protéger les données de recherche. La plupart des autres laboratoires de recherche avaient installé de tels systèmes vers la fin des années 1960, mais le AI Lab, à l'insistance de Stallman et des autres hackers, était demeuré une zone non-sécurisée.

Pour Stallman, l'opposition au système de sécurité était éthique et pratique. Du côté éthique, il soulignait que l'art du « hacking » reposait sur l'ouverture et la confiance intellectuelles. Du côté pratique, il faisait valoir que la structure interne de l'*ITS* était construite sur cet esprit d'ouverture et que toute tentative d'inverser cette conception demandait une refonte majeure.

« Les hackers ayant écrit l'*ITS* avaient conclu que la protection de fichier était habituellement utilisée par un soi-disant administrateur système pour obtenir un pouvoir sur autrui », expliquerait Stallman par la suite. « Ils ne voulaient pas que quelqu'un ait un pouvoir sur eux de cette façon et ils n'ont donc pas

implémenté de telles fonctions. Le résultat était que, peu importe ce qui pouvait se casser dans le système, vous pouviez toujours le réparer. »[10]

Avec beaucoup de vigilance, les hackers parvinrent à conserver les machines du AI Lab sans artifice de sécurité. Tout près cependant, au laboratoire de sciences informatiques (*Laboratory for Computer Sciences* — LCS) du MIT, l'esprit sécuritaire des membres de la faculté gagna la partie. Le LCS installa son premier système basé sur mot de passe en 1977. Encore une fois, Stallman prit sur lui de corriger ce qu'il considérait comme un relâchement éthique. Accédant au code source qui contrôlait le système de mots de passe, il y intégra une commande logicielle envoyant un message à tout usager du LCS qui tentait de créer un mot de passe unique. Si un utilisateur utilisait *starfish* [étoile de mer] par exemple, le message renvoyait quelque chose comme: « Je vois que vous avez choisi le mot de passe 'starfish'. Je vous suggère d'utiliser le mot de passe « retour chariot ». C'est plus facile à taper, et cela valide aussi le principe qu'il ne devrait pas exister de mot de passe.[11]

Les usagers qui se servaient de « retour chariot » — c'est-à-dire les utilisateurs qui appuyaient sur la touche « retour », entrant en fait une chaîne de caractères vide — laissaient leur compte accessible à quiconque. Aussi angoissant que cela pouvait être pour certains, cela renforçait l'idée du hacker qu'un ordinateur de l'institut, de même que ses fichiers informatiques, appartenaient au public et non à des individus. Lors d'une entrevue en 1984 pour le livre *Hackers*, Stallman note avec fierté qu'un cinquième des employés du LCS acceptèrent cet argument et employèrent la chaîne de caractères vide comme mot de passe. [12]

Cette croisade de Stallman s'avérera finalement futile. Vers la fin des années 80, même les machines du AI Lab avaient des systèmes de sécurité à mot de passe. Malgré tout, cela représenta une étape importante dans la maturation personnelle et politique de Stallman. Pour un observateur objectif connaissant la future carrière de Stallman, cela révèle un point d'inflexion établi entre l'adolescent timide ayant peur de s'exprimer, y compris sur des questions de vie ou de mort, et l'adulte activiste transformant son agacement et sa persuasion en une occupation à temps plein.

En exprimant son opposition à la sécurité informatique, Stallman faisait appel aux forces qui avaient façonné son passé : la soif de connaissances, le dégoût de l'autorité et l'irritation contre les règles et procédures secrètes qui excluaient certaines personnes naïves. Il s'abreuvait aussi des principes éthiques qui allaient façonner sa vie adulte : la responsabilité communautaire, la confiance et l'esprit d'action directe du hacker. Exprimée en termes de programmation informatique, la chaîne de caractère vide représente la version 1.0 de la vision politique de Richard Stallman — incomplète en certains points, mais en grande partie, pleinement mature.

Avec le recul, Stallman hésite à donner trop de signification à un événement survenu si tôt dans sa carrière de hacker. « Au début, beaucoup de gens partageaient mon sentiment », dit-il. « Le grand nombre de personnes ayant adopté la chaîne de caractères vide comme mot de passe était un signe que beaucoup pensaient que c'était la meilleure chose à faire. J'étais simplement enclin à militer sur ce point. »

Quoiqu'il en soit, Stallman doit au AI Lab de l'avoir éveillé à l'esprit activiste. Adolescent, Stallman avait observé les évènements politiques avec peu d'idée de ce qu'un individu seul pouvait faire ou dire d'important. Jeune adulte, Stallman s'exprimait sur des sujets où il se sentait très confiant, des sujets tels que l'architecture logicielle, la responsabilité collective, et la liberté individuelle. « J'ai rejoint cette communauté qui avait un style de vie impliquant le respect de la liberté de l'autre », dit-il. « Il me fallut peu de temps pour me rendre compte que c'était une bonne chose. Il m'a fallu plus de temps pour réaliser que c'était un enjeu moral. »

Faire du *hacking* au AI Lab ne fut pas la seule activité à accroître l'estime de Stallman. À la moitié de sa seconde année à Harvard, Stallman avait rejoint les rangs d'une troupe de danse spécialisée dans les danses folkloriques. Ce qui était initialement une tentative pour rencontrer des femmes et agrandir son cercle social devint rapidement une nouvelle passion à côté du *hacking*. En dansant devant un public vêtu d'un costume de paysan balte, Stallman ne se sentait plus comme cet étrange enfant de dix ans sans coordination motrice dont les essais au football américain finirent en frustration. Il se sentait confiant, agile et vivant. Pour un bref instant, il ressentit même l'illusion d'un lien émotionnel. Il trouva vite amusant d'être devant un public, et il ne fallut que peu de temps pour qu'il prenne autant goût aux représentations qu'à l'aspect social de la chose.

Bien que la danse et le *hacking* aient peu amélioré le statut social de Stallman, cela l'aida à surmonter le sentiment d'étrangeté qui avait assombri sa vie pré-Harvardienne. Au lieu de se lamenter sur sa nature bizarre, il trouva le moyen de la célébrer. En 1977, au cours d'un colloque de science-fiction, il tomba sur une femme vendant des boutons faits sur mesure. Excité, Stallman en commanda un avec l'inscription « Destituer Dieu ».

Pour Stallman, l'expression « destituer Dieu » fonctionnait à plusieurs niveaux. Athée depuis son enfance, Stallman vit cela comme une tentative d'ouverture d'un « second front » dans le débat en cours sur la religion. « À l'époque, tout le monde se demandait si Dieu était vivant ou mort », se souvient Stallman. « 'Destituer Dieu' abordait le problème sous un angle différent. Si Dieu était aussi puissant pour créer le monde et ensuite ne rien faire pour corriger les problèmes, pourquoi voudrions-nous adorer un tel Dieu? Ne serait-il pas mieux de le traduire en justice? »

En même temps, « destituer Dieu » était une boutade satirique sur l'Amérique et son système politique. Le scandale du Watergate des années 70 affecta profondément Stallman. Enfant, Stallman avait grandi en doutant de l'autorité. Maintenant adulte, sa défiance était consolidée par la culture de la communauté hacker du AI Lab. Pour les hackers, le Watergate n'était qu'une manifestation shakespearienne des luttes de pouvoir quotidiennes qui rendaient la vie si difficile à ceux ne bénéficiant d'aucun privilège. C'était une parabole excessive comparée à ce qui se passait quand le peuple échangeait la liberté et l'ouverture pour la sécurité et la commodité.

Remonté par une confiance grandissante, Stallman portait le bouton avec fierté. Les gens assez curieux pour le questionner sur le sujet recevaient tous le même laïus bien préparé : « Mon nom est Jéhovah », disait Stallman. « J'ai un plan précis pour sauver l'Univers, mais pour des raisons de sécurité divine, je ne peux vous dire de quoi il s'agit. Vous allez devoir mettre votre foi en moi car je vois le tableau, mais pas vous. Vous savez que je suis bon parce que je vous l'ai dit. Si vous ne me croyez pas, je vous mettrai sur ma liste d'ennemis et vous jetterai dans une fosse où l'Office du Revenu de l'Enfer vérifiera vos impôts pour l'éternité. »

Ceux qui interprétèrent le jeu comme une parodie littérale des audiences du Watergate ne comprirent que la moitié du message. Pour Stallman, l'autre moitié était quelque chose que seuls ses camarades hackers semblaient comprendre. Cent ans après que Lord Acton ait averti que le pouvoir absolu corrompait absolument, les américains semblaient avoir oublié la première partie du truisme d'Acton : le pouvoir lui-même corrompt. Plutôt que de souligner les multiples exemples de corruptions mineures, Stallman était satisfait d'exprimer son indignation envers un système entier qui, en premier lieu, faisait confiance au pouvoir.

« Je me suis dit, pourquoi m'arrêter au menu fretin », dit Stallman, se souvenant du bouton et de son message. « Si nous y allions contre Nixon, pourquoi ne pas y aller contre Monsieur Suprême [*Mr. Big*]. Telles que je vois les choses, tout être ayant le pouvoir et qui en abuse mérite qu'on le lui retire. »

Notes

1. *Richard Stallman: High School Misfit, Symbol of Free Software, MacArthur-certified Genius* (1999) de Michael Gross
2. Carmine DeSapio a la particularité suspecte d'être le premier patron italo-américain de Tammany Hall, la machine politique de New York. Pour plus d'informations sur DeSapio et la politique du New York d'après-guerre, voyez *Skinning the Tiger: Carmine DeSapio and the End of the Tammany Era,* - New York Affairs (1975):3:1.
3. Chess, un autre élève du SHP, décrit les protestations comme un « bruit de fond ». « Nous étions tous politisés », dit-il, « mais le SHP était important. Nous ne l'aurions jamais séché pour une manifestation. »
4. *Hackers* de Steven Levy — Penguin USA, 1984, p. 144. Levy y décrit en cinq pages environ la fascination qu'avait Gosper pour LIFE, un logiciel de jeu mathématique originellement créé par le mathématicien britannique John Conway. Je recommande très chaudement ce livre en tant que supplément, et peut-être même comme prérequis, à celui-ci.
5. Gerald Sussman, un membre de la faculté du MIT et hacker qui travailla au AI Lab avant Stallman,

conteste ce souvenir. Selon Sussman, les hackers ne cassaient jamais les portes pour libérer des terminaux.

6. Vous excuserez le trop bref résumé de la genèse de l'*ITS*, un système d'exploitation que bien des hackers considèrent toujours comme la quintessence de leur éthique. Pour en savoir plus sur la signification politique du logiciel, voyez *Architects of the Information Society: Thirty-Five Years of the Laboratory for Computer Science at MIT* (MIT Press, 1999) — Simson Garfinkel.
7. Cf note précédente.
8. *RMS Lecture at KTH (Sweden)*, (30 octobre 1986) — Richard Stallman http://www.gnu.org/philosophy/stallman-kth.html
9. Dans un courriel reçu peu après le dernier cycle d'édition du présent ouvrage, Stallman dit avoir trouvé également une partie de son inspiration politique au campus de Harvard. « Pendant ma première année à Harvard, dans un cours d'histoire chinoise, j'ai lu le récit de la première révolte contre la dynastie Chin », écrit-il. « Cette histoire n'est pas forcément crédible historiquement, mais c'était très émouvant. »
10. *Richard Stallman*(1986)
11. *Hackers* (Penguin USA, 1984) p. 417, de Steven Levy. J'ai modifié cette citation dont Levy utilise un extrait, pour illustrer plus directement comment le logiciel peut révéler la fausse sécurité du système. Levy utilise l'expression '[tel et tel]'
12. *Hackers* (Penguin USA, 1984) de Steven Levy, p. 417.

Chapitre V — Une flaque de liberté

Demandez à quelqu'un qui a passé plus d'une minute en présence de Richard Stallman et vous aurez la même impression : oubliez les cheveux longs. Oubliez l'attitude excentrique. La première chose que vous noterez est le regard. Un coup d'œil dans les yeux verts de Stallman et vous saurez que vous êtes en présence d'un vrai croyant.

Qualifier d'intense le regard de Stallman est un euphémisme. Les yeux de Stallman ne font pas que vous voir, ils regardent à travers vous. Même quand, momentanément, vos yeux s'égarent ailleurs par simple politesse, les yeux de Stallman restent fixés, comme deux faisceaux de photons crépitant sur votre visage.

C'est sans doute pour cela que l'on a souvent décrit Stallman en employant des qualificatifs à connotation religieuse. En 1998, dans un article de Salon.com intitulé *Le saint du logiciel libre*, Andrew Leonard décrit les yeux verts de Stallman comme « rayonnants du pouvoir d'un prophète de l'Ancien Testament [1] ». En 1999, un article du magazine *Wired* écrit que Stallman et sa barbe faisaient penser à Raspoutine [2] », tandis que le journal londonien *The Guardian* décrit son sourire comme celui « d'un disciple voyant Jésus [3] ».

De telles analogies faisaient mouche, mais rataient finalement la cible car elles ne parvenaient pas à rendre compte de la part vulnérable du personnage 'Stallman'. Observez le regard fixe de Stallman pendant un bon moment, et vous commencerez à noter des changements subtils. Ce qui semble au début une tentative d'intimider ou d'hypnotiser se révèle être, après le deuxième ou le troisième échange de regards, une tentative frustrée d'établir et maintenir le contact. Si, comme Stallman l'a supposé lui-même de temps en temps, sa personnalité est le produit de l'autisme ou du syndrome d'Asperger, ses yeux confirment certainement le diagnostic. Même à leur plus forte intensité, ils ont tendance à devenir nuageux et distants, comme les yeux d'un animal blessé se préparant à rendre l'âme.

Ma première rencontre personnelle avec le légendaire regard de Stallman remonte à mars 1999, au salon LinuxWorld de San Jose, en Californie. Étiquetée comme une « révélation » de la communauté Linux, cette convention était aussi reconnue comme l'événement ayant réintroduit Stallman dans la presse spécialisée. Déterminé à faire valoir sa véritable part de mérite, Stallman avait utilisé la convention pour enseigner aux spectateurs et journalistes rassemblés, l'histoire du Projet GNU et de ses objectifs ouvertement politiques.

En tant que journaliste envoyé pour couvrir l'évènement, je reçus ma propre documentation sur Stallman lors d'une conférence de presse annonçant la sortie de *GNOME 1.0*, une interface utilisateur graphique libre. Involontairement, j'appuyai sur toute une série de boutons d'alarme en lançant ma toute première question à Stallman lui-même : « Pensez-vous que la maturité de GNOME aura des conséquences sur la popularité commerciale du système d'exploitation Linux ? »

« Je vous demande s'il vous plaît d'arrêter d'appeler ce système d'exploitation 'Linux'», répondit Stallman, ses yeux zoomant immédiatement sur les miens. Le noyau Linux n'est qu'une petite partie du système d'exploitation. De nombreux programmes qui font partie du système d'exploitation que vous appelez Linux n'ont pas du tout été développés par Linus Torvalds. Ils ont été créés par les bénévoles du Projet GNU, contribuant au développement de ces programmes sur leur temps libre afin que les utilisateurs puissent avoir un système d'exploitation libre comme celui que nous avons aujourd'hui. Ne pas reconnaître les contributions de ces développeurs est, d'une part, impoli, et d'autre part, une mauvaise représentation de l'histoire. C'est pourquoi je vous demande, lorsque vous faites référence à ce système d'exploitation, de bien vouloir lui donner son véritable nom : GNU/Linux. »

Notant ces mots sur mon calepin, je remarquai un silence surnaturel dans la pièce où nous étions entassés. Quand je finis par lever les yeux, je trouvai ceux de Stallman qui m'attendaient, impassibles. Timidement, un second journaliste posa une question, en s'assurant d'utiliser le terme « GNU/Linux » au lieu de Linux. Miguel de Icaza, le leader du projet GNOME, répondit. Ce ne fut qu'à la moitié de la réponse d'Icaza que les yeux de Stallman se détachèrent des miens. A cette seconde, je sentis un léger frisson parcourir mon dos. Lorsque Stallman commença à faire la leçon à un autre journaliste sur une erreur de

diction, je sentis une pointe de soulagement culpabilisante. Au moins, il n'était pas en train de me regarder, m'étais-je dit.

Pour Stallman, ce genre de face-à-face servait à quelque chose. Vers la fin de la première convention LinuxWorld, la plupart des journalistes savaient qu'il ne fallait pas utiliser le terme « Linux » devant lui, et wired.com était prêt à publier une histoire comparant Stallman à un révolutionnaire pré-stalinien effacé des livres d'histoire par les hackers et les entrepreneurs qui désiraient minimiser l'importance des objectifs trop politisés du Projet GNU [4]. D'autres articles suivirent, et bien que peu de journalistes utilisaient l'appellation GNU/Linux dans leurs écrits, la plupart étaient prompts à reconnaître Stallman comme l'instigateur du mouvement pour créer un système d'exploitation libre, 15 ans plus tôt.

Je ne rencontrai Stallman que dix-sept mois plus tard. Pendant ce temps, Stallman visiterait la Silicon Valley une fois de plus en août pour la convention *1999 LinuxWorld Show*. Bien qu'il ne fût pas invité à donner un discours, Stallman réussit tout de même à laisser la plus belle phrase de l'événement. Alors qu'il acceptait, au nom de la *Free Software Foundation* (Fondation pour le logiciel Libre — FSF), le prix Linus Torvalds pour services rendus à la communauté — un prix qui empruntait son nom à celui du créateur de Linux —, Stallman ironisa : « Donner le prix 'Linus Torvalds' à la FSF est comme donner le prix 'Han Solo' à la Rébellion. »

Cette fois par contre, le commentaire ne trouva pas grand écho dans les médias. Au milieu de la semaine suivante, *Red Hat inc.*, un acteur proéminent de GNU/Linux, entra à la bourse. La nouvelle confirma ce que plusieurs journalistes tels que moi soupçonnaient déjà : « Linux » était sur toutes les lèvres de Wall Street, tout comme « commerce électronique » [*e-commerce*] et « point-com » l'étaient auparavant. Alors que le marché boursier abordait le passage à l'an 2000 comme une hyperbole approche son asymptote verticale, toute discussion considérant la question des logiciels libres ou de l' *open source* sous son aspect politique tomba dans l'oubli.

Peut-être est-ce pour cela que lorsque le LinuxWorld enchaîna ses deux premières conventions par une troisième en août 2000, Stallman y était manifestement absent.

Ma deuxième rencontre avec Stallman et son regard légendaire eut lieu peu après ce troisième LinuxWorld. Apprenant que Stallman allait être dans la Silicon Valley, j'organisai une entrevue au déjeuner à Palo Alto en Californie. L'endroit de la rencontre semblait ironique, non seulement à cause de son absence à la convention, mais aussi à cause de la localité. Non loin de Redmond, dans l'état de Washington, peu de villes offrent un tel témoignage à la valeur économique du logiciel propriétaire. Je m'y rendis en voiture depuis Oakland, curieux de savoir comment, ayant passé le plus clair de sa vie à combattre une culture possédant une prédilection pour l'avarice et l'égoïsme, Stallman se débrouillait dans une ville où même les petites maisonnettes valent autour d'un demi-million de dollars.

Je suivis les indications que Stallman m'avait fournies jusqu'à ce que j'arrive au bureau directeur de Art.net, une « association d'artistes virtuels » à but non lucratif. Située dans une maison bordée de haies dans un coin du nord de la ville, les installations de Art.net étaient agréablement vétustes. Tout à coup, l'idée de voir Stallman tapi au cœur de Silicon Valley ne sembla plus si étrange.

Je retrouvai Stallman assis dans une pièce sombre en train de taper sur son ordinateur portable gris. Il me fixa aussitôt que j'entrai dans la pièce, me donnant ainsi le plein feu de son regard de 200 watts. Alors qu'il m'offrit un réconfortant « bonjour » je lui retournai la pareille. Avant même que ne sortent les mots cependant, ses yeux avaient déjà retrouvé l'écran du portable.

« Je termine un article sur l'esprit du 'hacking' », dit Stallman, ses doigts toujours au travail. « Regardez ».

Je jetai un coup d'œil. La pièce était faiblement éclairée, et le texte d'un vert blanchi sur fond noir, l'inverse du jeu de couleurs que la plupart des logiciels de traitement de texte utilisent ; il fallut donc un moment pour que mes yeux s'y ajustent. Avant qu'ils ne le soient, j'étais en train de lire le compte rendu d'un récent dîner à un restaurant coréen. Avant le repas, Stallman fit une découverte intéressante : la personne responsable de mettre la table laissa six baguettes au lieu des deux habituelles devant la place de Stallman. Alors que la plupart auraient ignoré des paires supplémentaires, Stallman y releva un défi : trouver une façon d'utiliser les six baguettes à la fois. Comme beaucoup de logiciels bricolés par des hackers, la solution était

tout aussi brillante que ridicule à la fois. C'est ainsi qu'il décida d'utiliser cet exemple pour bien l'illustrer.

Alors que je lisais l'extrait, je sentis que Stallman me regardait, absorbé. Je levai la tête pour lire un demi sourire fier et enfantin sur son visage. Alors que je le complimentais pour son texte, mon commentaire suscita à peine mieux que la levée d'un sourcil.

« Je serai prêt à partir dans un instant », dit-il.

Stallman reprit le clavier de son portable. Sa machine était un bloc gris, tout le contraire des modèles élancés des portables récents qui semblaient être les préférés des programmeurs à la récente convention LinuxWorld. Au-dessus du clavier s'en trouvait un plus léger et plus petit, témoin des mains vieillissantes de Stallman. À la fin des années 80, lorsque Stallman mettait de 70 à 80 heures de travail par semaine pour écrire les premiers outils et logiciels libres pour le Projet GNU, la douleur aux mains devenait si insoutenable qu'il lui fallut engager un dactylographe. A ce moment là, Stallman s'en remettait à un clavier qui nécessite moins de pression qu'un clavier d'ordinateur typique.

Stallman tend à ignorer tout stimuli extérieur lorsqu'il travaille. À observer ses yeux rivés à l'écran alors que dansent ses doigts, on en venait rapidement à penser à deux amis de vieille date pris dans une conversation intense.

La séance se termina avec quelques grands coups sur le clavier et le lent démontage du portable.

« Prêt pour le dîner ? », demanda Stallman.

Nous marchâmes jusqu'à ma voiture. Se plaignant d'une entorse, Stallman suivit lentement en boitant. Il blâma la blessure d'un tendon de son pied gauche. La blessure avait trois ans et avait empiré à un point tel que Stallman, grand amateur de danse folklorique, avait été obligé d'abandonner toute activité de danse. « J'aime intrinsèquement la danse folklorique », se lamenta-t-il. « Ne plus pouvoir danser est une vraie tragédie pour moi. »

Le corps de Stallman témoignait de cette tragédie. Le manque d'exercice le laissait joufflu et ventru, un ventre qui se voyait moins l'année précédente. On pouvait voir que le gain de poids était dramatique car lorsque Stallman marchait, il courbait son dos tel une femme enceinte qui essaie de s'accommoder du poids supplémentaire.

La marche fut davantage retardée lorsque Stallman s'arrêta pour sentir le parfum des roses. Visant une floraison particulièrement belle, il chatouilla les pétales centrales de son prodigieux nez, inspira profondément et recula avec un soupir de satisfaction.

« Hum, *rhinophytophilia* [5] », dit-il en se frottant le dos.

Le trajet jusqu'au restaurant dura moins de trois minutes. À la recommandation de Tim Ney, ancien directeur exécutif de la FSF, je laissais à Stallman le choix du restaurant. Alors que certains journalistes étaient prompts à viser le style de vie monacal de Stallman, à la vérité, il était un véritable épicurien lorsqu'il s'agissait de nourriture. L'un des avantages d'être un missionnaire voyageant pour la cause du logiciel libre est d'avoir droit aux différents mets délicieux de partout à travers la planète. « Visitez pratiquement n'importe quelle ville d'importance au monde, et il est fort probable que Richard connaisse le meilleur restaurant en ville », raconta Ney. « Richard est très fier de reconnaître ce qu'il y a sur le menu et ainsi commander pour tous. »

Pour le repas de ce jour, Stallman avait choisi un restaurant cantonnais *dimsum* à deux coins de University Avenue, la principale artère de Palo Alto. Le choix était influencé en grande partie par le voyage récent en Chine de Stallman qui incluait une conférence dans la province de Guangdong, et par son aversion pour la cuisine hunanaise et sichuanaise, plus épicées. « Je ne suis pas un grand amateur de cuisine épicée », admit Stallman.

Nous arrivâmes quelques minutes après 11h00 et nous fûmes déjà sujets à une attente de vingt minutes. Connaissant l'aversion des hackers pour le temps perdu, je retins mon souffle un moment, craignant un éclat d'émotions. Contre toute attente, Stallman prit la nouvelle avec complaisance.

« C'est dommage que nous n'ayons pu trouver quelqu'un pour se joindre à nous », me dit-il. « C'est toujours plus agréable de manger avec un groupe de personnes. »

Pendant cette attente, Stallman s'exerça à quelques pas de danse. Ses gestes étaient habiles. Nous discutâmes de l'actualité. La seule chose que regretta Stallman, en ayant raté la conférence LinuxWorld, était d'avoir manqué le lancement de la *GNOME Foundation*. Soutenue par Sun Microsystems et IBM, cette fondation était pour Stallman une bonne illustration du fait que l'économie du logiciel libre et l'économie libérale de marché n'ont pas besoin d'être mutuellement exclusives. Néanmoins, Stallman demeure insatisfait du message qui en était ressorti.

« De la manière dont ce fut présenté, les compagnies parlaient de Linux sans jamais mentionner le Projet GNU. », dit-il.

Ce genre de déception ne faisait que contraster avec l'accueil chaleureux rencontré outremer, surtout en Asie, nota Stallman. Un survol rapide des voyages de Stallman en l'an 2000 relatait bien la popularité grandissante du message porté par le mouvement du logiciel libre. Entre ses visites en Inde, en Chine et au Brésil, Stallman avait passé 12 des 115 jours de ce périple sur le sol américain. Ses voyages lui donnèrent l'opportunité de voir comment le concept du logiciel libre se traduisait dans différentes langues et dans d'autres cultures.

« En Inde, beaucoup sont intéressés par les logiciels libres parce qu'ils y voient un moyen de construire une infrastructure informatique sans dépenser beaucoup », rapporta Stallman. « En Chine, le concept met du temps à s'enraciner. Comparer le logiciel libre à la liberté de parole est plus compliqué lorsqu'il n'y a justement pas de liberté de parole. Tout de même, le niveau d'intérêt porté au logiciel libre était profond lors de ma dernière visite. »

La conversation se tourna vers Napster, la compagnie de logiciels de San Mateo en Californie devenue une célébrité dans les médias ces derniers mois. La compagnie avait mis sur le marché un outil informatique controversé qui laissait les mordus de musique échantillonner et télécharger la musique d'autres mordus de musique. Grâce au pouvoir attractif de l'Internet, ce soi-disant logiciel pair-à-pair (*peer-to-peer* ou P2P) était devenu *de facto* un véritable juke-box en ligne, donnant le moyen à tout amateur de musique d'écouter de la musique en format MP3 sur son ordinateur sans payer de droits, au grand dam des compagnies de disques.

Bien que basé sur la plate-forme d'un logiciel propriétaire, le système Napster attestait un argument longtemps soutenu par Stallman, à savoir qu'une fois qu'une oeuvre se retrouvait sous forme digitale — en d'autres mots, lorsqu'il n'était plus question de dupliquer des sons ou des atomes mais de dupliquer de l'information — la naturelle propension humaine à partager une œuvre devenait plus difficile à restreindre. Au lieu d'imposer des restrictions supplémentaires, les dirigeants de Napster avaient plutôt décidé de tirer avantage de cette propension. En donnant aux amateurs de musique un lieu central pour échanger des fichiers musicaux, la compagnie avait misé sur sa capacité à diriger le trafic des usagers vers d'autres opportunités commerciales.

Le succès soudain du modèle de Napster sema la crainte chez les compagnies de disques traditionnelles, et pour cause : quelques jours seulement avant ma rencontre avec Stallman à Palo Alto, la juge Marilyn Patel de le *U.S. District Court* émit une ordonnance contre le service d'échange de fichiers à la demande de la *Recording Industry Association of America* (Association Américaine de l'Industrie du Disque — RIAA). Cette ordonnance fut par la suite suspendue par l' *U.S. Ninth District Court of Appeals* (Cour d'appel du 9ème district américain), mais vers le début de 2001 la cour d'appel trouva également la compagnie de San Mateo en bris avec la loi sur les droits d'auteur [6], une décision que la porte-parole de la RIAA, Hillary Rosen, déclara comme une « victoire pour la communauté créatrice et le commerce électronique licite [7]. »

Pour les hackers, tels Stallman, le modèle commercial de Napster était inquiétant sous plusieurs aspects. La volonté de la compagnie de s'approprier des principes hackers éprouvés comme le partage de fichiers et la propriété communautaire de l'information, alors que l'on vendait un service utilisant un logiciel propriétaire, renvoyait un message plutôt affligeant. Ayant déjà de la difficulté à passer son propre message bien articulé dans les médias, Stallman était plutôt -- on le comprend bien -- réticent quand vint le moment de parler de cette compagnie. Il admit tout de même avoir appris une chose ou deux du phénomène social Napster.

« Avant Napster, je croyais qu'il pouvait être acceptable pour les gens de ne distribuer qu'en privé des

œuvres du domaine du divertissement », dit Stallman. « Le nombre de gens qui trouvent Napster utile, par contre, me montre bien que le droit de redistribuer des copies non seulement de voisin à voisin mais bien auprès du grand public, est essentiel et ne peut donc être retiré. »

À peine eut-il fini cette phrase que la porte du restaurant s'ouvrit et nous fûmes invités à entrer par l'hôtesse. En quelques secondes, nous nous assîmes à côté d'un grand mur paré de miroirs dans un coin du restaurant.

Le menu était également un formulaire de commande, et Stallman en était à cocher les cases avant même que notre eau ne soit servie. « Rouleaux de crevettes frites enveloppés de tofu », fit Stallman à haute voix. « La texture du tofu est si intéressante. Je crois que nous devrions en commander. »

Ce dernier commentaire nous amena à une discussion sur la cuisine chinoise et sa récente visite en Chine. « La cuisine chinoise est absolument exquise », commenta Stallman, sa voix gagnant une pointe d'émotion pour la première fois depuis ce matin. « Il y a tellement de produits différents que je n'ai jamais vus aux États-Unis, des produits locaux faits de champignons et de légumes de la région. C'en était à un point tel que je tenais un journal pour retracer chaque repas merveilleux ! »

La conversation se tourna vers la cuisine coréenne. Durant le même mois de juin 2000 de sa tournée asiatique, Stallman alla visiter la Corée du Sud. Son arrivée enflamma les médias locaux en raison d'une conférence coréenne sur les logiciels à laquelle participait le fondateur de Microsoft, Bill Gates, cette même semaine. En dehors de sa photo au-dessus de celle de Gates sur la première page du plus grand journal de Séoul, Stallman rapporta que le summum du voyage était la nourriture. « J'avais un bol de *naeng myun*, qui sont des nouilles froides », dit-il. « Ces nouilles procurent une sensation intéressante. Dans la plupart des régions, on n'utilise pas tout à fait le même genre de nouilles pour le *naeng myun*, et je peux dire avec une certitude absolue que c'était le *naeng myun* le plus exquis que j'aie jamais goûté. »

Le terme « exquis » était toute une louange de la part de Stallman. Je le savais, car quelques instants après avoir entendu sa rhapsodie des *naeng myun*, je sentis les faisceaux lasers de ses yeux en train de brûler le dessus de mon épaule droite.

« Il y a une femme des plus exquises assise juste derrière vous », dit Stallman.

Je me tournai pour regarder, voyant en un clin d'œil le dos d'une femme. La dame était jeune, quelque part dans la mi-vingtaine, et portait une robe blanche cousue de paillettes. Elle et son compagnon de déjeuner étaient en train de finir de régler la note. Alors que les deux se levèrent de table pour quitter le restaurant, je pus me rendre compte, sans qu'il ne me vît, que ses yeux perdirent soudainement de leur intensité.

« Oh, non ! » dit-il. « Ils sont partis. Et dire que je ne la reverrai probablement jamais. »

Après un bref soupir, Stallman se recomposa. Cet instant me donna l'opportunité de discuter de sa réputation vis-à-vis du sexe opposé. C'était une réputation parfois contradictoire. Nombre de hackers rapportaient que Stallman accueillait les dames d'un baisemain [8]. Cependant, travaillant sur la connexion entre l'amour libre et le logiciel libre, la journaliste Annalee Newitz présente un Stallman rejetant les valeurs familiales traditionnelles. Dans un article du 26 mai 2000 paru dans Salon.com, elle dresse le portrait d'un hacker dans le genre Lothario, Stallman lui ayant affirmé : « Je crois en l'amour, mais pas en la monogamie »[9].

Le menu de Stallman s'abaissa un brin lorsque j'amenai le sujet. « Eh bien, La plupart des hommes semblent vouloir du sexe et ont une attitude plutôt méprisante envers les femmes », dit-il. « Même les femmes avec qui ils sont liés. Je ne comprends pas cela du tout. »

Je lui mentionnai le passage du livre de 1999, *Open Sources*, dans lequel il confessait avoir voulu nommer le défunt noyau GNU en l'honneur d'une amoureuse de l'époque. Elle s'appelait Alix, un nom qui cadrait parfaitement avec la convention des développeurs d'Unix voulant qu'un « x » termine le nom de tout noyau -- « Linux » par exemple. Comme la dame était une administratrice de système Unix, Stallman dit que cela aurait été un hommage plus touchant. Malheureusement, nota-t-il, le développeur principal renomma ce noyau HURD [10]. Bien que Stallman et son amoureuse rompirent, l'histoire instigue automatiquement une autre question : pour toutes les imageries médiatiques qui le dépeignent comme un fanatique au regard fou, Richard Stallman serait-il un grand romantique irrécupérable, un Don Quichotte chargeant sur des moulins à

vent corporatifs dans le dessein d'impressionner une éventuelle Dulcinée?

« Je n'essayais pas vraiment d'être romantique », dit-il, au souvenir de l'épisode d'Alix. « C'était plutôt une taquinerie. Je veux dire... c'était romantique, mais c'était aussi une taquinerie, vous voyez? C'eût été une belle surprise. »

Pour la première fois de toute la matinée, Stallman sourit. J'abordai le sujet du baisemain. « Oui, je le fais », répondit-il. « Je trouve que c'est une manière d'offrir de l'affection que bien des femmes vont apprécier. C'est l'occasion de donner de l'affection et d'en être apprécié. »

L'affection était quelque chose qui fuyait la vie de Richard Stallman, et il était péniblement sincère lorsque la question était soulevée. « Il n'y a vraiment pas eu beaucoup d'affection dans ma vie, sauf dans ma tête », dit-il. Dès lors, la discussion devint rapidement inconfortable. Après quelques réponses à un seul mot, Stallman finit par lever son menu, coupant court à l'interrogatoire.

« Voudriez-vous du *shimai* ? », demanda-t-il.

Lorsque le repas arriva, la conversation dura tout le temps des différents services. Nous discutions de l'affection notoire du hacker pour la cuisine chinoise, les excursions hebdomadaires dans le district de Chinatown de Boston pendant ses années de programmeur, embauché au AI Lab, et la logique sous-jacente des langues chinoises et de leurs systèmes d'écritures propres. Chacune de mes questions rencontrait une réponse bien informée de la part de Stallman.

« J'ai entendu des gens parler shanghaïen la dernière fois que j'étais en Chine », raconta-t-il. « C'était intéressant à entendre. C'était très différent [du mandarin]. Je leur ai fait dire des mots apparentés en mandarin et shanghaïen. Dans certains cas, vous voyez la ressemblance, mais une question qui me turlupinait était de savoir si l'intonation serait similaire. Elle ne l'est pas. Ça, ça m'intéresse parce qu'il y a une théorie qui dit que les tonalités ont évolué à la suite d'ajout de syllabes qui ont été perdues et remplacées. Les effets ont survécu dans les tons. Si cela est exact, et j'ai vu affirmer que cela s'est produit au cours de l'histoire, les dialectes doivent avoir divergé avant la perte de ces dernières syllabes. »

Le premier plat, une assiette de gâteaux aux navets sautés, arriva. Stallman et moi prenions un instant pour couper les larges rectangles qui sentaient le chou bouilli mais avaient le goût de beignets de pommes de terre frits dans du lard.

Je ramenai son exclusion sociale sur le plancher, en me demandant si l'adolescence de Stallman avait conditionné sa propension à prendre des positions impopulaires, surtout le combat qu'il mène depuis 1994 auprès des usagers informatiques pour qu'ils utilisent le terme « GNU/Linux » au lieu de « Linux ».

« Je crois que ça m'a aidé », répond-il, mâchant un *dumpling*. « Je n'ai jamais compris ce que la pression des pairs pouvait faire aux autres. Je pense que la raison est que je me sentais tellement rejeté que pour moi, il n'y avait rien à gagner à suivre les tendances. Ça n'aurait fait aucune différence. Je serais demeuré tout aussi exclu, alors je n'ai rien suivi. »

Stallman parla de ses goûts musicaux comme exemple de ses tendances anticonformistes. Adolescent, lorsque la plupart de ses collègues de classe écoutaient du *Motown* et de l' *acid rock*, Stallman préférait la musique classique. Le souvenir lui rappela un épisode comique de ses années estudiantines. Après le passage des Beatles à l'émission *Ed Sullivan Show* en 1964, la plupart des élèves s'étaient précipités pour acheter les plus récents singles et albums des Beatles. À ce moment précis, Stallman décida de boycotter les *Fab Four*.

« J'aimais quelques chansons populaires pré-Beatles», relata-t-il. « Mais je n'aimais pas les Beatles. Je détestais particulièrement la manière folle dont réagissaient les gens. C'était à qui aurait une panoplie Beatles pour les aduler le plus! »

Lorsque son boycott des Beatles n'eut prise sur personne, Stallman chercha d'autres moyens pour souligner la mentalité de troupeau de ses pairs. Stallman raconta avoir brièvement imaginé de créer un groupe musical dans le but de satiriser le groupe de Liverpool.

« Je voulais appeler ça *Tokyo Rose and the Japanese Beetles*. »

Étant donné son amour pour la musique folklorique internationale, je lui demandai s'il avait une

affinité similaire pour Bob Dylan et les autres musiciens folkloriques du début des années 60. Stallman hocha la tête. « J'aimais Peter, Paul et Mary », dit-il. « Ça me rappelle un grand 'filk'. »

Lorsque je lui demandai une définition du *filk*, Stallman m'expliqua le concept : un *filk*, dit-il, est une chanson populaire dont les paroles ont été remplacées par des paroles parodiées. Le processus d'écriture d'un *filk*, appelé « filking », était une activité populaire chez les hackers et aficionados de science-fiction. Parmi les *filks* classiques, on retrouvait *On Top of Spaghetti*, une révision de *On Top of Old Smokey*, et *Yoda* du maître *filk* Al Yankovic, sa chanson reprenant *Lola* des *Kinks* et revue à la sauce *Guerre des étoiles*.

Stallman me demanda si je serais intéressé d'entendre un *filk*. Sitôt que je dis oui, la voix étonnamment claire de Stallman entonne :

How much wood could a woodchuck chuck,
If a woodchuck could chuck wood?
How many poles could a polak lock,
If a polak could lock poles?
How many knees could a negro grow,
If a negro could grow knees?
The answer, my dear,
Is stick it in your ear.
The answer is to stick it in your ear...

[Combien de bois pourrait couper une marmotte,
Si une marmotte pouvait couper du bois ?
Combien d'armes pourrait croiser un polonais,
Si un polonais pouvait croiser les armes ?
Combien de genoux pourrait faire pousser un nègre,
Si un nègre pouvait faire pousser des genoux ?
La réponse, très cher(e),
C'est fous-toi le dans l'oreille.
La réponse, c'est de se le foutre dans l'oreille...]

La chanson se termina et les lèvres de Stallman se courbèrent en un demi-sourire presque enfantin. Je regardai les tables environnantes. Les familles asiatiques qui se régalaient de leur déjeuner du dimanche faisaient peu de cas de cet alto barbu qui se trouvait parmi eux.[1] Après un moment d'hésitation, je finis par sourire également.

« Voulez-vous cette dernière boulette de maïs? » demanda Stallman, les yeux étincelants. Avant même que je puisse briser son élan, Stallman s'en empara avec ses deux baguettes et la leva fièrement. « Peut-être devrais-je être celui qui prendra la boulette de maïs », poursuivit-il.

La nourriture disparue, notre conversation suivit la dynamique habituelle d'une entrevue. Stallman se cala bien dans sa chaise et prit sa tasse de thé. Nous reprîmes le sujet de Napster et sa relation avec le mouvement du logiciel libre. Les principes du logiciel libre devaient-ils s'appliquer à des domaines similaires telle que la publication musicale? demandai-je.

« C'est une erreur de transposer une réponse d'une chose à l'autre », répondit Stallman, différenciant les chansons et les logiciels. « La bonne approche c'est d'examiner chaque type d'œuvre et d'en tirer les conclusions que vous pouvez. »

Quand il s'agissait d'oeuvres protégées par les droits d'auteur, Stallman dit diviser le monde en trois catégories. La première suppose une œuvre « fonctionnelle » — par exemple les logiciels informatiques, les dictionnaires et les manuscrits. La seconde catégorie comprend les oeuvres pouvant être décrites comme des « témoignages » — comme des documents scientifiques ou historiques. De tels travaux ont une fonction qui pourrait être mise à mal si des auteurs autant que des lecteurs étaient libres de modifier l'oeuvre à volonté. La dernière catégorie comprend les oeuvres d'expression personnelle — journal intime, autobiographie. Modifier de tels documents reviendrait à changer les souvenirs d'une personne ou son point de vue, ce que Stallman considérait injustifiable d'un point de vue éthique.

De ces trois catégories, la première devrait conférer aux utilisateurs le droit illimité d'en faire des versions modifiées, alors que la seconde et la troisième devraient moduler ce droit en accord avec les volontés de son auteur original. Peu importe la catégorie cependant, la liberté de copier et redistribuer de manière non commerciale devrait s'appliquer intégralement et en tout temps, insista Stallman. Si cela signifie de laisser le droit aux internautes d'imprimer une centaine de copies d'un article, d'une image, d'une chanson ou d'un livre et ensuite distribuer par courriel les copies à une centaine d'étrangers, alors qu'il en soit ainsi. « Il est évident que la redistribution privée doit être permise, parce que seul un état policier peut arrêter cela», dit-il. « Il est antisocial de s'immiscer dans les relations entre les personnes et leurs amis. Napster m'a convaincu que nous avons de même besoin de permettre — nous devons permettre — la redistribution non commerciale au grand public pour l'unique plaisir de la chose : il y a tant de gens qui veulent le faire et trouvent cela si utile! »

Lorsque je demandai si les cours de justice accepteraient un aspect si permissif, Stallman m'interrompit.

« La question n'est pas la bonne», dit-il. « C'est-à-dire que vous changez complètement de sujet en passant du point de vue éthique à l'interprétation de la loi. Et ce sont deux questions totalement différentes dans ce même champ. Il est inutile de sauter de l'une à l'autre. Les cours interpréteraient les lois existantes essentiellement de manière impitoyable, parce que c'est ainsi que les lois ont été achetées par les éditeurs. »

Le commentaire offrait une perspective sur la philosophie politique de Stallman : bien que le système judiciaire supportait la capacité des entreprises à traiter les droits d'auteurs comme une version adaptée au logiciel d'un titre de propriété, cela ne signifiait pas que les usagers devraient jouer le jeu selon ces règles. La liberté était une question éthique, pas une question de loi. « Je pose un regard au-delà de ce que les lois existantes sont, et tourne ce regard vers ce qu'elles devraient être », dit-il. « Je n'essaie pas de faire une ébauche de la loi. Je me demande ce que devrait faire la loi. Je considère que la loi prohibant le partage de copies avec vos amis comme étant l'équivalent moral des lois *Jim Crow*. Cela ne mérite pas le respect. »

L'évocation de Jim Crow amène une autre question. Quelle influence ou inspiration lui laissaient les leaders politiques du passé? Comme le mouvement des droits civils des années 1950 et 1960, ses tentatives de motiver un changement social se basent sur un appel à des valeurs intemporelles : la liberté, la justice et le fair-play.

Stallman partagea son attention entre mon analogie et une mèche de cheveux particulièrement emmêlée. Quand je prolongeais l'analogie au point où je le comparais au Dr Martin Luther King Jr, Stallman m'interrompit après avoir détaché une mèche qu'il mit dans sa bouche.

« Je ne fais pas partie de sa ligue, mais je joue au même jeu », dit-il tout en mâchant.

Je suggérai Malcolm X pour un autre point de comparaison. Comme l'ancien porte-parole de *Nation of Islam*, Stallman avait cultivé cette réputation de courtisan de la controverse, aliénant des alliés potentiels et prêchant l'autosuffisance au-delà de l'intégration culturelle.

Mâchant une autre mèche, Stallman rejeta la comparaison. « Mon message est plus proche de celui de King », dit-il. « C'est un message universel. C'est un message d'une ferme condamnation de certaines pratiques qui maltraitent les autres. Ce n'est pas un message de haine pour quiconque. Et cela ne vise pas un petit groupe de personnes. J'invite tout le monde à valoriser la liberté et à l'obtenir. »

Malgré cela, une attitude suspicieuse envers les alliances politiques demeurait un trait de caractère fondamental de Stallman. Concernant son dégoût notoire du terme *open source*, sa réticence à participer à des projets récents pour former des coalitions semblait compréhensible. Ayant passé les deux dernières décennies à faire campagne au nom du logiciel libre, le capital politique de Stallman semblait très investi dans ce sens. Toujours était-il que des commentaires ironiques du genre de celui d'« Han Solo » en 1999 lors du LinuxWorld n'avaient fait que renforcer sa réputation dans l'industrie du logiciel qui le dépeignait comme un conservateur à tout crin qui se refusait à suivre toute tendance politique ou du marché.

« J'admire et respecte Richard pour tout le travail qu'il a accompli », confia Robert Young, président de la compagnie Red Hat, résumant la nature politique paradoxale de Stallman. « Ma seule critique est que parfois Richard traite ses amis de pire manière que ses ennemis. »

Son refus de chercher des alliances rendait tout aussi perplexe lorsque l'on considérait les intérêts politiques de Stallman en dehors du mouvement du logiciel libre. Visitez ses bureaux au MIT, et vous trouverez aussitôt un magasin de liquidation d'articles de journaux de gauche couvrant les abus des droits civils à travers le globe. Visitez son site web, et vous y trouverez des diatribes sur le *Digital Millenium Copyright Act* [loi sur les droits d'auteur du millénaire numérique], la guerre des drogues et l'Organisation Mondiale du Commerce.

Considérant son penchant pour l'activisme, je lui demandai pourquoi il n'avait pas cherché une voix plus puissante? Pourquoi n'avait-t-il pas utilisé sa vitrine dans le monde hacker pour donner plus de poids à sa voix politique plutôt que de la réduire.

Stallman laissa ses cheveux emmêlés tomber et considéra la question un moment.

« J'hésite à exagérer l'importance de cette petite flaque de liberté », répondit-il, « parce qu'il est très important de travailler à l'amélioration de la société et à la liberté dans des domaines plus connus et conventionnels. Je n'oserais dire que le logiciel libre est aussi important que ces domaines. C'est la responsabilité que j'ai prise parce que ça m'est tombé dans les bras et que j'y ai vu une façon de faire quelque chose. Mais, par exemple : mettre fin à la brutalité policière, gagner la guerre contre la drogue, mettre un terme aux racismes de tout poil toujours d'actualité, aider les gens à vivre une vie plus confortable, protéger les droits de ceux qui ont recours à l'avortement, nous protéger de la théocratie, ce sont toutes des causes importantes, beaucoup plus que ce que je fais. J'aurais aimé savoir quoi faire pour y prendre part. »

À nouveau, Stallman présentait son activité politique à la mesure de la confiance qu'il s'accordait à lui-même. Compte-tenu du temps qu'il lui avait fallu pour développer les principes fondamentaux du mouvement du logiciel libre, Stallman hésitait à s'embarquer pour une cause ou suivre une mode qui pouvaient l'emmener vers des terrains inconnus.

« J'aurais aimé savoir comment contribuer efficacement à ces causes plus importantes : j'en serais très fier si je le pouvais, mais elles sont très difficiles et bien des gens probablement meilleurs que moi y ont travaillé et y ont peu accompli », dit-il. « Mais de mon point de vue, pendant que certains se défendent contre ces grandes menaces visibles, j'en ai vu une autre qui restait délaissée. Alors je suis allé me battre contre elle. Elle n'est pas aussi grande, mais j'étais le seul à en découdre. »

Mâchant une dernière mèche de cheveux, Stallman proposa qu'on paie l'addition. Toutefois, avant que le serveur ne la retirât, Stallman sortit un billet de couleur blanche et le jeta par-dessus la pile. Le billet était de toute évidence contrefait, et je ne pus m'empêcher de l'examiner. Assurément, c'était un billet de contrefaçon. Au lieu d'arborer l'image de George Washington ou d'Abraham Lincoln, le côté frontispice du billet arborait l'image d'un cochon en caricature. Au lieu de la mention « États-Unis d'Amérique », la bannière au-dessus du cochon se lisait « États-Unis d'Avarice ». Le billet était de zéro dollar, et lorsque le serveur prit l'argent, Stallman prit bien soin de lui tirer sa manche.

« J'ai ajouté un zéro d'extra pour votre pourboire », dit Stallman, encore ce demi-sourire sur ses lèvres.

Le serveur, interdit ou troublé par l'allure du billet, sourit et repartit.

« Je crois que cela veut dire que nous sommes libres de partir », conclut-il.

Notes

1. Andrew Leonard, *The Saint of Free Software* [Le saint du logiciel libre] Salon.com (août 1998)
2. Leander Kahney, *Linux's Forgotten Man* [L'Homme oublié de Linux], Wired News (5 mars 1999)
3. *Programmer on moral high ground; Free software is a moral issue for Richard Stallman believes in freedom and free software* [Un programmeur aux valeurs morales élevées ; le logiciel libre est un sujet d'ordre moral car Richard Stallman croit à la liberté et au logiciel libre], London Guardian (6 novembre 1999). Ce ne sont que quelques extraits de nombreuses comparaisons religieuses. À cette date, la comparaison la plus extrême doit être créditée à Linus Torvalds qui, dans son autobiographie, écrit : « Richard Stallman est le dieu du logiciel libre » — Cf. Linus Torvalds et David Diamond, *Just for Fun: The Story of an Accidental Revolutionary*, Harper Collins Publisher Inc, 2001, p. 58 [*Il était une fois Linux. L'extraordinaire histoire d'une révolution accidentelle*, Paris, Eyrolles, 2001].

Une mention honorable ira à Larry Lessig qui, dans une description de Richard Stallman en pied de page, — Larry Lessig, *The Future of Ideas*, Random House, 2001, p. 270 — le compare à Moïse :

> [...] tel Moïse, il y eut un autre leader, Linus Torvalds, qui finalement emmena le mouvement vers la terre promise en facilitant le développement de la dernière partie de ce casse-tête qu'était le système d'exploitation. Tel Moïse, également, Stallman est tout autant respecté que vivement critiqué par les alliés du mouvement. Il ne pardonne pas, et il est de surcroît une inspiration pour certains, un leader d'importance critique pour la culture moderne. J'ai un profond respect pour les principes et l'engagement de cet individu extraordinaire, bien que j'aie aussi un grand respect pour ceux qui ont assez de courage pour mettre en doute ses préceptes et ensuite en subir sa colère.

Lors d'une dernière entrevue avec Stallman, je lui demandai ce qu'il pensait des comparaisons religieuses. « Certaines personnes me comparent en effet à un vieux prophète de l'Ancien Testament, et la raison en est que ces prophètes dénonçaient certaines pratiques sociales inadéquates. Ils ne faisaient aucun compromis à la moralité. Ils ne pouvaient être achetés, et ils étaient habituellement méprisés. »

4. Cf note 2.
5. À cet instant, je croyais que Stallman s'en référait au nom scientifique de la fleur. Des mois plus tard, j'apprenais que *rhinophytophilia* est en fait une référence humoristique à l'activité sexuelle, c'est-à-dire Stallman mettant son nez dans une fleur et savourant cet instant. Pour une autre histoire de fleur, tout aussi comique, voyez : http://www.stallman.org/texas.html.
6. Cecily Barnes et Scott Ard, *Court Grants Stay of Napster Injunction*, [La justice entérine l'injonction contre Napster] News.com (28 juillet 2000). http://news.cnet.com/news/0-1005-200-2376465.html
7. *A Clear Victory for Recording Industry in Napster Case*, [Une nette victoire pour l'industrie du disque dans l'affaire Napster] communiqué de presse (12 février 2001) de la RIAA. http://www.riaa.com/PR_story.cfm?id=372
8. Mae Ling Mak, *A Mae Ling Story* [Une histoire de Mae Ling](17 décembre 1998). http://www.crackmonkey.org/pipermail/crackmonkey/1998q4/003006.html. À cette date, Mak est la seule personne que j'aie trouvée qui veuille être citée à l'égard de cette pratique, bien que j'aie également entendu la même chose de sources féminines différentes. Mak, en dépit de son aversion à l'exprimer, s'est tout de même résolue à mettre ses appréhensions de côté et a ainsi dansé avec Stallman lors de la convention LinuxWorld de 1999. http://www.linux.com/interact/potd.phtml?potd_id=44.
9. Annalee Newitz, *If Code is Free Why Not Me?* [Si le code (informatique) est libre, pourquoi pas moi?] Salon.com (26 mai 2000). http://www.salon.com/tech/feature/2000/05/26/free_love/print.html.
10. Richard Stallman, *The GNU Operating System and the Free Software Movement*, [Le système d'exploitation GNU et le mouvement du logiciel libre] Open Sources (O'Reilly & Associates, Inc., 1999) p. 65
11. Pour d'autres *filks* de Stallman, visitez http://www.stallman.org/doggerel.html. Pour entendre Stallman chanter la chanson *The Free Software Song* [La chanson du logiciel libre] visitez http://www.gnu.org/music/free-software-song.html

Chapitre VI — La Commune d'Emacs

Le AI Lab des années 1970 était un endroit unique à tous les niveaux. Des projets d'avant-garde et des chercheurs de haut niveau lui donnaient une position hautement appréciée dans le monde des sciences informatiques. La culture hacker interne et sa politique anarchique lui conférèrent aussi une dimension rebelle mythique. C'est seulement plus tard, après que les scientifiques et superstars du logiciel eurent quitté l'endroit, que les hackers prirent pleinement conscience du monde unique et éphémère où ils vécurent autrefois.

« C'était un peu comme le jardin d'Éden », raconte Stallman dans un article de *Forbes* en 1998 pour résumer le laboratoire et sa philosophie de partage de logiciel . « Jamais nous n'avions pensé à ne pas coopérer »[1].

De telles analogies d'ordre mythique, bien qu'extrêmes, soulignent un fait important : le neuvième étage du 545 Tech Square était, pour beaucoup, plus qu'un lieu de travail. Pour les hackers comme Stallman, c'était un chez-soi.

Le mot « chez-soi » est un terme bien pesé dans le lexique de Stallman. Par moquerie à l'encontre de ses parents, Stallman, à ce jour, se refuse à nommer chez-soi tout autre domicile avant celui de *Currier House*, le dortoir où il vécut pendant son temps à Harvard. Notable est sa façon de décrire son départ de l'endroit en des termes tragi-comiques. Une fois, alors qu'il retraçait ses années là-bas, Stallman raconta que son seul regret était d'avoir été mis à la porte. Ce n'est que lorsque je lui demandai ce qui précipita ce départ que je réalisai que j'étais tombé dans un piège typique de Stallman.

« À Harvard, il ont cette politique de vous inviter à partir si vous réussissez trop de cours », répondit Stallman.

Sans dortoir ni désir de retourner à New York, Stallman suivit le chemin tracé par Greenblatt, Gosper, Sussman et bien d'autres hackers avant lui. S'inscrivant au MIT en tant qu'étudiant diplômé, Stallman loua un appartement près de Cambridge, mais considéra rapidement le AI Lab lui-même, comme son réel domicile. Lors d'un discours en 1986, Stallman évoqua ses souvenirs de cette époque :

> J'ai peut-être habité au laboratoire plus souvent que d'autres, parce que chaque année ou deux, pour une raison quelconque, j'étais sans domicile et je passais donc quelques mois à y vivre. Et je l'ai toujours trouvé très confortable, ainsi qu'agréable et frais durant l'été. Mais il n'était pas inhabituel d'y voir des gens s'endormir, encore une fois, à cause de leur enthousiasme : vous restez éveillé aussi longtemps que possible à programmer, parce que vous ne voulez pas vous arrêter. Et, lorsque vous êtes complètement épuisé, vous vous hissez sur la plus proche surface horizontale douillette. Une atmosphère très décontractée.[2]

L'atmosphère familière du lieu pouvait poser problème à l'occasion. Ce que certains voyaient comme un dortoir, d'autres le considéraient comme une fumerie électronique d'opium. Dans le livre de 1976 « Computer Power and Human Reason » [le pouvoir informatique et la raison humaine], le chercheur Joseph Weizenbaum offre une critique cinglante de l'informaticien-clochard [*computer bum*], le terme qu'il utilisait pour les hackers peuplant les salles d'informatique comme le AI Lab. « Leurs vêtements défraîchis, les cheveux malpropres et leurs visages barbus, ainsi que leurs coiffures défaites témoignent qu'ils sont oublieux de leur corps et du monde dans lequel ils évoluent », écrit Weizenbaum. « [Les informaticiens-clochards] n'existent, du moins engagés à ce point, que pour et par l'ordinateur ».[3]

Près d'un quart de siècle après cette publication, Stallman se hérisse lorsqu'il entend la description de cet « informaticien-clochard » par Weizenbaum, en parlant alors au présent comme si Weizenbaum lui-même était dans la pièce. « Il veut que les gens ne soient que des professionnels, le faisant pour l'argent et voulant ensuite s'éloigner et oublier dès que possible », raconte Stallman. « Ce qu'il voit comme un état normal des choses, je le vois comme une tragédie. »

La vie de hacker, cependant, n'était pas sans tragédie. Stallman voit sa transition de hacker du dimanche à celle d'habitant permanent du AI Lab comme une série d'infortunes ne pouvant être soulagées

que dans l'euphorie de la programmation. Comme il le dit lui-même, sa première malchance fut son diplôme de Harvard. Déterminé à continuer ses études de physique, il s'inscrivit alors au second cycle du MIT. Le choix de l'école était tout naturel. Non seulement cela lui fournit l'opportunité de suivre les pas des étudiants renommés du MIT : William Shockley (1936), Richard P. Feynman (1939), et Murray Gell-Mann (1951), mais cela le rapprocha aussi de deux miles du AI Lab et de son nouvel ordinateur, le PDP-10. « Mon attention se tournait vers la programmation, mais je continuais à penser que, peut-être, je pourrais faire les deux », confie-t-il.

Travaillant dur de jour dans les domaines scientifiques du second cycle universitaire, et programmant le soir dans les confins monastiques du AI Lab, Stallman essaya de maintenir un parfait équilibre. L'évasion de ce jeu de bascule était sa séance hebdomadaire avec la troupe de danse folklorique, sa seule sortie sociale lui garantissant au moins un minimum d'interaction avec le sexe opposé. Mais vers la fin de cette première année au MIT, un désastre se produisit. Une blessure au genou le força à abandonner la troupe. D'abord, Stallman vit cela comme un problème temporaire, et consacra le temps qu'il aurait passé à danser, ainsi libéré, à travailler davantage au AI Lab. À la fin de l'été, alors que son genou lui faisait toujours mal et que les cours recommençaient, il commença à s'inquiéter. « L'état de mon genou ne s'améliorait pas », se souvient-il, « ce qui voulait dire que je devais complètement arrêter la danse. J'étais désespéré. »

Sans dortoir ni danse, l'univers social de Stallman venait d'imploser. Tel un astronaute subissant les effets de l'apesanteur, il réalisa que sa capacité à interagir avec des non-hackers, en particulier les non-hackers féminins, s'était manifestement dégradée. Après seize semaines dans l'AI Lab, la confiance construite tranquillement durant ses 4 ans à Harvard s'était pratiquement volatilisée.

« Je sentais que j'avais perdu toute mon énergie », se souvient Stallman. « J'avais perdu l'énergie de vouloir faire quoi que ce soit hormis ce qui était tentant dans l'immédiat. L'énergie de vouloir faire autre chose était annihilée. J'étais au désespoir total. »

Stallman se retira encore plus du monde, consacrant entièrement son énergie à son travail au AI Lab. En octobre 1975, il abandonna ses études au MIT pour ne plus y revenir. La programmation, autrefois un passe-temps, était devenue sa vocation.

Rétrospectivement, Stallman juge inévitable cette transition d'étudiant à hacker. Tôt ou tard, croit-il, l'appel des sirènes de la programmation aurait été plus puissant que ses autres intérêts professionnels. « Avec la physique et les mathématiques, je ne pouvais jamais trouver le moyen de contribuer », dit-il en se souvenant de ses combats avant sa blessure au genou. « J'aurais bien été fier d'avancer dans l'un ou l'autre de ces domaines, mais je ne voyais jamais comment. Je ne savais pas par où commencer. Avec les logiciels, je voyais immédiatement comment écrire des programmes qui fonctionneraient et seraient utiles. Le plaisir que me procurait ce savoir m'amena à vouloir en faire plus. »

Stallman n'était pas le premier à assimiler programmation et plaisir. Beaucoup de hackers employés au AI Lab possédaient un tel curriculum universitaire incomplet. La plupart venaient compléter un diplôme en mathématiques ou en génie électrique et finalement troquaient carrière universitaire et ambition professionnelle pour l'exaltation pure et simple accompagnant la résolution de problèmes jusque-là inédits. Comme Saint Thomas d'Aquin, scolastique réputé pour avoir travaillé si longuement à son traité théologique qu'il en avait des visions spirituelles, les hackers parvenaient à des états transcendants à force de concentration mentale et d'épuisement physique. Bien que Stallman évitât les drogues, comme la majorité des hackers, il appréciait cette euphorie qui venait vers la fin d'une session de programmation de vingt heures.

Cependant, l'émotion la plus agréable était probablement la sensation d'accomplissement personnel. S'agissant de programmation, c'était l'élément naturel de Stallman. Une enfance remplie de longues soirées d'études lui a donné la faculté de travailler de longues heures avec peu de sommeil. Désadapté social depuis l'âge de 10 ans, il avait peu de difficultés à oeuvrer seul. De plus, en tant que mathématicien doté d'un don pour la logique et la prévoyance, il avait cette capacité à contourner les difficultés de conception, laissant la majorité des hackers tourner en rond.

« Il était particulier », se souvient Gerald Sussman, enseignant au MIT et ancien chercheur au AI Lab.. Le décrivant comme « un penseur clair et un concepteur clair », Sussman employa Stallman en qualité d'assistant de projet de recherche à partir de 1975. Le projet était complexe, nécessitant la création d'un programme d'intelligence artificielle qui pourrait analyser des diagrammes de circuits. Non seulement fallait-

il un expert en commandes Lisp, un langage de programmation construit spécifiquement pour des applications d'I.A., mais également comprendre comment un être humain aborderait la même tâche.

Lorsqu'il ne travaillait pas à des études officielles comme le programme d'analyse de circuit automatisé de Sussman, Stallman consacrait son temps à d'autres projets préférés. C'était l'intérêt majeur d'un hacker d'améliorer les infrastructures logicielles du laboratoire, et l'un de ses plus gros sujets favoris à cette époque était l'éditeur logiciel TECO.

L'histoire du travail de Stallman sur TECO est intimement liée à son futur leadership du mouvement du logiciel libre. C'est également un stade important dans l'histoire de l'évolution informatique, à un tel point qu'un bref résumé de celle-ci s'impose. Durant les années cinquante et soixante, alors que les ordinateurs commençaient à apparaître dans les universités, la programmation informatique était une procédure incroyablement abstraite. Pour communiquer avec la machine, les programmeurs créaient une série de cartes perforées où chacune représentait une commande logicielle unique. Ces derniers passaient ensuite ces cartes à un administrateur du système central qui les insérait une à une dans la machine, attendant qu'en ressorte une nouvelle série que le programmeur déchiffrait alors comme données de sortie. Cette procédure appelée « traitement par lots » (batch processing) était lourde et très longue. Elle était aussi sujette aux abus de pouvoir. Un des facteurs instinctif motivant l'aversion des hackers envers la centralisation était le pouvoir détenu par les opérateurs système qui dictaient l'ordre de priorité des lots à exécuter.

En 1962, les informaticiens et hackers participant au projet MAC du MIT, un précurseur du AI Lab, prirent des mesures pour alléger cette frustration. Le partage du temps, originellement appelé « vol de temps », donnait la possibilité à de multiples programmes de profiter des capacités opératoires de la machine. Les interfaces télétypes rendaient possible la communication avec un ordinateur, non pas avec des séries de cartes perforées, mais avec du texte. Un programmeur tapait les commandes et lisait ligne par ligne le résultat généré par la machine.

À la fin des années soixante, la conception des interfaces fit de grands bonds en avant. Lors d'une célèbre conférence en 1968, Doug Engelbart, scientifique travaillant alors au Stanford Research Institute, dévoila un prototype d'interface graphique moderne. Branchant un téléviseur à l'ordinateur ainsi qu'un dispositif de pointage qu'Englebart appelait une "souris", le scientifique créa un système encore plus interactif que celui à temps partagé développé au MIT. En utilisant l'affichage vidéo comme une imprimante très rapide, le procédé d'Engelbart permit à l'utilisateur de déplacer le curseur à l'écran et voir sa position mise à jour par l'ordinateur en temps réel. L'usager avait tout à coup la possibilité de placer du texte n'importe où à l'écran.

De telles innovations mettraient deux décennies supplémentaires avant de se retrouver sur le marché. Bientôt, dans les années 1970, les écrans vidéo allaient commencer à remplacer les télétypes comme terminaux d'affichage, créant ainsi le potentiel d'une édition en plein écran (par opposition au ligne par ligne).

Un des premiers programmes à profiter de l'édition à l'écran était le TECO du AI Lab, Acronyme de *Text Editor and COrrector* [éditeur et correcteur de texte], le programme fut optimisé par des hackers à partir du vieil éditeur télétype de la machine PDP-6.[4]

TECO représentait une amélioration substantielle des vieux éditeurs, mais avait encore ses inconvénients. Pour créer et éditer un document, un programmeur devait entrer une série de commandes logicielles spécifiant chaque édition. C'était un procédé abstrait. Contrairement aux traitements de texte modernes, qui mettent à jour le texte à chaque frappe au clavier, TECO demandait à l'utilisateur d'entrer une grande série d'instructions d'édition suivie d'une séquence de « fin de commande » simplement pour changer le texte. Avec le temps, un hacker devenait suffisamment habile pour écrire un document entier en mode d'édition, mais comme Stallman le soulignerait plus tard, la procédure nécessitait « une faculté mentale équivalente à jouer aux échecs les yeux bandés ».[5]

Pour faciliter la procédure, les hackers du AI Lab avaient élaboré un système qui affichait les modes « source » et « affichage » sur un écran divisé. En dépit de cette innovation, passer d'un mode à l'autre était tout de même fastidieux.

TECO n'était pas le seul éditeur en mode plein écran existant dans le monde de l'informatique à cette

époque. Pendant une visite au laboratoire d'intelligence artificielle de Stanford (le *Stanford Artificial Intelligence Lab*) en 1976, Stallman découvrit un programme d'édition appelé E. Le logiciel contenait une fonction interne donnant la possibilité à son utilisateur de mettre l'affichage à jour à chaque frappe d'une commande. Dans le langage informatique des années 1970, *E* était un des premiers logiciels rudimentaires d'édition WYSIWYG. Contraction de « ce que vous voyez est ce que vous obtenez » (*What You See Is What You Get*), WYSIWYG signifiait qu'un usager pouvait manipuler le fichier en naviguant dans le texte à l'écran, contrairement au travail avec un logiciel d'édition en arrière-plan.[6]

Impressionné par cette trouvaille, Stallman chercha une manière identique d'améliorer les fonctionnalités de TECO à son retour au MIT. Il trouva une fonction dénommée *Control-R*, écrite par Carl Mikkelson et nommée d'après la combinaison de clés qui la déclenchait. Le bidouillage de Mikkelson fit passer TECO de son mode d'exécution abstrait à un mode plus intuitif de touche par touche. Stallman améliora cette fonction de manière subtile mais significative. Il rendit d'autres commandes TECO, ou macros, accessibles par de nouvelles combinaisons à deux clés. Alors que les usagers entraient une série de commandes qui s'évaporaient de suite, le hack de Stallman donnait la possibilité de sauvegarder des macros sur un fichier et ensuite les solliciter à nouveau et à volonté. Le bidouillage de Mikkelson éleva TECO au rang d'éditeur WYSIWYG. « Ç'était une réelle avancée », relate Guy Steele, hacker au AI Lab à cette période.[7]

Au souvenir de Stallman, ce hack de macro déclencha une véritable explosion d'innovations. « Chacun écrivait sa propre collection de commandes d'édition à l'écran, une commande pour tout ce qu'il aimait normalement faire », se souviendra-t-il plus tard. « Les gens se les passaient et les amélioraient, les rendant plus puissantes et plus générales. Les séries de redéfinitions devinrent elles-mêmes peu à peu des programmes du système d'exploitation.[8] »

Tant de personnes trouvèrent l'innovation de la macro utile qu'ils l'incorporèrent à leurs propres programmes TECO et l'éditeur TECO devint secondaire devant la passion des macros qu'il avait inspiré. « Nous commencions à le classer mentalement comme langage de programmation plutôt qu'éditeur », raconte Stallman. Les utilisateurs ressentaient du plaisir à expérimenter leurs propres améliorations du logiciel et échanger de nouvelles idées.[9]

Deux ans après l'explosion, le taux d'innovations commençait à montrer des effets secondaires dangereux. La croissance explosive avait confirmé l'approche collaboratrice des hackers, suscitant l'enthousiasme, mais elle avait aussi conduit à un excès de complexité. « Nous avions un effet Tour de Babel », raconte Guy Steele.

L'effet menaçait de tuer l'esprit qui l'avait animé, poursuit Steele. Les hackers avaient conçu l'ITS pour faciliter l'aptitude des programmeurs à partager les connaissances et améliorer le travail de chacun. Cela signifiait pouvoir s'asseoir au bureau d'un autre programmeur, ouvrir son travail et faire commentaires et modifications directement dans le programme. « Parfois, la manière la plus simple de montrer à quelqu'un comment programmer ou déboguer était simplement de s'asseoir au terminal et de le faire à sa place », explique Steele.

La fonction de macro, après sa deuxième année, commençait à miner cette capacité. Avec leur volonté d'utiliser les nouvelles fonctionnalités du mode plein écran, les hackers avaient transformé leurs versions de TECO au point qu'en utilisant le terminal d'un autre, on devait habituellement passer la première heure à trouver ce que faisait telle ou telle macro.

Frustré, Steele pris sur lui de régler le problème. Il rassembla les quatre ensembles de macros et établit un diagramme des commandes les plus utiles. Alors qu'il implémentait le concept spécifié par ce diagramme, Steele dit qu'il attira l'attention de Stallman.

« Il commençait à regarder par-dessus mon épaule et demandait ce que je fabriquais », se souvient Steele.

Pour Steele, hacker discret qui interagissait rarement avec Stallman, ce souvenir se détache toujours. Au A.I.Lab., qu'un hacker regarde par-dessus l'épaule d'un autre pendant son travail était chose commune. Stallman, mainteneur TECO au laboratoire, jugea le travail de Steele « intéressant » et se mit rapidement à l'ouvrage pour l'achever.

« Comme j'aime le souligner, j'ai complété le premier 0,001 pourcent de la mise en oeuvre et Stallman fit le reste », dit Steele en riant.

Le nouveau nom du projet, Emacs, vint de l'aimable autorisation de Stallman. Acronyme de "édition de macros" [*editing macros*], il signifiait la transcendance de l'évolution qui avait eu lieu durant l'explosion des macros deux ans auparavant. Il profitait aussi d'une faille présente dans le lexique informatique. Notant le manque de programme de l'ITS commençant avec la lettre "e", Stallman choisit Emacs rendant ainsi possible sa référence avec une seule lettre. Encore une fois, le goût du hacker pour l'efficacité laissa sa marque.[10]

En développant un système standardisé de commandes macros, Stallman et Steele durent traverser une corde raide politique. Créant un programme standard, Stallman violait un principe hacker fondamental : « promouvoir la décentralisation ». Il menaçait également d'entraver la grande flexibilité qui avait alimenté en premier lieu l'explosion des innovations de TECO.

« D'un côté, nous tentions de créer encore un ensemble de commandes uniforme, de l'autre, nous voulions le garder entièrement ouvert parce que la capacité de programmation était importante », se souvient Steele.

Pour résoudre le problème, Stallman, Steele et les collègues hackers Dan Moon et Dan Weinreb limitèrent leurs efforts de standardisation aux commandes WYSIWYG qui contrôlaient l'apparence du texte à l'écran. Le reste de l'effort Emacs serait dédié à conserver son extensibilité de style 'jeu expérimental'.

Stallman se trouvait maintenant devant une énigme : si les usagers faisaient des changements sans les communiquer au reste de la communauté, l'effet Tour de Babel apparaîtrait ailleurs. S'en remettant à la doctrine hacker de partage d'innovation, Stallman incorpora un message dans le code source pour établir les conditions d'utilisation. Les utilisateurs étaient libres de modifier et redistribuer le code à la condition qu'ils donnent en retour les extensions qu'ils écrivaient. Stallman appela cela la "Mise en commun Emacs" (*Emacs Commune*). Tout comme TECO était devenu plus qu'un simple éditeur, Emacs était devenu plus qu'un simple logiciel. Pour lui, il s'agissait d'un contrat social. Dans une première note documentant le projet, Stallman décrit les termes du contrat. « EMACS », écrit-il, « a été distribué sur la base du partage communautaire, ce qui signifie que toute amélioration doit m'être retournée pour être incorporée et redistribuée.[11] »

Tout le monde n'accepta pas le contrat. L'innovation explosive continua au cours de la décennie, résultant en une foule de programmes identiques à Emacs avec différents degrés d'inter-compatibilité. Certains évoquaient leur relation avec l'Emacs original de Stallman au moyen d'appellations récursives humoristiques : SINE (Sine Is Not Emacs), Eine (Eine is not Emacs) et Zwei (Zwei Was Eine Initially). En tant que chef de file fidèle à l'éthique hacker, Stallman ne vit pas de raisons de stopper cette innovation par un harcèlement judiciaire. Néanmoins, le fait que quelques personnes prennent si avidement un logiciel du dépôt communautaire, le modifient et flanquent un nouveau nom au programme en résultant, montrait un manque stupéfiant de courtoisie.

Un tel comportement inconvenant reflétait l'image d'un autre comportement qui perturbait les développements dans la communauté hacker. La décision de Brian Reid en 1979 d'implémenter des « bombes à retardement » (« time Bomb ») dans Scribe, rendant possible à Unilogic de limiter l'accès au logiciel aux utilisateurs n'ayant pas payé, était un sombre présage pour Stallman. « Il considérait cela comme la chose la plus 'nazi' qu'il ait vue dans sa vie », se rappelle Reid. Malgré une renommée postérieure sur Internet en tant que co-créateur de la hiérarchie Usenet « alt », Reid assure qu'il doit toujours faire oublier cette décision de 1979, au moins aux yeux de Stallman. « Il disait que tous les logiciels devraient être libres et que la perspective de faire payer un programme était un crime contre l'humanité.[12] »

Quoique Stallman ait été impuissant pour influer sur les ventes de Reid, il possédait la capacité de contrer les autres formes de comportements jugées contraire à l'éthique hacker. Comme principal mainteneur du code source pour la « Communauté » Emacs, Stallman commença à exercer son pourvoir à des fins politiques. Au cours de l'étape finale de son conflit avec les administrateurs du laboratoire d'informatique au sujet des mots de passe, il entreprit une « grève »[13] du logiciel refusant d'envoyer la dernière version d'Emacs aux membres jusqu'à ce qu'ils rejettent le système de sécurité présent sur les ordinateurs. Le mouvement améliora peu la réputation grandissante qui qualifiait Stallman d'extrémiste, mais il passa le message : on pouvait compter sur les membres de la « Communauté » pour défendre les valeurs fondamentales du hacker.

« Beaucoup de gens étaient en colère contre moi, disant que je tentais de les prendre en otage ou de les faire chanter, ce que je faisais dans un sens », dirait plus tard Stallman à l'auteur Steven Levy. « Je m'étais violemment engagé contre eux car je pensais qu'ils s'étaient violemment engagés contre tout un chacun en général.[14] »

Avec le temps, Emacs devint un argument de vente de l'éthique hacker. Non seulement la flexibilité implémentée par Stallman dans le logiciel encourageait la collaboration, mais elle l'exigeait aussi. Les usagers qui ne suivaient pas les derniers développements de l'évolution d'Emacs ou qui ne renvoyaient pas leurs contributions à Stallman couraient le risque de manquer les derniers aboutissements, et il y en avait beaucoup. Vingt ans plus tard, les utilisateurs avaient modifié Emacs pour tant d'usages différents (l'utilisant comme tableur, calculateur, base de donnée et navigateur Web) que les derniers développeurs Emacs adoptèrent un lavabo plein à ras bord pour représenter sa fonctionnalité polyvalente. « C'est l'idée que nous voulions communiquer » dit Stallman, « la quantité de trucs qu'il contient est à la fois merveilleuse et effrayante ».

Les contemporains de Stallman au AI Lab sont plus charitables. Hal Abelson, un étudiant du MIT qui travailla avec lui durant les années soixante-dix et l'assisterait plus tard comme membre fondateur du conseil d'administration de la *Free Software Foundation* (FSF), décrit Emacs comme « une création absolument géniale ». En donnant aux programmeurs la possibilité d'ajouter de nouvelles bibliothèques et fonctions logicielles sans endommager le programme, déclare Abelson, Stallman à ouvert la voie à de futurs projets informatiques en collaboration à grande échelle. « Sa structure était si robuste que vous aviez des gens du monde entier qui collaboraient et contribuaient assez librement », dit Abelson, « je ne sais pas si cela avait été réalisé avant.[15] »

Guy Steele exprime une admiration identique. Actuellement chercheur pour Sun Microsystems, il se rappelle Stallman principalement comme un « programmeur génial à la capacité de générer du code relativement absent de bogue ». Quoique leurs personnalités ne s'accordassent pas exactement, ils collaborèrent suffisamment longtemps pour que Steele ait un aperçu du style intense de codage de Stallman. Il se rappelle un épisode notable à la fin des années soixante-dix quand les deux programmeurs firent équipe pour écrire la fonction « beau caractère » (pretty print) de l'éditeur. À l'origine conçue par Steele, la fonction « beau caractère » était une nouvelle propriété de frappe de touche qui reformatait le code source d'Emacs pour qu'il soit plus lisible et prenne moins de place tout en soulignant les qualités WYSIWYG du programme. La fonction était suffisamment stratégique pour susciter l'intérêt actif de Stallman et il fallut peu de temps pour que Steele écrivît que lui et Stallman préparaient une version améliorée.

« Nous nous sommes assis un matin », se rappelle Steele. « J'étais au clavier et lui à mes côtés », dit il. « Il voulait effectivement me laisser taper, mais il me disait aussi quoi taper. »

La session de programmation dura dix heures. Pendant tout ce temps, Steele raconte que ni lui ni Stallman n'ont fait une pause ou échangé quelques mots. À la fin de la séance, ils avaient réussi à réduire le code source de « beau caractère » juste en dessous de cent lignes. « Mes doigts étaient tout le temps sur le clavier », se souvient Steele, « C'était comme si nos deux pensées s'écoulaient vers l'écran. Il me disait ce que je devais taper et je le tapais. »

La durée de la réunion se révéla d'elle même quand Steele quitta finalement l'AI Lab Se tenant en dehors du bâtiment du 545 Tech Square, il fut surpris de se retrouver dans l'obscurité de la nuit. En tant que programmeur, Steele était habitué aux sessions marathon de codage. Mais il y avait quelque chose de différent dans cette séance. Travailler avec Stallman avait obligé Steele à inhiber tous les stimuli externes et focaliser son entière énergie mentale sur la tâche du moment. Avec le recul, Steele dit qu'il trouva la force d'esprit de Stallman à la fois exaltante et effrayante. « Ma première pensée, après coup, fut que c'était une grande expérience, très intense, et que je ne voulais plus jamais la reproduire au cours de ma vie. »

Notes

1. Josh McHugh, *For the Love of Hacking*, Forbes (10 août 1998). http://www.forbes.com/forbes/1998/0810/6203094a.html
2. Stallman, 1986.
3. Joseph Weizenbaum, *Computer Power and Human Reason: From Judgment to Calculation*, W. H.

Freeman, 1976, p.116.

4. D'après le Jargon File, le nom TECO est originellement tiré de « Tape Editor and Corrector ». http://www.tuxedo.org/~esr/jargon/html/entry/TECO.html.
5. Richard Stallman, *EMACS: The Extensible, Customizable, Display Editor*, AI Lab Memo (1979). Une version mise à jour de ce mémo au format HTML, de laquelle je tire cette citation, est trouvable à cette adresse : http://www.gnu.org/software/emacs/emacs-paper.html.
6. Richard Stallman, *Emacs the Full Screen Editor* (1987). http://www.lysator.liu.se/history/garb/txt/87-1-emacs.txt
7. *Ibidem*.
8. *Ibidem*.
9. *Ibidem*.
10. *Ibidem*.
11. Stallman, 1979, section 34 (#SEC34): http://www.gnu.org/software/emacs/emacs-paper.html#SEC34
12. Dans un interview du magazine online *MEME*, Stallman qualifia de rebutante la commercialisation de Scribe, mais il hésita à mentionner Reid nominativement. « Le problème est que personne n'a censuré ni puni cet étudiant pour ce qu'il a fait », affirma Stallman. « Au final, d'autres furent tentés de suivre son exemple ». Cf *MEME* 2.04 : http://memex.org/meme2-04.html.
13. Steven Levy, *Hackers*, Penguin USA, 1984, p.419.
14. *Ibidem*.
15. En écrivant ce chapitre, j'ai été amené à insister davantage sur la dimension sociale de EMACS que sur son aspect logiciel. Pour en savoir plus sur la partie logicielle, je recommande la lecture du mémo de 1979, et tout particulièrement la section intitulée « Research Through Development of Installed Tools » (#SEC27 : http://www.gnu.org/software/emacs/emacs-paper.html#SEC27). Non seulement cette partie est accessible pour les non spécialistes, mais elle illustre aussi dans quelle mesure interfèrent fortement, chez Stallman, la pensée politique et la conception de logiciels. En voici un exemple : « On ne serait pas parvenu à créer EMACS par des procédés aussi soignés soient-ils. En effet, par de tels procédés, on arrive seulement au but que l'on s'était fixé dès le départ. Ni moi ni personne n'avait imaginé un éditeur extensible jusqu'à ce que j'en crée un, et n'avait non plus mesuré sa valeur jusqu'à ce qu'on puisse l'essayer. EMACS existe parce que je me sentais libre de faire individuellement de petites améliorations utiles en suivant un chemin dont on ne voyait pas le bout ».

Chapitre VII — Une morale inflexible

Le 27 septembre 1983, les programmeurs se connectant au groupe de discussion Usenet net.unix-wizards reçurent un message peu habituel. Posté aux premières heures, à 12 heures 30 du matin pour être exact, et signé par rms@mit-oz, l'objet du message était laconique mais propre à retenir l'attention. « Nouvelle implémentation d'UNIX », pouvait-on lire. Cependant au lieu de présenter une version fraîchement disponible d'Unix, le premier paragraphe du message était en fait un appel à contribution :

> Commençant ce jour de Thanksgiving, je vais écrire un système logiciel complet compatible Unix appelé GNU (pour GNU N'est pas UNIX), et le distribuerai librement à tous ceux qui souhaitent l'utiliser. Des contributions en temps, en argent, en programmes et équipements sont grandement nécessaires.[1]

Pour un développeur Unix expérimenté, le message était un mélange d'idéalisme et d'incommensurable arrogance. Non seulement l'auteur s'engageait à reconstruire à partir de rien le système d'exploitation UNIX déjà mature, mais il proposait également de l'améliorer par endroit. Le nouveau système GNU, prédisait l'auteur, devra intégrer tous les composants essentiels : un éditeur de texte, un shell pour lancer des applications Unix-compatibles, un compilateur, « et diverses autres choses.[2] » Il devait contenir également beaucoup de caractéristiques séduisantes que les autres systèmes Unix n'offraient pas encore : une interface graphique utilisateur basée sur le langage de programmation Lisp, un système de fichiers à l'épreuve des plantages, et des protocoles réseaux construits à l'image du réseau interne du MIT.

« GNU sera capable d'exécuter des programmes Unix, mais ne sera pas identique à Unix », écrivait l'auteur. « Nous ferons toutes les améliorations utiles, sur la base de notre expérience avec d'autres systèmes d'exploitation. »

Prévoyant une réaction sceptique de la part de quelques lecteurs, l'auteur poursuivit l'exposé de son ébauche de système d'exploitation avec une brève note biographique intitulée « Qui suis-je? » :

> Je suis Richard Stallman, inventeur de l'éditeur EMACS souvent imité, actuellement au laboratoire d'intelligence artificielle du MIT. J'ai travaillé activement sur des compilateurs, des éditeurs, des débogueurs, des interpréteurs de commandes, ainsi que sur le système hétérogène de temps partagé et le logiciel d'exploitation des machines LISP. J'ai mis au point l'affichage indépendant du terminal pour ITS. En outre, j'ai mis en place un système de fichier à tolérance de panne et deux systèmes de fenêtrage pour machines LISP.[3]

Le destin n'a pas voulu que le projet fou de Stallman, le GNU, démarre ce jour de *Thanksgiving*. Cependant, en janvier 1984, Stallman tint une partie de ses promesses et s'immergea entièrement dans le monde du développement des programmes Unix. Pour un architecte logiciel nourri au lait de l'ITS, c'était comme concevoir des centres commerciaux de banlieue à la place de palais mauresques. Néanmoins, construire un système d'exploitation dans le style d'Unix avait ses avantages cachés. L'ITS était puissant, mais avait également un talon d'Achille : les hackers du MIT l'avaient conçu pour tirer parti spécifiquement de l'architecture des PDP de DEC. Quand les administrateurs du AI Lab choisirent de remplacer le puissant PDP-10 au début des années 80, le logiciel d'exploitation que les hackers avaient comparé à une cité grouillante d'activité ne fut plus, subitement, qu'une ville fantôme. Unix, à l'inverse, a été conçu pour le portage et la survie à long terme. À l'origine développé par les jeunes scientifiques d'AT&T, le programme avait échappé au contrôle des gestionnaires et avait trouvé sa terre d'élection dans le monde des systèmes informatiques universitaires au budget limité. Avec moins de ressources que leurs frères du MIT, les développeurs d'Unix avaient adapté le logiciel pour qu'il puisse piloter un assortiment bigarré de systèmes matériels : tout, du PDP-11 16 bits ─ une machine que les hackers du AI Lab jugaient destinée aux petits travaux ─ jusqu'aux unités centrales de 32 bits telles le VAX 11/780. Depuis 1983, quelques compagnies, et notamment Sun Microsystems, étaient même allées très loin dans le développement d'une nouvelle génération de micro-ordinateurs, des « stations de travail », pour tirer profit de ce système d'exploitation de plus en plus omniprésent.

Pour faciliter ce processus, les responsables de la conception des versions dominantes d'Unix ont veillé à conserver une couche supplémentaire d'abstraction entre le logiciel et la machine. Au lieu de tailler sur mesure le système d'exploitation pour tirer profit des ressources d'une machine spécifique ─ comme les hackers du AI Lab l'avaient fait avec l'ITS et le PDP-10 – les développeurs d'Unix ont favorisé une approche plus générique, indépendante du matériel. Au lieu de se concentrer exclusivement sur les composants physiques eux-mêmes, ils s'attachèrent davantage aux standards de communication ainsi qu'aux spécifications permettant l'intégration des nombreux sous-composants du système d'exploitation. De cette manière, ces développeurs créèrent un système rapidement adaptable à n'importe quel type de machine. Si un utilisateur chicanait sur une certaine portion, les standards permettaient de retirer un sous-composant spécifique pour le corriger ou le remplacer par quelque chose de mieux. Au final, ce qui manquait à Unix en termes d'architecture ou d'esthétique était largement compensé en termes de flexibilité et d'économie, d'où son adoption rapide.[4]

La fin du système ITS, que les hackers du AI Lab avaient si longtemps materné, décida Stallman à commencer le développement du système GNU. L'abandon de l'ITS fut un coup très rude pour lui. Survenu juste après l'épisode de l'imprimante laser Xerox, l'évènement montrait qu'à l'évidence la culture des hackers au laboratoire perdait de son immunité face aux pratiques commerciales du monde extérieur.

Comme le code logiciel qui le composait, les causes de la mort de l'ITS étaient profondément enracinées dans le passé. Le Département de la Défense américain, bailleur de fonds de longue date de la recherche informatique au MIT et autres institutions de haut niveau, dut revoir drastiquement son calendrier d'investissements durant les années 1970 afin de tenter de maîtriser les budgets qui avaient progressé en spirale ascendante pendant la guerre du Vietnam. Dans leur recherche désespérée de nouvelles ressources financières, les laboratoires et les universités se tournèrent vers le secteur privé. Dans le cas du AI Lab, trouver des investisseurs privés était facile. Étant à l'origine de certains des projets de science informatique les plus ambitieux de l'après-guerre, le laboratoire était devenu un incubateur de technologies plutôt énergique. En effet, dès 1980, la majorité du personnel, y compris beaucoup de hackers, partageait son temps entre l'institut et les projets commerciaux.

Ce qui, de prime abord, apparaissait comme un contrat favorable aux deux parties — les hackers pouvaient travailler sur les meilleurs projets et donnaient en retour au labo un regard privilégié sur les technologies informatiques les plus récentes — se révéla rapidement être un pacte faustien. Plus les hackers consacraient leur temps aux projets commerciaux de pointe, moins ils pouvaient se dévouer à la maintenance générale de l'infrastructure logicielle baroque du laboratoire. Bientôt, des compagnies commencèrent à débaucher les hackers pour monopoliser leur temps et leur attention. Avec moins de hackers pour gérer la boutique, programmes et machines voyaient s'accroître les intervalles de mise à jour. Pire, selon Stallman, le labo amorçait un « changement démographique ». Les hackers qui formaient autrefois une minorité agissante au sein du AI Lab étaient en train de perdre leur légitimité, tandis que « les professeurs et les étudiants, qui n'aimaient pas vraiment le [PDP-10], étaient aussi nombreux qu'auparavant ».[5]

Le point de rupture survint en 1982. Cette année-là l'administration du labo décida de changer son ordinateur principal, le PDP-10. Digital, la société qui le fabriquait, avait interrompu cette ligne de produit. La compagnie offrait bien toujours un puissant super-calculateur, nommé le KL-10, mais la nouvelle machine exigeait une réécriture complète (un nouveau « portage ») de l'ITS si les hackers voulaient continuer à faire tourner le même système d'exploitation. Effrayés de constater que le labo avait perdu une part importante de ses ressources internes en programmeurs de talent, les membres de la faculté optèrent pour Twenex, un système d'exploitation commercial développé par Digital. Minoritaires, les hackers n'eurent d'autre choix que s'incliner.

Quelques années plus tard, Stallman se souviendrait encore du discours des membres de la faculté : « Ils disaient : 'Sans hackers pour maintenir le système, nous allons droit au désastre ; il nous faut un logiciel commercial. Nous nous attendons à ce que la société [Digital] en assure la maintenance'. La suite a montré qu'ils se trompaient lourdement, mais c'est ce qu'ils ont fait.[6] »

Au début, les hackers considéraient le système Twenex comme un nouveau symbole de l'autorité qui ne demandait qu'à être détourné. Le nom même du système était une provocation. Officiellement dénommé TOPS-20 par DEC, il était le successeur du TOPS-10, un système d'exploitation commercial vendu avec le PDP-10. Bolt Beranek Newman en avait développé une version améliorée, dénommée Tenex, sur laquelle

était basé TOPS-20.[7] Stallman, à l'origine du nom Twenex, explique qu'il a proposé ce nom pour éviter d'utiliser celui de TOPS-20. « Le système était loin d'être au top, à tel point qu'il était inconcevable de l'appeler ainsi », se souvient Stallman, « c'est pourquoi j'ai décidé de glisser un 'w' dans le nom Tenex et de l'appeler Twenex. »

La machine sur laquelle tournait le système Twenex/TOPS-20 eut, par dérision, son propre surnom : Oz. Selon une légende, la machine hérita de ce surnom car elle avait besoin d'un petit PDP-11 pour faire fonctionner son terminal. Un hacker, voyant fonctionner pour la première fois le tandem KL-10-PDP-11 le compara au magicien pompeux du film le *Magicien d'Oz*. « Je suis le grand, le puissant Oz! », récita le hacker, « Ne faites pas attention au PDP-11 qui est derrière la console.[8] »

Si les hackers riaient en voyant le KL-10 pour la première fois, Twenex leur en passa rapidement l'envie. Si Twenex se prévalait d'une sécurité intégrée, elle était devenue une obsession pour les ingénieurs qui conçurent les utilitaires et les applicatifs. Auparavant, en ce qui concernait la sécurité au laboratoire de sciences informatiques, la gestion des mots de passe revenait à jouer au chat et à la souris, mais cela devint une guerre ouverte pour l'administration du système. Les administrateurs arguaient du fait que sans sécurité, le système Oz était davantage enclin aux pannes accidentelles. Les hackers répliquaient que de tels accidents pourraient être évités grâce à la relecture du code source. Malheureusement, le nombre de hackers ayant le temps et l'envie de procéder à cette vérification du code avait diminué à tel point que l'argument des administrateurs système prévalut.

Stallman réussit à déjouer les tentatives de prise de contrôle de ces derniers en contournant les mots de passe et provoquant délibérément des arrêts violents pour glaner des informations parmi les décombres. Après un « coup d'état » raté, Stallman envoya un message d'alerte à tout le personnel du AI Lab.

« Il y a eu une nouvelle tentative de prise de pouvoir », écrivit Stallman. « Jusqu'ici, les forces aristocratiques ont été défaites ». Pour protéger son identité, Stallman signa son message « Radio Free OZ ».

Le déguisement était bien mince. En 1982, l'aversion de Stallman pour les mots de passe et le secret était d'une telle notoriété que les utilisateurs externes au AI Lab employaient son compte comme tremplin pour accéder à ARPAnet - le réseau informatique financé par la recherche qui servira de base à l'Internet d'aujourd'hui. Parmi ces « touristes » du début des années 1980 il y avait Don Hopkins, un programmeur de Californie, qui apprit par la rumeur que tout ce qu'un intrus devait faire pour accéder au célèbre système ITS du MIT était d'ouvrir une session sous les initiales RMS et d'écrire le même monogramme de trois lettres lorsque le système demandait le mot de passe.

« Je suis éternellement reconnaissant au MIT de m'avoir laissé, ainsi qu'à beaucoup d'autres, utiliser librement leurs ordinateurs », dit Hopkins. « Cela en disait long pour beaucoup de gens. »

Cette prétendue politique de « touristes », qui avait été ouvertement tolérée par les gestionnaires du MIT pendant les années ITS[9], tourna court quand Oz devint le premier maillon raccordant le Labo à ARPAnet. Au début, Stallman poursuivit sa politique consistant à reprendre son identifiant de connexion comme mot de passe pour permettre aux utilisateurs extérieurs de suivre ses traces. Avec le temps, cependant, la fragilité d'Oz incita les administrateurs à interdire l'accès aux intrus qui, par pure maladresse ou malveillance délibérée, auraient pu endommager le système. Quand ces mêmes administrateurs ont par la suite exigé de Stallman qu'il cesse de diffuser son mot de passe, ce dernier refusa de le faire en invoquant son éthique personnelle et cessa complètement d'utiliser le système Oz.[10]

« [Lorsque] les mots de passe ont fait leur première apparition au laboratoire d'intelligence artificielle du MIT, j'ai [décidé] de suivre mon opinion selon laquelle il ne devrait y avoir aucun mot de passe », dirait plus tard Stallman. « Dans la mesure où je ne crois pas qu'il soit vraiment souhaitable d'avoir une quelconque sécurité sur un ordinateur, je n'ai pas à collaborer avec ce régime sécuritaire.[11] »

Au début des années 1980, son refus de s'incliner devant le grand et puissant Oz a symbolisé la tension croissante entre les hackers et les gestionnaires du AI Lab. Cette tension passa cependant au second plan, derrière le conflit qui fit rage au sein même de la communauté des hackers. Avant l'arrivée du KL-10, cette dernière s'était déjà divisée en deux camps : le premier regroupé autour d'une compagnie de logiciel appelée Symbolics, Inc., le second rallié au principal rival de Symbolics, Lisp Machines, Inc. (LMI). Les deux compagnies étaient en compétition pour le lancement de la machine LISP, un dispositif conçu pour tirer

pleinement parti du langage de programmation LISP.

Créé par un pionnier, John McCarthy, chercheur en intelligence artificielle au MIT à la fin des années 50, LISP est un langage élégant, bien adapté aux programmes chargés d'effectuer de nombreuses opérations de traitement et de tri. Le nom du langage est la contraction de *LISt Processing* (traitement de liste). Après le départ de McCarthy pour le laboratoire d'intelligence artificielle de Stanford, les hackers du MIT ont perfectionné le langage en créant un dialecte local dénommé MACLISP. Ce nom faisait référence au projet *MAC*, un projet de recherche du DARPA qui donna naissance au AI Lab et au laboratoire de sciences informatiques. Mené par l'archi-hacker Richard Greenblatt, les programmeurs du AI Lab ont construit en totalité un système d'exploitation basé entièrement sur LISP, nommé le *Lisp Machine operating system* (le système d'exploitation de machine LISP). En 1980, le projet de machine LISP avait abouti à la création de deux sociétés commerciales: Symbolics, dirigée par Russell Noftsker, un ancien administrateur du AI lab, et Lisp Machines, Inc. (LMI), dirigée par Greenblatt.

Le logiciel de la machine LISP avait été créé par les hackers, ce qui signifiait qu'il appartenait au MIT mais restait disponible pour toute personne souhaitant en faire une copie, comme le voulait la coutume. Mais un tel système limitait tout espoir d'obtenir un privilège commercial pour une compagnie désirant établir un monopole de distribution du logiciel. Pour acquérir d'autres avantages et doper les fonctionnalités du système d'exploitation afin d'attirer les clients, les compagnies recrutèrent des hackers du AI Lab et les firent travailler sur les divers composants du système d'exploitation de la machine LISP en dehors de l'influence du laboratoire.

La plus agressive dans cette stratégie était la compagnie Symbolics. Vers la fin de 1980, elle avait embauché quatorze programmeurs du AI Lab comme conseillers à temps partiel pour développer sa version de la machine LISP. En dehors de Stallman, le reste des programmeurs s'était mis au service de LMI.[12]

Au début, Stallman participa aux tentatives des deux sociétés pour commercialiser des machines LISP, même si cela lui demandait plus de travail. Toutes deux avaient acquis du MIT la licence du code source du système d'exploitation de la machine LISP, et c'était le travail de Stallman de mettre à jour celle du laboratoire pour suivre les dernières innovations. Bien que le contrat entre Symbolics et le MIT donnait à Stallman le droit d'examiner ─ mais pas de copier ─ le code source de Symbolics, Stallman précise qu'un accord tacite entre la direction de Symbolics et le AI Lab lui accordait la possibilité d'emprunter des bouts de code attrayants suivant la coutume des hackers.

Le 16 mars 1982, une date que Stallman se rappelle bien car c'est celle de son anniversaire, des cadres de Symbolics décidèrent d'en finir avec cet accord à l'amiable. Le coup était en grande partie stratégique. LMI, le concurrent direct sur le marché des machines LISP, utilisait essentiellement une copie de celle du AI Lab. Pour ne pas favoriser le développement d'un tel rival, l'exécutif de Symbolics choisit d'appliquer la licence à la lettre. Si le AI Lab voulait que son système d'exploitation reste en phase avec le système d'exploitation de Symbolics, le laboratoire devait basculer sur une machine de Symbolics et arrêter sa collaboration avec LMI.

En tant qu'administrateur de la machine LISP du laboratoire, Stallman était scandalisé. Considérant cette annonce comme un « ultimatum », il répondit en coupant la liaison par onde courte qui reliait Symbolics au Laboratoire. Il ne travailla plus jamais sur la machine de Symbolics et fit immédiatement allégeance à LMI. « Pour moi, le AI Lab était un pays neutre, comme la Belgique durant la Première Guerre Mondiale », dit Stallman. « Si l'Allemagne envahit la Belgique, la Belgique déclare la guerre à l'Allemagne aux côtés de la Grande-Bretagne et de la France. »

Les circonstances de la prétendue « guerre de Symbolics » de 1982-1983 varient fortement en fonction de la source des témoignages. Quand les cadres de Symbolics notèrent que leurs dernières améliorations apparaissaient systématiquement dans la machine LISP du AI Lab et, par extension, dans celle de LMI, ils installèrent un programme « espion » sur le terminal de l'ordinateur de Stallman. Ce dernier affirme cependant qu'il réécrivait les fonctionnalités à partir de zéro, tirant parti de la clause de droit de regard de la licence, mais prenant soin également de produire un code source aussi différent que possible. Mais les cadres de Symbolics virent les choses autrement et plaidèrent leur cause auprès de l'administration du MIT. Selon le livre paru en 1994, *The Brain Makers: Genius, Ego, and Greed, and the Quest for Machines That Think* (Les fabricants de cerveau : Le génie, l'ego et l'avarice, et la quête de machines

pensantes), écrit par Harvey Newquist, l'administration répondit par un avertissement à Stallman menaçant de l'écarter du projet de machine LISP.[13] Pourtant, d'après Stallman, les administrateurs du MIT l'ont soutenu: « je n'ai jamais été menacé », dit-il. « J'ai cependant modifié ma façon de faire. Pour plus de sûreté, j'ai cessé de lire leur code source. J'ai seulement consulté la documentation et écrit le code en me basant sur elle. »

Quelle qu'en fût l'issue, la querelle renforça la résolution de Stallman. Sans lecture du code source, Stallman remplit les blancs à sa manière et enrôla des membres du AI Lab pour fournir un flux soutenu de rapports de bogues. Il s'assura également que des programmeurs de LMI avaient un accès direct à ces changements. « J'allais punir Symbolics même si c'était la dernière chose que je devais faire », dit Stallman.

De tels témoignages sont riches d'enseignement. Non seulement jettent-ils la lumière sur la nature belliqueuse de Stallman, mais ils reflètent également l'intensité de l'émotion provoquée par le conflit. Selon d'autres sources provenant de Newquist, Stallman devint si furieux qu'il envoya un email menaçant « de se barder de dynamite et d'aller se faire sauter dans les bureaux de Symbolics ».[14] Bien que Stallman prétende n'avoir aucun souvenir de ce courriel et qu'il en ait toujours parlé comme d'une « méchante rumeur », il reconnaît que de telles pensées lui ont bien traversé l'esprit : « j'ai sans doute eu le fantasme de me suicider et de détruire au passage leur bâtiment », indique Stallman. « Je pensais que ma vie était finie.[15] »

Un tel degré de désespoir est dû, pour beaucoup, à ce que Stallman ressentit comme une « destruction » de son « foyer » – c'est-à-dire l'abandon de la sub-culture fermée des hackers au AI Lab. Par la suite, dans un entretien par courrier électronique, Stallman se compara à la figure historique d'Ishi, le dernier survivant des Yahis, une tribu du nord-ouest de la Côte Pacifique, éliminée pendant les guerres indiennes des années 1860 et 1870. Cette analogie élève l'entêtement de Stallman à un niveau épique, à la limite du mythe. En réalité, cependant, cela masque les tensions entre Stallman et ses camarades hackers avant le schisme entre Symbolics et LMI. Au lieu de voir en Symbolics une force exterminatrice, plusieurs des collègues de Stallman l'ont finalement considérée comme une offre pertinente. En commercialisant la machine LISP, cette compagnie permit aux principes hackers, ceux auxquels obéissaient les ingénieurs en conception de logiciels, de sortir de la tour d'ivoire du AI Lab et de pénétrer le domaine commercial où prévalaient plutôt les principes des gestionnaires. Au lieu de voir en Stallman un résistant, beaucoup de hackers l'ont considéré comme un gêneur anachronique.

Stallman ne conteste pas cette interprétation des événements. Cela dit, il y avait encore un autre mobile à sa réaction d'hostilité face à « l'ultimatum » de Symbolics. Bien avant que Symbolics ait embauché la majeure partie des hackers du AI Lab, Stallman indique que plusieurs d'entre eux, qui rejoindraient plus tard Symbolics, l'évitaient. « Je n'étais plus invité à aller à Chinatown », se souvient-il. « La coutume inaugurée par Greenblatt était que si vous partiez dîner, vous demandiez autour de vous ou envoyiez des messages pour savoir si quelqu'un au laboratoire souhaitait vous accompagner. Non seulement ils ne m'invitaient plus, mais quelqu'un m'avoua plus tard qu'on l'avait pressé de me mentir pour aller dîner sans moi, en cachette. »

Bien que Stallman ait ressenti de la colère envers les hackers à l'origine de cette forme mesquine d'ostracisme, la polémique avec Symbolics avait fait naître une nouvelle forme de colère, la colère d'une personne sur le point d'être expropriée. Lorsque Symbolics cessa de lui transmettre les modifications du code source, Stallman répondit en allant s'enterrer dans les bureaux du MIT pour y réécrire chaque nouvelle fonctionnalité ou outil logiciel à partir de zéro. Aussi frustrant que cela eût pu paraître, ce travail garantissait aux futurs utilisateurs de machines LISP un accès libre aux mêmes fonctionnalités que les utilisateurs de Symbolics.

Cela renforça également le statut légendaire de Stallman au sein de la communauté hacker. Déjà connu pour son travail sur Emacs, la capacité de Stallman à produire seul un travail équivalent à celui de toute l'équipe des programmeurs de Symbolics – équipe au sein de laquelle on comptait plus d'un hacker légendaire – reste l'un des plus grands exploits humains de l'ère de l'information, ou de toute autre ère dans ce domaine. Qualifiant ce hacking de véritable « coup de maître », et Stallman lui-même de « John Henry virtuel de la programmation », l'auteur Steven Levy note que nombre de ses rivaux de chez Symbolics n'avaient d'autre choix que de saluer à contre-cœur leur ancien camarade idéaliste. Levy rapporte les propos de Bill Gosper, un hacker qui finirait par travailler pour Symbolics dans les bureaux de la compagnie à Palo Alto, exprimant son émerveillement pour le travail fourni par Stallman durant cette période :

> Devant quelque chose écrit par Stallman, il pourrait m'arriver de trouver cela mauvais (c'est improbable, mais on pourrait m'en convaincre), cependant j'exprimerai toujours mon admiration : 'mais attendez une minute! Stallman n'avait personne avec qui débattre là-bas des nuits entières. Il a travaillé seul! C'est incroyable que quelqu'un ait pu réaliser cela dans son coin!'[16] »

Les mois passés à lutter contre Symbolics laissèrent à Stallman un mélange de fierté et de profonde tristesse. En tant que libéral convaincu dont le père avait servi pendant la Seconde Guerre Mondiale, Stallman n'est pas un pacifiste. De bien des manières, la guerre contre Symbolics représentait le rite de passage pour lequel Stallman s'était préparé depuis qu'il avait rejoint le personnel du AI Lab une décennie plus tôt. Cependant, cette guerre coïncida simultanément avec la traumatisante destruction de la culture hacker qui avait entouré Stallman depuis son adolescence. Un jour, raconte-t-il, alors qu'il faisait une pause dans l'écriture du code, Stallman éprouva un sentiment bouleversant au moment de traverser la salle des machines du laboratoire. Là, il fit face à la silhouette massive du PDP-10 abandonné. Pris de peur à la vue des voyants éteints, voyants qui autrefois clignotaient silencieusement pour indiquer l'état du programme en cours, l'émotion qu'il ressentit n'était pas différente de celle provoquée par la vision du cadavre bien conservé d'un être cher.

« Je me mis à pleurer en plein milieu de la salle des machines », dit-il. « Voyant cette machine là, morte, sans personne pour s'en occuper, cela me rappela à quel point ma communauté avait été détruite. »

Stallman n'aurait guère de temps pour porter le deuil. En dépit de la fureur dont il avait fait preuve et tout le travail qu'il avait consacré à son élaboration, la machine LISP n'était qu'un épisode annexe par rapport aux grandes batailles qui se livraient sur le marché des nouvelles technologies. L'implacable progression de la miniaturisation des ordinateurs voyait apparaître des microprocesseurs plus récents, plus puissants, et qui allaient bientôt incorporer toutes les capacités matérielle et logicielle de la machine comme une métropole moderne absorbant un antique village désert.

Surfant sur cette vague du microprocesseur, des centaines – des milliers ─ de logiciels commerciaux, chacun protégé par un patchwork de licences utilisateur et de clauses de confidentialité, ont rendu impossible au hacker l'examen ou le partage du code source. Les licences étaient brutes et mal taillées, mais en 1983 elles étaient devenues assez solides pour satisfaire les tribunaux et pour effrayer quiconque voudrait les contourner. Le logiciel, autrefois une sorte de garniture que la plupart des fabricants d'ordinateurs offraient pour rendre plus savoureux leurs très coûteux systèmes informatiques, devenait rapidement le plat principal. Dans leur faim croissante pour de nouveaux jeux et programmes, les utilisateurs en oublaient de demander la recette, comme on la demande traditionnellement après chaque repas.

Nulle part cet état de fait n'était plus évident qu'au royaume des ordinateurs individuels. Les compagnies telles Apple et Commodore faisaient la fortune de jeunes millionnaires en vendant des machines avec des logiciels d'exploitation intégrés. Ignorant tout de la culture hacker et de son dégoût pour le logiciel pré-compilé, la plupart des utilisateurs n'ont pas éprouvé le besoin de protester quand ces compagnies ont fourni leurs programmes sans les accompagner des fichiers contenant le code source. Quelques anarchistes adeptes de l'éthique hacker tentèrent de répandre cette éthique sur le nouveau marché, mais la plupart du temps ce dernier gratifiait les programmeurs assez prompts pour écrire de nouveaux programmes et assez prudents pour en garantir les droits d'auteur par des protections légales.

Un des plus notoires de ces programmeurs était Bill Gates, sorti d'Harvard deux ans avant le jeune Stallman. Gates était un entrepreneur débutant, co-dirigeant de la société de logiciel Micro-Soft ─ qui s'écrirait plus tard Microsoft ─ basée à Albuquerque. Bien que Stallman ne l'ait pas su alors, sept ans avant l'envoi de son message au newsgroup de net.unix-wizards, Gates avait adressé une lettre ouverte à la communauté des programmeurs. Écrite en 1976, à l'attention des utilisateurs de PC copiant les programmes de Micro-Soft, la « lettre ouverte aux amateurs » de Gates dénonçait la notion du développement communautaire du logiciel.

« Qui peut se permettre d'effectuer un travail professionnel pour rien? », demandait Gates. « Quel amateur peut-il investir trois années-homme dans la programmation, trouver tous les bogues, documenter son produit, et le distribuer gratuitement?[17] »

Bien que peu de hackers au AI Lab ne l'aient vue, la lettre de Gates est néanmoins l'illustration du

changement d'attitude envers les logiciels à la fois de la part des compagnies les commercialisant et de la part des développeurs de programmes commerciaux. Pourquoi traiter le logiciel comme un produit sans valeur quand le marché en a décidé autrement? Des années 1970 aux années 1980, la vente du logiciel était devenue plus qu'un moyen d'amortir les coûts : c'était devenu un enjeu politique. Au moment où l'administration Reagan s'empressait de démanteler nombre de règlements fédéraux et suspendait les programmes échafaudés durant le demi-siècle qui avait fait suite à la Grande Dépression, plus d'un programmeur considérait l'éthique des hackers comme anti-compétitive et, par extension, anti-américaine. Au mieux, c'était un retour aux attitudes anti-entreprises de la fin des années 1960 et du début des années 1970. Comme un banquier de Wall Street découvrant un de ses vieux tee-shirts caché entre ses chemises de chez Cardin et ses costumes trois-pièces, beaucoup d'informaticiens ont considéré l'éthique des hackers comme un rappel embarrassant d'une époque idéaliste.

Pour un homme qui avait vécu toutes les années 1960 comme un retour fâcheux aux années 1950, Stallman n'imaginait pas vivre à l'écart de ses pairs. Cependant, en tant que programmeur habitué à travailler avec les meilleures machines et le meilleur logiciel, Stallman dut faire face à ce qu'il appela un « choix moral inflexible » : soit il se débarrassait de ses objections morales contre les logiciels « propriétaires » ─ le terme par lequel Stallman et ses camarades hackers désignaient tout programme soumis à un copyright ou avec une licence d'utilisateur finale restreignant les droits de copie ou modification ─, soit il consacrait son existence à l'élaboration d'un sytème alternatif de logiciels non propriétaires. Après les longs mois d'épreuve avec Symbolics, Stallman se sentait davantage enclin à choisir la seconde solution. « Certes, j'aurais pu être obligé de cesser complètement de travailler sur des ordinateurs », indique Stallman. « Je n'ai aucune qualification spéciale, mais je suis sûr que j'aurais pu devenir serveur. Pas dans un restaurant chic, probablement, mais j'aurais bien pu me trouver une place de serveur quelque part. »

Devenir serveur ─ c'est-à-dire laisser tomber totalement la programmation ─ aurait signifié rejeter une activité, l'écriture de programmes d'ordinateur, qui lui avait donné tellement de plaisir. Depuis son entrée à Cambridge, il était facile pour Stallman de se souvenir des longues périodes où la programmation lui procurait une satisfaction unique. Plutôt que tout lâcher, Stallman décida de s'accrocher.

Stallman, athée, rejette les notions telles le destin, le dharma ou l'appel divin, censées influencer l'existence. Néanmoins, il estime que la décision de combattre les logiciels propriétaires et de créer un système d'exploitation pour aider les autres à faire de même était une décision tout à fait naturelle de sa part. Après tout, son obstination, son intuition et sa virtuosité dans l'écriture du code formaient une combinaison toute personnelle qui l'avait conduit à cette croisée des chemins dont personne n'avait soupçonné l'existence. En relatant sa prise de décision dans un chapitre du livre *Open Sources* paru en 1999, Stallman fait référence aux mots du sage juif Hillel :

> Si je ne suis pas pour moi, qui sera pour moi? Si je ne suis que pour moi, qui suis-je? Si pas maintenant, quand?[18]

En public, Stallman évite toute référence religieuse et relate sa prise de décision en termes pragmatiques. « Je me suis demandé : que pourrais-je faire, moi, développeur de système d'exploitation, pour améliorer la situation? Ce n'est qu'après avoir longuement examiné la question que j'ai réalisé qu'un développeur de système d'exploitation était exactement ce qu'il fallait pour résoudre le problème. »

Comme l'indique Stallman, une fois cette décision prise, tout « se mettait en place ». Il renoncerait à l'utilisation des logiciels qui l'ont forcé à compromettre ses convictions morales, tout en consacrant sa vie à la création d'un logiciel qui permettrait aux autres de suivre aisément le même chemin. S'engageant à bâtir un logiciel d'exploitation libre « ou mourir en essayant... le plus vieux possible », plaisantait Stallman, il démissionna du MIT en janvier 1984, pour construire le GNU.

Sa démission l'empêchait de travailler dans le cadre légal du MIT. Cependant Stallman avait encore assez d'amis et d'alliés au AI Lab pour conserver l'accès à son bureau. Il eut ainsi la possibilité d'obtenir des contrats de consulting externe pour assurer les phases initiales du projet GNU. En démissionnant, cependant, Stallman a tué dans l'oeuf toute discussion sur un éventuel conflit d'intérêt ou sur l'appartenance à l'institut du logiciel en question. L'homme, que la crainte de l'isolement social avait conduit au début de sa vie d'adulte à s'intégrer de plus en plus profondément au sein du AI Lab, construisait maintenant un pare-feu légal entre lui et cet environnement.

Durant les premiers mois, Stallman travailla également à l'écart de la communauté Unix. Bien que son annonce au groupe net.unix-wizards ait attiré des messages de sympathie, au début, peu de volontaires se connectèrent pour se joindre à la croisade.

« La réaction de la communauté était assez homogène », se rappelle Rich Morin, alors chef d'un groupe d'utilisateurs Unix. « Les gens ont dit : 'Oh, quelle grande idée. Montrez-nous votre code. Montrez-nous qu'on peut le faire'. »

Dans le plus pur style hacker, Stallman commença par rechercher les programmes et les outils existants qui pouvaient être convertis en programmes et outils GNU. Un des premiers était un compilateur appelé *VUCK*, qui convertissait des programmes écrits en langage C, un langage informatique très populaire, en code compréhensible pour une machine. Traduit du néerlandais, l'acronyme du programme signifiait « Outils de compilation de l'université libre ». Plein d'optimisme, Stallman demanda à l'auteur du programme si le programme était libre. Quand l'auteur l'informa que l'expression « université libre » faisait référence à la *Vrije Universiteit in Amsterdam* (Université Libre d'Amsterdam), Stallman fut déçu.

« Il m'a répondu en se moquant que si l'université était libre, le compilateur, lui, ne l'était pas », se rappelle Stallman. « J'ai donc décidé que mon premier programme pour le projet GNU serait un compilateur multi-langage et multi-plateforme.[19] »

Par la suite, Stallman trouva un compilateur du langage PASTEL, écrit par des programmeurs du *Lawrence Livermore National Lab*. A la connaissance de Stallman, le compilateur était alors libre d'être copié et modifié. Malheureusement, le programme avait un défaut de conception majeur : il chargeait l'intégralité de chaque programme en mémoire centrale, occupant cet espace précieux au détriment d'autres tâches du système. Sur des gros ordinateurs ce défaut de conception était pardonnable. Sur des systèmes Unix il constituait une barrière infranchissable, car les machines fonctionnant sous Unix étaient trop petites pour manipuler les fichiers de grand taille qui étaient générés. Dans un premier temps, Stallman accomplit des progrès substantiels, ajoutant au compilateur un frontal compatible C. Mais avant l'été il en était arrivé à la conclusion qu'il lui fallait finalement réécrire entièrement un nouveau compilateur à partir de zéro.

En septembre 1984, Stallman stoppa le développement du compilateur jusqu'à la fin du trimestre et se lança à la recherche d'un fruit plus accessible. Il commença le développement d'une version GNU d'Emacs, le programme qu'il avait lui-même maintenu pendant une décennie. La décision était stratégique. Au sein du monde Unix, les deux programmes d'édition disponibles étaient vi, écrit par Bill Joy, cofondateur de Sun Microsystems, et ED, écrit par Ken Thompson un scientifique des laboratoires Bell (et co-créateur d'Unix). Tous les deux étaient utiles et populaires, mais ni l'un ni l'autre n'offraient les possibilités d'extension sans limites d'Emacs. En réécrivant Emacs pour les utilisateurs d'Unix, Stallman tenait là sa meilleure chance de faire preuve de son talent. Cela donnait également l'occasion aux utilisateurs d'Emacs de se familiariser avec la mentalité de Stallman.

En y repensant, Stallman soutient qu'il n'avait pas saisi le caractère stratégique de cette décision : « j'ai voulu un Emacs et j'ai profité de l'occasion pour en développer un. »

De nouveau, la peur de réinventer la roue a aiguisé son efficace instinct de hacker. En écrivant une version Unix d'Emacs, Stallman se trouva bientôt dans les traces de James Gosling, étudiant diplômé de Carnegie Mellon, auteur d'une version en C nommée Gosling Emacs ou GOSMACS. La version d'Emacs de Gosling incluait un interpréteur qui exploitait une version simplifiée du langage LISP appelée MOCKLISP. Déterminé à construire GNU Emacs sur une base proche de LISP, Stallman emprunta copieusement les innovations de Gosling. Bien que ce dernier ait placé GOSMACS sous copyright et vendu les droits à UniPress, une compagnie logiciel financée par des fonds privés, Stallman cita les allégations d'un camarade developpeur qui avait participé aux premières phases d'écriture de l'interprète MOCKLISP. Selon ce programmeur, Gosling, alors qu'il était encore doctorant à Carnegie, avait assuré à ses premiers collaborateurs que leur travail demeurerait accessible. Cependant, quand UniPress eut vent du projet de Stallman, la compagnie menaça de durcir l'application du copyright. À nouveau, Stallman se heurtait à la perspective de devoir tout rebâtir à partir de rien.

En reconstituant le code source de l'interpréteur de Gosling à partir du programme compilé, Stallman créa un interpréteur en LISP pleinement fonctionnel, rendant inutile l'utilisation de l'interpréteur original de Gosling. Néanmoins, l'idée que des concepteurs puissent liquider leurs droits – et, en amont, l'idée même

qu'un concepteur disposât du droit de vendre quelque chose - offusquait Stallman. Dans un discours de 1986 à l'Institut Technique Royal suédois, Stallman cita l'incident d'UniPress comme un nouvel exemple des dangers liés aux logiciels propriétaires :

« Parfois je pense que l'une des meilleures choses que je pourrais sans doute faire dans ma vie est de m'emparer d'un stock complet d'exemplaires d'un logiciel propriétaire sous secret commercial et de commencer à en distribuer des copies au coin d'une rue. Ainsi, il n'y aurait plus de secret commercial », maugréa Stallman. « Ce serait peut-être une manière beaucoup plus efficace de donner aux gens un nouveau logiciel libre que de l'écrire moi-même ; mais les gens seraient encore trop lâches pour le prendre.[20] »

Sur le long terme, en dépit du stress qu'elle générait, la controverse sur les innovations de Gosling aida Stallman et le mouvement pour le logiciel libre. Stallman fut forcé de mettre à jour les faiblesses de la Commune Emacs [cf. chap. 6] et du contrat de confiance informel à l'origine de ces contrecoups problématiques. Cela obligea également Stallman à préciser les objectifs politiques du mouvement du logiciel libre. Après la sortie de GNU Emacs en 1985, Stallman publia *Le manifeste GNU*, un complément à l'annonce originale de septembre 1983. Stallman inclut dans le document une large section consacrée aux nombreux arguments employés par les programmeurs des secteurs privés et universitaires pour justifier la prolifération des logiciels propriétaires. L'un de ces arguments, « Les programmeurs ne méritent-ils pas une récompense pour leur créativité? », reçut une réponse qui contenait toute la colère de Stallman à la suite de l'affaire Gosling :

« Si quelque chose mérite une récompense, c'est la contribution à la société », écrivit Stallman. « La créativité peut être un apport à la société, mais uniquement dans le cas [sic] où cette dernière est libre d'en utiliser le produit. Si des programmeurs méritent d'être récompensés pour la création de programmes novateurs, ils méritent de même d'être punis s'ils limitent l'utilisation de ces programmes.[21] »

Avec la publication de GNU Emacs, le projet GNU avait enfin du code à montrer. Il eut également tous les soucis propres à n'importe quelle entreprise diffusant du logiciel. En effet, comme de plus en plus de concepteurs Unix commençaient à jouer avec le logiciel, l'argent, les cadeaux, et les demandes de bandes ont commencé à affluer. Pour gérer les aspects commerciaux du projet GNU, Stallman réunit quelques-uns de ses collègues et forma la *Free Software Foundation* (FSF), une organisation à but non lucratif visant à accompagner la réalisation du projet GNU. Avec Stallman comme président et divers alliés hackers comme membres du conseil, la FSF représentait la vitrine publique du projet GNU.

Robert Chassell, à l'époque programmeur chez *Lisp Machines, Inc.*, est devenu l'un des cinq membres du conseil de la FSF à la suite d'une conversation dînatoire avec Stallman. Chassell fut également le trésorier de l'organisation, un rôle initialement modeste mais qui s'est rapidement développé.

« Je pense qu'en 1985 le total de nos transactions, recettes et dépenses, étaient approximativement de 23.000 dollars », se rappelle Chassell. « Richard avait son bureau, et nous optimisions tout l'espace. Je mettais tout le fourbi, surtout les bandes, sous mon bureau. Cela dura jusqu'à ce que, un peu plus tard, LMI nous prête un local qui nous permit de stocker les bandes et d'autres choses de ce type. »

En plus de fournir une vitrine, la FSF attira d'autres programmeurs désabusés. Le marché d'Unix qui avait semblé si collégial, même au moment de l'annonce initiale du projet GNU par Stallman, devenait de plus en plus concurrentiel. Dans leur tentative d'assurer leur emprise sur les clients, les compagnies commençaient à verrouiller l'accès au code source d'Unix, une tendance qui ne fit qu'accroître le nombre de requêtes concernant les projets de logiciels GNU en cours. Les magiciens d'Unix qui avaient par le passé considéré Stallman comme un idiot bruyant commençaient maintenant à le voir comme le Cassandre du logiciel.

« Un bon nombre de gens ne réalisent pas, jusqu'à ce que cela leur arrive, la frustration d'avoir travaillé plusieurs années sur un programme pour le voir finalement disparaître », dit Chassell, résumant par là les sentiments et les opinions des correspondants écrivant à la FSF pendant les premières années. « Puis, quand cela se reproduit, vous commencez à vous dire : 'Eh! attendez une minute...'. »

C'est l'expérience de sa propre spoliation qui décida Chassel à prendre part à la FSF. Avant LMI, Chassell avait été engagé pour écrire un livre d'introduction sur Unix pour Cadmus, Inc., une compagnie de logiciels située dans la région de Cambridge. Quand Cadmus a coulé, emportant dans sa chute les droits du

livre, Chassell a tenté en vain de les racheter.

« Pour ce que j'en sais, ce livre repose toujours sur une étagère quelque part, inutilisable, incopiable, simplement retiré du système », indique Chassell. « C'était vraiment une bonne introduction, si je puis me permettre. Cela n'aurait peut-être pris que trois ou quatre mois pour le transformer en une introduction parfaitement adaptée à GNU/Linux aujourd'hui. Tout ce travail, hormis ce qu'il me reste en mémoire, a été perdu. »

Condamné à voir son travail partir à la poubelle pendant que son ex-employeur luttait contre la faillite, Chassell affirme qu'il a alors eu un avant-goût de la colère qui conduisit Stallman aux bord de l'apoplexie. « Ce qu'il y avait pour moi de plus clair, c'était que si vous voulez avoir une vie bien remplie, vous ne tenez pas à en voir certains fragments disparaître à jamais », explique Chassell. « Cette idée même d'avoir la liberté d'entrer, de rectifier quelque chose et de le modifier, quoi que cela puisse être, fait vraiment une différence. On se réjouit alors, après avoir vécu quelques années, du fait que ce que l'on a accompli reste valable. Parce que sans cette liberté, votre travail est bon à être écarté, rejeté, abandonné et, en fin de compte, il ne vous en reste plus aucune trace. C'est comme perdre un peu de votre vie. »

Notes

1. Richard Stallman, *Initial GNU Announcement* (septembre 1983).
2. *Ibidem*.
3. *Ibidem*.
4. Marschall Kirk McKusik, *Twenty Years of Berkeley Unix, Open Sources*, O'Reilly & Associates, Inc., 1999, p. 38.
5. Richard Stallman (1986)
6. *Ibidem*.
7. Sources multiples : un interview de Richard Stallman, un courriel de Gerald Sussmann, et le Jargon File 3.0.0. http://www.clueless.com/jargon3.0.0/TWENEX.html
8. http://www.as.cmu.edu/~geek/humor/See_Figure_1.txt
9. *MIT AI Lab Tourist Policy*, in : http://catalog.com/hopkins/text/tourist-policy.html
10. Richard Stallman (1986).
11. *Ibidem*.
12. H. P. Newquist, *The Brain Makers: Genius, Ego, and Greed in the Quest for Machines that Think*, Sams Publishing, 1994, p. 172.
13. *Idem*, p.196.
14. *Ibidem*. L'anecdote racontée par Newquist, fut corroborrée par plusieurs membres de l'executif de Symbolics. Newquist écrit : « Le message causa une vague d'excitation et d'interrogation chez les employés de Symbolics, mais en fin de compte, personne ne prenait au sérieux le coup de colère de Stallman. »
15. http://www.as.cmu.edu/~geek/humor/See_Figure_1.txt
16. Steven Levy, *Hackers*, Penguin USA, 1984, p. 426.
17. Bill Gates, *An Open Letter to Hobbyists*, 3 février 1976. Pour voir une copie de cette lettre : http://www.blinkenlights.com/classiccmp/gteswhine.html
18. Richard Stallman, *Open Sources*, O'Reilly & Associates, Inc., 1999, p. 56. Stallman ajoute une note de bas de page à ce propos : « Je suis athée, je ne suis aucun chef religieux, mais parfois je les admire pour ce qu'ils disent ».
19. Richard Stallman (1986)
20. *Ibidem*.
21. Richard Stallman, *The GNU Manifesto*, 1985. http://www.gnu.org/gnu/manifesto.html

Chapitre VIII — Saint Ignucius

Le *Maui High Performance Computing Center* (Centre de Calcul Haute Performance de Maui, Hawaï — MHPCC) est situé dans un bâtiment de plain-pied, dans les collines rouges poussiéreuses juste au dessus de la ville de Kihei. Entouré par les paysages et les habitations des multi-millionnaires du club de golf de Silversword, le centre représente l'ultime gâchis de ressources scientifiques. Loin des confinements étroits de Tech Square, ou même des tentaculaires métropoles de recherche d'Argonne, Illinois, et Los Alamos, Nouveau Mexique, le MHPCC ressemble à un endroit où les scientifiques passent plus de temps sur leur bronzage que sur leurs travaux de recherches post-doctorales.

Cette image n'est qu'à moitié vraie. Bien que les chercheurs du MHPCC profitent effectivement des opportunités récréatives locales, ils prennent néanmoins leur travail au sérieux. Selon Top500.org, un site web qui référence les plus puissants supercalculateurs sur la planète, la machine IBM SP Power3 détenue par le MHPCC s'adjuge le score de 837 milliards d'opérations à virgule flottante par seconde (FLOP), entrant de fait dans le top 25 des plus puissants ordinateurs au monde. Propriété commune de l'Université d'Hawaï et de l'U.S. Air Force, la machine répartit ses cycles de calcul entre de nombreuses et lourdes tâches, liées à la logistique militaire et à la recherche physique dans les hautes températures.

En termes simples, le MHPCC est un lieu unique, un lieu où la culture cérébrale de la science et de l'ingénierie coexiste paisiblement avec la culture détendue des îles hawaïennes. Un slogan sur le site Internet du centre en 2000 résume tout : « Calculer au paradis ».

Ce n'est pas vraiment le genre d'endroit où l'on s'attendrait à trouver Richard Stallman, un homme qui, en regardant la magnifique vue de la toute proche passe de Maui à travers la fenêtre du bureau d'un membre de l'équipe, murmure une brève critique : « Trop de soleil ». Reste que, en tant qu'émissaire entre deux paradis de l'informatique, Stallman a un message à faire passer, même si cela implique de soumettre sa fragile peau laiteuse de hacker aux dangers d'une exposition tropicale.

La salle de conférence est déjà bondée lorsque j'arrive pour assister à la présentation de Stallman. La proportion entre les deux sexes est sensiblement plus équilibrée qu'à la conférence de New York : 85% masculin et 15% féminin. Mais elle reste globalement similaire. La moitié environ de l'assistance porte des pantalons kakis et des tee-shirts à logo. L'autre moitié semble retournée aux sources hawaïennes. Vêtus des ostentatoires chemises à fleurs si populaires dans ce coin du monde, leurs visages sont bronzés à l'extrême. Les seuls indices révélant leur statut de geek sont les gadgets : téléphones Nokia, Palm Pilots, et ordinateurs portables Sony VAIO.

Il va sans dire que Stallman, qui se tient devant cette assemblée habillé d'un simple tee-shirt bleu, d'un *baggy* marron et de chaussettes blanches, fait tache. L'éclairage au néon de la salle de conférences souligne la couleur maladive de cette peau qui ne voit le soleil que rarement. Sa barbe et ses cheveux sont suffisants pour provoquer des rivières de sueur, y compris du plus frais des cous hawaïens. Avec en quelque sorte le mot « métropolitain » tatoué sur son front, Stallman ne pourrait pas paraître plus étranger même s'il le souhaitait.

Alors que Stallman s'affaire du côté de la scène, quelques membres du public portant des tee-shirts au logo de *Maui FreeBSD Users Group* (MFUG) s'empressent de mettre en place l'équipement audio et vidéo. FreeBSD, un rejeton libre de la distribution logicielle de Berkeley, la vénérable version Unix académique des années 1970, est techniquement un compétiteur du système d'exploitation GNU/Linux. Cela n'empêche que, dans le monde du logiciel libre, les discours de Stallman sont documentés avec une ferveur proche de celle vue pour le groupe *Grateful Dead*, et sa légendaire armée d'archivistes amateurs. En tant que locaux du logiciel libre, il échoit aux membres du MFUG de s'assurer que les collègues programmeurs à Hamburg, Mumbai, et Novosibirsk ne ratent rien des dernières perles de la sagesse de RMS.

L'analogie avec *Grateful Dead* est brillante. Stallman l'utilise souvent pour décrire les possibilités de business inhérentes au modèle du logiciel libre. En refusant de restreindre la possibilité pour les fans d'enregistrer leurs concerts publics, les membres de *Grateful Dead* devinrent plus qu'un groupe de rock. Ils devinrent le centre d'une communauté tribale gravitant autour de leur musique. Au cours du temps, cette

communauté tribale devint si importante et dévouée, que le groupe put se passer de contrat avec une maison de disque, pour se financer avec les seuls tournées et concerts *live*. En 1994, pour leur dernière année de tournée, les Grateful Dead levèrent 52 millions de dollars grâce aux seules entrées des concerts.[1]

Alors que peu nombreuses sont les entreprises privées de logiciel ayant réussi à égaler un tel succès financier, l'aspect tribal de la communauté du logiciel libre est une raison pour laquelle beaucoup de personnes, dans la seconde moitié des années 1990, commencèrent à accepter l'idée que la publication du code source pourrait être une bonne chose. En espérant pouvoir constituer leurs propres troupes de dévots loyaux, des entreprises telles que IBM, Sun Microsystems, et Hewlett Packard en vinrent à accepter la lettre, sinon l'esprit, du message de Stallman sur le logiciel libre. En décrivant la GPL comme la « Magna Carta » de l'industrie des technologies de l'information, Evan Leibovitch, le chroniqueur informatique de ZDNet, voit l'affection grandissant pour tout ce qui est GNU comme quelque chose de plus qu'une simple tendance. « Ce changement sociétal permet aux utilisateurs de reprendre le contrôle de leur avenir », écrit Leibovitch. « Tout comme la *Magna Carta* donnait des droits aux sujets de l'Empire britannique, la GPL assure les droits et libertés au nom des utilisateurs de logiciels informatiques ».[2]

L'aspect tribal de la communauté du logiciel libre explique aussi pourquoi quarante programmeurs dépareillés, qui pourraient travailler sur des projets de physique ou encore parcourir l'Internet à la recherche d'informations météorologiques pour le windsurfing, ont préféré s'enfermer dans une salle de conférence afin d'écouter le discours de Richard Stallman.

Contrairement à la conférence de New York, Stallman n'est pas présenté par un tiers. Il ne propose d'ailleurs pas d'auto-présentation. Quand les gens de FreeBSD finissent par lancer leur équipement, Stallman s'avance simplement, commence à parler, et recouvre toute autre voix dans l'assistance par la sienne.

« La plupart du temps, quand les gens s'intéressent à la question des règles qu'une société devrait privilégier pour l'utilisation du logiciel, ceux qui s'occupent d'en débattre font partie d'entreprises vendant des logiciels, et ils considèrent la question d'un point de vue intéressé », dit Stallman, ouvrant son discours. « Quelles règles pouvons-nous imposer à tout un chacun afin qu'il soit obligé de nous payer un maximum d'argent? J'ai eu la chance, dans les années 1970, de faire partie d'une communauté de programmeurs qui s'échangeaient leurs logiciels. C'est pourquoi je préfère considérer systématiquement la question sous un angle différent et me demander quelle sorte de règles rendent-elles possible une société qui soit bonne pour ceux qui en font partie? J'en arrive à des conclusions complètement différentes. »

Encore une fois, Stallman revient rapidement sur l'anecdote de l'imprimante laser Xerox, se réservant un instant pour reproduire le même jeu qu'à New York, en prenant l'audience à partie. Il consacre aussi quelques minutes à l'explication du nom *GNU/Linux*.

« Certains me demandent : 'Pourquoi faire tant de bruit pour redonner à GNU le crédit que Linux lui a subtilisé? Après tout, le plus important est que le but soit atteint, pas que l'on sache grâce à qui il a été atteint'. Et bien, cela serait un sage conseil si c'était vrai. Mais le but n'était pas de construire un système d'exploitation ; le but est de répandre la liberté vers les utilisateurs d'ordinateur. Et pour cela, il nous faut rendre possible le fait de pouvoir tout faire avec son ordinateur, en toute liberté.[3] »

Et Stallman d'ajouter : « Il reste énormément de travail à faire. »

Pour certains dans le public, tout cela n'est pas nouveau. Pour d'autres, c'est un peu obscur. Quand un membre du clan des tee-shirts commence à piquer du nez, Stallman arrête son discours et demande à quelqu'un de réveiller l'étourdi.

« Quelqu'un m'a dit une fois que ma voix était si relaxante qu'il se demandait si je n'étais pas un quelconque guérisseur », dit Stallman, provoquant les rires du public. « Je pense que ça veut dire que je peux vous faire gentiment plonger dans un sommeil doux et paisible. Et certains d'entre vous pourraient en avoir besoin. Peut-être que je ne devrais pas m'opposer à ça. Si vous avez besoin de dormir, alors faites-le coûte que coûte. »

Le discours s'achève avec une discussion rapide sur les brevets logiciels, un problème de plus en plus omniprésent et qui touche à la fois l'industrie du logiciel, et la communauté du logiciel libre. Comme dans le cas *Napster*, les brevets logiciels reflètent la nature inopportune de la transposition de concepts créés pour le monde physique au nouvel univers des technologies de l'information.

La différence entre la protection d'un programme sous copyright et la protection d'un programme sous brevet logiciel est subtile mais importante. Dans le cas du copyright, un créateur de logiciel peut restreindre la reproduction du code source, mais pas celle de l'idée ou de la fonctionnalité relayée par ce code source. En d'autres termes, si un développeur décide de ne pas utiliser un logiciel suivant les termes initiaux posés par son créateur, il reste libre de rétro-ingénierer le programme. C'est-à-dire dupliquer la fonctionnalité originale du programme en écrivant un nouveau code source depuis une feuille blanche. Une telle habitude de copie des idées est chose commune dans l'industrie du logiciel commercial, où les entreprises isolent souvent leurs équipes de rétro-ingénierie afin de se prémunir des accusations d'espionnage ou de malhonnêteté des développeurs. Dans le jargon du développement logiciel moderne, les entreprises évoquent cette technique en employant le terme d'ingénierie en « salle propre ».

Les brevets logiciels fonctionnent différemment. Selon le bureau américain des brevets, des entreprises ou individus sont libres d'enregistrer des brevets pour des algorithmes innovants, à la condition de soumettre leurs propositions à la relecture publique. En théorie, cela permet au possesseur du brevet d'exploiter son invention en lui assurant un monopole limité à vingt ans à compter de la date de la rédaction du brevet. En pratique, la révélation de ces informations par le détenteur du brevet n'a qu'une valeur restreinte, puisque la façon de fonctionner du programme est souvent évidente. Contrairement au copyright, un brevet donne à son possesseur la capacité de décapiter les développeurs indépendants travaillant sur des logiciels aux fonctionnalités similaires ou identiques.

Dans l'industrie du logiciel, où 20 années peuvent couvrir le cycle de vie entier d'un marché, les brevets ont une importance stratégique. Alors que des entreprises comme Microsoft et Apple s'affrontaient sur le copyright, l'apparence et le ressenti de diverses technologies, les industries actuelles de l'Internet utilisent les brevets pour placer les applications individuelles et les modèles de business sur écoute. L'exemple le plus connu est la tentative par Amazon, en 2000, de breveter son processus d'achat en ligne "one-click". Pour la plupart des entreprises, cependant, les brevets logiciels sont devenus des armes défensives, avec des accords bidirectionnels équilibrant un portefeuille de brevets par rapport à l'autre, sur un qui-vive général avec de faux airs de détente corporatiste. Malgré tout, dans quelques cas notables de cryptage informatique ou d'algorithmes d'imagerie, des vendeurs de logiciels ont réussi avec succès à museler toute technologie rivale.

Pour Stallman, la question des brevets logiciels souligne, s'il en était besoin, l'importance d'une vigilance de tout instant de la part des hackers. Cela souligne aussi l'importance de d'élever les avantages politiques des programmes libres au dessus de leurs avantages commerciaux. En montrant du doigt la capacité des brevets logiciels à créer des zones hermétiques dans un marché, Stallman dit que la performance compétitive et le prix, deux domaines où des systèmes d'exploitation libres tels que GNU/Linux ou FreeBSD détiennent déjà un avantage conséquent sur leurs contreparties commerciales, sont de la poudre aux yeux comparés aux vastes problèmes de la liberté des utilisateurs et des développeurs.

« Ce n'est pas que nous soyons incapables d'écrire de meilleurs logiciels », dit Stallman. « Le fait est que nous n'en avons pas le droit : on nous interdit d'aider le public. Alors, que va-t-il se produire quand les utilisateurs se retrouveront confrontés aux lacunes du logiciel libre? Et bien, si le mouvement *open source* les a persuadés que ces libertés sont bonnes car elles permettent des logiciels plus puissants et fiables, ils diront probablement : « Vous ne tenez pas vos promesses. Ce logiciel n'est pas plus puissant. Il lui manque cette fonctionnalité. Vous m'avez menti ». Mais s'ils sont en accord avec le mouvement du logiciel libre, dans le fait que la liberté est importante en elle-même, alors ils diront, 'Comment ce fait-il que ces gens osent m'empêcher d'accéder à cette fonctionnalité et à ma liberté'. Et avec ce type de réponse, nous pourrons peut-être survivre aux coups qui nous atteindront lorsque ces brevets exploseront. »

Bien entendu, de tels commentaires suscitent un certain degré d'agitation. La plupart des avocats de l' *open source* sont autant, si ce n'est plus, acerbes que Stallman quand il s'agit de s'opposer aux brevets logiciels. Reste que la logique sous-jacente de l'argument de Stallman — à savoir que les avocats de l' *open source* insistent plus sur les avantages fonctionnels du logiciel libre que sur les avantages politiques — est incontestable. Au lieu de souligner le sens politique du logiciel libre, les avocats de l' *open source* ont choisi de mettre l'accent sur l'efficacité technique du modèle de développement de type hacker. Se référant à la puissance des comités de relecture, l'argumentaire de l'*open source* dépeint des programmes tels que GNU/Linux ou FreeBSD comme étant mieux construits, mieux inspectés et, par extension, plus fiables pour

l'utilisateur lambda.

Il ne s'agit pas de dire que le terme *open source* n'a pas d'implications politiques. Pour les défenseurs de l' *open source*, le terme sert à deux objectifs. D'abord, cela élimine la confusion associée au mot « libre », un mot que les entreprises interprètent comme signifiant « gratuit ». Ensuite, cela permet aux entreprises d'examiner le phénomène du logiciel libre sur un plan technique, et non éthique. Eric Raymond, cofondateur de l'Open Source Initiative, un des plus importants hackers à avoir adopté ce terme, a effectivement résumé sa frustration de suivre Stallman sur le chemin de la politique dans un article publié en 1999, intitulé *Fermez vos gueules et montrez-leur le code* :

> La réthorique de RMS est très séduisante pour le type de personnes que nous sommes. Nous, les hackers, sommes des penseurs et des idéalistes qui répondent automatiquement aux appels, aux « principes », à la « liberté » et aux « droits ». Même si nous sommes en désaccord avec certaines parties de son programme, nous voudrions que la réthorique de Stallman fonctionne, nous pensons qu'elle devrait fonctionner mais nous avons tendance à être perdus et incrédules quand elle n'a aucun impact sur 95% de la population qui n'est pas câblée de la même façon que nous.[4]

Parmi ces 95%, écrit Raymond, on retrouve le noyau des managers, investisseurs et utilisateurs simples d'ordinateur qui, au travers du poids clair des chiffres, tendent à décider la direction générale du marché du logiciel commercial. Sans moyen pour séduire ces gens, argumente Raymond, les programmeurs sont condamnés à développer leur idéologie en marge de la société :

> Quand RMS insiste pour que nous parlions des « droits des utilisateurs d'ordinateur », il nous fait une proposition dangereusement attractive qui nous conduirait à répéter les erreurs du passé. Nous devrions rejeter cette idée -- non parce qu'elle est fausse en principe, mais parce que ce type de discours, appliqué au logiciel, ne convainc personne sauf nous. En fait, cela déroute et refoule la plupart des gens étrangers à notre culture.[5]

En regardant Stallman dérouler son argumentaire politique en personne, il est difficile de voir quoi que ce soit de confus ou repoussant. Son apparence physique pourrait certes sembler repoussante, mais son message est logique. Quand un membre du public demande si, en évitant les logiciels propriétaires, les utilisateurs de logiciels libres renoncent aux joies des plus récentes avancées de la technologie, Stallman répond à la question selon ses propres convictions. « Je pense que la liberté est plus importante que les avancées technologiques », dit-il. « J'opterai toujours pour un logiciel libre légèrement en retard du point de vue technologique au détriment d'un logiciel propriétaire plus avancé, parce que je ne renoncerai pas à ma liberté pour une telle chose. Ma règle est que si je ne peux pas le partager librement, je ne le prends pas. »

De telles réponses, cependant, renforcent la nature quasi-religieuse du discours de Stallman. Comme un juif mangeant uniquement kasher ou un mormon refusant de boire de l'alcool, Stallman entoure sa décision d'utiliser du logiciel libre des couleurs de la tradition et de la croyance personnelles. Comme les évangélistes du logiciel, Stallman se refuse à inculquer de force ces préceptes sur son public. Mais encore une fois, le public de Stallman quitte rarement la salle sans savoir où se trouve le vrai chemin menant à la vertu logicielle.

Comme pour accompagner son message, Stallman ponctue son discours d'un rituel inhabituel. Sortant d'un sac plastique une robe noire, Stallman l'enfile. D'un second sac, il sort un disque informatique jaune réfléchissant et le place sur sa tête. Le public laisse échapper un rire soudain.

« Je suis St. Ignucius de l'Église Emacs », dit Stallman, levant sa main droite en simulant une bénédiction. « Je bénis ton ordinateur, mon enfant. »

Stallman habillé en St. Ignucius. Photo par Wouter van Oortmerssen.

Les rires deviennent applaudissements à tout rompre après quelques secondes. Alors que le public applaudit, le disque informatique sur la tête de Stallman accroche la lumière d'un spot de la scène, révélant une auréole parfaite. En un clin d'œil, Stallman passe du statut d'étranger bizarre à celui d'icône religieuse russe.

« Emacs était initialement un éditeur de texte », dit Stallman, expliquant la panoplie. « En fin de compte, c'est devenu un style de vie pour beaucoup, et pour certains une religion. Nous appelons cette religion l'Église Emacs. »

Le sketch est un moment léger d'auto-dérision, une réponse humoristique aux nombreuses personnes qui pourraient voir l'ascétisme logiciel de Stallman comme une forme déguisée de fanatisme religieux. C'est aussi le signe qu'il se sent définitivement comme à la maison. C'est comme si, en enfilant cette robe et cette auréole, Stallman rassurait finalement le public en disant : « C'est normal de rire. Je sais que je suis bizarre. »

Évoquant plus tard le personnage de St. Ignucius, Stallman raconte qu'il y avait d'abord pensé en 1996, longtemps après la création d'Emacs, mais bien avant l'émergence du terme *open source* et de la bataille pour la direction des communautés de hackers qui l'a engendrée. À l'époque, dit Stallman, il cherchait une façon de « se moquer de lui-même », pour rappeler à ceux qui voulaient bien l'entendre que, même s'il est obstiné, il n'était pas ce fanatique que certains dépeignaient. Ce n'est que plus tard, ajoute Stallman, que d'autres ont utilisé le personnage comme une façon pratique d'incarner sa réputation d'idéologue du logiciel, comme le fit Eric Raymond en 1999 dans une entrevue sur le site *linux.com* :

> Quand je dis que RMS calcule son action, je ne le rabaisse pas et je ne l'accuse pas de malhonnêteté. Je dis que, comme tout bon communicant, il a des sorties théâtrales. L'avez-vous déjà vu dans son accoutrement de St. Ignucius, bénissant le logiciel avec un disque en guise d'auréole? La plupart du temps, c'est fait inconsciemment ; il a juste appris à trouver le degré de stimulus énervant qui marche, qui permet de garder l'attention du public sans (généralement) lui faire perdre les pédales.[6]

Stallman n'est pas en accord avec l'analyse de Raymond. « C'est juste une façon de rire de moi-même », dit-il. « Si certains y voient plus que cela, c'est un reflet de leurs problèmes, pas des miens. »

Ceci dit, Stallman avoue aimer faire l'intéressant. « Vous rigolez? », dit-il à un moment. « J'adore être le centre d'attention ». Pour faciliter ce processus, Stallman raconte qu'il s'est inscrit une fois à *Toastmaster*, une organisation qui aide ses membres à fasciner leur public -- avec un travail sur l'expression orale que Stallman recommande chaleureusement à tous. Il possède une présence scénique qui rendrait jaloux la plupart des acteurs et rappelle dans un sens les vaudevilliens du passé. Quelques jours après la présentation au MHPCC, je fis allusion à sa prestation à LinuxWorld 1999, et lui demandai s'il avait un complexe de Groucho — c'est-à-dire le refus de faire partie de tout club qui aurait envie de le voir parmi ses membres. La réponse de Stallman est immédiate : « Non, mais j'admire Groucho Marx pour de nombreuses raisons, il m'a sûrement inspiré par certains aspects. Mais j'ai aussi été influencé par Harpo sur d'autres points. »

L'influence de Groucho Marx est évidente dans son affection de toujours pour les sous-entendus. Encore une fois, double sens et jeux de mots sont des traits communs chez les hackers. L'aspect le plus similaire à Groucho Marx pourrait être le sérieux avec lequel il lance ses blagues. La plupart arrivent si

furtivement, sans même l'indice d'un mouvement de sourcil ou d'un sourire retroussé, qu'on en vient presque à se demander si Stallman rit plus de son public que son public ne rit de lui.

En voyant les membres du MHPCC rire devant la parodie de St. Ignucius, ces considérations s'évaporent. Bien que n'étant pas à proprement parler un *one man show*, Stallman a de toute évidence ce qu'il faut pour tenir une pièce pleine d'ingénieurs en haleine. « Être un saint dans l'Église Emacs ne requiert pas le célibat, mais il est essentiel de s'astreindre à une vie de pureté morale », explique-t-il au public de Maui. « Vous devez exorciser les systèmes d'exploitation diaboliques et installer saintement un système opérationnel libre. Puis, vous devez installer uniquement des logiciels libres sur cette base. Si vous suivez cette discipline pour la vie, alors vous serez un saint de l'Église Emacs, et vous pourrez peut-être même avoir une auréole. »

Le sketch de St. Ignucius s'achève avec une brève blague pour les initiés. Sur la plupart des systèmes Unix et dérivés, le premier programme concurrent d'Emacs est Vi, que l'on prononce vi-aille, un éditeur de texte développé par un ancien étudiant de l'*UC Berkeley*, actuellement ingénieur en chef de Sun Microsystems, Bill Joy. Avant de reposer son « auréole », Stallman se moque du logiciel rival. « Les gens me demandent parfois si c'est un péché dans l'Église Emacs d'utiliser Vi », dit-il, « Utiliser une version libre de Vi n'est pas un péché, c'est une pénitence. Alors hackez bien. »

Après une brève séance de questions-réponses, les membres du public s'attroupent autour de Stallman. Certains demandent des autographes. « Je vais signer ça », dit Stallman, tenant devant lui une impression faite par une femme de la GNU General Public License, « mais seulement si vous promettez d'utiliser le terme GNU/Linux au lieu de Linux et dites à tous vos amis de faire de même. »

La remarque ne fait que confirmer une observation personnelle. Contrairement à tout autre personnage public ou homme politique, Stallman ne s'arrête jamais. A part le personnage de St. Ignucius, l'idéologue que vous voyez sur scène est le même dans la vie de tous les jours. Plus tard dans la soirée, durant une conversation à table, un programmeur évoque son affinité pour les logiciels *open source*. Stallman, entre deux bouchées, rabroue le convive : « Vous voulez dire logiciel libre. C'est la façon correcte d'y référer. »

Au cours de la séance de questions-réponses, Stallman admet jouer parfois le pédagogue. « Beaucoup affirment : 'Bon, invitons d'abord des gens à rejoindre la communauté, puis enseignons-leur ce que veut dire liberté.' Cela pourrait être une stratégie raisonnable mais, dans les faits, tout le monde invite des gens à rejoindre la communauté alors que presque personne ne s'occupe de parler de liberté une fois qu'ils sont là. »

Selon Stallman, le résultat ressemble à une ville du tiers-monde. Les gens arrivent, espérant faire fortune, ou au moins participer à une culture ouverte et dynamique, mais ceux qui détiennent le vrai pouvoir ne cessent jamais de dresser de nouveaux plans et pièges — par exemple les brevets logiciels — pour maintenir les masses hors jeu. « Il y a des millions de personnes arrivant et construisant des bidonvilles, mais personne ne s'intéresse à la seconde étape : sortir les gens de ces bidonvilles. Si vous pensez que parler des libertés logicielles est une bonne stratégie, s'il vous plaît, attelez-vous à la seconde étape. Nombreux sont ceux travaillant sur la première étape. Il nous faut plus de volontaires s'attelant à la seconde. »

Travailler sur la « seconde étape » signifie faire assimiler le fait que la liberté, et non la soumission, est le problème fondamental du mouvement du logiciel libre. Ceux qui espèrent réformer l'industrie du logiciel privatif depuis l'intérieur sont sur un chemin sans issue. « Le changement depuis l'intérieur est risqué », dit Stallman. « À moins de travailler à l'échelon d'un Gorbatchev, vous allez être neutralisé. »

Des mains se lèvent. Stallman désigne un membre du clan des tee-shirts. « Sans brevet logiciel, comment pensez-vous pouvoir gérer l'espionnage industriel? »

— « Voyez-vous, ces deux points n'ont vraiment rien à voir l'un avec l'autre », dit Stallman.

— « Mais je veux dire, si quelqu'un essaie de voler un morceau de logiciel d'une autre société. »

Stallman recule comme s'il est touché par un jet mortel. « Attendez un instant », dit Stallman. « Volé? Excusez-moi, il y a tellement de parti pris dans cet argument que la seule chose que je puisse faire est de le refuser. Les sociétés qui développent du logiciel propriétaire, entre autres choses, conservent énormément de secrets commerciaux, et cela ne changera sûrement jamais. À l'époque, même dans les années 1980, la plupart des programmeurs ne savaient même pas qu'il existait des brevets logiciels et n'y accordaient aucune attention. Ce qu'il se passait, c'est que les gens publiaient les idées intéressantes, et s'ils

ne faisaient pas partie du mouvement du logiciel libre, ils gardaient leurs petits secrets pour eux. Aujourd'hui, ils brevettent ces idées générales et gardent toujours secrets les petits détails. Jusque-là dans ce que vous décrivez, les brevets ne font vraiment aucune différence dans un sens ou dans l'autre. »

« Mais ça n'influence pas leur publication », dit un autre membre du public sautant sur l'occasion, avec une voix tremblante dès le début de son intervention.

« Mais si, cela influence », dit Stallman. « Leur publication vous dit que c'est une idée hors de portée du reste de la communauté durant vingt ans. Bon sang, mais comment cela peut-il être bénéfique? En plus, leur publication est écrite d'une manière si compliquée à lire, afin d'obscurcir l'idée et de rendre le brevet aussi général que possible, qu'il devient fondamentalement inutile de s'intéresser aux informations publiées pour apprendre quoi que ce soit. La seule raison de s'intéresser aux brevets consiste à apprendre de mauvaises nouvelles au sujet de ce que vous n'avez pas le droit de faire. »

Le public devient silencieux. Le discours, qui a débuté à 15h15, approche de la sonnerie des 17h, et la plupart des auditeurs remuent sur leur siège, impatients de débuter le week-end. Sentant cette fatigue, Stallman regarde les membres du public et met fin à la séance. « Donc il semble que nous en ayons fini », dit-il, enchaînant avec l'expression des commissaires-priseurs pour décourager toute question de dernière minute : « une fois, deux fois, adjugé ». Quand il voit que personne ne lève de main, Stallman conclut avec son expression habituelle :

— « Hackez bien. »

Notes

1. *Grateful Dead Time Capsule: 1985-1995 North American Tour Grosses*. http://www.accessplace.com/gdtc/1197.htm
2. Evan Leibovitch, *Who's Afraid of Big Bad Wolves*, ZDNet Tech Update (15 décembre 2000). http://techupdate.zdnet.com/techupdate/stories/main/0Y
3. Pour des raisons narratives, l'auteur a hésité à plonger dans les détails lorsqu'il a décrit la définition complète pour Stallman de la « liberté » logicielle. Le site Internet du Projet GNU liste quatre points fondamentaux : La liberté d'utiliser un programme, pour quelque but que ce soit (liberté 0). La liberté d'étudier la façon dont un programme fonctionne, et de l'adapter à vos besoins (liberté 1). La liberté de distribuer des copies d'un programme afin d'aider votre prochain (liberté 2). La liberté d'améliorer un programme, et de publier ces améliorations au public, afin que la communauté entière en bénéficie (liberté 3). Pour plus d'informations, veuillez consulter la Définition du logiciel libre : http://www.gnu.org/philosophy/free-sw.html.
4. Eric Raymond, *Shut Up and Show Them the Code*, online essay, (28 juin 1999). http://www.tuxedo.org/~esr/writings/shut-up-and-show-them.html. Lien NON VALABLE L ADRESSE EST : http://www.catb.org/~esr/writings/shut-up-and-show-them.html.
5. *Ibidem*.
6. *Guest Interview: Eric S. Raymond*, Linux.com (18 mai 1999). http://www.linux.com/interviews/19990518/8/

Chapitre IX — La licence publique générale GNU

Au printemps 1985, Richard Stallman était parvenu à la première étape du projet GNU : une version libre (basée sur Lisp) d'*Emacs*. Cependant, pour atteindre cet objectif, il fut confronté à deux problèmes. Dans un premier temps, il dut reconstruire *Emacs* pour le rendre interopérable puis, dans un second temps, il dut réorganiser la Communauté Emacs de manière similaire.

La dispute avec UniPress avait révélé une faille dans le contrat social de la Commune Emacs. Là où les utilisateurs s'en remettaient à l'excellente expertise de Stallman, les règles de la commune tenaient bon. Dans les domaines où Stallman n'occupait plus la position de hacker alpha (sur les systèmes Unix pré-1984 par exemple) les personnes et les entreprises étaient libres d'inventer leurs propres règles.

La tension entre la liberté de modifier le code et la liberté pour les auteurs d'exercer leurs privilèges s'est développée bien avant GOSMACS. Le Copyright Act signé en 1976 avait mis à niveau la loi américaine sur le copyright, en étendant cette protection légale aux logiciels. Selon la section 102(b) de ce document, les individus et les entreprises possédaient dorénavant le droit d'exercer leur copyright sur les « termes » d'un logiciel et non les « processus ou méthodes effectivement implémentés dans le programme [1] ». Si l'on essaye de traduire, les programmeurs et entreprises avaient dorénavant le droit de traiter les logiciels comme des livres ou des chansons. D'autres programmeurs pouvaient s'inspirer du travail sous copyright, mais pour le copier ou en réaliser un dérivé non-parodique, ils devaient en premier lieu obtenir la permission de l'auteur original. Bien que cette nouvelle loi garantisse que même les logiciels sans notices de copyright soient dotés d'une protection copyright, les programmeurs s'assurèrent vite de la protection de leurs droits en ajoutant de telles notices à leurs logiciels.

D'abord, Stallman les vit arriver avec inquiétude. Rares étaient les logiciels qui n'empruntaient pas des bouts de code source issus de tel ou tel programme préexistant, et pourtant, d'un coup de stylo, le président du congrès avait donné la possibilité aux programmeurs de s'assurer des droits exclusifs sur des logiciels construits par des communautés. On injectait aussi une dose de formalisme dans ce qui était auparavant un système informel. Même si les hackers pouvaient prouver comment les traînées de sang laissées par des logiciels remontaient et trouvaient leurs origines quelques années auparavant, si ce n'est quelques décennies, les ressources et l'argent qu'engloutissaient chaque bataille juridique sur les notices de copyright étaient hors de portée pour eux. En termes simples, des disputes qui se réglaient auparavant entre hackers se réglaient dorénavant entre avocats. Dans un tel système, les entreprises, et non les hackers, détenaient un avantage automatique.

Ceux qui défendaient le copyright logiciel avaient leurs contre arguments : sans copyright, les produits risquaient de se retrouver dans le domaine public. Mettre une notice de copyright sur un produit servait aussi en tant que preuve de qualité. Les développeurs et entreprises qui liaient leur nom au copyright engageaient aussi leur réputation. Finalement, c'était un contrat autant qu'une déclaration de propriété. En utilisant le copyright comme une forme flexible de licence, un auteur pouvait abandonner certains droits en échange de certains comportements de la part de l'utilisateur. Par exemple, un auteur pouvait renoncer à bloquer les copies non autorisées dans les cas où l'utilisateur renonçait à créer des dérivés commerciaux du produit.

C'est ce dernier argument qui adoucit probablement la réticence de Stallman face aux notices de copyright logiciel. En se penchant sur les années qui débouchèrent sur le projet GNU, Stallman rapporte qu'il a commencé à ressentir la nature bénéfique du copyright aux alentours de la sortie d'*Emacs 15.0*, la dernière mise à jour importante d' *Emacs* avant le Projet GNU. « J'avais vu des courriels avec des notices de copyright auxquelles étaient ajoutées de simples licences 'copies conformes permises' », se souvient Stallman. « Elles furent à coup sûr [une] inspiration. »

Pour *Emacs 15*, Stallman définit un copyright qui donnait aux utilisateurs le droit de faire et redistribuer des copies. Il donnait aussi aux utilisateurs le droit de créer des versions modifiées, sans toutefois pouvoir se rendre propriétaires exclusifs de ces copies modifiées, comme ce fut le cas pour GOSMACS.

Bien qu'utile pour codifier le contrat social de la Commune Emacs, dit Stallman, la licence *Emacs 15*

restait trop « informelle » pour les objectifs du Projet GNU. Peu après avoir commencé à travailler sur une version GNU d'*Emacs*, Stallman commença à consulter les autres membres de la Free Software Foundation (FSF) sur la façon d'arrêter les termes de la licence. Il consulta aussi les avocats qui l'avaient aidé à mettre en place la FSF.

Mark Fischer, un avocat de Boston spécialisé dans les lois sur la propriété intellectuelle, se rappelle de ses échanges de l'époque avec Stallman concernant la licence. « Richard avait des opinions très fortes concernant la façon dont cela devait fonctionner », dit Fischer. « Il avait deux principes. Le premier était de rendre le logiciel aussi ouvert que possible. Le second était d'encourager les autres à suivre les mêmes pratiques de publication. »

Encourager les autres à suivre les mêmes pratiques de publication voulait dire fermer la faille qui avait permis à des versions privées d'*Emacs* de voir le jour. Pour combler cette faille, Stallman et ses collègues du logiciel libre trouvèrent une solution : les utilisateurs seraient libres de modifier *GNU Emacs* tant qu'ils publiaient leurs modifications. En plus, les produits « dérivés » en résultants devaient automatiquement porter la même licence GNU Emacs. La nature révolutionnaire de cette dernière clause mit du temps à s'imposer. À l'époque, pour Fisher la licence GNU Emacs était un simple contrat. Cela donna un coût à l'utilisation de *GNU Emacs*. Au lieu d'argent, Stallman facturait aux utilisateurs l'accès à leurs propres modifications futures. Ceci dit, Fischer se souvient que ces termes de contrat étaient inédits.

« Je pense que demander ce prix aux autres était, sinon unique, très inhabituel à l'époque », dit-il.

La licence GNU Emacs fit ses débuts lorsque Stallman finit par publier *GNU Emacs* en 1985. Suite à cette publication, Stallman voulut connaître les retours de l'ensemble de la communauté hacker sur les moyens d'améliorer les termes de la licence. Parmi ceux qui répondirent se trouvait le futur activiste du logiciel John Gilmore, alors consultant chez Sun Microsystems. Dans le cadre de son emploi de consultant, il avait porté *Emacs* sur SunOS, la version maison d'Unix chez Sun. A cette occasion, Gilmore avait publié ses modifications comme le requérait la Licence GNU Emacs. Au lieu de voir la licence comme une responsabilité, Gilmore la concevait comme une expression claire et concise de l'esprit hacker. « Jusqu'alors, la plupart des licences étaient très informelles », se souvient Gilmore.

En illustration de ce non-formalisme, Gilmore cite une licence du milieu des années 1980 pour *trn*, un lecteur de nouvelles écrit par Larry Wall, un hacker qui trouva plus tard la célébrité en tant que créateur de l'utilitaire Unix *patch* et du langage de script *Perl*. Dans l'espoir de trouver un équilibre entre la courtoisie usuelle des hackers et le droit d'un créateur à pouvoir dicter les conditions de publication des versions commerciales, Wall utilisait la notice de copyright jointe comme une sonnante tribune éditoriale.

> Copyright (c) 1985, Larry Wall
> Vous pouvez copier entièrement ou en partie le kit trn tant que vous ne tentez pas d'en faire de l'argent, ou de prétendre que vous l'avez écrit [2].

De telles déclarations, bien que reflétant l'éthique hacker, exprimaient aussi la difficulté de traduire la nature floue et informelle de cette éthique vers le langage rigide et juridique du copyright. En écrivant la Licence GNU Emacs, Stallman avait fait plus que bloquer la faille permettant les dérivés commerciaux. Il avait traduit l'éthique hacker dans des termes compréhensibles à la fois par les avocats et par les hackers. Sollicité lors d'une conversation sur Usenet, Gilmore envoya un courriel à Stallman en novembre 1986, proposant des modifications :

> « Vous devriez probablement enlever EMACS de la licence et le replacer par LOGICIEL ou autre chose. Bientôt, on l'espère, *Emacs* ne sera pas la pièce principale du système GNU, et la licence s'appliquera à l'ensemble [3]. »

Gilmore n'était pas le seul à suggérer une approche plus générale. Fin 1986, Stallman lui-même travaillait sur la prochaine étape du Projet GNU, un débogueur de code source, et cherchait un moyen de retoucher la Licence Emacs afin qu'elle puisse s'appliquer aux deux logiciels. La solution de Stallman était de supprimer toutes les références explicites à *Emacs* et convertir la licence en un parapluie légal général pour le Projet GNU. La Licence Publique Générale GNU, GPL en abrégé (*General Public Licence*), était née.

En mettant au point la GPL, Stallman suivit la convention logicielle qui consiste à utiliser des

chiffres décimaux pour indiquer les versions prototypes et des nombres entiers pour indiquer les versions matures. En 1989, quatre ans après avoir défini le premier prototype, Stallman publia la version 1.0 de la GPL, la définissant comme licence officielle de *GNU Emacs* et du débogueur GNU, dit GDB, la seconde majeure incursion de Stallman dans l'univers de la programmation Unix. La licence contenait un préambule dévoilant ses intentions politiques :

> La Licence Publique Générale est faite pour s'assurer que vous ayez la liberté de donner ou vendre des copies de logiciel libre, que vous receviez le code source ou que vous puissiez y accéder si vous le souhaitez, que vous puissiez modifier le logiciel ou en inclure des parties dans de nouveaux logiciels libres, et que vous sachiez que vous pouvez faire toutes ces choses.
>
> Pour protéger vos droits, nous devons imposer des restrictions qui empêchent quiconque de vous disputer ces droits, ou même de vous demander d'y renoncer. Ces restrictions se traduisent par certaines responsabilités pour vous si vous distribuez des copies du logiciel, ou si vous le modifiez [4].

En créant la GPL, Stallman avait été forcé de faire un ajustement supplémentaire aux dogmes informels de l'ancienne Communauté Emacs. Là où il avait autrefois demandé que les membres de la Commune publient toutes leurs modifications, Stallman demandait à présent la publication dans les seuls cas où les programmeurs faisaient circuler publiquement, comme Stallman le faisait,leurs versions modifiées. En d'autres termes, les développeurs qui modifiaient *Emacs* pour leur simple usage personnel n'étaient plus tenus à envoyer les modifications du code source à Stallman. Dans ce qui deviendrait un rare compromis dans la doctrine du logiciel libre, Stallman tranchait dans le coût du logiciel libre. Les utilisateurs pouvaient innover sans que Stallman ne les observe par dessus l'épaule tant qu'ils n'interféraient pas avec Stallman et le reste de la communauté hacker pour la distribution du même logiciel.

Après coup, selon Stallman, le compromis de la GPL était motivé par son propre mécontentement vis à vis de l'aspect *Big Brother* du contrat social de la Communauté Emacs originelle. Bien qu'il aimât observer les systèmes des autres hackers, la pensée que quelque futur mainteneur de code source puisse utiliser ce pouvoir à des fins condamnables le força à adoucir la GPL.

« C'était mal de demander aux gens de publier toutes leurs modifications », dit Stallman. « C'était mal de leur demander de se référer à un développeur privilégié. Ce genre de centralisation et de privilège pour un seul individu n'était pas cohérent avec une société dans laquelle tous avaient les mêmes droits. »

Au fil des hacks, la GPL figure comme l'un des meilleurs coups de Stallman. Elle a créé un système de propriété collective à l'intérieur des habituels murs propriétaires de la loi sur le copyright. Mais surtout, elle a démontré la similarité intellectuelle entre code législatif et code logiciel. Implicitement, dans le préambule de la GPL réside un message profond : au lieu de considérer la loi sur le copyright logiciel avec suspicion, les hackers devraient plutôt la voir comme un système de plus ne demandant qu'à être hacké.

« La GPL s'est vraiment développée comme n'importe quel module de logiciel libre, avec une vaste communauté discutant sa structure, son respect ou les transgressions qu'ils observaient, et la nécessité d'ajustements ou de compromis légers pour une plus grande acceptation », dit Jerry Cohen, un autre avocat ayant aidé Stallman sur la genèse de la licence. « Le processus s'est très bien déroulé et la GPL dans ses diverses versions est passée du scepticisme général, voire quelquefois des réactions hostiles, à un assentiment unanime. »

En 1986, dans une entrevue pour le magasine *Byte*, Stallman résume la GPL en des termes imagés. En plus de proclamer les valeurs des hackers, dit Stallman, les lecteurs devraient aussi « la voir comme une forme de jiu-jitsu intellectuel, destiné à retourner contre les voleurs de logiciels le système légal que ceux-ci avaient mis en place [5] ». Des années plus tard, Stallman décrit la création de la GPL en des termes moins hostiles. "Je pensais à des problématiques qui étaient à la fois politiques et légales », dit-il. « J'ai dû faire ce qui pouvait être soutenu par le système législatif dans lequel nous sommes. Dans l'esprit, le boulot consistait à légiférer les bases d'une nouvelle société, mais puisque je n'étais pas un gouvernement, je ne pouvais changer aucune loi. J'ai tâché de le faire en m'appuyant sur le système législatif existant qui n'avait pas été conçu pour quoi que ce soit de semblable. »

À l'époque où Stallman soupesait les problèmes éthiques, politiques et légaux associés au logiciel

libre, un hacker californien nommé Don Hopkins lui envoya un manuel pour le microprocesseur 68000. Hopkins, un hacker Unix, comme lui passionné de science-fiction, lui avait emprunté le manuel quelques temps auparavant. En guise de remerciement, Hopkins avait décoré l'enveloppe retour avec des autocollants trouvés lors d'une convention locale de science-fiction. Un autocollant en particulier attira l'attention de Stallman. On y lisait : « Copyleft (L), Tous Droits Renversés ». Suivant la publication de la première version de la GPL, Stallman rendit hommage à l'autocollant, et surnomma la licence du logiciel libre « Copyleft ». Au cours du temps, le surnom et son symbole, un « C » renversé, devinrent le raccourci général de la FSF pour toute méthode de copyright « rendant un programme libre et requérant que toute version modifiée ou étendue du programme soit de même un logiciel libre ».

Un jour, le sociologue allemand Max Weber avança que toutes les grandes religions sont construites autour de la « routinisation » ou de l'« institutionnalisation » du charisme. Chaque religion ayant du succès, argumente Weber, convertit le charisme ou le message du leader religieux originel en un appareillage social, politique et éthique, plus aisément compréhensible et traduisible au cours du temps.

Bien que non religieuse en tant que telle, la GPL se qualifie sûrement comme un exemple de ce processus de « routinisation » à l'oeuvre dans le monde moderne décentralisé du développement logiciel. Depuis son introduction, des programmeurs et entreprises qui ont par ailleurs montré peu de loyauté ou d'allégeance à Stallman ont volontiers accepté le marché de la GPL à sa juste mesure. Quelques-uns ont même accepté la GPL comme mécanisme préemptif de protection pour leurs propres logiciels. Même ceux qui rejettent le contrat de la GPL comme étant trop contraignant reconnaissent son influence.

Keith Bostic était un hacker de ce dernier groupe, un employé de l'Université de Californie à l'époque de la sortie de la GPL 1.0. Le département de Bostic, le *Computer Systems Research Group* (Groupe de Recherche en Systèmes Informatiques — CSRG), avait été impliqué dans le développement d'Unix depuis la fin des années 1970 et était responsable de nombreuses parties clef d'Unix, dont le protocole réseau TCP/IP, la pierre angulaire de la communication Internet moderne. A la fin des années 1980, AT&T, le propriétaire originel de la marque Unix, commença à se concentrer pour commercialiser Unix, et s'intéressa à la *Berkeley Software Distribution*, dite BSD, la version académique d'Unix développée par Bostic et ses pairs à Berkeley, la considérant comme étant la clef d'une technologie commerciale.

Bien que le code source BSD de Berkeley fût partagé entre chercheurs et développeurs commerciaux, avec une licence pour ce code, cette commercialisation représentait un problème. Le code de Berkeley était entremêlé avec du code propriétaire AT&T. Par conséquent, les distributions Berkeley n'étaient disponibles que pour ceux qui avaient déjà une licence Unix chez AT&T. Comme AT&T augmentait le prix de ces licences, ce compromis, qui avait en premier lieu semblé inoffensif, devint de plus en plus lourd à supporter.

Engagé en 1986, Bostic s'était donné comme projet personnel de porter BSD sur l'ordinateur PDP-11 de chez DEC. C'est durant cette période, dit Bostic, qu'il entra en contact avec Stallman lors de ses incursions occasionnelles sur la Côte Ouest. « Je me souviens de discussions animées avec Stallman à propos du copyright, alors qu'il utilisait des stations de travail empruntées au CSRG », dit Bostic. « Nous allions dîner ensuite et continuions à discuter de copyright en mangeant. »

En fin de compte, ces arguments trouvèrent des oreilles réceptives, mais pas de la manière dont Stallman l'eut souhaité. En juin 1989, Berkeley sépara son code réseau du reste de la distribution propriété d'AT&T et le distribua sous la licence de l'Université de Californie. Les termes du contrat étaient libéraux. Tout ce qu'un licencié devait faire était d'aider l'université avec des publicités vantant des programmes dérivés [6]. En contraste avec la GPL, les dérivés propriétaires étaient possibles. Un seul problème empêcha l'adoption rapide de la licence : la publication du code réseau BSD n'était pas celle d'un système d'exploitation complet. Les gens pouvaient étudier le code, mais il ne pouvait être exécuté qu'en conjonction avec d'autres codes aux licences propriétaires.

Au cours des années suivantes, Bostic et les autres employés de l'Université de Californie travaillèrent à combler les parties manquantes pour transformer BSD en un système d'exploitation complet, librement distribuable. Bien que retardé par une bataille juridique avec *Unix Systems Laboratories* — la succursale d'AT&T qui possédait la propriété de la marque Unix — la démarche porta finalement ses fruits au début des années 1990. Bien avant cette date, cependant, de nombreux utilitaires de Berkeley firent leur apparition dans le Projet GNU de Stallman.

« Je pense qu'il est hautement improbable que nous ayons avancé aussi bien que nous l'avons fait sans l'influence de GNU », dit Bostic, rétrospectivement. « C'était une chose sur laquelle ils travaillaient vraiment dur, et nous aimions l'idée. »

À la fin des années 1980, la GPL commençait à exercer un effet gravitationnel sur la communauté du logiciel libre. Il n'était pas nécessaire à un logiciel de porter la GPL pour se qualifier de libre — il suffit de constater le cas des utilitaires BSD — mais mettre un programme sous GPL envoyait un message définitif. « Je pense que l'existence même de la GPL a encouragé les gens à penser clairement au fait qu'ils faisaient du logiciel libre et aux méthodes de publication de leur travail », dit Bruce Perens, le créateur d' *Electric Fence* (un utilitaire Unix populaire) et futur leader de l'équipe de développement de Debian GNU/Linux. Quelques années après la publication de la GPL, Perens décida d'abandonner la licence maison d' *Electric Fence* en faveur du copyright de Stallman vérifié par des avocats. « C'était en fait très facile à faire », se souvient Perens.

Rich Morin, le programmeur qui avait considéré l'annonce de GNU par Stallman avec un certain scepticisme, se souvient avoir été impressionné par le nombre de logiciels qui commençèrent à s'amasser sous la bannière de la GPL. En tant que leader d'un groupe d'utilisateurs de SunOS, une des tâches principales de Morin durant les années 1980 avait été d'envoyer des cassettes contenant les meilleurs utilitaires gratuits ou libres. Ce travail impliquait souvent de devoir téléphoner aux auteurs des logiciels pour vérifier si leurs programmes étaient protégés par copyright ou s'ils avaient été publiés dans le domaine public. Aux alentours de 1989, dit Morin, il commença à remarquer que les meilleurs logiciels tombaient généralement sous la licence GPL. « En tant que distributeur de logiciel, dès que je voyais les lettres GPL, je savais que j'étais comme à la maison », se souvient Morin.

Pour compenser les efforts qu'il mettait dans la compilation des cassettes pour le groupe d'utilisateurs Sun, Morin demandait un peu d'argent aux destinataires. À présent, avec les logiciels qui migraient sous GPL, Morin mettait subitement deux fois moins de temps à faire ses cassettes, faisant même un peu de profit au passage. Sentant l'opportunité commerciale, Morin transforma son hobby en une entreprise : *Prime Time Freeware*.

De telles exploitations commerciales s'inscrivaient parfaitement dans le cadre du développement du logiciel libre. « Quand nous parlons de logiciel libre, nous nous référons à la liberté, pas à la gratuité », prévient Stallman dans le préambule de la GPL. A la fin des années 1980, Stallman avait trouvé un simple moyen mnémotechnique : « Ne pensez pas libre comme dans bière gratuite, mais comme dans liberté d'expression. »

La plupart des entreprises ignorèrent les sollicitations de Stallman. Cependant, pour quelques entrepreneurs, la liberté associée au logiciel libre était la même liberté que celle associée aux marchés libres. Sortez la propriété logicielle de l'équation, vous obtenez une situation où même la plus petite des sociétés de logiciels est en mesure de rivaliser avec les IBM et les DEC du monde entier.

Un des premiers entrepreneurs à saisir ce concept fut Michael Tiemann, un développeur étudiant à l'Université de Stanford. Durant les années 1980, Tiemann avait suivi le Projet GNU comme un aspirant musicien Jazz suivant son artiste préféré. Ce n'est cependant pas avant la publication du compilateur C GNU, dit GCC, en 1987, qu'il commença à comprendre le plein potentiel du logiciel libre. Définissant GCC comme une « bombe », Tiemann dit que l'existence même de ce logiciel souligne la détermination de Stallman en tant que programmeur.

« Tout comme chaque écrivain rêve d'écrire LE grand roman américain, chaque programmeur à l'époque des années 1980 parlait d'écrire LE grand compilateur américain », se souvient Tiemman. « Tout à coup, Stallman l'avait fait. C'était une leçon de modestie. »

« Si on évoque individuellement les points où l'on était en échec, GCC en faisait partie », enchérit Bostic. « Personne n'avait de compilateur à l'époque, avant que GCC n'entre dans la danse. »

Au lieu d'entrer en concurrence avec Stallman, Tiemann décida de développer son travail. La version originelle de GCC pesait 110.000 lignes de code, mais Tiemann se souvient d'un logiciel relativement facile à comprendre. Si facile en fait qu'il lui fallut moins de cinq jours pour le maîtriser et une autre semaine pour le porter sur une nouvelle plateforme matérielle, le microprocesseur *National Semiconductor 32032*. Au cours

de l'année suivante, Tiemann commença à jouer avec le code source, créant un compilateur natif pour le langage C++. Un jour, au cours d'un exposé aux Laboratoires Bell, il croisa des développeurs de chez AT&T luttant pour atteindre le même objectif.

« Il y avait 40 ou 50 personnes dans la salle, et j'ai demandé combien travaillaient sur le compilateur natif », se souvient Tiemann. « Mon hôte me dit que l'information était confidentielle, mais ajouta que si je regardais dans la salle, je pouvais m'en faire une bonne idée générale. »

Peu après, poursuit Tiemann, une ampoule s'alluma dans sa tête. « J'avais travaillé sur ce projet pendant six mois. Je me suis simplement dit, que ce soit moi ou le code, voilà un niveau d'efficacité que le marché libéralisé devrait être prêt à récompenser. »

Tiemann trouva une inspiration supplémentaire dans le Manifeste GNU, qui, bien que fustigeant l'avarice de certaines entreprises de logiciels, encourage les autres à considérer les avantages du logiciel libre du point de vue de l'utilisateur. En supprimant le pouvoir du monopole de la question du logiciel commercial, la GPL offre la possibilité aux acteurs les plus astucieux de se battre sur le marché des services et du consulting, les deux niches les plus profitables du marché des logiciels.

Dans un essai daté de 1999, Tiemann rappelle l'impact du Manifeste de Stallman. « Ça ressemblait à de la polémique socialiste, mais j'y ai vu quelque chose de différent. J'y ai vu un business-plan caché [7]. »

Faisant équipe avec John Gilmore, un autre fan du Projet GNU, Tiemann lança un service de consulting logiciel dédié à la personnalisation des logiciels GNU. Nommé *Cygnus Support*, l'entreprise signa son premier contrat de développement en 1990. À la fin de l'année, l'entreprise avait engrangé 725.000 dollars en contrats de support et de développement. *GNU Emacs*, *GDB* et *GCC* étaient les trois grands outils orientés développeurs, mais ils ne furent pas les seuls développés par Stallman durant la première décennie du Projet GNU. En 1990, Stallman avait aussi développé des versions GNU du Bourne Shell (rebaptisé le *Bourne Again Shell*, dit *BASH*), *YACC* (rebaptisé *Bison*) et *awk* (rebaptisé *gawk*). Comme *GCC*, chaque logiciel GNU devait être construit pour pouvoir fonctionner sur de multiples systèmes, et non sur une seule plateforme propriétaire. Dans le processus qui rendait les logiciels plus flexibles, Stallman et ses collaborateurs les rendirent souvent plus utiles aussi.

Rappelant l'approche universaliste de GNU, Morin, de chez *Prime Time Freeware*, désigne un paquet logiciel essentiel, quoique terre à terre, intitulé *hello*. « C'est le programme *hello world*, qui consiste en cinq lignes de C, empaqueté comme s'il s'agissait d'une distribution GNU », dit Morin. « Par conséquent, il a toute l'information *Texinfo* et *configure*. Il a tous les bons outils d'ingénierie logicielle que le Projet GNU a inventé pour permettre le portage vers les autres environnements de façon aisée. C'est un travail extrêmement important, et cela n'affecte pas seulement tous les logiciels de Stallman, mais aussi tous les autres logiciels du Projet GNU. »

Selon Stallman, améliorer les logiciels venait en second lieu, après les avoir construits. « Pour chaque morceau je pourrais trouver ou non une façon de l'améliorer », dit Stallman lors de son entrevue pour *Byte*. « Dans une certaine mesure, je bénéficie de la réimplémentation, qui permet l'amélioration de beaucoup de systèmes. Dans une autre mesure, c'est parce que je suis depuis longtemps dans le milieu et que j'ai travaillé sur de nombreux autres systèmes. J'ai donc de nombreuses idées à mettre à l'épreuve [8]. »

Malgré tout, alors que les outils GNU marquèrent la fin des années 1980, la réputation de Stallman acquise au AI Lab pour sa minutie dans la conception devint légendaire à travers la communauté du développement logiciel toute entière.

Jeremy Allison, utilisateur de Sun à la fin des années 1980, et programmeur destiné à lancer son propre projet de logiciel libre, *Samba*, dans les années 1990, se souvient de cette réputation avec amusement. À la fin des années 1980, Allison commença à utiliser *Emacs*. Inspiré par le modèle de développement communautaire de ce programme, dit-il, il proposa une modification du code source, pour la voir rejetée par Stallman.

« C'était comme le titre d'une mauvaise publication », dit Allison. « 'Dieu répond aux prières d'un enfant : Non.' »

La stature grandissante de Stallman en tant que programmeur étaient toutefois contrebalancée par ses

combats en tant que gestionnaire de projet. Bien que le Projet GNU avançât de succès en succès dans la création d'outils de développement, son incapacité à produire un noyau fonctionnel, le policier central réglant la circulation et qui détermine quel programme a accès au microprocesseur dans tout système Unix, commençait à soulever des mécontentements avant même la fin des années 1980.

Comme pour la plupart des développements du Projet GNU, Stallman avait initié le développement du noyau en cherchant un programme existant à modifier. Une lecture des *GNUsletters* du Projet GNU à la fin des années 1980 révèle que cette approche, tout comme l'essai initial de construire *GCC* sur *Pastel*, était tout sauf idéale. Une *GNUsletter* de janvier 1987 rapporta que le Projet GNU travaillait à mettre à niveau *TRIX*, un noyau Unix développé au MIT. En février 1988, selon une autre newsletter, le Projet GNU avait changé son fusil d'épaule pour *Mach*, un « micro-noyau » léger développé à *Carnegie Mellon*. Ceci dit, le développement officiel du noyau GNU ne commenca pas avant 1990 [9].

Les délais dans le développement du noyau n'étaient qu'un des soucis pesant sur l'esprit de Stallman à cette époque. En 1989, *Lotus Development Corporation* poursuivit en justice un des ses rivaux, *Paperback Software International*, pour avoir copié les commandes de menus du logiciel tableur populaire, Lotus 1-2-3. L'action en justice de Lotus, ajoutée à la bataille Apple-Microsoft sur « l'aspect et le ressenti », offraient un arrière-plan problématique au Projet GNU. Bien que les deux actions fussent en dehors de son champ, elles tournaient toutes deux autour des systèmes d'exploitation et applications développés pour les ordinateurs personnels, et non pas les systèmes matériels compatibles Unix — elles menaçaient de produire un effet paralysant sur la culture toute entière du développement logiciel. Déterminé à faire quelque chose, Stallman recruta quelques amis programmeurs et composa une publicité dans les magazines pour s'élever contre ces actions en justice. En plus de ces publicités il organisa un groupe de protestation contre les entreprises menant ces actions. Appelé « Ligue pour la liberté de programmer », le groupe mena des actions devant les bureaux de Lotus Inc. et devant le tribunal de Boston qui accueillait le procès Lotus.

Ces manifestations étaient remarquables [10]. Elles illustrent la nature évolutive de l'industrie du logiciel. Les applications avaient tranquillement remplacé les systèmes d'exploitation comme cheval de bataille. Dans sa quête inachevée pour construire un système d'exploitation libre, le Projet GNU semblait désespérément en retard. En effet, le fait même que Stallman ait ressenti le besoin de mettre en place une équipe pour contrer les poursuites juridiques sur « l'aspect et le ressenti » avait renforcé cette idée d'obsolescence aux yeux de certains observateurs.

En 1990, la Fondation John D. et Catherine T. MacArthur valida le statut de génie accordé à Stallman en l'élevant au rang de membre de la confrérie MacArthur, le rendant ainsi récipiendaire de la fameuse *Genius Grant*, une récompense de 240.000 dollars pour avoir lancé le Projet GNU et avoir donné corps à la philosophie du logiciel libre, ce qui dissipa quelques préoccupations à court terme. D'abord et surtout, cela donna à Stallman, alors bénévole de la FSF et subvenant à ses besoins grâce à des contrats de consulting, la possibilité de dédier davantage de son temps à l'écriture de code source pour GNU [11].

Étonnamment, le prix permit aussi à Stallman de voter. Quelques mois auparavant, un incendie dans l'appartement de Stallman l'avait dépouillé de ses quelques possessions de jeunesse. Stallman se présentait alors lui-même comme « squatteur » .

Il est intéressant de noter que le succès final du Projet GNU et du mouvement pour le logiciel libre en général vint d'un des voyages de Stallman. En 1990, il rendit visite à l'Université polytechnique d'Helsinki en Finlande. Au sein du public se trouvait Linus Torvalds, alors agé de 21 ans, futur développeur du noyau Linux — le noyau libre permettant de combler la faille la plus large du Projet GNU.

Alors étudiant à l'université voisine d'Helsinki, Torvalds considéra Stallman avec amusement. « Je vis, pour la première fois de ma vie, le stéréotype du hacker barbu à cheveux longs », se souvient dans son autobiographie [12]. « Nous n'en avons pas beaucoup à Helsinki [13]. »

Bien que peu familier avec le côté « sociopolitique » de la stratégie de Stallman, Torvalds en apprécia cependant la logique sous-jacente : aucun programmeur n'écrit de code sans commettre d'erreur. En partageant le code source, les hackers plaçaient l'amélioration d'un programme avant toute motivation comme la défense de l'ego ou l'avarice.

Comme de nombreux programmeurs de sa génération, Torvalds ne s'était pas fait les dents sur des

ordinateurs *mainframe* comme l'IBM 7094, mais sur un assortiment bigarré de systèmes informatiques artisanaux. En tant qu'étudiant à l'université, Torvalds avait fait sa découverte de la programmation PC sur Unix, avec le MicroVAX de l'université. Cette progression pas à pas avait offert à Torvalds une perspective différente sur les barrières liées à l'accès aux machines. Pour Stallman, les barrières majeures étaient la bureaucratie et les privilèges. Pour Torvalds, c'était la géographie et l'hiver rude d'Helsinki. Obligé de cheminer à travers l'Université d'Helsinki à seule fin de se connecter sur son compte Unix, Torvalds commença rapidement à chercher une façon de pouvoir se connecter depuis les limites chauffées de son appartement situé hors du campus.

Cette quête conduit Torvalds vers le système d'exploitation Minix, une version allégée d'Unix développée à des fins pédagogiques par le professeur d'université hollandais Andrew Tanenbaum. Le programme était suffisamment léger pour pouvoir résider dans la mémoire d'un PC 386, la machine la plus puissante que Torvalds pouvait s'offrir, mais il lui manquait encore certaines fonctionnalités nécessaires. Il manquait en particulier à Minix l'émulation du terminal, la fonctionnalité permettant à Torvalds de mimer le comportement d'un terminal universitaire, rendant possible la connexion vers le MicroVAX depuis sa maison.

Durant l'été 1991, Torvalds réécrivit Minix en partant d'une page blanche, ajoutant d'autres fonctionnalités dans le processus. A la fin de l'été, Torvalds parlait de son travail en cours comme le « *GNU Emacs* des programmes d'émulation de terminal [14] ». Se sentant sûr de lui, il sollicita un groupe de discussion Minix pour trouver des copies des standards POSIX, les spécifications qu'un logiciel doit suivre pour être compatible avec Unix. Quelques semaines plus tard, Torvalds envoyait un message rappelant étrangement le message originel de Stallman sur GNU en 1983 :

> Bonjour à tous ceux ici utilisant minix-
> Je crée un système d'exploitation (libre, juste un passe-temps, ça ne sera pas aussi grand et professionnel que GNU pour clones 386 (486) AT). Ça a mijoté depuis avril, et commence à être prêt. J'aimerais des retours sur ce que les gens aiment ou n'aiment pas dans Minix en général, puisque mon système d'exploitation y ressemble d'une certaine façon (même arrangement physique du système de fichiers (pour des raisons pratiques) entre autres choses) [15].

Le message reçut quelques réponses et après un mois, Torvalds avait publié la version 0.01 du système d'exploitation — c'est-à-dire la toute première version convenant à la lecture extérieure — sur un site Internet FTP. Dans le mouvement, Torvalds avait trouvé un nom pour ce nouveau système. Sur son propre disque dur, il avait sauvegardé le programme sous le nom *Linux*, un nom qui respecte la convention logicielle qui consiste à donner à chaque variante d'Unix un nom finissant avec la lettre X. Trouvant le nom trop « egotistique », Torvalds le changea en *Freax*, pour finalement voir le responsable du site FTP le renommer avec le premier nom.

Bien que Torvalds ait décidé de construire un système d'exploitation complet, lui et les autres développeurs savaient à l'époque que la plupart des outils fonctionnels requis pour cela étaient déjà disponibles, grâce au travail de GNU, BSD, et des autres développeurs du logiciel libre. Un des premiers outils dont l'équipe de développement de Linux tira avantage fut le compilateur C GNU, un outil rendant possible l'exécution de programmes écrits en langage C.

Intégrer *GCC* améliora la performance de Linux. Mais cela souleva aussi des problèmes. Bien que la puissance « virale » de la GPL ne s'appliquât pas au noyau Linux, la volonté de Torvalds à emprunter GCC pour l'intégrer dans sa propre distribution logicielle libre le contraignait en quelque sorte à laisser aux autres la possibilité d'emprunter en retour. Comme Torvalds le dirait plus tard : « Je m'étais élevé sur les épaules de géants [16]. Sans surprise, il commença à se demander ce qui pouvait advenir si d'autres personnes venaient vers lui en cherchant une aide similaire. Une décennie après sa décision, Torvalds fait écho à Robert Chassel de la FSF lorsqu'il résume ses pensées du moment :

> Vous mettez six mois de votre vie dans ce travail, vous voulez le rendre public et en retirer quelque chose en retour, mais vous ne voulez pas que les gens puissent en tirer avantage. Je voulais que les gens puissent voir [Linux], et qu'ils puissent faire des modifications jusque dans ses entrailles. Mais je voulais aussi m'assurer que ce que j'obtenais en retour était la possibilité de voir ce que les gens en faisaient. Je voulais pouvoir avoir un accès constant aux sources afin que, s'ils faisaient des

améliorations, je puisse reproduire ces améliorations moi-même [17].

Quand vint le temps de publier la version 0.12 de Linux, la première intégrant un compilateur *GCC*, Torvalds décida de déclarer son allégeance au mouvement du logiciel libre. Il remisa l'ancienne licence et la remplaça par la GPL. La décision provoqua un florilège de portages, alors que Torvalds et ses collaborateurs s'intéressaient à d'autres programmes de GNU pour les inclure dans la croissante marmite Linux. Après quelques années, les développeurs de Linux proposèrent leur première version de publication, Linux 1.0, comprenant des versions complètement modifiées de *GCC*, *GDB*, et un hôte pour les outils BSD.

En 1994, cette distribution logicielle agglomérée avait obtenu suffisamment de respect dans le monde des hackers, au point que certains observateurs se demandèrent si Torvalds n'avait pas lâché la poule aux oeufs d'or en optant pour la GPL durant les premiers mois du projet. Dans le premier numéro du *Linux Journal*, l'éditeur Robert Young s'assit avec Torvalds pour un entretien. Quand Young demanda au programmeur finlandais s'il regrettait d'avoir abandonné la propriété exclusive du code source de Linux, Torvalds dit non. « Même avec une compréhension totale de la situation », Torvalds considérait la GPL « comme une des meilleures stratégies de conception » adoptée durant les premiers pas du projet Linux [18].

Que la décision ait été prise sans aucun appel ou déférence à Stallman ou à la FSF parle en faveur de la portabilité grandissante de la GPL. Bien que cela prît quelques années à Stallman pour le reconnaître, le boom du développement de Linux rappelait les souvenirs d'*Emacs*. Cette fois cependant, l'innovation initiatrice de l'explosion n'était pas un hack logiciel comme *Control-R* mais la nouveauté de pouvoir faire fonctionner un système similaire à Unix sur l'architecture PC. Les motivations peuvent avoir été différentes, mais le résultat final convient sûrement aux attentes éthiques : une système d'exploitation complet et fonctionnel composé entièrement de logiciels libres.

Comme son message initial au groupe de discussion *comp.os.minix* l'indique, il faudra quelques mois avant que Torvalds considère Linux comme autre chose qu'un ersatz en attendant que les développeurs GNU sortent le noyau *HURD*. Cette volonté initiale de ne pas voir Linux en termes politiques fut un véritable camouflet pour la FSF.

En ce qui concerne Torvalds, il était le dernier dans une longue lignée d'enfants démontant et ré-assemblant des choses pour s'amuser. Cependant, en résumant le succès galopant d'une réussite qui aurait pu passer le reste de ses jours sur un vieux disque dur abandonné, Torvalds se félicite d'avoir, dans sa jeunesse, eu la sagesse d'abandonner le contrôle sur son code et d'accepter le marché qu'offrait la GPL.

Torvalds se rappelle du discours de Stallman à l'Université polytechnique en 1991, ainsi que de sa décision subséquente d'opter pour la GPL. Il écrit : « Je n'ai peut-être pas vu la lumière, mais je pense que quelque chose dans son discours s'est ancré en moi [19]. »

Notes

1. Hal Abelson, Mike Fischer, et Joanne Costello, *Software and Copyright Law*, version mise à jour (1998). http://www.swiss.ai.mit.edu/6805/articles/int-prop/software-copyright.html
2. *Trn Kit README*. http://www.za.debian.org/doc/trn/trn-readme
3. John Gilmore, citation d'un courriel à l'auteur.
4. Richard Stallman, et al., *GNU General Public License: Version 1* (février 1989). http://www.gnu.org/copyleft/copying-1.0.html
5. David Betz et Jon Edwards, *Richard Stallman discusses his public-domain [sic] Unix-compatible software system with BYTE editors*, BYTE (juillet 1996) (Republié sur le site Internet du Projet GNU : http://www.gnu.org/gnu/byte-interview.html.). Cette interview présente un aspect intéressant, si ce n'est candide, des attitudes politiques de Stallman durant les premières années du Projet GNU. Elle est aussi utile pour retracer l'évolution de la rhétorique de Stallman. Décrivant l'objectif de la GPL, Stallman dit, « J'essaye de modifier la façon dont les gens abordent le savoir et la technologie en général. Je pense qu'essayer d'accaparer la propriété du savoir, essayer de contrôler la façon dont les gens peuvent y accéder et le redistribuer, est du sabotage. » Mettons cela en rapport avec une déclaration de l'auteur en Aout 2000: « Je vous conjure de ne plus utiliser l'expression 'propriété intellectuelle' dans vos réflexions. Elle vous entraîne à la confusion car elle fait l'amalgame entre

copyrights, brevets, et marques déposées. Ces choses sont tellement différentes dans leurs effets qu'il devient stupide d'essayer d'en parler en les confondant. Si vous entendez parler quelqu'un à propos de propriété intellectuelle, sans les guillemets, alors il ne pense pas très clairement, et vous ne devriez pas le suivre sur ce terrain. »

6. L'« odieuse clause de publicité » de l'Université de Californie devint par la suite un problème. Cherchant une alternative moins restrictive à la GPL, certains hackers utilisèrent celle de l'Université de Californie, en remplaçant « Université de Californie » par le nom de leur propre institution. Le résultat fut que les logiciels libres qui empruntaient du code à des douzaines d'autres logiciels libres devaient citer les publicités de douzaines d'institutions différentes. En 1999, après une décennie de lobbyisme de la part de Stallman, l'Université de Californie accepta d'abandonner cette clause. Cf *The BSD License Problem* sur http://www.gnu.org/philosophy/bsd.html
7. Michael Tiemann, *Future of Cygnus Solutions: An Entrepreneur's Account*, Open Sources (O'Reilly & Associates, Inc., 1999): 139. http://www.oreilly.com/catalog/opensources/book/tiemans.html
8. Richard Stallman, *Byte* (1986).
9. *HURD History*. http://www.gnu.org/software/hurd/history.html
10. Selon un communiqué de la Ligue pour la Liberté de Programmer, les manifestations furent remarquables pour avoir introduit le premier chant de protestation en hexadécimal : - 1-2-3-4, jetez les avocats par la fenêtre ; - 5-6-7-8, innover et non légiférer ; - 9-A-B-C, 1-2-3 n'est pas pour moi ; - D-E-F-O, l'aspect et le ressenti doivent s'en aller.
(http://lpf.ai.mit.edu/Links/prep.ai.mit.edu/demo.final.release)
11. J'utilise le terme « écriture » assez vaguement. En effet, à l'époque du prix MacArthur, Stallman avait commencé à souffrir de douleurs chroniques dans ses mains et dictait son travail à des sténographes employés par la FSF. Bien que certains aient pensé que ces douleurs étaient le résultat de troubles musculosquelettiques, une maladie commune chez les programmeurs, Stallman n'en est pas sûr à 100%. « Ce n'était PAS le syndrome du tunnel carpien », écrit-il. « Mon problème aux mains était localisé dans les mains, pas dans les poignets ». Stallman a depuis appris à travailler sans forcer sur ses poignets après avoir opté pour un clavier au toucher plus sensible.
Dorothea Salo, une lectrice, ajoute le commentaire suivant : « Les troubles musculo-squelettiques sont un terme générique couvrant une vaste variété de problèmes nerveux, musculaires et ligamenteux, causés par les mouvements répétitifs comme ceux que l'on retrouve lorsque l'on tape au clavier. Stallman n'avait probablement pas le syndrome du tunnel carpien, mais il avait sûrement une quelconque forme de trouble musculo-squelettique. (La description des ses symptômes est vague, mais le syndrome du tunnel cubital pourrait correspondre). »
12. Voir Linus Torvalds et David Diamond, *Just For Fun: The Story of an Accidentaly Revolutionary* (HarperCollins Publishers, Inc., 2001). Trad. fr.: Linus Torvalds et David Diamond, *Il était une fois Linux. L'extraordinaire histoire d'une révolution accidentelle*, Paris, Osman Eyrolles Multimédia, 2001.
13. Linus Torvalds et David Diamond, *Just For Fun: The Story of an Accidentaly Revolutionary* (HarperCollins Publishers, Inc., 2001): 58-59.
14. Linus Torvalds et David Diamond, *Just For Fun: The Story of an Accidentaly Revolutionary* (HarperCollins Publishers, Inc., 2001): 78.
15. *Linux 10th Anniversary*. http://www.linux10.org/history/
16. Linus Torvalds et David Diamond, *Just For Fun: The Story of an Accidentaly Revolutionary* (HarperCollins Publishers, Inc., 2001): 96-97.
17. Linus Torvalds et David Diamond, *Just For Fun: The Story of an Accidentaly Revolutionary* (HarperCollins Publishers, Inc., 2001): 94-95.
18. Robert Young, *Interview with Linus, the Author of Linux*, Linux Journal (1er mars 1994). http://www.linuxjournal.com/article.php?sid=2736
19. Linus Torvalds et David Diamond, *Just For Fun: The Story of an Accidentaly Revolutionary* (HarperCollins Publishers, Inc., 2001): 59.

Chapitre X — GNU/Linux

En 1993, le mouvement du logiciel libre arriva à un croisement. D'un point de vue optimiste, tous les signes indiquaient le succès de la culture « hacker ». *Wired*, un nouveau magazine offrant des articles sur le cryptage des données, Usenet, et la liberté des logiciels, se vendait bien dans les librairies. L'Internet, d'abord jargon utilisé uniquement par les hackers et professeurs universitaires, avait trouvé le chemin menant au dictionnaire. Même le président Clinton l'utilisait. L'ordinateur personnel, jouet pour amateur au début, avait acquis un respect global, donnant accès aux logiciels écrits par des hackers à toute une nouvelle génération d'utilisateurs. Et alors que le Projet GNU n'était pas encore finalisé comme un système d'exploitation totalement libre, les utilisateurs curieux pouvaient toujours essayer Linux en attendant.

De tout point de vue, les nouvelles étaient bonnes, ou le semblaient-elles. Après une décennie de combat, les hackers et leurs valeurs commençaient finalement à être acceptés par la société en général. Ils étaient assimilés.

Mais l'étaient-ils vraiment ? Pour un esprit pessimiste, chaque signe d'intégration avait son mauvais côté. Bien sûr, être un hacker était soudain à la mode, mais être à la mode, était-ce bon pour une communauté qui prospérait sur l'exclusion ? Bien sûr, la Maison Blanche ne tarissait pas d'éloges au sujet d'Internet, allant même jusqu'à avoir son propre nom de domaine, *whitehouse.gov*, mais tout cela était accompagné par des entreprises, des comités de censures, et des organismes de répression qui essayaient de dompter la culture Far West d'Internet. Bien sûr, les PC étaient plus puissants, mais en inondant le marché avec ses puces, Intel avait créé une situation dans laquelle les éditeurs de logiciels propriétaires avaient le pouvoir. Pour chaque utilisateur rejoignant la cause du logiciel libre grâce à Linux, des centaines, voire des milliers, démarraient Microsoft Windows pour la première fois.

Enfin, il y avait l'étrange nature de Linux. Non restreint par des failles de conception (comme GNU) ou des problèmes légaux (comme BSD), l'évolution à haute vitesse de Linux a été si inattendue, son succès si accidentel, que les programmeurs les plus proches du code ne savaient qu'en faire. Étant plus un album de compilations qu'un système d'exploitation, il était constitué d'un mélange des plus grands compilateurs à la sauce « hacker » : tout depuis GCC, GDB et glibc (le projet de librairie C nouvellement développé par GNU) à X (un système d'interface graphique basé sur Unix développé par le Laboratoire de Science Informatique du MIT) en passant par des outils développés par BSD comme BIND (le Démon de Nommage pour Internet de Berkeley, qui permet d'utiliser des noms de domaines faciles à mémoriser au lieu d'adresses IP) et TCP/IP. La pierre angulaire étant bien sûr le noyau Linux — lui-même une version retaillée et surchargée de Minix. Plutôt que de refaire un système d'exploitation à partir du début, Torvalds et son équipe de programmeurs, croissant rapidement, avaient suivi le vieil adage de Picasso : « les bons artistes empruntent ; les grands artistes volent ». Ou comme Torvalds lui-même le transcrirait à propos de son propre succès : « Je suis juste une personne très oisive qui aime avoir les honneurs pour le travail que d'autres font [1]. »

Une telle fainéantise, bien que d'une admirable efficacité, était troublante d'un point de vue politique. Cela soulignait le manque d'idéologie de Torvalds. À la différence des développeurs de GNU, Torvalds n'avait pas bâti un système d'exploitation avec le but de fournir un outil de travail aux autres hackers ; il avait bâti quelque chose avec quoi il pouvait s'amuser. Tout comme Tom Sawyer peinturlurant une clôture, le génie de Torvalds repose moins sur ses talents de visionnaire que sur sa capacité à recruter d'autres hackers pour accélérer le développement.

Cependant, une question troublante n'était toujours pas posée par la réussite de Torvalds et ses recrues : qu'était exactement Linux ? Était-ce une manifestation de la philosophie du logiciel libre d'abord exprimée par Stallman dans le Manifeste de GNU ? Ou bien était-ce simplement une habile compilation de logiciels que n'importe quel utilisateur, pareillement motivé, pourrait assembler sur son propre système domestique ?

Vers fin 1993, un nombre grandissant d'utilisateurs de Linux tendait vers la seconde définition et commençaient à brasser leurs propres variations de Linux. Ils eurent même le culot de les mettre en bouteille et vendre leurs variations (ou *distributions*) à d'autres amateurs d'Unix. Les résultats étaient pour le moins médiocres.

« C'était bien avant Red Hat et les autres distributions commerciales », se souvient Ian Murdock, alors étudiant en sciences informatiques à l'université de Purdue. « En lisant un magazine Unix, vous trouviez toutes ces pubs de la taille d'une carte de visite proclamant 'Linux'. La plupart étant des opérations sous-marines, ne voyant aucun problème à glisser un peu de leur code dans le mélange. »

Murdock, un programmeur Unix, se souvient avoir été « balayé » par Linux après l'avoir téléchargé et installé pour la première fois sur son PC domestique. « C'était juste beaucoup de plaisir », dit-il. « Ça m'a donné envie de m'impliquer ». Néanmoins, l'explosion de distributions de mauvaise qualité finit par plomber son enthousiasme naissant. Il décida donc que la meilleure façon de s'impliquer serait de faire une version de Linux propre de tout additif. Murdock entreprit donc de lister tous les meilleurs logiciels libres disponibles avec l'intention de les intégrer à sa distribution. « Je voulais faire quelque chose qui ferait honneur à Linux », dit Murdock.

Dans l'objectif « d'attirer l'attention », Murdock posta son projet sur l'Internet, y compris sur le fil Usenet *comp.os.linux*. L'une des toutes premières réponses émanait de *rms@ai.mit.edu*. En tant que hacker, Murdock reconnut l'adresse aussitôt. Il s'agissait de Richard M. Stallman, un homme que Murdock connaissait à l'époque comme initiateur du projet GNU et surtout comme « le hacker parmi les hackers ». Voyant l'adresse dans sa boîte email, Murdock fut ébahi. Pourquoi donc Stallman, une personne dirigeant son propre projet de système d'exploitation, s'intéresserait-il à ses pleurnicheries concernant Linux ?

Murdock ouvrit le message.

« Il disait que la FSF commençait à regarder Linux de près et que faire un système Linux pourrait aussi l'intéresser. Pour Stallman, nos objectifs étaient compatibles dans leurs principes. »

Le message représentait une volte-face dramatique pour Stallman. Jusqu'en 1993, Stallman s'était contenté de ne pas se mêler des affaires de la communauté Linux. En fait, il avait complètement déconsidéré le système d'exploitation rebelle à son arrivée dans le monde Unix en 1991. Après avoir reçu la première alerte d'un système de type Unix fonctionnant sur PC, Stallman dit qu'il délégua la tâche d'examiner ce système à un ami. Stallman se rappelle : « Il me rapporta que le programme était modelé sur system-V, qui était une version inférieure d'Unix. Il m'a aussi dit qu'il n'était pas portable. »

Le rapport de cet ami était correct. Fait pour fonctionner sur des machines à base d'Intel 386, Linux était fortement ancré à sa plate-forme bon marché. Ce que cet ami ne rapporta pas, par contre, était l'énorme avantage de Linux en tant que seul système d'exploitation librement modifiable sur le marché. En d'autres termes, tandis que Stallman passa les trois années suivantes à lire les rapports de bogues de son équipe HURD, Torvalds raflait les programmeurs qui allaient ensuite porter Linux sur de nouvelles plate-formes.

En 1993, l'incapacité du Projet GNU à fournir un noyau fonctionnel créait de nombreux problèmes au sein même du Projet GNU aussi bien que dans le mouvement du logiciel libre au sens large. Un article du magazine *Wired*, écrit par Simson Garfinkel en Mars 1993, déclarait le projet GNU « enlisé » malgré le succès de nombreux autres outils du projet [2]. Les membres du projet et sa branche sans but lucratif, la FSF, se souviennent que l'atmosphère était encore pire que l'article de Simson Garfinkel ne le laissait entendre. « Il était très clair, au moins pour moi à cette époque, qu'il y avait une fenêtre pour introduire un nouveau système d'exploitation », dit Chassell. « Et dès l'instant que cette fenêtre était fermée, les gens étaient devenus moins intéressés. C'est exactement ce qu'il s'est produit, en fait [3]. »

Les efforts du Projet GNU durant la période 1990-1993 ont fait couler beaucoup d'encre. Tandis que certains blâment Stallman pour ces difficultés, Eric Raymond, un des premiers membres de l'équipe GNU Emacs et plus tard critique de Stallman, dit que le problème était nettement institutionnel. « La FSF devint arrogante », dit Raymond. Ils se sont écartés de leur but, qui était de faire un système d'exploitation fonctionnel, pour faire un système d'exploitation expérimental ». Pire encore, « Ils pensaient que rien hors de la FSF ne pouvait les atteindre. »

Murdock, moins proche des mécanismes internes du Projet GNU, porte un jugement moins sévère. « Je pense que le problème est qu'ils étaient un peu trop ambitieux et qu'ils persistèrent à prendre de bonnes initiatives pour en corriger de mauvaises », dit-il. « Les micro-noyaux à la fin des années 1980 et au début des années 1990 étaient un sujet à débat. Malheureusement, c'est à ce moment que le Projet GNU commença à planifier son propre noyau. Ils finirent avec beaucoup de travail réalisé, mais il leur en aurait fallu faire

encore plus pour revenir en arrière. »

Stallman cite de nombreux problèmes expliquant les délais. Les procès Lotus et Apple avaient amené des distractions politiques, qui, couplées à l'incapacité de Stallman à taper, lui rendaient la tâche difficile pour aider l'équipe du HURD. Stallman cite également la mauvaise communication entre différentes parties du Projet GNU. « Nous avions beaucoup de travail à faire pour avoir un environnement de débogage fonctionnel », se souvient-il. « Et les gens qui maintenaient GDB à l'époque n'étaient pas très coopératifs ». Le plus gros problème étant, dit Stallman, que lui et les autres membres du Projet GNU sous-estimaient la difficulté d'étendre le micro-noyau *Mach* à un noyau Unix complet.

« Je me disais, OK, la partie [de Mach] communiquant avec la machine a déjà été déboguée », dit Stallman, se souvenant des déboires de l'équipe du HURD dans un discours en 2000. « Avec cette avance, nous devions pouvoir finir le travail plus vite. Malgré cela, il s'est révélé que déboguer ces programmes asynchrones et multithreadés était vraiment difficile. Il y avait des bogues temporels qui souillaient les fichiers, et ça n'était pas drôle. Le résultat final est qu'il a fallu de très nombreuses années pour avoir une version de test [4] ».

Qu'importe l'excuse, ou les excuses, le succès des concurrents, à savoir l'équipe du noyau Linux, créa une situation de forte tension. Bien sûr, le noyau Linux avait été licencié sous GPL, mais comme Murdock lui-même l'avait précisé, l'intention de traiter Linux comme un pur système d'exploitation libre était loin de faire l'unanimité. Vers fin 1993, le nombre des utilisateurs de Linux était passée d'une douzaine de fans de Minix à quelque chose entre 20 000 et 100 000 [5]. Ce qui n'était au début qu'un hobby était maintenant un marché mûr pour l'exploitation. Comme Winston Churchill observant les troupes soviétiques marchant dans Berlin, Stallman ressentit, on le comprend, des émotions mitigées à l'heure de la célébration de la « victoire » de Linux [6].

Bien qu'en retard à la fête, Stallman avait toujours du mordant. Aussitôt que la FSF annonça qu'elle fournirait son argent et son support moral au projet logiciel de Murdock, d'autres offres de support commencèrent à affluer. Murdock nomma le nouveau projet Debian, compression du prénom de sa femme, Deborah, et du sien, et en quelques semaines, la première version de la distribution fut sortie. « [Le soutien de Richard] catapulta Debian d'un petit projet à quelque chose qui retint l'attention de la communauté pratiquement en l'espace d'une nuit », dit Murdock.

En janvier 1994, Murdock publia le *manifeste de Debian*. Écrit dans l'esprit du *manifeste de GNU* de Stallman, publié une décennie plus tôt, il expliquait l'importance de travailler intimement avec la FSF. Murdock écrit :

> La Free Software Foundation joue un rôle extrêmement important dans le futur de [la distribution] Debian. Par le simple fait qu'ils vont la distribuer, un message est envoyé au monde que Linux n'est pas un produit commercial et qu'il ne devrait jamais l'être, mais cela ne signifie pas que Linux ne sera jamais capable de concourir commercialement. Pour ceux qui ne seraient pas d'accord, je vous défie de justifier le succès de GNU Emacs et de GCC, qui ne sont pas des logiciels commerciaux mais qui ont pourtant eu un certain impact sur le marché commercial.
>
> Le temps est venu de se concentrer sur le futur de Linux en lieu et place de l'objectif destructeur qui est de s'enrichir aux dépends de toute la communauté Linux et de son avenir. Le développement et la distribution de Debian ne sont peut-être pas des réponses aux problèmes que j'ai mentionnés dans ce Manifeste, mais j'espère au moins attirer suffisamment l'attention sur ces problèmes pour qu'ils puissent être résolus.

Peu après la publication du Manifeste, la FSF fit sa première demande importante. Stallman voulait que Murdock appelle sa distribution « GNU/Linux ». Au début, dit Murdock, Stallman voulait utiliser le terme « Lignux », comme Linux avec GNU en son cœur, mais un bref test du terme sur Usenet et dans certains groupes de hackers entraîna suffisamment de noms d'oiseaux pour que Stallman change pour l'expression moins maladroite « GNU/Linux ».

Bien que certains virent la démarche de Stallman consistant à préfixer « GNU » comme une tentative tardive pour gagner du crédit, Murdock pensait différemment. Rétrospectivement, il y vit plutôt une tentative

de contrecarrer la tension grandissante entre le Projet GNU et les développeurs du noyau Linux. « Il y avait l'amorce d'une rupture », se souvient Murdock. « Richard était inquiet. »

La plus sérieuse rupture, dit Murdock, portait sur *glibc*. Raccourci de GNU C Library, *glibc* est le paquet qui permet aux programmeurs de faire des *appels système* adressés au noyau. Durant les années 1993-1994, *glibc* fut un goulot d'étranglement problématique au développement de Linux. Les nouveaux utilisateurs ajoutaient tellement de nouvelles fonctions au noyau Linux que les mainteneurs de *glibc* furent vite submergés par les suggestions de modification. Frustrés par les délais et la réputation de traîne-savate du Projet GNU, certains développeurs Linux suggérèrent la création d'un *fork*, c'est-à-dire une version de *glibc* spécifique à Linux et développée en parallèle.

Dans le monde des hackers, les *forks* sont d'intéressants phénomènes. Bien que l'éthique des hackers permette à un programmeur de faire ce qu'il veut avec le code source d'un programme, la plupart des hackers préfèrent reverser leurs innovations dans un fichier central du code source, ou « arbre », pour s'assurer de la compatibilité avec les programmes des autres. « Forker » *glibc* si tôt dans le développement de Linux aurait signifié la perte des contributions potentielles de centaines, voire de milliers de développeurs de Linux. Cela aurait signifié aussi une incompatibilité croissante entre Linux et le système GNU que Stallman et l'équipe GNU espéraient toujours développer.

En tant que leader du Projet GNU, Stallman avait déjà eu l'expérience des effets négatifs des *forks* logiciels en 1991. Un groupe de développeurs d'Emacs travaillant pour une entreprise nommée Lucid furent déçus devant le refus de Stallman de reverser leurs modifications au code source de GNU Emacs. Le fork avait donné naissance à une version parallèle, Lucid Emacs, et beaucoup de ressentiment [7].

D'après Murdock, Debian basait son travail sur un *fork* similaire de *glibc*, ce qui motiva Stallman à insister sur l'ajout du préfixe GNU quand Debian publia sa distribution logicielle. « Le *fork* a depuis convergé. Néanmoins, à l'époque il y avait une préoccupation selon laquelle la communauté Linux, dans la mesure où elle se serait considérée comme autre chose qu'une partie de la communauté GNU, pouvait entraîner une désunion. »

Stallman corrobore les souvenirs de Murdock. En fait, selon lui, il y avait des forks naissants à tous les principaux composants de GNU. Au début, Stallman considérait ces forks comme le produit de raisins acides. À l'opposé de la dynamique rapide et informelle de l'équipe du noyau Linux, les mainteneurs du code source de GNU tendaient à être plus lents et circonspects dans leurs changements qui pourraient affecter la viabilité à long terme d'un programme. Ils étaient par contre moins hésitants quant à critiquer sévèrement le code des autres. Le temps passant, à la lecture des emails des développeurs de Linux, Stallman commença à sentir un manque latent de conscience du Projet GNU et de ses objectifs.

« Nous avons découvert que les gens qui se considéraient eux-mêmes utilisateurs de Linux se moquaient du Projet GNU », dit Stallman. « Ils disaient, 'Pourquoi devrais-je m'en préoccuper? Je me moque du Projet GNU. Ça marche pour moi. Ça marche pour la plupart des utilisateurs de Linux, et rien d'autre n'a d'importance pour nous'. Et cela était vraiment une surprise étant donné que ces personnes utilisaient à la base une variante du système GNU, et ils s'en inquiétaient si peu. Ils se moquaient de GNU plus que quiconque. »

Tandis que certains vécurent comme un renversement politique la description de Linux comme une variante de GNU, Murdock, déjà sympathisant à la cause des logiciels libres, considérait comme raisonnable la requête de Stallman consistant à appeler la version de Debian « GNU/Linux ». « C'était davantage la recherche d'une unité que celle d'une reconnaissance », dit-il.

Des demandes de nature plus technique s'ensuivirent. Bien que Murdock fût arrangeant sur les problèmes d'ordre politique, il était plus rigide concernant la structure et le modèle de développement du logiciel lui-même. Ce qui avait commencé comme une démonstration de solidarité devint vite le miroir des conflits dans les autres projets GNU.

« Je peux vous dire avoir eu ma part de désaccords avec lui », dit Murdock avec un rire. « Honnêtement, Richard peut être une personne avec qui il est difficile de travailler. »

En 1996, après sa sortie de l'Université de Purdue, Murdock décida de passer les rênes du projet Debian alors en plein essor. Il avait déjà cédé les tâches administratives à Bruce Perens, le hacker bien connu

pour son travail sur *Electric Fence*, un utilitaire Unix licencié sous GPL. Perens, tout comme Murdock, était un programmeur Unix qui était tombé amoureux de GNU/Linux aussitôt que les propriétés Unix de Linux étaient devenues apparentes. Tout comme Murdock, Perens sympathisa avec le courant politique de Stallman et la FSF, quoique de manière plus éloignée.

« Je me souviens qu'après que Stallman eut publié le Manifeste de GNU, GNU Emacs et GCC, j'avais lu un article disant qu'il travaillait en tant que consultant pour Intel », dit Perens, évoquant ses premières impressions sur Stallman à la fin des années 1980. « Je lui ai écrit pour lui demander comment il pouvait défendre les logiciels libres d'un côté et travailler pour Intel de l'autre. Il m'a répondu : 'Je travaille en tant que consultant pour produire des logiciels libres'. Il était parfaitement courtois et j'ai alors pensé que sa réponse était l'évidence même. »

En tant que développeur important de Debian, par contre, Perens observait avec stupéfaction les batailles entre Stallman et Murdock concernant la conception de Debian. À son accession à la tête de l'équipe de développement, Perens prit la décision de mettre de la distance entre Debian et la FSF. « J'ai décidé que nous ne voulions pas de micro-direction dans le style de Richard », dit-il.

Selon Perens, Stallman fut pris à contrepied par la décision mais eut la sagesse de faire avec. « Il nous donna du mou puis envoya un message disant que nous avions vraiment besoin d'une relation. Il demanda que nous appelions le système GNU/Linux et que nous en restions là. Je décidai que c'était convenable. Je pris la décision unilatéralement. Tout le monde soupira de soulagement. »

Le temps passant, Debian allait développer une réputation de version hacker de Linux, aux côtés de Slackware, une autre distribution populaire construite durant la même période 1993-1994. Malgré tout, hors du royaume des systèmes pour hackers, Linux commençait à faire parler de lui dans le marché des Unix commerciaux. En Caroline du Nord, une entreprise Unix nommée Red Hat réorganisait ses affaires pour se concentrer sur Linux. Le président directeur général était Robert Young, l'éditeur original du *Linux Journal* qui, en 1994, avait demandé à Linus Torvalds s'il regrettait d'avoir licencié le noyau sous GPL. Pour Young, la réponse de Torvalds eut un profond impact sur sa vision de Linux. Au lieu de chercher une tactique traditionnelle pour caler GNU/Linux sur le marché, Young commença à se demander ce qui se passerait si une entreprise adoptait la même démarche que Debian, c'est à dire assembler un système d'exploitation en n'utilisant que des logiciels libres. Cygnus Solutions, l'entreprise montée par Michael Tiemann et John Gilmore en 1990, démontrait déjà la possibilité de vendre des logiciels libres en se basant sur leur qualité et leur adaptabilité. Et si Red Hat avait la même approche avec GNU/Linux ?

« Dans la tradition des scientifiques occidentaux, nous nous tenions sur des épaules de géants », dit Young, citant à la fois Torvalds et Sir Isaac Newton avant lui. « En affaires, cela se traduit par éviter de réinventer la roue. La beauté du modèle [de la GPL] est que vous placez votre code dans le domaine public [8]. Si vous êtes un vendeur de logiciel indépendant, que vous essayez de concevoir une application et avez besoin d'utiliser un composeur pour modem, et bien... pourquoi réinventer les composeurs ? Vous pouvez juste chiper PPP de Red Hat Linux et l'utiliser au coeur de votre outil pour composeurs de modems. Si vous avez besoin d'une collection d'outils graphiques, vous n'avez pas besoin de réécrire votre bibliothèque graphique. Téléchargez juste GTK. Soudainement, vous avez la capacité de réutiliser ce qu'il y a de mieux. Et soudainement, votre objectif en tant que fabricant de logiciels est moins tourné sur la gestion des logiciels et davantage sur l'écriture de logiciels adaptés aux besoins de vos clients. »

Références

1. Torvalds a cité cette phrase à de nombreuses reprises. La référence la plus notable est dans le livre de Eric Raymond, *La Cathédrale et le Bazar* (mai 1997) http://www.tuxedo.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/index.html.
2. Simson Garfinkel, *Stallman est-il stoppé ?*, *Wired* (mars 1993)
3. La théorie de Chassell au sujet d'une fenêtre de 36 mois pour le lancement d'un nouveau système d'exploitation n'est pas unique au Projet GNU. Au début des années 1990, les versions libres de la Berkeley Software Distribution étaient entravées par les avocats du Unix System Laboratories restreignant la distribution des logiciels dérivés de BSD. Alors que de nombreux utilisateurs considéraient les dérivés de BSD comme FreeBSD et OpenBSD comme étant supérieurs à

GNU/Linux tant sur les performances que sur la sécurité, le nombre d'utilisateurs de FreeBSD et OpenBSD ne représentent qu'une fraction de la population totale d'utilisateurs de GNU/Linux. Pour une analyse du succès relatif de GNU/Linux en comparaison d'autres systèmes libres, voir l'essai du hacker néo-zélandais, Liam Greenwood, *Pourquoi Linux est-il une réussite ?* (1999). http://www.freebsddiary.org/linux.php

4. Discours du Maui High Performance Computing Center (Cf. chap. 8). Dans les courriels suivant ce discours, j'ai demandé à Stallman ce qu'il voulait dire exactement par « bogues temporels ». Stallman expliqua qu'utiliser l'expression « erreurs temporelles » était le meilleur moyen de résumer le problème. Il offrit une explication technique montrant l'influence des erreurs temporelles sur les performances d'un système d'exploitation : les « erreurs temporelles » arrivent dans un système asynchrone où des tâches parallèles peuvent en principe se dérouler dans n'importe quel ordre. Et certains ordres particuliers peuvent conduire à des problèmes. Imaginez qu'un programme A fasse X, et qu'un programme B fasse Y, où X et Y sont toutes deux des sous-routines qui examinent et mettent à jour une structure de données. Presque toujours, l'ordinateur exécutera X avant Y ou bien Y avant X, et il n'y aura donc aucun problème. À de rares occasions, par hasard, l'ordonnanceur laissera le programme A s'exécuter jusqu'à mi-parcours de X, puis lancera B qui fera Y. Ainsi, Y sera fait alors que X est à moitié fait. Étant donné que les processus mettent à jour la même structure de données, ils vont interférer. Par exemple, peut être que X a déjà examiné la structure de données et il ne verra pas qu'il y a eu une modification. Il y aura une faille non reproductible, car dépendante de nombreux facteurs aléatoires (quand l'ordonnanceur décide d'exécuter tel programme et durant un temps donné). La façon d'éviter ce genre d'erreur est d'utiliser un verrou pour s'assurer que X et Y ne se passent pas au même moment. Les programmeurs écrivant des systèmes asynchrones connaissent l'importance des verrous, mais parfois ils ne réalisent pas la nécessité d'un verrou à un endroit particulier ou sur une structure de données particulière. Alors le programme présente une erreur temporelle.
5. Les chiffres concernant le nombre des utilisateurs de GNU/Linux sont au mieux approximatifs, c'est pourquoi je donne un si grand écart. Le chiffre 100 000 provient du site « Milestones » de Red Hat , http://www.redhat.com/about/corporate/milestones.html
6. J'ai écrit cette analogie à Winston Churchill avant que Stallman ne m'envoie de lui-même son propre commentaire sur Churchill : « La seconde guerre mondiale et la détermination nécessaire pour vaincre étaient de forts souvenirs quand je grandissais. Des déclarations telles que celles de Churchill, 'Nous les combattrons sur les zones de débarquement, nous les combattrons sur les plages... nous ne nous rendrons jamais', ont toujours résonné dans mon esprit. »
7. Jamie Zawinski, un ancien programmeur de Lucid qui allait devenir le chef de l'équipe de développement de Mozilla, a un site Internet qui documente le fork Lucid/GNU Emacs, intitulé, « Le schisme Lemacs/FSFmacs ». http://www.jwz.org/doc/lemacs.html
8. Young utilise le terme « domaine public » de manière erronée ici. Le domaine public signifie « non protégé par copyright ». Les programmes protégés sous GPL sont par définition protégés par copyright.

Chapitre XI — Open Source

En novembre 1995, Peter Salus, membre de la *Free Software Foundation* (FSF) et auteur en 1994 d'un ouvrage intitulé *A Quarter Century of Unix*, lança un appel à communication à destination des membres de la liste de discussion « système » du Projet GNU. Salus allait être président de la future conférence sur le « logiciel librement redistribuable » qui se tiendrait à Cambridge au Massachusetts. Prévue pour février 1996 et sponsorisée par la FSF, la manifestation s'annonçait comme la première à être complètement dédiée au logiciel libre et, dans une démonstration d'unité avec les autres développeurs de logiciel libre, elle accueillait tous les articles concernant « tout aspect de GNU, Linux, NetBSD, 386BSD, FreeBSD, Perl, Tcl/tk, et autres outils dont le code est accessible et redistribuable ». Salus écrivit :

> Durant les quinze dernières années, les logiciels gratuits et bon marché sont devenus monnaie courante. Cette conférence permettra la réunion de développeurs de logiciels librement redistribuables de différents types, et des éditeurs (via divers médias) de ce type de logiciels. Il y aura des tutoriels et des articles de référence, ainsi que les interventions de Linus Torvalds et Richard Stallman [1].

Une des premières personnes à recevoir le message de Salus fut Eric S. Raymond, membre du comité de la conférence. Bien qu'il ne soit pas le dirigeant d'un projet ou d'une entreprise, Raymond s'était construit une bonne réputation dans la communauté des hackers en tant que contributeur majeur à *GNU Emacs* et comme éditeur du *Nouveau dictionnaire hacker*, une version papier du fichier *Jargon* de la communauté, vieux d'une décennie.

Pour Raymond, la conférence de 1996 était un évènement bienvenu. Actif dans le Projet GNU durant les années 1980, Raymond avait pris ses distances avec ce dernier en 1992 en dénonçant, comme beaucoup avant lui, le style « micro-management » de Stallman. « Richard a initié une discussion houleuse quand j'ai fait des modifications non autorisées alors que je nettoyais les librairies LISP d'Emacs », se souvient Raymond. « Ça m'a tellement frustré que j'ai décidé de ne plus travailler avec lui. »

Malgré cette désillusion, Raymond resta actif dans la communauté du logiciel libre. À tel point que lorsque Salus proposa d'associer pour la conférence Stallman et Torvalds en tant qu'intervenants principaux, Raymond appuya volontiers l'idée. Avec Stallman représentant le contingent des plus vieux et plus sages, et Torvalds représentant le corps plus jeune et énergique des hackers Linux, l'association des deux signifiait une démonstration symbolique d'unité qui ne pouvait être que bénéfique, surtout pour les jeunes hackers ambitieux (c'est-à-dire âgés de moins de 40 ans) tels que Raymond. « J'avais pour ainsi dire un pied dans chaque camp », dit Raymond.

Au moment de la conférence, la tension entre les deux camps était devenue plus palpable. Les deux groupes avaient cependant quelque chose en commun : cette conférence était leur première opportunité de voir le *wunderkind* finlandais en chair et en os. Étonnamment, Torvalds se révéla être un orateur charmant et volubile. Avec seulement un léger accent suédois, Torvalds surprit les membres du public avec son esprit [2]. Encore plus surprenant, dit Raymond, fut le fait que Torvalds n'épargnait pas les autres hackers importants, y-compris le plus auguste d'entre eux, Richard Stallman. À la fin de la conférence, le style moitié hacker et moitié mauvais élève de Torvalds remportait les suffrages des conférenciers, tous âges confondus.

« C'était un moment clef », se rappelle Raymond. « Avant 1996, Richard était le seul à avoir assez de crédibilité pour être le leader de la culture toute entière. Ceux qui le désapprouvaient ne le faisaient pas en public. La personne qui brisa ce tabou fut Torvalds. »

La rupture ultime avec ce tabou devait arriver vers la fin de la conférence. Lors d'une discussion sur la domination croissante du marché par Microsoft Windows ou sur un sujet similaire, Torvalds admit être un amateur du programme de présentations PowerPoint de Microsoft. Du point de vue des puristes de l'ancienne école du logiciel, ce genre de propos était l'équivalent d'un curé déclarant son amour pour une femme en plein sermon. Du point de vue de Torvalds et de ses supporters dont le nombre allait grandissant, c'était du bon sens. Pourquoi se priver de bons logiciels propriétaires, juste pour une idéologie? Être un hacker ne doit pas être une souffrance, il s'agit de produire des résultats.

« C'était une chose plutôt choquante à dire », se souvient Raymond. « Mais encore une fois, il a pu le

faire car en 1995 et 1996, il a rapidement acquis de la notoriété. »

Stallman, de son côté, ne se remémore aucune tension lors de cette conférence en 1996, mais il se souvient d'avoir par la suite été piqué au vif lorsque l'impertinence de Torvalds fut portée aux nues. « Il y avait une partie de la documentation Linux où l'on incitait les gens à imprimer les standards de codage de GNU puis à les déchirer », dit Stallman, citant un exemple parmi d'autres. « OK, donc il est en désaccord avec certaines de nos conventions. Ce n'est pas un problème, mais il a choisi une manière étonnamment méchante pour le dire. Il aurait pu simplement dire 'Voilà la façon avec laquelle je pense qu'il faut mettre en forme votre code.' Parfait. Il ne devrait pas y avoir d'hostilité ici. »

Pour Raymond, l'accueil chaleureux fait aux propos de Torvalds par les autres hackers ne fit que confirmer ses soupçons. La ligne de séparation entre les développeurs de Linux et ceux de GNU/Linux était majoritairement générationnelle. Beaucoup de hackers Linux, comme Torvalds, avaient grandi dans un monde dominé par le logiciel propriétaire. A moins qu'un programme soit clairement inférieur, la plupart ne voyaient pas d'objection à accepter un logiciel pour des raisons liées uniquement à sa licence. Quelque part dans l'univers du logiciel libre se cachait un programme que les hackers pourraient un jour transformer en une alternative libre à PowerPoint. Jusqu'à ce que ce moment arrive, pourquoi reprocher à Microsoft d'avoir eu l'initiative de développer ce programme et de s'en être réservé les droits?

En tant qu'ancien membre du Projet GNU, Raymond entrevit et agita la tension entre Stallman et Torvalds. Lors de la décennie ayant succédé au lancement du Projet GNU, Stallman s'était construit une réputation impressionnante en tant que programmeur. Il s'était aussi construit une réputation d'intransigeance en terme de conception de logiciel et de gestion humaine. Peu avant la conférence de 1996, la FSF subit de nombreuses démissions, imputables pour la plupart à Stallman. Brian Youmans, un employé de la FSF engagé par Salus à la suite de ces démissions, se souvient de la scène : « À un moment, Peter [Salus] était le seul employé restant à travailler dans les locaux. »

Pour Raymond, les démissions ne firent que confirmer ses soupçons : les récents retards tels que HURD et les problèmes tels que le schisme *Lucid-Emacs* illustraient les difficultés habituellement rencontrées dans la gestion des projets de logiciel, mais pas dans celle de la gestion du développement de code. Peu après la conférence sur « le logiciel librement redistribuable », Raymond commença à travailler sur son propre projet de logiciel, un utilitaire *popmail* appelé *fetchmail*. S'inspirant de Torvalds, Raymond publia son programme avec la promesse de mettre à jour le code source aussi vite et souvent que possible. Quand les utilisateurs commencèrent à envoyer des rapports d'erreurs et des suggestions de nouvelles fonctionnalités, Raymond, qui avait d'abord anticipé un vaste fatras, trouva le logiciel résultant du processus étonnamment robuste. Analysant le succès de l'approche adoptée par Torvalds, Raymond tira une rapide conclusion : en utilisant Internet comme sa « boîte de Petri » et la surveillance rigoureuse de la communauté hacker comme un moyen de sélection naturelle, Torvalds avait créé une évolution du modèle, libre de planification centralisée.

De plus, Raymond réalisa que Torvalds avait trouvé une façon de contourner la loi de Brooks. D'abord exprimée par Fred P. Brooks, manager du projet OS/360 chez IBM, et auteur du livre *Mythical Man-Month* en 1975, la loi de Brooks postulait qu'ajouter des développeurs à un projet ne faisait qu'entraîner des retards additionnels. Croyant comme la plupart des hackers que le logiciel, comme la soupe, gagne à avoir un nombre limité de cuisiniers, Raymond sentit quelque chose de révolutionnaire à l'œuvre. En invitant de plus en plus de cuisiniers dans la cuisine, Torvalds avait effectivement trouvé une manière de rendre *meilleur* le logiciel en résultant [3].

Raymond coucha ses observations sur le papier. Il les mit en forme comme un discours, qu'il lit rapidement devant un groupe d'amis et voisins dans le comté de Chester, en Pennsylvanie. Intitulé *La Cathédrale et le Bazar*, le discours mettait en contraste le style de management du Projet GNU avec le style de management de Torvalds et des hackers du noyau. Raymond dit que les réactions furent enthousiastes, mais qu'elles le furent encore plus au printemps suivant, lors du *Linux Kongress* de 1997, une rencontre d'utilisateurs Linux en Allemagne.

« Au Kongress, ils me firent une ovation debout à la fin du discours », se rappelle Raymond. « J'ai interprété cela comme un symbole pour deux raisons. D'une part, cela voulait dire qu'ils étaient enthousiasmés par ce qu'ils entendaient. D'autre part, cela voulait dire qu'ils étaient enthousiasmés même

après avoir entendu le discours au travers de la barrière linguistique. »

Finalement, Raymond convertit ce discours en article, toujours intitulé *La Cathédrale et le Bazar*. Ce papier tirait son nom de l'analogie centrale de Raymond. Les programmes GNU étaient des cathédrales, monuments impressionnants et planifiés de façon centralisée selon l'éthique hacker, conçus pour résister aux assauts du temps. Linux, d'un autre côté, ressemblait plus à un grand bazar bruyant, un projet logiciel développé à travers les dynamiques floues et décentralisées d'Internet...

Implicitement, dans chaque analogie se cachait une comparaison entre Stallman et Torvalds. Là où Stallman incarnait le modèle classique de l'architecte de cathédrale, c'est à dire un « sorcier » programmeur qui pouvait disparaître pendant dix-huit mois et revenir avec quelque chose comme le Compilateur C GNU, Torvalds était plutôt un génial organisateur de soirée. En laissant les autres diriger la discussion sur la conception de Linux et en n'intervenant que lorsque le débat requérait un arbitre, Torvalds avait créé un mode de développement reflétant parfaitement son caractère relax. Pour Torvalds, la tâche managériale la plus importante consistait à ne pas imposer de contrôle mais à laisser les idées affluer.

Raymond résume : « Je pense que le hack le plus astucieux et lourd de conséquence de Linus n'est pas le noyau Linux en lui-même, mais plutôt l'invention du modèle de développement associé » [4].

En résumant les secrets du succès managérial de Torvalds, Raymond avait lui-même lancé un coup d'état. Parmi le public au *Linux Kongress* se trouvait Tim O'Reilly, éditeur de *O'Reilly & Associates*, entreprise spécialisée dans les manuels de logiciels et les livres ayant attrait au logiciel (et éditeur de ce livre dans sa version originale). Après avoir entendu le discours de Raymond au Kongress, O'Reilly l'invita rapidement à donner le même lors de la conférence Perl inaugurale de son entreprise, qui aurait lieu plus tard dans l'année à Monterey, Californie.

Bien que cette conférence fut supposée se concentrer sur *Perl*, un langage de script créé par un hacker Linux, Larry Wall, O'Reilly promit à Raymond qu'on y traiterait d'autres technologies logicielles libres. Étant donné l'intérêt commercial croissant pour Linux et *Apache*, un serveur web libre très populaire, O'Reilly comptait profiter de l'occasion pour informer sur le rôle du logiciel libre dans la création de toute l'infrastructure d'Internet. Des langages populaires tels que *Perl* et *Python* aux programmes obscurs comme *BIND* (le Berkeley Internet Naming Daemon), un logiciel qui permet aux utilisateurs de remplacer les obscures adresses IP par des noms de domaines faciles à retenir (par exemple amazon.com) et *sendmail*, le programme de courriel le plus populaire sur Internet, le logiciel libre était devenu un phénomène émergent. Telle une colonie de fourmis bâtissant un magnifique nid, grain de sable après grain de sable, la seule chose qui faisait défaut était la conscience collective communautaire. O'Reilly vit dans le discours de Raymond un bon moyen de motiver cette prise de conscience : faire définitivement accepter le fait que le développement de logiciel libre ne se résumait pas au Projet GNU. Les langages de programmation comme *Perl* et *Python*, ainsi que les logiciels Internet comme *BIND*, *sendmail*, et *Apache*, prouvaient que le logiciel libre était déjà omniprésent et influent. O'Reilly promit aussi à Raymond un accueil encore plus chaleureux que celui du *Linux Kongress*.

Et il avait raison : « Cette fois là, j'ai eu droit à l'ovation debout avant le discours », dit Raymond en riant.

Comme prévu, le public n'était pas composé des seuls hackers, mais on y comptait d'autres personnes intéressées par le pouvoir croissant du mouvement pour le logiciel libre. L'un des contingents était un groupe de chez Netscape, la startup de *Mountain View* en Californie, qui arrivait en fin de course dans sa guerre de trois ans avec Microsoft pour le contrôle du marché des navigateurs Internet.

Intrigués par le discours de Raymond et impatients de récupérer les parts de marché perdues, les envoyés de Netscape rapportèrent le message à leur maison mère. Quelques mois plus tard, en Janvier 1998, l'entreprise annonça ses intentions de rendre public le code source de son navigateur web vedette, *Navigator*, dans l'espoir de profiter de l'aide des hackers dans les développements futurs.

Quand Jim Barksdale, le PDG de Netscape, cita le traité de Raymond *La Cathédrale et le Bazar* comme ayant majoritairement influencé la décision de l'entreprise, cela éleva immédiatement Raymond au rang de célébrité parmi les hackers. Déterminé à ne pas rater l'occasion, Raymond voyagea sur la Côte Ouest pour donner des interviews, conseiller les employés de Netscape, et finalement prendre part à la fête

célébrant la publication du code source de *Netscape Navigator*. Le nom de code du code source de *Navigator* était *Mozilla*: une référence à la taille gargantuesque du programme (30 millions de lignes de code) et à son héritage. Développé comme version propriétaire sur la base de *Mosaic*, le navigateur web créé par Marc Andreessen à l'université de l'Illinois, *Mozilla* était une preuve de plus que lorsqu'il fallait construire de nouveaux programmes, la plupart des développeurs préféraient se baser sur des programmes plus anciens et modifiables.

Durant son passage en Californie, Raymond arriva aussi à caser une visite chez VA Research, une entreprise basée à Santa Clara, commercialisant des stations de travail avec la distribution logicielle GNU/Linux pré-installée. Initié par Raymond, la réunion était intime. La liste des invités incluait le fondateur de VA, Larry Augustin, quelques employés de VA, et Christine Peterson la présidente de l'Institut Foresight, un *think tank* de la Silicon Valley spécialisé dans les nanotechnologies.

« L'ordre du jour de la réunion se centrait sur un point: comment tirer avantage de la décision de Netscape afin que d'autres entreprises aient envie de l'imiter? ». Raymond ne se souvient pas de toute la discussion, mais il se rappelle de la première plainte formulée. Malgré les meilleurs efforts de Stallman et des autres hackers pour rappeler aux gens que le mot « libre » dans logiciel libre signifiait liberté et non gratuité, le message ne passait toujours pas. La plupart des businessmen, après avoir entendu le terme pour la première fois, l'interprétaient comme synonyme de « coût zéro », écartant rapidement toute autre interprétation. Tant que les hackers ne trouvaient pas une façon de transcender cette dissonance cognitive, le mouvement pour le logiciel libre avait encore du chemin devant lui, même après l'épisode Netscape.

Peterson, dont l'organisation avait pris une part active dans l'avancement de la cause du logiciel libre, proposa une alternative: *open source*.

Rétrospectivement, Peterson dit qu'elle a pensé au terme *open source* alors qu'elle parlait de la discussion à propos de Netscape avec un ami travaillant dans les relations publiques de l'industrie du logiciel. Elle ne se souvient pas où elle a trouvé ce terme ou si elle l'a emprunté à un autre domaine, mais ce qui est sûr est que son ami n'aimait pas cette proposition [5].

Durant cette réunion, d'après Peterson, la réaction était totalement différente. « J'hésitais à le suggérer », se souvient elle. « Je n'avais pas vraiment d'emprise sur le groupe, alors j'ai commencé à l'utiliser l'air de rien, sans souligner que c'était une nouvelle expression ». À la surprise de Peterson, le terme pris racine. A la fin de l'entrevue, la plupart des présents, y-compris Raymond, semblaient satisfaits par ce terme.

Selon lui, Raymond commença à utiliser le terme *open source* publiquement en tant que substitut à logiciel libre seulement un jour ou deux après la fête inaugurale de *Mozilla*, au moment où O'Reilly prévoyait une rencontre pour parler du logiciel libre. Nommant cette rencontre « le Sommet du Freeware », O'Reilly voulait attirer l'attention des médias et du public sur les autres projets méritants qui avaient eux aussi incité Netscape à publier *Mozilla*. « Tous ces gars avaient beaucoup en commun, et j'étais surpris qu'ils ne se connussent pas tous les uns les autres », dit O'Reilly. « Je voulais aussi montrer au monde à quel point la culture du logiciel libre avait déjà eu un gros impact. Les gens passaient à côté d'une grande partie de la tradition du logiciel libre. »

En mettant au point sa liste d'invités, O'Reilly pris une décision ayant des conséquences politiques à long terme. Il décida de restreindre la liste aux développeurs de la Côte Ouest tels que Wall, Eric Allman, créateur de *sendmail*, et Paul Vixie, créateur de *BIND*. Il y avait des exceptions, bien sûr : Raymond, résident en Pennsylvanie, était déjà sur place grâce au lancement de *Mozilla* et fut vite invité. Tout comme Guido van Rossum, de Virginie, créateur de *Python*. « Frank Willison, mon rédacteur en chef et champion de Python dans l'entreprise, l'invita sans même me demander », se souvient O'Reilly. « J'étais content de l'avoir là, mais au début, c'était uniquement un simple rassemblement local. »

Pour certains, le refus d'inclure le nom de Stallman sur la liste relevait de l'outrage. « J'ai décidé de ne pas y aller à cause de ça », dit Perens, se souvenant du sommet. Raymond, qui s'y rendit, argumenta en défaveur d'une invitation pour Stallman. La rumeur prit de l'ampleur du fait qu'O'Reilly, hôte de l'évènement, s'était publiquement opposé à Stallman sur le problème des copyrights pour les manuels de logiciel. Stallman avait dit que les manuels pour les logiciels libres devaient être de même librement copiables, redistribuables et modifiables. O'Reilly, lui, rétorquait qu'un marché à valeur ajoutée pour les livres non libres augmentait l'utilité du logiciel libre, en le rendant plus accessible à un public plus vaste. Les deux s'étaient également

querellé à propos du titre de l'évènement, Stallman insistant pour « Logiciel Libre » et contre « Freeware », moins lourd de sens politique.

Après coup, O'Reilly ne considère pas sa décision d'omettre le nom de Stallman dans la liste des invités comme une offense. « À l'époque, je n'avais jamais rencontré Richard en personne, mais dans nos échanges de courriels, il avait été inflexible et réticent à s'engager au dialogue. Je voulais m'assurer que la tradition GNU soit représentée lors du meeting. J'ai donc invité John Gilmore et Michael Tiemann, que je connaissais personnellement, et dont je savais qu'ils étaient très attachés aux valeurs de la GPL mais semblaient davantage prêts à se prêter à un échange franc sur les forces et faiblesses des divers projets et traditions du logiciel libre. Étant donné tout le brouhaha qui allait suivre, je regrette effectivement de ne pas avoir invité Richard, mais je ne pense vraiment pas que cela doive être interprété comme un manque de respect envers le Projet GNU ou envers Richard en personne. » Qu'il y ait eu offense ou pas, O'Reilly et Raymond s'accordent sur le fait que l'expression *open source* remporta l'adoption de suffisamment de participants au congrès pour être qualifiée de succès. Les présents partagèrent leurs idées et expériences, et échangèrent sur les moyens d'améliorer l'image du logiciel libre. Chercher comment souligner les succès du logiciel libre était d'une importance clef, tout spécialement dans le domaine de l'infrastructure Internet, au lieu de jouer la carte de l'opposition de GNU/Linux face à Microsoft Windows. Mais comme ce fut le cas au cours de la réunion chez VA, la discussion se concentra bientôt sur les problèmes associés au terme « logiciel libre ». O'Reilly se souvient d'un commentaire particulièrement révélateur de la part de Torvalds, présent au meeting :

« Linus venait juste d'emménager dans la Silicon Valley à ce moment, et il expliqua comment il n'avait appris que peu de temps auparavant que le mot 'libre' avait deux sens en anglais : libre comme dans 'libre' et libre comme dans 'gratuit'. »

Michael Tiemann, fondateur de Cygnus, proposa une alternative à la problématique expression 'logiciel libre' : *sourceware*. « Personne n'en était vraiment emballé », se souvient O'Reilly. « C'est alors qu'Eric a lancé le terme *open source* . »

Bien que l'expression fut attrayante pour certains, l'accord en faveur d'un changement de la terminologie officielle était loin d'être unanime. À la fin de la conférence, qui ne durait qu'une journée, les présents mirent les trois expressions — *logiciel libre*, *open source*, et *sourceware* — au vote. Selon O'Reilly, 9 des 15 présents votèrent pour *open source*. Bien que certains se chicanaient encore sur le terme, tous les présents s'accordèrent pour l'employer dans leurs futures déclarations à la presse. « Nous voulions sortir avec un message de solidarité », dit O'Reilly.

L'expression ne mit pas longtemps à intégrer le lexique national. Peu après le sommet, O'Reilly accompagna des membres à une conférence de presse pour des journalistes du *New York Times*, du *Wall Street Journal*, et d'autres journaux importants. Après quelques mois, la tête de Torvalds apparaissait en couverture du magasine *Forbes*, avec celles de Stallman, du créateur de *Perl*, Larry Wall, et du chef d'équipe d**Apache** ***Brian Behlendorf, apparaissant en pages intérieures. L****open source* pouvait s'attaquer au monde des affaires.

Pour les présents au sommet tels que Tiemann, le message de solidarité était la chose la plus importante. Bien que son entreprise ait obtenu de bons succès en vendant des outils et services autour des logiciels libres, il ressentait la difficulté que les programmeurs et les entrepreneurs rencontraient.

« Il ne fait aucun doute que l'utilisation du terme "libre" était problématique dans nombre de situations », dit Tiemann. « L**open source** ***s'est positionné comme une solution attractive et sage pour les marchés. Le logiciel libre se positionnait comme étant moralement juste. Pour le meilleur ou pour le pire, nous nous sommes dit qu'il était plus avantageux de s'aligner avec la foule de l****open source**. »

Du côté de Stallman, la réponse à cette nouvelle expression mettait du temps à venir. Selon Raymond, Stallman pensa un moment à utiliser le terme, pour mieux le rejeter. « Je le sais car j'avais des conversations directes avec lui à ce propos », dit Raymond.

À la fin de 1998, Stallman avait arrêté sa position : *open source*, bien que pratique dans l'optique de faire passer les avantages techniques du logiciel libre, encourageait aussi les intervenants à négliger la question des libertés logicielles. Étant donné ce problème, Stallman restait fidèle à l'expression *logiciel libre*.

Résumant sa position à la Convention-expo LinuxWorld en 1999, un évènement étiqueté par Torvalds lui-même comme la « grand-messe » de la communauté Linux, Stallman implora ses frères hackers de résister à l'attraction d'un compromis facile.

« Parce que nous avons montré ce que nous savons faire, nous n'avons pas à être prêts à tout pour pouvoir travailler avec des entreprises ou compromettre nos objectifs », dit Stallman durant une discussion de groupe. « Laissez-les faire des offres, et nous accepterons. Nous ne voulons pas changer ce que nous faisons pour leur proposer de nous aider. Vous pouvez faire un pas vers un but, puis un autre, et ainsi de suite, et vous finirez par atteindre votre but. Ou bien vous pouvez prendre une demi-mesure, ce qui signifie que vous ne pourrez plus jamais faire de nouveau pas, et que vous n'arriverez jamais à vos fins. »

Cependant, avant même le LinuxWorld, Stallman montrait déjà de plus en plus de détermination à discréditer ses compères qui se montraient davantage conciliants. Quelques mois après le sommet sur le Freeware, O'Reilly organisa sa deuxième conférence annuelle sur *Perl*. Cette fois-ci, Stallman était dans le public. Durant une discussion de groupe au sujet de la décision d'IBM d'utiliser le serveur web libre *Apache* dans ses offres commerciales, Stallman, à l'aide d'un microphone du public, intervint dans le débat avec une tirade hostile à John Ousterhout, présent sur scène, et créateur du langage de script *Tcl*. Stallman qualifia Ousterhout de « parasite » de la communauté du logiciel libre pour avoir commercialisé une version propriétaire de *Tcl* via sa propre startup, Scriptics. « Je ne pense pas que Scriptics soit nécessaire pour la suite de l'existence de *Tcl* », dit Stallman, sifflé par les autres membres du public[6].

« Ce n'était pas une jolie scène à voir », se souvient Rich Morin de chez Prime Time Freeware. « John a fait des choses très respectables: *Tcl*, *Tk*, *Sprite*. C'est un vrai contributeur. »

Malgré son affection pour la personne de Stallman et ses positions, Morin ressentit de l'empathie pour ceux troublés par le comportement dissonant de Stallman.

La sortie de Stallman à la conférence Perl fit fuir momentanément un autre sympathisant potentiel, Bruce Perens. En 1998, Eric Raymond proposa de lancer l' *Open Source Initiative*, dite *OSI*, une organisation qui régulerait l'utilisation du terme *open source* et offrirait une marche à suivre pour les entreprises dans leurs propres programmes. Raymond engagea Perens pour tracer cette définition [7].

Perens allait par la suite s'opposer à l'*OSI*, regrettant que l'organisation se soit mise en place en réaction à Stallman et à la FSF. Reste que, au regard du besoin d'une définition du logiciel libre en dehors des murs de la FSF, Perens comprend pourquoi d'autres hackers peuvent ressentir le besoin de se distancer. « J'aime et j'admire vraiment Richard », dit Perens. « Je pense que Richard ferait mieux son travail s'il avait plus d'équilibre. Cela implique de se retirer du monde du logiciel libre pour quelques mois. »

Les énergies monomaniaques de Stallman eurent peu d'influence pour contrer la stratégie des amis de l' *open source* en matière de relations publiques. En Août 1998, quand le fabriquant de processeurs Intel acquis des actions du vendeur GNU/Linux Red Hat, un article associé du *New York Times* décrivait l'entreprise comme un produit du mouvement « connu alternativement comme logiciel libre et *open source* » [8]. Six mois plus tard, un article de John Markoff sur Apple Computer annonçait dans son titre l'adoption par l'entreprise du serveur *open source Apache* [9].

Cette tendance allait coïncider avec l'importance croissante des entreprises qui adoptaient effectivement le terme *open source*. A la fin du mois d'août 1999, Red Hat, une entreprise qui se prévalait volontiers de l' *open source*, vendait ses actions au Nasdaq. En Décembre, VA Linux -- anciennement VA Research -- voyait sa propre introduction en bourse battre des sommets historiques. Ouvrant à 30 dollars l'action, le prix transperça vite la barre des 300 dollars avant de se stabiliser au niveau de 239 dollars. Les actionnaires suffisamment chanceux pour s'être engagés au prix de départ, et étant restés jusqu'au bout, avaient obtenu une hausse de 698% du prix de leurs titres, un record pour le Nasdaq.

On trouvait parmi ces heureux actionnaires Eric Raymond, qui, en tant que membre de l'entreprise depuis le lancement de *Mozilla*, avait reçu 150.000 actions VA Linux. Abasourdi par la prise de conscience que son texte contrastant les styles managériaux de Stallman et Torvalds lui ait rapporté 36 millions de dollars en revenus, Raymond écrivit une suite à l'article. Dans cette suite, Raymond traita de la relation entre l'éthique hacker et la richesse économique :

Les journalistes me demandent souvent ces jours-ci si je pense que la communauté *open source* sera

> corrompue par l'arrivée de massives sommes d'argent. Je leur réponds ce que je pense, et qui est ceci : la demande commerciale pour les programmeurs a été si intense pendant tellement longtemps que quiconque pouvant être attiré par l'argent est déjà parti depuis longtemps. Notre communauté s'est sélectionnée elle-même pour d'autres préoccupations — la réalisation, l'honneur, la passion artistique, etc [10].

Que de tels commentaires aient ou non relayé les soupçons selon lesquels Raymond et les autres ayant proposé l'*open source* n'étaient là que pour l'argent, ils confirmèrent le dernier message de la communauté *open source* : tout ce qu'il faut pour vendre le concept du logiciel libre, c'est un visage ami et un message raisonnable. Au lieu d'affronter le marché la tête la première comme Stallman l'a fait, Raymond, Torvalds, et les autres leaders de la communauté des hackers avaient choisi une attitude plus relaxée -- ignorant le marché dans certains domaines, s'en servant dans d'autres. Au lieu de jouer le rôle de lycéens mauvais élèves, ils avaient joué la carte de la célébrité, augmentant leur pouvoir dans le processus.

« Dans ses mauvais jours, Richard croit que Linus Torvalds et moi avons conspiré pour nous accaparer sa révolution », dit Raymond. « Le rejet de l'expression *open source* par Richard et sa création délibérée d'une séparation idéologique vient selon moi d'un étrange mélange d'idéalisme et de territorialisme. Il y a des gens qui pensent que tout est du à l'ego de Richard. Je ne le pense pas. C'est plutôt parce qu'il s'associe personnellement avec l'idée du logiciel libre qu'il voit toute menace envers cette idée comme une menace personnelle. »

Ironiquement, le succès de l'*open source* et de ses supporters comme Raymond ne diminua pas le rôle de Stallman en tant que leader. À tout le moins, cela fournit à Stallman de nouveaux adeptes à convertir. Reste que la charge territoriale de Raymond fut une sacrée attaque. Il y a de nombreux exemples où Stallman, en dépit de ses habitudes, a sorti les armes au nom de ses principes : on peut citer son rejet initial du noyau Linux, ou encore son refus actuel, en tant que figure politique, à s'aventurer hors de l'univers des questions de logiciel.

Mais encore une fois, comme le montre ce récent débat autour de l'*open source*, lorsque Stallman a sorti les armes, il y a généralement trouvé matière à gagner du terrain. « Un des traits de caractères principaux de Stallman est le fait qu'il ne bouge pas », dit Ian Murdock. « S'il le fallait, il attendrait même dix ans pour que les gens se rallient à son point de vue. »

Murdock, personnellement, trouve cette nature inébranlable rafraîchissante et précieuse. Stallman n'est peut être plus le leader solitaire du mouvement pour le logiciel libre, mais il est toujours l'étoile du berger de la communauté. « On sait toujours qu'il sera constant dans ses avis », dit Murdock. « La plupart des gens ne sont pas ainsi. Que vous soyez ou non d'accord avec lui, vous devez vraiment respecter cela. »

Notes

1. Peter Salus, *FYI-Conference on Freely Redistributable Software, 2/2, Cambridge* (1995) (archivé par Terry Winograd). http://hci.stanford.edu/pcd-archives/pcd-fyi/1995/0078.html
2. Bien que Linus Torvalds soit finlandais, sa langue maternelle est le suédois. *The Rampantly Unofficial Linus FAQ* offre une brève explication : La Finlande a une minorité signifiante (environ 6%) de sa population parlant suédois. Il s'appellent eux-mêmes *finlandssvensk* ou *finlandssvenskar* et se considèrent finlandais. Nombre de leurs familles ont vécu en Finlande durant des siècles. Le suédois est l'une des deux langues officielles de la Finlande. http://tuxedo.org/~esr/faqs/linus/
3. La loi de Brooks est le résumé succinct de la citation suivante tirée du livre de Brooks :

   > Puisque la construction de logiciel est de façon inhérente un problème de systèmes — un exercice en interactions complexes — l'effort de communication est énorme, et il devient rapidement la principale cause de la perte de temps engendrée par le partitionnement. Ajouter plus de personnel allonge et ne réduit pas l'emploi du temps.

   Fred P. Brooks, *The Mythical Man-Month* (Addison Wesley Publishing, 1995).
4. Eric Raymond, *The Cathredral and the Bazaar* (1997).
5. Malcolm Maclachlan, *Profit Motive Splits Open Source Movement*, TechWeb News (26 août 1998). http://content.techweb.com/wire/story/TWB19980824S0012

6. *Ibidem*.
7. Bruce Perens http://oreilly.com/catalog/opensources/book/perens.htmlv
8. Amy Harmon, *For Sale: Free Operating System, New York Times* (28 septembre 1998). http://www.nytimes.com/library/tech/98/09/biztech/articles/28linux.html
9. John Markoff, *Apple Adopts `Open Source' for its Server Computers*, *New York Times* (17 mars 1999). http://www.nytimes.com/library/tech/99/03/biztech/articles/17apple.html
10. Eric Raymond, *Surprised by Wealth*, Linux Today (10 décembre 1999). http://linuxtoday.com/news_story.php3?ltsn=1999-12-10-001-05-NW-LF

Chapitre XII — Un bref voyage dans l'enfer du Hacker

Richard Stallman regarde fixement, sans ciller, au travers du pare-brise d'une voiture de location, en attendant que le feu passe au vert alors que nous cheminons dans le centre ville de Kihei.

Nous nous rendons tous deux à la ville voisine de Pa'ia, où nous sommes censés rencontrer des programmeurs de logiciels et leurs épouses à l'occasion d'un dîner, d'ici environ une heure.

A peu près deux heures se sont écoulées depuis le discours de Stallman au Maui High Performance Center, et Kihei, une ville qui semblait si accueillante avant cette prise de parole, semble à présent profondément repoussante. Comme la plupart des citées balnéaires, Kihei est une étude unidimensionnelle d'extension banlieusarde. En conduisant sur la rue principale, avec son infinie succession d'échoppes de hamburgers, d'agences immobilières, et de magasins de bikinis, il est difficile de ne pas se sentir telle une bouchée recouverte d'acier transitant dans le tube digestif d'un ver solitaire commercial géant. Ce sentiment est exacerbé par le manque de rues attenantes. Avec comme unique solution d'aller tout droit, le trafic avance par à-coups. Deux cents mètres devant, un feu passe au vert. A peine commençons-nous à bouger que le feu passe de nouveau à l'orange.

Pour Stallman, résident permanent de la côte est, l'idée de passer à Hawaï la plus grande part d'une journée ensoleillée coincé dans les embouteillages suffit à déclencher une embolie. Encore pire est de savoir qu'avec quelques rapides bifurcations à droite il y a moins d'un kilomètre, cette situation aurait tout bonnement pu être évitée. Malheureusement, nous sommes dépendants du conducteur qui nous précède, un programmeur du laboratoire qui connaît le chemin et a décidé de nous mener à Pa'ia via la route touristique, au lieu de l'autoroute de Pilani toute proche.

« C'est affreux », dit Stallman entre des soupirs frustrés. « Pourquoi n'a-t-on pas pris l'autre itinéraire? »

A nouveau, le feu devant nous passe au vert. A nouveau, nous avançons péniblement de quelques longueurs de voiture. Le processus se répète durant dix minutes supplémentaires, jusqu'à ce que nous atteignions une grande intersection promettant un accès à l'autoroute adjacente.

Le conducteur nous précédant n'y pense pas et continue tout droit.

« Pourquoi ne tourne-t-il pas ?», grogne Stallman, agitant ses bras, frustré. « Est-ce que tu peux y croire à ça? »

Je m'abstiens de répondre. Je trouve que le fait même d'être assis dans le siège passager d'une voiture conduite par Stallman, à Maui en plus, est suffisamment incroyable comme ça. Il y a deux heures de cela, je ne savais même pas que Stallman avait son permis de conduire. Maintenant, en écoutant le violoncelle de Yo-Yo Ma jouant les notes de basse sombres de « Appalachian Journey » sur l'autoradio, et en regardant le coucher de soleil sur notre gauche, je fais de mon mieux pour me fondre dans le paysage.

Dès qu'une autre occasion de bifurquer se présenta, Stallman mit son clignotant en marche, espérant que le conducteur devant nous s'en rende compte. Pas de chance. Encore une fois, nous peinons à avancer, et nous nous retrouvons bloqués après quelques centaines de mètres. À présent, Stallman est livide.

« C'est comme s'il nous ignorait délibérément », dit-il, gesticulant et pantomimant comme un sémaphore sur le pont d'un porte-avion dans une vaine tentative pour attirer l'oeil de notre guide. Ce dernier ne réagit pas et, durant les cinq minutes suivantes, nous ne voyons qu'une portion de son crâne dans le rétroviseur.

Je regarde au travers de la vitre de Stallman. Les îles proches de Kahoolawe et Lanai offrent un cadre parfait au coucher de soleil. C'est une vue à couper le souffle. Le genre à rendre ce type de situation un peu plus acceptable si vous êtes un hawaïen natif, je suppose. J'essaye d'attirer l'attention de Stallman sur le phénomène, mais, à présent obsédé par l'inattention du conducteur qui nous précède, il n'en fait que peu de

cas.

Quand le conducteur passe un nouveau feu vert, ignorant complètement un « Autoroute Pilani, prochaine à droite », je grince des dents. Je me souviens d'un avertissement anticipé du programmeur BSD Keith Bostic. « Stallman ne supporte pas bien les idiots », m'avait-il prévenu. «Si quelqu'un dit ou fait quelque chose de stupide, il le regardera dans les yeux et dira : 'C'est stupide' ».

En voyant ce conducteur indolent devant nous, je réalise que c'est la stupidité, et non le dérangement, qui rend Stallman furieux à présent.

« C'est comme s'il avait choisi cette route sans la moindre idée sur la manière la plus efficace de se rendre là bas », dit Stallman.

Le mot « efficace » flotte dans l'air comme une mauvaise odeur. Peu de choses irritent autant l'esprit hacker que l'inefficacité. C'était l'inefficacité d'avoir à vérifier l'imprimante Xerox deux à trois fois par jour qui déclencha l'enquête initiale de Stallman au sujet du code source du pilote. C'était l'inefficacité d'avoir à ré-écrire des outils logiciels détournés par les vendeurs de logiciels commerciaux qui amena Stallman à se battre contre *Symbolics* et à lancer le Projet GNU. Si, comme Jean-Paul Sartre le disait, l'enfer c'est les autres, l'enfer hacker c'est la réitération des erreurs stupides des autres, et ce n'est pas exagérer que de dire que la vie entière de Stallman a été une tentative de sauver l'humanité de ces profondeurs brûlantes.

Cette métaphore de l'enfer devient des plus évidentes alors que nous avançons lentement dans ce paysage. Avec sa multitude de boutiques, de parkings, et de feux de circulation mal réglés, Kihei ressemble moins à une ville qu'à un gros programme logiciel mal planifié. Au lieu de rediriger le trafic et de distribuer les véhicules le long des rues adjacentes et des voies rapides, les architectes de la ville ont décidé de tout faire passer dans une seule artère. D'un point de vue hacker, être dans une voiture au milieu de ce fatras revient à écouter à plein volume un CD de crissements d'ongles sur un tableau noir.

« Les systèmes imparfaits rendent furieux les hackers », observe Steven Levy. Voilà un autre avertissement dont j'aurais dû me souvenir avant de monter en voiture avec Stallman. « C'est une raison pour laquelle les hackers détestent généralement conduire des voitures : le système de feux rouges programmés aléatoirement et de rues à sens unique bizarrement agencées cause des délais tellement diablement 'non-nécessaires' [l'emphase vient de Levy] que leur instinct est de réarranger les panneaux, d'ouvrir les boîtiers des feux rouges... de repenser tout le système »[1].

Plus frustrante encore est la traîtrise de notre guide. Au lieu de chercher un raccourci intelligent, ce que n'importe quel hacker ferait d'instinct, il a plutôt choisi de suivre le jeu des architectes de la ville. Comme Virgile dans *L'enfer* de Dante, notre guide est déterminé à nous faire faire le tour complet de l'enfer hacker, que nous soyons d'accord ou pas.

Avant que je puisse faire cette remarque à Stallman, le conducteur met finalement son clignotant à droite. Les épaules crispées de Stallman se relaxent un peu, et durant un instant la tension dans la voiture se dissipe. Elle revient cependant au galop alors que la voiture devant nous ralentit. Des panneaux "Travaux" encadrent la rue, et bien que l'autoroute Pilani ne soit qu'à quelques centaines de mètres, la route à deux voies censée nous en offrir l'accès est bloquée par un bulldozer arrêté et deux gros tas de terre.

Il faut quelques secondes à Stallman pour comprendre ce qu'il se passe alors que notre guide entame un demi-tour laborieux en cinq manœuvres devant nous. Quand il voit le bulldozer et les panneaux "Accès bloqué" juste devant, Stallman finit par exploser.

« Pourquoi, pourquoi, pourquoi ?», gémit-il, lançant sa tête en arrière. « Vous auriez dû savoir que cette rue était bloquée! Vous auriez dû savoir que cet itinéraire ne fonctionnerait pas! Vous avez fait ça délibérément! »

Le conducteur finit sa manoeuvre et nous croise en sens inverse, retournant vers l'artère principale. Ce faisant, il secoue sa tête et nous adresse un signe d'excuse. Ajouté à une grimace toutes dents dehors, la gestuelle du guide révèle une touche de frustration de continental mais tempérée par une dose protectrice de fatalisme insulaire. Au travers des vitres fermées de notre voiture de location, on lit un message succinct : « Hé, c'est Maui ; qu'est-ce qu'on peut y faire? »

Stallman n'en peut plus.

« Putain mais ne souris pas! » hurle-t-il, baissant la vitre par la même occasion. « C'est ta putain de faute. Tout aurait été tellement plus simple si on s'y était pris à ma façon. »

Stallman insiste sur les mots « ma façon » en agrippant le volant et se balançant en avant à deux reprises. L'image de Stallman au travers de ce cadre vacillant est celle d'un enfant piquant un accès de colère dans un siège de voiture, une image soutenue par le ton de sa voix. À mi-chemin entre colère et angoisse, il semble au bord des larmes.

Heureusement, les larmes ne viennent pas. Comme un orage d'été, la colère s'éteint presque aussi vite qu'elle est apparue. Après quelques ronchonnements, Stallman met la marche arrière et réalise son propre demi-tour. Quand nous retrouvons la rue principale, son visage est aussi impassible que lorsqu'il quittait l'hôtel il y a une demi-heure.

Moins de cinq minutes plus tard, nous atteignons le croisement suivant. Il offre alors un accès facile à l'autoroute, et, en quelques secondes, nous fonçons vers Pa'ia à une vitesse relaxante. Le soleil qui irradiait tout à l'heure d'une lueur dorée sur l'épaule gauche de Stallman brûle à présent d'un beau rouge orangé dans notre rétroviseur. Il étale ses couleurs sur la double rangée d'arbres wili wili qui défilent devant nous des deux côtés de l'autoroute.

Durant les vingt minutes suivantes, le seul bruit dans notre voiture, à part le ronronnement des pneus et du moteur, est le son d'une violoncelle et d'un trio de violons jouant les accords mélancoliques d'un morceau folk appalachien.

Notes

1. Voir Steven Levy, *Hackers* (Penguin USA [paperback], 1984): 40.

Chapitre XIII — Continuer le combat

Pour Richard Stallman, le temps ne soigne peut-être pas toutes les blessures, mais il se révèle être un allié pratique.

Quatre ans après *La Cathédrale et le Bazar*, Stallman s'irrite encore de la critique de Raymond. Il grogne aussi face à l'élévation de Linus Torvalds au rang de hacker le plus célèbre. Il se souvient d'un tee-shirt populaire qui commença à apparaître aux conventions Linux aux alentours de 1999. Fait comme une parodie de l'affiche publicitaire originale de *Star Wars*, le tee-shirt montrait Torvalds brandissant un sabre laser à la façon de Luke Skywalker, alors que la tête de Stallman se trouvait au sommet de R2D2. Ce tee-shirt tape toujours sur les nerfs de Stallman, non seulement parce qu'il le représente comme un sous-fifre de Torvalds, mais aussi parce qu'il élève Torvalds au rang de leader dans la communauté du logiciel libre et de l'*open source*, un rôle que Torvalds lui-même craint d'endosser. C'est « ironique », dit Stallman, abattu. « Porter le sabre est exactement ce que Torvalds refuse de faire. Il concentre l'attention de tous en tant que symbole du mouvement, et puis il refuse de se battre. À quoi bon tout cela? »

Mais encore une fois, c'est ce refus de « porter le sabre » de la part de Torvalds qui a permis à Stallman de sauver sa réputation d'arbitre éthique de la communauté hacker. Malgré ses griefs, Stallman se doit d'admettre que les dernières années ont été plutôt bonnes, que ce soit pour lui ou pour son organisation. Relégué à la périphérie lors du succès inattendu de GNU/Linux, Stallman s'est néanmoins réapproprié cette initiative avec succès. Son emploi du temps des interventions entre janvier 2000 et décembre 2001 comprenait des arrêts sur les six continents et des visites dans des pays où la notion de liberté logicielle porte de lourds sous-entendus : la Chine et l'Inde par exemple.

En dehors de la tribune publique, Stallman a aussi appris à imposer son pouvoir en tant que commissaire de la GNU-GPL. Durant l'été 2000, alors que l'air s'échappait rapidement de la bulle créée par l'offre publique d'achat de VA Linux en 1999, Stallman et la FSF remportèrent deux victoires majeures. En Juillet 2000, Trolltech, une entreprise de logiciels norvégienne développant *Qt*, une suite d'outils graphiques précieux pour la distribution GNU/Linux, annonça qu'elle allait enregistrer ses logiciels sous licence GPL. Quelques semaines plus tard, Sun Microsystem, une compagnie qui jusque-là avait tenté maladroitement de suivre le mouvement *open source* sans céder le contrôle total de ses droits logiciels, se laissa finalement fléchir et annonça qu'elle aussi allait publier la nouvelle suite *OpenOffice* sous double licence : la *Lesser GNU Public Licence* (Licence Publique GNU Limitée -- LGPL) et la *Sun Industry Standards Source Licence* (Licence Source des Standards Industriels Sun -- SISSL).

L'importance accordée à chaque victoire résidait dans le fait que Stallman était peu intervenu dans ces batailles. Dans le cas de Trolltech, Stallman avait simplement joué le rôle de pontife du logiciel libre. En 1999, l'entreprise avait proposé une licence qui était sensée respecter les conditions posées par la FSF, mais après un examen attentif, Stallman y releva des incompatibilités légales qui auraient rendu impossible l'inclusion de *Qt* dans les programmes logiciels sous licence GPL. Fatigués de combattre Stallman, les dirigeants de Trolltech décidèrent finalement de séparer *Qt* en deux versions : l'une sous licence GPL, et l'autre sous licence QPL, permettant ainsi aux développeurs de contourner les problèmes de compatibilité cités par Stallman.

Dans le cas de Sun, ces derniers souhaitaient jouer selon les règles de la FSF. A la conférence Open Source-O'Reilly de 1999, le co-fondateur et directeur scientifique de Sun Microsystems, Bill Joy, se fit l'avocat de la licence *Community Source* de son entreprise. [Il s'agissait] essentiellement d'un compromis permettant aux utilisateurs de copier et modifier les logiciels propriété de Sun, sans toutefois pouvoir faire payer pour ces logiciels tant que l'on n'a pas négocié un accord de royalties avec Sun. Un an après le discours de Joy, le vice-président de Sun Microsystems, Marco Boerries, se trouvait sur la même scène pour détailler le nouveau compromis de licence concernant *OpenOffice*, une suite d'applications bureautiques définie spécialement pour la distribution logicielle GNU/Linux.

« Je peux l'épeler en trois lettres », dit Boerries. « GPL. »

À l'époque, selon Boerries, la décision de son entreprise avait moins à voir avec Stallman qu'avec la

marche en avant des programmes protégés par la GPL. « Ce qui s'est passé, fondamentalement, fut la prise de conscience que différents produits attiraient différentes communautés, et que la licence utilisée dépend du type de communauté que l'on souhaite attirer. Avec *OpenOffice* il était évident que nous avions le plus de corrélation avec la communauté GPL »[1].

De tels commentaires soulignent la force trop souvent ignorée de la GPL et, indirectement, le génie politique de l'homme qui a joué un rôle majeur dans sa création. « Il n'est pas un avocat sur terre qui aurait proposé la GPL telle quelle », dit Eben Moglen, professeur de droit à la Columbia University, conseiller général de la FSF. « Mais elle fonctionne. Et elle fonctionne grâce aux principes de conception de Richard. »

Ancien programmeur professionnel, Moglen retrouve jusqu'en 1990 des traces de son travail bénévole avec Stallman, alors que ce dernier avait demandé son aide juridique pour une affaire privée. Moglen, qui travaillait à l'époque avec l'expert en cryptage Phillip Zimmerman durant ses batailles juridiques avec le gouvernement fédéral[2], fut honoré par cette demande.

« Je lui ai dit que j'avais utilisé *Emacs* chaque jour de ma vie, et qu'il faudrait énormément de conseil juridique de ma part pour rembourser cette dette. »

Depuis lors, Moglen, peut-être plus que quiconque, a eu la meilleure chance d'observer le transfert de la philosophie hacker de Stallman dans la sphère juridique. Selon lui, l'approche de Stallman face au code juridique et son approche du code logiciel sont essentiellement les mêmes. « Je dois dire, en tant qu'avocat, que l'idée selon laquelle ce qu'il faudrait faire avec un texte de loi est d'en supprimer tous les bogues, n'a pas vraiment de sens », dit Moglen. « Il y a de l'incertitude dans tout processus juridique, et les avocats cherchent à tirer le bénéfice de ces incertitudes à l'avantage de leur client. L'objectif de Richard est complètement à l'opposé. Son but est de supprimer l'incertitude, ce qui est tout bonnement impossible. Il est intrinsèquement impossible de mettre en forme une licence qui puisse contrôler toutes les circonstances dans tous les systèmes juridiques à travers le monde entier. Mais si jamais vous essayez de le faire, il faudrait vous y prendre à la façon de Richard. Et l'élégance, la simplicité résultant de ce qui est ainsi élaboré, atteint quasiment l'objectif fixé. À partir de là, avec un peu de travail d'avocat, vous pouvez aller loin. »

Chargé d'argumenter en faveur des activités de Stallman, Moglen comprend la frustration des alliés potentiels. « Richard est un homme qui ne fait aucune concession sur les problématiques qu'il considère comme fondamentales », dit-il, « et il prend mal le fait de jouer sur les mots ou même seulement la recherche de flou artistique, ce que la société demande souvent d'un grand nombre de personnes. »

Du fait de la réticence de la FSF à peser sur des questions hors du champ du développement de GNU et du renforcement de la GPL, Moglen consacra son énergie débordante au service de la Fondation *Electronic Frontier*, une organisation offrant une aide juridique aux ennemis des nouveaux copyrights, comme Dmitri Skylarov. En 2000, Moglen servit aussi comme conseiller direct pour un groupe de hackers qui s'étaient regroupés pour diffuser *deCSS*, le programme de décryptage de DVD. Malgré le silence de son client principal dans les deux cas, Moglen apprit à apprécier la ténacité de Stallman. Au cours des années passées, il y a eu bien des fois où je suis allé voir Richard pour lui dire : « 'Il faut faire ci. Il faut faire ça. Voilà la situation stratégique. Voilà le prochain mouvement. Voilà ce qu'il faut faire'. Et la réponse de Richard a toujours été : 'Nous n'avons pas à faire quoi que ce soit.' Il suffit d'attendre. Ce qui doit être fait le sera. »

« Et vous savez quoi? », ajoute Moglen, « En général il avait raison. »

De tels commentaires désavouent l'auto-critique de Stallman : « Je ne suis pas doué pour le jeu », dit-il, répondant aux nombreux détracteurs anonymes qui le voient comme un stratège perspicace. « Je ne suis pas doué pour prévoir et anticiper ce que les autres vont faire. Ma démarche a toujours été de me concentrer sur la Fondation et de dire 'Rendons-la aussi forte que nous le pouvons'. »

La popularité croissante de la GPL et sa force gravitationnelle continue sont la plus belle récompense pour la Fondation créée par Stallman et ses collègues de GNU. Bien qu'il ne soit à présent plus en mesure de se qualifier de « dernier vrai hacker », Stallman peut cependant se prévaloir d'avoir construit le canevas éthique du mouvement. Que les programmeurs actuels se sentent ou non à l'aise au sein de ce canevas n'est pas la question. Le fait qu'ils aient le choix est le plus grand legs de Stallman.

Cela dit, parler de legs semble un peu prématuré pour l'instant. Stallman, âgé de 48 ans au moment de l'écriture de ce livre, a encore quelques années devant lui pour ajouter ou soustraire à ce legs. Reste que la

nature autonome du mouvement pour le logiciel libre rend attirante l'idée d'examiner la vie de Stallman en dehors de ses batailles quotidiennes avec l'industrie du logiciel, et davantage d'un point de vue historique et prestigieux.

A son crédit, Stallman refuse tout prétexte à spéculer. « Je n'ai jamais été capable d'établir des plans détaillés sur l'avenir », dit Stallman, énonçant prématurément sa propre épitaphe. « J'ai juste dit : 'Je vais me battre. Qui sait jusqu'où cela me mènera?' »

Il est évident qu'en choisissant ses combats, Stallman a éloigné ceux-là même qui autrement auraient été ses plus grands champions. C'est aussi une preuve de sa nature vertueuse et franche de voir nombre de ses anciens ennemis politiques argumenter en sa faveur lorsqu'ils y sont bien obligés. Cependant, le tiraillement entre Stallman l'idéologue et Stallman le génie hacker amène un biographe à se demander de quelle manière Stallman sera considéré lorsque sa propre personnalité ne sera plus là pour baliser le chemin...

Dans les versions préliminaires de ce livre, j'ai intitulé cela la question des « cent ans ». Espérant dégager une vision objective de Stallman et de son travail, j'ai demandé à diverses têtes pensantes de l'industrie du logiciel de s'extraire du contexte actuel et de se mettre dans la position d'un historien se penchant sur le mouvement du logiciel libre dans cent ans. Du point de vue actuel, il est aisé de voir des similarités entre Stallman et ces américains du passé qui, bien que marginaux au cours de leur carrière, ont acquis une dimension historique une fois plus vieux. Des similitudes se trouvent facilement chez Henry David Thoreau, philosophe transcendentaliste et auteur de *La désobéissance civile*, et John Muir, fondateur du Sierra Club, père du mouvement environnementaliste moderne. De même, on peut penser à William Jennings Bryan, dit le « Grand Roturier », leader du mouvement populiste, ennemi des monopoles, et qui, bien qu'il fût un homme puissant, semble être tombé dans l'oubli.

Bien que n'étant pas le premier à considérer le logiciel comme un bien public, Stallman aura assurément une place dans les livres d'histoire grâce à la GPL. À partir de là, il semble intéressant de prendre du recul et d'examiner le legs de Richard Stallman en dehors du contexte actuel. La GPL sera-t-elle toujours utilisée par les programmeurs en 2102, ou aura-t-elle été reléguée aux oubliettes depuis longtemps? L'expression *Free Software* sera-t-elle demain aussi obsolète que l'expression *Free Silver* l'est aujourd'hui, ou sera-t-elle considérée comme étonnamment visionnaire au vu des évènements politiques à venir?

Prédire le futur est un sport risqué, mais la plupart des gens, quand on leur pose la question, semblent prêts à saisir leur chance. « Dans cent ans, Richard et quelques autres mériteront plus qu'une place dans les livres d'histoire » , dit Moglen. « Ils seront considérés comme les personnages principaux du récit. »

Les « quelques autres » que Moglen nomme pour les chapitres des futurs livres d'histoire incluent John Gilmore, le conseiller de Stallman pour la GPL et père de la Fondation *Electronic Frontier*, et Theodor Holm Nelson, dit Ted Nelson, auteur en 1982 du livre *Literary Machines*. Selon Moglen, Stallman, Nelson, et Gilmore se singularisent historiquement, mais de manière différente. Il accorde à Nelson, communément reconnu pour avoir proposé le terme *hypertexte*, le mérite d'avoir identifié la complexité de la possession de l'information à l'âge digital. Gilmore et Stallman, entre temps, ont eu le grand mérite d'avoir identifié les effets politiques néfastes du contrôle de l'information et d'avoir mis sur pied des organisations pour lutter contre ces effets : la Fondation *Electronic Frontier* dans le cas de Gilmore et la FSF dans le cas de Stallman. Des deux, cependant, Moglen voit les activités de Stallman comme plus personnelles et moins politiques dans leur forme.

« Richard était unique, du fait que, très tôt, les implications éthiques des logiciels non-libres lui étaient particulièrement claires », dit Moglen. « Cela a un rapport avec la personnalité de Richard, et de nombreuses personnes, lorsqu'elles écriront à propos de lui, tenteront de décrire cette personnalité comme un épiphénomène, voire même comme un handicap dans l'oeuvre de sa vie. »

Gilmore, qui considère sa position entre l'erratique Nelson et l'irrascible Stallman comme un genre d'« honneur mitigé », approuve néanmoins l'argument de Moglen. Gilmore écrit :

> Mon souhait est que les écrits de Stallman résistent autant que ceux de Thomas Jefferson ; c'est un auteur très clair, y compris dans ses principes... Que Richard devienne ou non aussi influent que Jefferson le fut, dépendra de l'importance que les abstractions nommées « droits civiques » prendront dans cent ans face aux abstractions que nous nommons « logiciels » ou « restrictions techniquement

imposées ».

Un autre élément de l'héritage de Stallman qu'il ne faut pas considérer à la légère, écrit Gilmore, est le modèle collaboratif de développement logiciel initié par le projet GNU. Bien qu'il se révèle parfois imparfait, ce modèle est néanmoins devenu un standard au sein de l'industrie du développement logiciel. Tout compte fait, dit Gilmore, ce modèle de développement collaboratif pourrait devenir plus important que le Projet GNU, la licence GPL, ou tout logiciel développé par Stallman :

> Avant l'Internet, il était particulièrement difficile de collaborer à distance sur des logiciels, même au sein d'équipes qui se connaissaient et se faisaient confiance. Richard a ouvert la voie du développement collaboratif, notamment avec les travaux entrepris par des volontaires auparavant désorganisés et qui se rencontraient rarement. Il n'a construit aucun des outils de base nécessaires à cette tâche (le protocole TCP, les listes d'email, *diff* et *patch*, les fichiers *tar*, RCS ou CVS ou remote-CVS), mais il a utilisé ceux qui étaient disponibles pour former des groupes sociaux de programmeurs qui pouvaient ainsi travailler ensemble avec efficacité.

Lawrence Lessig, professeur de droit à Stanford et auteur en 2001 du livre *The Future of Ideas* est aussi dithyrambique [que Gilmore]. Comme beaucoup d'universitaires en droit, Lessig voit la GPL comme l'un des remparts majeurs des "biens digitaux communs" comme on les appelle aujourd'hui, c'est-à-dire le vaste ensemble des logiciels, des standards réseau et de télécommunication, sous contrôle de la communauté, qui ont déclenché la croissance exponentielle de l'Internet lors des trois dernières décennies. Plutôt que de placer Stallman parmi les autres pionniers de l'Internet, tels Vannevar Bush, Vinton Cerf, et J. C. R. Licklider, qui ont poussé les autres à avoir une vision plus large de la technologie informatique, Lessig voit l'impact de Stallman comme plus personnel, introspectif et, finalement, unique :

> [Stallman] a fait passer le débat des faits aux valeurs. Il a montré ce qui était en jeu, et a construit un outil pour porter en avant ces idées... Ceci dit, je ne saurais le placer dans le même contexte que Cerf ou Licklider. L'innovation est différente. Il ne s'agit pas seulement d'un certain type de code, ou d'avoir permis l'Internet. [Il s'agit] bien plutôt d'avoir montré au gens la valeur d'une certaine conception de l'Internet. Je ne pense pas qu'il y ait quelqu'un d'autre dans sa catégorie, que ce soit dans le passé ou dans le futur.

Tout le monde ne considère pas l'héritage de Stallman comme étant gravé dans le marbre, bien entendu. Eric Raymond, l'avocat de l' *open source*, pense que le rôle de leader de Stallman a diminué de façon significative depuis 1996. Raymond voit des signes moins évidents lorsqu'il regarde dans la boule de cristal pour 2102 :

> Je pense que les oeuvres de Stallman (GPL, *Emacs*, GCC) seront perçues comme des travaux révolutionnaires, aux sources de la technologie de l'information. Je pense que l'histoire sera moins tendre avec certaines des théories portées par RMS, et pas sympathique du tout avec ses tendances personnelles à la territorialité et au culte de la personnalité.

Stallman lui-même ne voit pas que des bons présages :

> Ce que l'histoire dira du Projet GNU, d'ici vingt ans, dépendra de qui gagnera la bataille pour la possession du savoir public. Si nous perdons, nous ne serons qu'une note de bas de page. Si nous gagnons, il n'est pas sûr que les gens reconnaîtront le rôle de la distribution logicielle GNU : s'ils pensent qu'il s'agit du système « Linux », ils se construiront une fausse image de ce qu'il s'est passé, et pourquoi.
>
> Mais même si nous gagnons, ce que les gens apprendront de cette histoire dans cent ans dépendra de la politique dominante.

À la recherche de sa propre analogie avec l'histoire du 19ème siècle, Stallman évoque John Brown, le militant abolitionniste considéré comme un héros d'un côté de la ligne Mason Dixon et comme un fou de l'autre côté.

La révolte des esclaves de John Brown ne put jamais se réaliser, mais, durant son procès, une demande nationale pour l'abolition de l'esclavage vit effectivement le jour. Au cours de la guerre civile, John Brown était un héros ; cent ans après, et pour la grande part des années 1900, les livres d'histoire enseignèrent qu'il était fou. Durant la période où la ségrégation était légale, la bigoterie se montrant sans honte, les États-Unis acceptaient en partie l'histoire que le Sud voulait faire passer, et les manuels d'histoire recelaient bien des contrevérités à propos de la guerre civile et des évènements qui y sont relatifs.

De telles comparaisons illustrent à la fois la nature transversale du travail de Stallman, et le caractère bivalent de sa réputation. Il serait difficile de voir la réputation de Stallman abaissée au même niveau d'infamie que celle de Brown le fut après la Reconstruction. Stallman, malgré ses occasionnelles analogies guerrières, n'a jamais poussé à la violence. Reste qu'il est facile d'imaginer un futur où les idées de Stallman finiraient au rebut. En façonnant la cause du logiciel libre non pas comme un mouvement de masse, mais comme une collection de batailles personnelles contre les forces de la tentation propriétaire, Stallman semble avoir créé une situation sans victoire possible, spécialement pour ses nombreux acolytes faisant preuve de la même obstination.

Encore une fois, c'est cette même volonté inébranlable qui pourrait se révéler être le plus grand legs de Stallman. Moglen, observateur privilégié durant les dix dernières années, met en garde ceux qui se méprennent sur la personnalité de Stallman en la considérant comme contre-productive ou épiphénoménale face aux « artefacts » de sa vie. Sans cette personnalité, dit Moglen, il y aurait bien peu d'artefacts à considérer. En tant qu'ancien greffier auprès de la Cour Suprême, il ajoute :

> Ecoutez, le plus grand homme pour qui j'ai jamais travaillé était Thurgood Marshall. Je savais ce qui avait fait de lui un grand homme. Je savais pourquoi il avait été capable de changer le monde dans la mesure de ses capacités. Il serait un peu osé de faire une comparaison, car les deux ne pourraient pas être plus différents : Thurgood Marshall était un homme dans la société, représentant un peuple d'exclus faisant néanmoins partie de cette société, à laquelle il tenait lui aussi malgré tout à appartenir. Son habileté, c'était l'habileté sociale. Mais il restait intègre. Aussi différents soient-ils à tout autre égard, la personne avec qui je le compare désormais le plus, c'est Stallman : entier, compact, fait de la substance qui compose les étoiles, jusqu'au-boutiste.

Pour essayer de confirmer cette image, Moglen pense à un moment partagé au printemps 2000. Le succès du rachat de VA Linux résonnait toujours dans le milieu des affaires, et une demi-douzaine de numéros dediés au logiciel libre faisaient la une des gazettes. Cerné par cet ouragan d'articles et de récits, chacun appelant à des commentaires, Moglen se souvient avoir déjeuné avec Stallman et s'être senti comme réfugié dans l'oeil du cyclone. Durant l'heure suivante, dit-il, la conversation revint tranquillement sur un seul sujet : le renforcement de la GPL.

« Nous étions assis là à parler de ce que nous allions faire à propos de problèmes en Europe de l'est, et de la réaction à avoir lorsque les questions relatives à la propriété de contenu commenceraient à toucher le logiciel libre », se souvient Moglen. « Alors que nous parlions, j'imaginais un instant à quoi nous pouvions ressembler aux yeux des passants. Nous voilà, deux petits barbus anarchistes, à comploter et à planifier les prochaines étapes. Et, bien sûr, Richard était en train de défaire des noeuds dans ses cheveux, les plongeant dans sa soupe, et se comportant à sa manière habituelle. Quiconque écoutant notre conversation aurait pensé que nous étions fous, mais je savais : je savais que la révolution était là, à cette table. Voilà ce qui la fait avancer, et voici l'homme qui la fait avancer. »

Pour Moglen, ce moment-là plus que tout autre mettait en lumière la simplicité fondamentale du style de Stallman.

« C'était drôle », se souvient Moglen. « Je lui ai dit : 'Richard, tu sais, toi et moi sommes les deux seuls à ne gagner aucun argent sur le dos de cette révolution'. Et puis j'ai payé pour le repas, parce que je savais qu'il n'en avait pas les moyens. »

Notes

1. Marco Boerries, entretien avec l'auteur (July, 2000).
2. Pour plus d'informations sur les travaux légaux de Zimmerman, lisez *Crypto* par Steven Levy, p.

287-288. Dans la version originale du livre que vous avez en main, je rapportais que Moglen aidait Zimmerman contre la National Security Agency (NSA). Selon Levy, Zimmerman faisait l'objet d'une enquête de la part de l'U.S. Attorney's Office et des douanes, pas de la NSA.

Épilogue — Écrasante solitude

Écrire la biographie d'une personne vivante c'est un peu comme réaliser une pièce de théâtre. Le drame qui se déroule sur scène est souvent pâle au regard de celui qui a lieu dans les coulisses.

Dans l' *Autobiographie de Malcom X*, Alex Haley donne aux lecteurs un rare épisode de ce genre. Se débarrassant un instant du rôle de narrateur objectif, Haley écrit l'épilogue du livre à la première personne. Cet épilogue explique comment un journaliste indépendant, initialement considéré comme un « outil » et un « espion » par le porte-parole de la Nation de l'Islam, a réussi à composer avec les barrières personnelles et politiques afin de coucher la vie de Malcom X sur le papier.

Alors que j'hésite à comparer ce livre avec l' *Autobiographie de Malcom X*, je dois ma gratitude envers Haley pour cet épilogue franc. Durant ces 12 derniers mois, il a servi comme un manuel d'instructions pour gérer un sujet biographique qui a construit toute sa carrière sur le fait d'être désagréable. Depuis le début, j'avais imaginé conclure cette biographie par un épilogue similaire, autant en guise d'hommage à Haley que pour permettre aux lecteurs de comprendre comment ce livre a pu voir le jour.

La petite histoire a commencé dans un appartement d'Oakland, puis elle se poursuivit à travers les diverses localités mentionnées dans ce livre : Silicon Valley, Maui, Boston, et Cambridge. En fin de compte, c'est l'histoire de deux villes : New York, la capitale mondiale de l'édition littéraire, et Sebastopol en Californie, la capitale de l'édition littéraire du Comté de Sonoma.

Tout débute en avril 2000. A cette époque, j'écrivais des articles pour le malchanceux site Internet BeOpen.com. Une de mes premières missions fut un entretien téléphonique avec Richard M. Stallman. L'entretien fut un tel succès que Slashdot[1] le référença en première page dans sa liste journalière d'articles. Quelques heures après, les serveurs chauffaient chez BeOpen alors que les lecteurs cliquaient pour visiter le site.

En principe, l'histoire aurait dû se terminer là. Trois mois après ce premier entretien, alors que j'assistai à la conférence O'Reilly sur l'*open source* à Monterey en Californie, je reçu le courrier électronique suivant de la part de Tracy Pattison, manager des droits étrangers pour une grande maison d'édition new-yorkaise :

> Pour: sam@BeOpen.com
> Sujet: Entretien RMS
> Date: Lundi 10 Juillet 2000 15:56:37 -0400
>
> Cher M. Williams,
>
> J'ai lu votre entretien avec Richard Stallman sur BeOpen avec un grand intérêt. Je m'intéresse à RMS et son travail depuis pas mal de temps maintenant, et j'ai été ravie de lire votre contribution dans laquelle je pense vraiment que vous avez fait du bon travail pour capturer un peu de l'esprit de ce que Stallman essaye de faire avec GNU-Linux et la Fondation pour le Logiciel Libre.
>
> Ce que j'adorerais faire, cependant, c'est d'en lire davantage, et je ne pense pas être la seule dans ce cas. Pensez-vous qu'il y ait des informations et/ou des sources supplémentaires pour compléter et actualiser votre entretien et l'adapter comme une biographie ? Peut-être inclure des informations plus anecdotiques sur sa personnalité et son histoire qui pourraient vraiment intéresser et éclairer les lecteurs en dehors du cercle des programmeurs hardcore?

Tracy terminait le message en me demandant de lui téléphoner pour parler plus longuement de cette idée. C'est ce que je fis. Tracy me dit que son entreprise lançait une nouvelle série de livres électroniques, et qu'elle cherchait des récits qui puissent attirer des lecteurs adeptes de nouveautés. Le format du livre électronique était de 30.000 mots, soit environ 100 pages, et elle avait suggéré à ses supérieurs l'idée de profiler un acteur majeur de la communauté hacker. Ses chefs apprécièrent la proposition, et dans ses recherches de personnes intéressantes à profiler, elle avait trouvé l'entretien de Stallman sur BeOpen. D'où

son courrier électronique.

C'est là que Tracy me demanda : serais-je d'accord pour étendre l'entretien en un profil complet?

Ma réponse fut immédiate : oui. Avant d'accepter, Tracy suggéra que je mette au point une proposition de récit qu'elle pourrait présenter à ses supérieurs. Deux jours après, je lui envoyai une proposition mise en forme. Une semaine plus tard, Tracy me contacta à nouveau par courrier électronique: ses chefs avaient donné le feu vert.

Je dois admettre que penser obtenir de Stallman sa participation à un projet de livre électronique était un peu prématuré de ma part. En tant que journaliste couvrant le mouvement *open source*, je savais que Stallman n'était pas commode. J'avais déjà reçu à ce moment-là une demi-douzaine de courriers électroniques dénonçant mon emploi du terme « Linux » au lieu de « GNU/Linux ».

Ceci dit, je savais aussi que Stallman recherchait des moyens pour diffuser son message vers le grand public. Il se pouvait qu'en lui présentant le projet de cette manière, il fût plus réceptif. Sinon, je pouvais toujours m'en remettre à la grande quantité de documents, d'entretiens, et de conversations enregistrées en ligne que Stallman avait laissé traîner sur l'Internet et faire une biographie non-autorisée.

Durant mes recherches, j'ai trouvé un essai intitulé *Liberté, ou Copyright ?*, écrit par Stallman et publié en Juin 2000, dans la *MIT Technology Review*. L'essai critiquait les livres électroniques pour leur quantité de péchés en matière de logiciels. Non seulement les lecteurs devaient utiliser des logiciels propriétaires pour pouvoir les lire, se lamentait Stallman, mais les méthodes employées pour empêcher les copies non-autorisées étaient exagérément rudes. Au lieu de télécharger un fichier transférable au format HTML ou PDF, les lecteurs téléchargeaient un fichier encrypté. Fondamentalement, acheter un livre électronique signifiait acheter une clef non-transférable permettant de traduire le contenu encrypté. Toute tentative d'ouvrir un livre électronique sans la clef autorisée signifiait enfreindre le *Digital Millennium Copyright Act*, la loi de 1998 dédiée à soutenir l'application du copyright sur l'Internet. Des pénalités similaires étaient promises à ceux des lecteurs qui convertiraient le contenu du livre dans un format de fichiers ouvert, même si leur seule intention était de pouvoir lire le livre sur un autre ordinateur chez eux. Contrairement à un livre normal, les lecteurs ne détenaient plus le droit de prêter, copier, ou revendre un livre électronique. Ils n'avaient que le droit de le lire sur une machine autorisée, nous avertissait Stallman:

> Nous avons toujours les mêmes anciennes libertés en utilisant des livres papier. Mais si les livres électroniques venaient à remplacer les versions papier, cette exception ne ferait pas long feu. Avec « l'encre électronique », qui rend possible de télécharger un nouveau texte sur une feuille de papier apparemment imprimée, même les journaux pourraient devenir éphémères. Imaginez : plus de marchands de livres d'occasion, plus de prêt de livres à des amis ; plus d'emprunt à la bibliothèque locale, plus de « fuites » qui pourraient donner à quelqu'un la possibilité de lire sans payer (et, au jugé des publicités pour Microsoft Reader, plus d'achat anonyme de livre non plus). C'est ça le monde que les éditeurs envisagent pour nous[2].

Il va sans dire que cet essai souleva certaines interrogations. Ni Tracy ni moi n'avions parlé du logiciel que son entreprise allait utiliser, ou même du type de copyright auquel l'utilisation du livre électronique serait soumise. J'évoquai l'article de la *MIT Technology Review*, et demandai à Tracy si elle pouvait me fournir des informations sur la politique de son entreprise concernant les livres électroniques. Tracy me promit de me contacter à nouveau.

Impatient de commencer, je décidai d'appeler Stallman malgré tout, et lui présenter l'idée du livre. Quand je le fis, il exprima immédiatement de l'intérêt et du souci. « Est-ce que tu as lu mon article sur les livres électroniques? », me demanda-t-il.

Quand je lui dis: « oui, j'ai lu l'article et j'attends des nouvelles de l'éditeur », Stallman énonça deux conditions : d'une part, il ne voulait pas soutenir un mécanisme de licence de livre électronique envers lequel il s'opposait fondamentalement, et d'autre part, il ne voulait pas se taire s'il collaborait au livre. « Je ne veux pas participer à quoi que ce soit qui me ferait apparaître comme un hypocrite », dit-il.

Pour Stallman, le problème du logiciel passait après celui du copyright. Il affirma qu'il était prêt à passer outre le logiciel que l'éditeur ou ses distributeurs utilisaient, tant que l'entreprise précisait dans le

copyright que les lecteurs étaient autorisés à faire et distribuer des copies intégrales du contenu du livre électronique. Stallman désigna *The Plant* de Stephen King comme modèle possible. En effet, en juin 2000, King annonça sur son site Internet officiel qu'il allait auto-publier *The Plant* sous forme d'épisodes. Selon cette déclaration, le coût total du livre serait de 13 dollars, répartis sur une série de chapitres à 1 dollar. Tant qu'au moins 75% des lecteurs payaient pour chaque chapitre, King promis de continuer à publier les nouveaux épisodes. En août, le plan semblait fonctionner, alors que King avait publié les deux premiers chapitres et que le troisième était en cours.

« Je serais prêt à accepter quelque chose comme ça », dit Stallman. « Tant qu'il est aussi autorisé de faire des copies exactes ».

Je fis suivre cette information à Tracy. J'étais convaincu qu'elle et moi pourrions trouver un arrangement équitable. J'appelai ensuite Stallman et convins d'un premier entretien pour le livre. Stallman accepta de me rencontrer sans demander à nouveau où en était la question du copyright. Peu après le premier entretien, je fonçai vers le second (à Kihei cette fois), m'arrangeant pour rencontrer Stallman avant son départ pour 14 jours de congés à Tahiti.

C'est au cours des vacances de Stallman que la mauvaise nouvelle arriva de la part de Tracy. Le département des affaires juridiques de son entreprise ne voulait pas infléchir sa politique de copyright sur les livres électroniques. Les lecteurs qui voudraient rendre leur livre transférable devraient soit craquer le code d'encryptage, soit convertir le livre dans un format ouvert tel que HTML. Dans les deux cas, ils enfreindraient la loi et s'exposeraient à des sanctions juridiques.

Avec deux entretiens frais dans ma besace, je ne voyais pas d'autre manière d'écrire un livre sans prendre en compte ces données nouvelles. J'arrangeai rapidement un voyage à New York pour rencontrer mon agent et Tracy afin de voir si un compromis était possible.

A mon arrivée à New York, je rencontrai mon agent, Henning Guttman. C'était notre premier entretien face à face, et Henning semblait pessimiste concernant nos chances de forcer un compromis du côté de l'éditeur. Les grandes maisons d'édition bien établies voyaient déjà le format livre électronique avec suffisamment de suspicion, et n'étaient pas dans l'état d'esprit idéal pour expérimenter les termes du copyright en rendant plus facile pour les lecteurs d'éviter de payer. Cependant, en tant qu'agent se spécialisant dans les livres sur la technologie, Henning était surpris par la nature nouvelle de mon problème. Je lui parlai des deux entretiens que j'avais déjà collectés et de la promesse faite à Stallman de ne pas publier le livre de manière qui le « ferait apparaître comme un hypocrite ». Convenant que j'étais lié du point de vue éthique, Henning suggéra d'en faire notre argument de négociation.

En opposant ce fait, dit Henning, nous pourrions toujours adopter la stratégie du bâton et de la carotte. La carotte serait la publicité qui viendrait avec la publication d'un livre électronique respectant l'éthique interne de la communauté hacker. Le bâton serait l'ensemble des risques associés à la publication d'un livre ne la respectant pas. Neuf mois avant que Dmitri Skylarov devienne la "cause célèbre" de l'Internet, nous savions que ce n'était qu'une question de temps avant qu'un programmeur entreprenant révèle comment craquer les livres électroniques. Nous savions aussi que voir une maison d'édition majeure publier un livre électronique protégé par encrytage et concernant Richard M. Stallman était l'équivalent logiciel d'écrire « Volez Ce Livre Electronique » sur la couverture.

Après ma rencontre avec Henning, j'appelai Stallman. Espérant rendre la carotte encore plus appétissante, je discutai avec lui de certains compromis potentiels. Et si l'éditeur publiait le livre sous une double licence, un peu comme Sun Microsystems l'avait fait avec *Open Office*, la suite bureautique libre ? L'éditeur pourrait ensuite publier des versions commerciales du livre électronique sous son format maison, tirant avantage des revenus qui venaient avec le logiciel dédié, tout en publiant une version librement distribuable sous le format moins esthétique HTML.

Stallman dit qu'il ne s'opposait pas à la double licence, mais qu'il n'aimait pas l'idée de rendre la version libre inférieure à la version restreinte. Par ailleurs, dit-il, cette idée était trop compliquée. Les licences doubles convenaient pour le cas *Open Office* uniquement parce qu'il n'avait eu aucun contrôle sur la prise de décision. Dans le cas présent, dit Stallman, il avait une possibilité de contrôler le résultat. Il pouvait refuser de coopérer.

Je fis quelques autres suggestions sans obtenir plus d'effet. A peu près la seule chose que je pus obtenir de Stallman fut une concession pour que le copyright du livre électronique restreigne toute forme d'échange de fichier à la « distribution non-commerciale ».

Avant de raccrocher, Stallman suggéra que je dise à l'éditeur que je lui avait promis que le résultat serait libre. Je lui répondis que je ne pouvais pas accepter cette déclaration, mais que je considérais le livre comme non-achevable sans sa coopération. Apparemment satisfait, Stallman raccrocha avec sa phrase traditionnelle: « Hacke bien. »

Henning et moi rencontrâmes Tracy le lendemain. Elle annonça que son entreprise était prête à publier des extraits non-encryptés mais qu'elle limiterait ces extraits à 500 mots. Henning lui signifia que cela ne serait pas suffisant pour m'affranchir de l'obligation éthique envers Stallman. Tracy évoqua les obligations contractuelles de son entreprise envers divers marchands en ligne tels que Amazon.com. Même si l'entreprise décidait d'ouvrir le contenu du livre électronique pour cette fois seulement, elle s'exposait au risque de voir ses partenaires dénoncer une rupture de contrat. Excluant tout changement d'opinion de la part des directeurs ou de Stallman, la décision m'échouait. Je pouvais utiliser mes entretiens et m'opposer à mes engagements avec Stallman, ou bien je pouvais plaider l'éthique journalistique et m'affranchir de l'accord oral pour écrire le livre.

A la suite de cette réunion, mon agent et moi nous installâmes dans un pub de la troisième avenue. J'utilisai son téléphone cellulaire pour appeler Stallman, laissant un message puisque personne ne décrochait. Henning s'en alla un instant, me laissant le temps de mettre mes idées au clair. Quand il revint, il tenait son téléphone à la main.

« C'est Stallman », dit Henning.

La conversation tourna mal dès le début. Je lui relayai le commentaire de Tracy concernant les obligations contractuelles de l'éditeur.

« Eh bien », dit Stallman abruptement. « Qu'est-ce qui pourrait faire que je m'intéresse à leurs obligations contractuelles? »

Parce que demander à une maison d'édition majeure de s'exposer à une bataille juridique avec ses distributeurs au nom d'un livre électronique de 30.000 mots est une bien grande chose, suggérais-je.

« Est-ce que tu comprends? », dit Stallman. « C'est exactement la raison pour laquelle je fais ça. Je veux une victoire qui soit un message. Je veux qu'ils aient à faire un choix entre les libertés et leurs pratiques commerciales habituelles. »

Alors que les mots « une victoire qui soit un message » faisaient écho dans ma tête, je sentis mon attention s'envoler momentanément vers le trafic piétonnier sur le trottoir. En rentrant dans le bar, j'avais constaté avec joie que le lieu était à moins d'un bloc du carrefour immortalisé par la chanson des Ramones en 1976, "53rd and 3rd", une chanson que j'avais toujours aimé jouer durant mon époque de musicien. Tel le prostitué éternellement frustré décrit par cette chanson, je pouvais sentir les choses s'écrouler aussi vite qu'elles s'étaient construites. L'ironie était palpable. Après des semaines d'écoute attentive des lamentations des autres, je me retrouvais dans la position d'essayer d'obtenir le plus rare des festins: un compromis de Richard Stallman.

Alors que je continuais à parler, plaidant la position de l'éditeur et révélant ma sympathie grandissante pour cette dernière, Stallman, tel un animal, sentit le sang et attaqua.

« Alors c'est tout? Tu vas simplement me baiser? Tu vas simplement te soumettre à leur volonté? »

Je soulevai à nouveau la question du double copyright.

« Tu veux dire licence », dit sèchement Stallman.

"Oui, licence. Copyright. Peu importe.", dis-je, me sentant subitement tel un thon blessé répandant une large traînée de plasma dans l'eau.

« Ah, mais putain pourquoi n'as-tu pas fait ce que je t'avais dit! », cria-t-il.

J'avais du soutenir la position de l'éditeur jusqu'à l'extrême, car dans mes notes, j'ai réussi à consigner

la châtaigne finale de Stallman : « Ca m'est égal. Ce qu'ils font est mal. Je ne peux pas supporter le mal. Au revoir. »

Dès que je reposai le téléphone, mon agent fit glisser une Guiness fraîchement servie vers moi. « Je me suis dit que tu en aurais sans doute besoin », dit-il en riant. « Je pouvais te voir trembler sur la fin. »

Je tremblais effectivement. Ce tremblement ne cesserait qu'après une bonne moitié de Guiness. Ca faisait bizarre de m'entendre être qualifié d'émissaire du « mal ». Plus étrange encore, sachant qu'il y avait trois mois, j'étais dans un appartement d'Oakland, cherchant l'idée de mon prochain récit. A présent, j'étais assis à un endroit du monde que je ne connaissais qu'au travers de chansons rock, rencontrant des directeurs d'édition et buvant une bière avec un agent que je n'avais rencontré qu'hier. Tout cela était trop surréaliste, comme si je voyais ma vie mise sous la forme d'un montage de cinéma.

A ce moment, mon compteur interne d'absurdité pris le relais. Le tremblement initial laissa la place à des rires convulsifs. Pour mon agent, je devais ressembler à un de ces auteurs fragiles subissant une rupture émotionnelle intime. A mes yeux, je commençais à peine à apprécier la beauté cynique de ma situation. Contrat ou pas contrat, j'avais déjà le commencement d'un sujet plutôt bon. Il ne s'agissait que de trouver un endroit où le raconter. Quand mes rires convulsifs se calmèrent enfin, je levai mon verre pour porter un toast.

« Bienvenue sur le front, mon ami », dis-je, trinquant avec mon agent. «Il se peut même que tu aimes ça. »

Si cette histoire avait été une pièce de théâtre, c'est à ce moment là que je placerais un intermède romantique. Déprimée par la nature tendue de notre rencontre, Tracy nous invita avec Henning à aller boire quelques verres avec elle et certains de ses collègues de bureau. Nous quittâmes alors le bar sur la troisième avenue, pour nous diriger vers East Village, où nous rattrapâmes Tracy et ses amis.

Une fois rendu sur place, je parlai avec Tracy, évitant soigneusement d'évoquer le travail. Notre discussion fut plaisante et relaxée. Avant de nous quitter, nous décidâmes de nous revoir le soir suivant. A nouveau, la conversation fut agréable, à tel point que le livre électronique sur Stallman était devenu un lointain souvenir.

Une fois rentré à Oakland, je pus appeler divers journalistes amis ou connus. Je leur racontai mes misères. La plupart me tancèrent pour avoir trop cédé à Stallman dans la négociation préliminaire. Un ancien professeur en école de journalisme me suggéra d'ignorer le commentaire de Stallman sur les « hypocrites » et d'écrire mon histoire malgré tout. Ceux connaissant Stallman et son savoir des médias m'exprimèrent leur sympathie mais offrirent tous la même réponse : à toi de voir.

Je décidai de mettre le livre en attente. Malgré les entretiens, je n'avançais pas. Par ailleurs, cela me donna l'opportunité de parler à Tracy sans avoir d'abord à passer par Henning. Aux alentours de Noël, nous échangions les visites : tantôt elle venait sur la côte ouest, tantôt j'allais à New York. Le jour précédant la nouvelle année, je lui fis ma demande. Décidant où nous installer, je choisis de venir à New York. En Février, j'emballai mon ordinateur portable et toutes mes notes préliminaires liées à la biographie de Stallman, et nous nous envolâmes pour l'aéroport JFK. Tracy et moi étions mariés le 11 Mai. Merci aux contrats d'édition ratés.

Durant l'été, je commençai à arranger mes notes d'entretiens sous la forme d'un article pour un magazine. D'un point de vue éthique, je me sentais en droit de le faire, puisque les termes originaux encadrant ces entretiens ne stipulaient rien pour la presse papier traditionnelle. Pour être tout à fait honnête, j'étais aussi plus à l'aise pour écrire sur Stallman après huit mois de silence radio. Depuis notre conversation téléphonique en Septembre, je n'avais reçu que deux courriers électroniques de Stallman. Les deux me châtiaient pour avoir utilisé « Linux » au lieu de « GNU/Linux » dans une paire d'articles pour le magazine en ligne *Upside Today*. A part ça, ce fut le silence. En juin, environ une semaine après son discours à l'université de New York, je pris l'initiative d'écrire un article de magazine de 5000 mots sur Stallman. Cette fois, les mots affluèrent. La distance avait aidé à restaurer mon sens perdu de la perspective émotionnelle, je suppose.

En juillet, une année complète après le courrier originel de Tracy, je reçu un appel d'Henning. Il me dit que O'Reilly & Associates, une maison d'édition basée à Sebastopol, Californie, était intéressée pour publier l'histoire de Stallman sous la forme d'une biographie. La nouvelle me ravit. De toutes les maisons d'édition de par le monde, O'Reilly, la même entreprise qui avait publié *La Cathédrale et le Bazar* d'Eric

Raymond, semblait la plus attentive aux problèmes qui avaient tué le livre électronique précédent. En tant que journaliste, je m'étais beaucoup servi du livre *Open Sources* de O'Reilly comme d'une référence historique. Je savais aussi que divers chapitres du livre, dont celui écrit par Stallman, avaient été publiés avec des notices de copyright autorisant la redistribution. De telles informations seraient utiles si le problème de la publication électronique faisait à nouveau surface.

Bien entendu, le problème revit le jour. J'appris par Henning que O'Reilly voulait publier la biographie à la fois sous forme papier traditionnelle, mais aussi dans sa nouvelle offre de service payante *Safari Tech Books Online*. La licence utilisateur de Safari impliquerait des restrictions spécifiques[3], m'avertit Henning, mais O'Reilly souhaitait permettre un copyright qui donnait aux utilisateurs le droit de copier le texte, quelque soit le médium employé. Fondamentalement, en tant qu'auteur, j'avais le choix entre deux licences: la Licence Open Publication ou la Licence Documentation Libre GNU.

Je consultai alors le contenu et l'histoire de chaque licence. La Licence Open Publication (OPL)[4] donne aux lecteurs le droit de reproduire et distribuer une réalisation, en partie ou entièrement, de manière « physique ou électronique », tant que la copie garde la Licence Open Publication. Elle permet aussi les modifications de la réalisation, sous réserve de certaines conditions. Enfin, la Licence Open Publication contient plusieurs options, qui, si elles sont choisies par l'auteur, peuvent limiter la création de versions "significativement modifiées" ou de dérivés sous la forme de livres sans l'accord préalable de l'auteur.

La Licence Documentation Libre GNU (GFDL)[5], quant à elle, permet la reproduction et la distribution d'un document sous n'importe quelle forme, tant que les versions résultantes portent la même licence. Elle permet aussi la modification du document sous certaines conditions. Contrairement à l'OPL, cependant, elle ne donne pas aux auteurs la possibilité de restreindre certaines modifications. Elle ne donne pas non plus aux auteurs le droit de rejeter certaines modifications qui pourraient résulter en un produit littéraire concurrent. Elle requiert cependant que certaines informations soient inscrites sur la couverture et le dos du livre si une personne différente du propriétaire du copyright souhaite publier plus de cent copies d'un travail protégé.

Durant mon processus de recherche sur les licences , je me suis aussi assuré de visiter la page du site Internet du Projet GNU intitulée « Diverses Licences et Commentaires à leur Propos »[6]. Sur cette page, j'ai trouvé une critique de Stallman concernant la Licence Open Publication. Elle concernait la création de versions modifiées et la capacité de l'auteur à choisir une des options de l'OPL restreignant les modifications. Si un auteur ne souhaitait choisir aucune de ces options, il avait intérêt à utiliser plutôt la GFDL, remarquait Stallman, puisque cela minimisait le risque de voir activées ces options non-souhaitées dans des versions modifiées du document.

L'importance accordée aux modifications dans ces deux licences reflétait leur but originel : donner aux propriétaires de manuels logiciels une chance d'améliorer leurs produits et de publier ces améliorations au bénéfice du reste de la communauté. Puisque mon livre n'était pas un manuel, j'accordai peu d'intérêt aux clauses de modifications dans les deux licences. Ma seule considération était d'offrir aux lecteurs le droit de copier ou d'échanger le contenu, la même liberté dont ils auraient bénéficié en achetant une version imprimée du livre. Considérant ces deux licences convenables pour mon objectif, je signai le contrat avec O'Reilly dès qu'il me fut envoyé.

Reste que la notion de modifications non restreintes m'intriguait. Dans les premières négociations avec Tracy, j'avais présenté les mérites d'une licence de type GPL dans le cadre du contenu d'un livre électronique. Au pire, avais-je dit, la licence garantirait beaucoup de publicité favorable pour le livre électronique. Au mieux, elle inciterait les lecteurs à participer au processus d'écriture. En tant qu'auteur, j'étais prêt à laisser les autres amender mon travail tant que mon nom restait en position principale. Par ailleurs, il pouvait même être intéressant d'observer l'évolution du livre. J'imaginais les versions futures comme des versions en ligne du "Talmud", avec mon texte originel comme pilier central, entouré d'enluminures et de commentaires de contributeurs dans les marges.

Mon idée tirait son inspiration du Projet Xanadu (*http://www.xanadu.com*), le concept logiciel légendaire originellement conçu par Ted Nelson en 1960. Au cours de la conférence O'Reilly sur l'*open source* en 1999, j'avais vu la première démonstration de *Udanax*, le dérivé *open source* du projet, et j'avais été impressionné. Durant un passage de la démonstration, *Udanax* affichait côte à côte un document mère et une

version modifiée, dans une mise en forme similaire sur deux colonnes en format texte. Avec un simple clic on pouvait introduire des lignes reliant chaque phrase du document mère à celle correspondante dans le second. Une version électronique de la biographie de Richard M. Stallman ne devait pas nécessairement être compatible avec Udanax, mais étant donné de telles possibilités technologiques, pourquoi ne pas donner aux utilisateurs une possibilité de s'amuser[7] ?

Quand Laurie Petrycki, mon rédacteur chez O'Reilly, me donna le choix entre l'OPL et la GFDL, je caressai à nouveau ce rêve. En Septembre 2001, le mois où je signai le contrat, les livres électroniques étaient quasiment passés de mode. De nombreuses maisons d'édition, dont celle de Tracy, mettaient un terme à leurs séries de livres électroniques pour cause de manque d'intérêt du public. Je devais me poser la question. Si ces entreprises avaient traité les livres électroniques non pas comme une façon de publier, mais comme une façon de créer une communauté, est-ce que ces collections auraient survécu?

Après avoir signé le contrat, j'informai Stallman que le projet de livre était à nouveau sur les rails. J'évoquai le choix que O'Reilly me donnait entre la Licence Open Publication et la Licence Documentation Libre GNU. Je lui dis que je penchais vers l'OPL, car je ne voyais aucune raison de donner aux adversaires de O'Reilly la possibilité de publier le même livre sous une couverture différente. Stallman me répondit, argumentant en faveur de la GFDL, faisant remarquer que O'Reilly l'avait utilisée à plusieurs reprises dans le passé. Malgré les évènements de l'année passée, je proposai alors un arrangement. Je choisirais la GFDL si cela me permettait de faire plus d'entretiens et si Stallman aidait O'Reilly à faire la publicité du livre. Stallman accepta de participer à plus d'entretiens, mais dit que sa participation à des évènements promotionnels dépendrait du contenu du livre. Considérant cela comme juste, je convins alors d'un entretien le 17 décembre 2001 à Cambridge.

Je plaçai la rencontre de manière à coïncider avec un voyage d'affaire de mon épouse Tracy à Boston. Deux jours avant le départ, celle-ci me suggéra d'inviter Stallman à dîner.

« Après tout », dit-elle, « c'est lui qui nous a fait nous rencontrer. »

J'écrivis un courrier électronique à Stallman, qui répondit promptement, acceptant l'offre. Après avoir roulé vers Boston le lendemain, je pris Tracy à son hôtel, puis nous bifurquâmes en direction du MIT. En arrivant à Tech Square, nous surprîmes Stallman au milieu d'une conversation, alors que nous frappions à sa porte.

« Excusez-moi », dit-il, tenant la porte ouverte de suffisamment loin afin que Tracy et moi ne puissions que difficilement entendre son interlocuteur. C'était une jeune femme, aux alentours de 25 ans à mon avis, nommée Sarah.

« Je me suis permis d'inviter une autre personne à ce dîner », dit Stallman, nous mettant devant le fait accompli, me lançant ce même sourire félin qu'il m'avait fait dans ce restaurant de Palo Alto.

Pour être honnête, je n'étais pas vraiment surpris. La rumeur selon laquelle Stallman avait une nouvelle petite amie m'était parvenue quelques jours auparavant, par l'intermédiaire de la mère de Stallman. « En fait, ils sont allés au Japon ensemble le mois dernier quand Richard est allé recevoir le Prix Takeda », m'avait-elle alors annoncé[8].

Sur le chemin vers le restaurant, j'appris les circonstances de la première rencontre entre Richard et Sarah. Curieusement, les circonstances était très familières. Travaillant sur son propre livre de fiction, Sarah avait entendu parler de Stallman et de quel personnage intéressant il était. Elle décida rapidement de s'en inspirer pour l'un des personnages de son livre, et, dans ses recherches, elle arrangea un entretien avec lui. A partir de là, les choses s'enchaînèrent. Les deux étaient ensemble depuis le début 2001, dit-elle.

« J'admirais vraiment la façon avec laquelle Richard avait construit un mouvement politique tout entier afin de traiter un problème profondément personnel », dit Sarah, expliquant son attraction vers Stallman.

Ma femme lança immédiatement la question : « Quel est ce problème? »

« Une solitude écrasante. »

Durant le dîner, je laissai les femmes faire la conversation et passer le plus clair du temps à essayer de repérer des indices pouvant révéler si les douze derniers mois avaient adouci Stallman de façon significative.

Je ne vis rien qui puisse le suggérer. Bien que plus charmeur que dans mes souvenirs (un charme néanmoins gâché, en quelque sorte, par le nombre de fois où le regard de Stallman semblait fixé sur les seins de mon épouse), Stallman conservait ce même niveau d'acidité. A un moment, ma femme entama un emphatique « Dieu nous en garde » pour se voir instantanément rétorquer une réprimande typique de Stallman :

« Je suis désolé d'avoir à te l'apprendre, mais Dieu n'existe pas », dit-il.

Plus tard, quand le dîner fut terminé et Sarah partie, Stallman semblait avoir un peu baissé la garde. Alors que nous marchions vers un marchand de livres voisin, il admit que les douze derniers mois avaient énormément changé son regard sur la vie. « e pensais que j'allais être seul pour toujours », dit-il. « Je suis heureux de m'être trompé ».

Avant de nous quitter, Stallman me tendit sa « carte de plaisir », une carte de visite avec son adresse, son numéro de téléphone, et ses passe-temps favoris (« échanger des bons livres, la bonne cuisine, les danses et les musiques exotiques ») afin que je puisse le contacter pour un entretien final.

Le lendemain, après un nouveau repas d'origine douteuse, Stallman semblait encore plus amoureux que le soir précédent. Se souvenant de ses débats avec le dortoir de la Currier House sur les avantages et les inconvénients des sérums d'immortalité, Stallman exprima le souhait que les scientifiques puissent un jour trouver la clef de l'immortalité. « Maintenant que je commence enfin à être heureux dans ma vie, je souhaite que ça dure », dit-il.

Lorsque j'évoquai le commentaire de Sarah sur « l'écrasante solitude », Stallman ne trouvait pas de relation entre la solitude physique et spirituelle, et la solitude des hackers. « L'envie de partager le code a un rapport avec l'amitié, mais l'amitié à un niveau bien moindre », dit-il. Plus tard, cependant, quand le sujet refit surface, Stallman admit que la solitude, ou la crainte de la solitude éternelle, a joué un rôle majeur, comme carburant de sa motivation durant les premiers jours du Projet GNU.

« Ma fascination pour les ordinateurs n'est la conséquence de rien d'autre », dit-il. « Je n'aurais pas été moins fasciné par les ordinateurs même si j'avais été populaire et que les femmes s'attroupaient autour de moi. Cependant, il est vrai que le fait de ne pas avoir de maison, d'en trouver une et de la perdre, d'en trouver une autre pour la voir détruite, m'a affecté profondément. Celle que j'ai perdu, c'est le dortoir, celle qui a été détruite, c'est le AI Lab.. La précarité de n'avoir aucune maison ni communauté fut très forte. Cela m'a donné envie de me battre pour retrouver cela. »

Après l'entretien, je ne pus m'empêcher d'éprouver un sentiment curieux. En entendant Sarah décrire ce qui l'avait attirée vers Stallman, et en entendant Stallman décrire lui-même les sentiments qui l'entraînèrent à porter la cause du logiciel libre, j'étais ramené à mes propres motivations qui présidaient à l'écriture de ce livre. Depuis juillet 2000, j'avais appris à apprécier tant les aspects attractifs que repoussants de la personne de Richard Stallman. Comme Eben Moglen avant moi, je sentis qu'abaisser cette personne au rang d'épiphénomène ou d'élément perturbateur à l'intérieur du mouvement global du logiciel libre serait une grave erreur. En de nombreux points de vue, les deux se définissent mutuellement à tel point qu'ils en deviennent indistinguables.

Cependant, je ne suis pas sûr que tous les lecteurs ressentent ce même niveau d'affinité avec Stallman : en effet, après avoir lu ce livre, certains pourront même ne ressentir aucune affinité. Cependant je suis sûr que la plupart seront d'accord. Peu d'individus offrent un portait aussi singulier que Richard M. Stallman. C'est mon souhait le plus sincère que, avec ce portrait initial complété et avec l'aide de la GFDL, d'autres ressentiront une urgence similaire d'y ajouter leur propre perspective.

Notes

1. http://www.slashdot.org), le site populaire des « nouvelles pour nerds » propriété de VA Software, Inc.. Anciennement VA Linux Systems et, avant cela, VA Research
2. Voir *Freedom—Or Copyright?* (Mai 2000) http://www.technologyreview.com/articles/stallman0500.asp.
3. Voir "Safari Tech Books Online; Subscriber Agreement: Terms of Service". http://safari.oreilly.com/mainhlp.asp?help=service

4. Voir « Safari Tech Books Online; Subscriber Agreement: Terms of Service ». http://safari.oreilly.com/mainhlp.asp?help=service
5. Voir « The GNU Free Documentation License: Version 1.1 » (mars 2000). http://www.gnu.org/copyleft/fdl.html
6. Voir http://www.gnu.org/philosophy/license-list.html
7. Quiconque souhaitant "porter" ce livre vers *Udanax*, la version libre de *Xanadu*, recevra un soutien enthousiaste de ma part. Pour en savoir plus sur cette intéressante technologie, visitez *http://www.udanax.com*.
8. Hélas, ce n'est qu'après que Stallman soit allé au Japon que j'entendis parler de la décision de la Fondation Takeda de récompenser Stallman, Linus Torvalds et Ken Sakamura, avec son tout premier prix pour « Réalisation Techno-Entreprenariale pour le Bien-Être Socio-Économique ». Pour plus d'information à propos de cette récompense et du prix de 1 millions de dollars l'accompagnant, visitez le site de Takeda (http://www.takeda-foundation.jp).

Annexe A — Terminologie

Dans la plupart des cas, j'ai choisi d'utiliser le terme GNU/Linux en référence au système d'exploitation libre et Linux pour désigner spécifiquement le noyau sur lequel repose le système. L'exception la plus remarquable à cette règle se trouve dans le chapitre 9 : dans la dernière partie de ce chapitre, je décris les débuts de l'évolution de Linux comme un dérivé de Minix. On peut admettre en toute sécurité que durant les deux premières années de son développement, le système d'exploitation de Torvalds et ses collègues avait peu en commun avec la vision de Stallman du système GNU, même s'il se mit à partager les composants vitaux de GNU, comme le GNU C Compiler et le GNU Debugger.

Cette décision bénéficie aussi du fait qu'avant 1993, Stallman voyait peu d'intérêt à insister sur le crédit.

D'aucuns pourraient considérer comme arbitraire la décision d'utiliser le terme GNU/Linux pour les versions suivantes du même système. Je tiens à préciser qu'il ne s'agissait en aucun cas d'une condition nécessaire pour gagner la coopération de Stallman quant à l'écriture de ce livre. Cette décision est personnelle, d'une part du fait de la nature modulaire du système et de la communauté qui l'entoure, et, d'autre part, à cause de la nature apolitique de Linux. Étant donné qu'il s'agit d'une biographie de Richard Stallman, il semblait inapproprié de définir le système d'exploitation en termes apolitiques.

Dans les phases finales du livre, quand il devînt clair qu'O'Reilly & Associates serait l'éditeur, Stallman demanda effectivement que « GNU/Linux » et non « Linux » soit utilisé si O'Reilly voulait son appui pour la promotion du livre une fois publié. Une fois informé de cette demande, je relayai ma décision exposée ci-dessus et laissai Stallman juger si le livre en résultant remplissait la condition ou non. À l'heure où j'écris ces mots, je n'ai aucune idée du jugement de Stallman.

Une situation similaire entoure les expressions *logiciel libre* et *open source*. Là encore, j'ai opté pour le plus politique *logiciel libre* pour décrire les logiciels dont le code source est librement copiable et modifiable. Bien que plus populaire, j'ai choisi de n'utiliser *open source* qu'en référence aux groupes et marchés qui promeuvent son usage. Ces termes sont complètement interchangeables sauf dans certains cas, et, en prenant cette décision, j'ai suivi le conseil de Christine Peterson, généralement créditée pour l'usage de cette expression : « L'expression *logiciel libre* devrait toujours être utilisée dans les circonstances où elle convient le mieux », écrit Peterson. « [*Open source*] est devenu populaire principalement parce qu'un nouveau terme était nécessaire, et non parce qu'il est idéal ».

Annexe B — Hack, Hacker et Hacking

Pour comprendre le sens exact du mot "hacker", il est bon de se pencher sur son étymologie au fil des années.

The New Hacker Dictionnary, un abrégé en ligne du jargon des programmeurs, liste officiellement neuf connotations différentes du mot "hack" et autant pour le mot "hacker". Un essai accompagnant cette publication cite Phil Agre, un hacker du MIT qui avertit les lecteurs de ne pas être déboussolés par l'apparente flexibilité du mot. "Hacker n'a qu'une seule signification", argumente Agre. "Une signification profonde et subtile qui défie l'articulation".

Indépendamment de l'étendue de la définition, la plupart des hackers modernes attribuent l'origine au MIT, où le terme naquit en tant que jargon étudiant dans les années cinquante. En 1990, le musée du MIT composa un journal documentant le phénomène des hackers. D'après ce journal, les étudiants de l'institut durant les années cinquante utilisaient le mot "hack" tout comme un étudiant moderne utiliserait le mot "bidouille". Suspendre un vieux coucou à une fenêtre de dortoir était un hack, mais quelque chose de dur ou malveillant, comme par exemple, bombarder d'oeufs les fenêtres d'un dortoir rival ou la dégradation d'une statue du campus, dépassait les limites. Un esprit d'amusement inoffensif, créateur, était implicite dans la définition de hack.

Cet état d'esprit allait inspirer la forme verbale : "hacker". Un étudiant des années cinquante qui aurait passé une bonne partie de son après-midi à parler au téléphone ou à démonter une radio aurait pu vous décrire son activité par "hacker". Là encore, un interlocuteur moderne substituerait la forme verbale pour "bidouille"-"bidouiller" ou bien "bricoler" pour décrire la même activité.

À mesure que les années cinquante passèrent, le mot "hack" acquit un sens plus précis, plus rebelle. Le MIT des années cinquante était extrêmement compétitif, et le "hackage" apparut à la fois comme réaction et extension à cette culture. Bidouillages et blagues devinrent soudain une façon de relâcher la pression, de faire un pied de nez à l'administration du campus, et de laisser libre cours à son esprit et son comportement créatifs, que le rigoureux programme de l'Institut étouffait. Avec ses myriades de couloirs et de conduits de ventilation souterrains, l'Institut offrait beaucoup d'endroits à explorer pour l'étudiant non décontenancé par des portes fermées ou des panneaux "entrée interdite". Les étudiants commencèrent à nommer "spéléo-hacking" leurs explorations hors-pistes. Au sol, le système téléphonique du campus offrait des possibilités similaires. Expérimentations nonchalantes et rigueur apprirent aux étudiants comment faire d'amusantes farces. S'inspirant de la plus traditionnelle course dans les tunnels, les étudiants nommèrent vite cette nouvelle activité "hack téléphonique."

La force combinée du jeu créatif et de l'exploration sans restriction allait servir de base pour les prochaines mutations du terme "hacker". Les premiers hackers informatiques, se décrivant comme tels au début des années soixante sur le campus du MIT, faisaient partie d'un groupe estudiantin de la fin des années cinquante appelé le *Tech Model Railroad Club*. Le groupe à l'origine du circuit électrique alimentant le Railroad Club y formait une fine équipe appelée le *Signals and Power (S&P) Committee*. Le circuit lui-même était un agencement sophistiqué de relais et interrupteurs similaires à ceux qui contrôlaient le central téléphonique du campus. Pour le diriger, un des membres du groupe devait juste composer les commandes sur un téléphone et observer les trains obéissant à ses ordres.

Les jeunes ingénieurs électriciens responsables de la construction de ce système considéraient leur activité avec un esprit similaire au "hacking téléphonique". En adoptant le terme "hacking", ils commencèrent à l'affiner encore davantage. Du point de vue des hackers de S&P, utiliser un relais de moins pour commander une partie du chemin de fer signifiait en avoir un de plus pour s'amuser plus tard. Le sens de "hacking" passa donc de façon subtile d'un synonyme de jeu pour couler le temps à jeu passe-temps améliorant la performance ou l'efficacité générale du système du Railroad Club. Bientôt, les membres du comité de S&P utilisèrent fièrement "hacking" pour désigner le fait d'améliorer et de remodeler le circuit électrique du chemin de fer, et ils nommèrent "hackers" ceux qui pratiquaient cette activité.

Etant donnée leur affinité pour l'électronique sophistiquée, sans parler de leur rébellion envers les

portes fermées et les panneaux "accès interdit", en peu de temps, les hackers eurent vent d'une nouvelle machine sur le campus. Appelée TX-0, il s'agissait d'un des tout premiers ordinateurs commerciaux. A la fin des années cinquante, la bande de S&P avait migré en masse à la salle de contrôle du TX-0, apportant leur esprit de jeu créatif sur la machine. Le royaume grand ouvert de la programmation allait entraîner une nouvelle mutation étymologique. "Hacker" ne signifiait plus alors souder des bouts de circuits disparates, mais monter des programmes informatique sans prêter attention aux procédures "officielles" d'écriture des programmes. Cela se rapportait aussi a l'amélioration des logiciels existants qui avaient tendance à s'approprier trop de ressources système. En phase avec l'origine du mot, cela signifiait aussi écrire d'inutiles programmes juste pour s'amuser.

Un exemple classique de cette extension de "hacking" est le jeu Spacewar, le premier jeu video interactif. Développé par des hackers du MIT au début des années soixante, Spacewar combinait toutes les caractéristiques du hacking traditionnel : il était espiègle et aléatoire, n'ayant d'autre but que de passer le temps pendant la nuit pour la douzaine de hackers qui adoraient y jouer. D'un point de vue logiciel, c'était une innovation monumentale en termes de programmation. Le jeu était aussi entièrement libre. Les hackers l'ayant réalisé pour le plaisir, ils ne voyaient pas de raison de cacher leur création et la partageaient alors avec d'autres programmeurs. A la fin des années soixante, Spacewar était devenu le loisir favori de nombreux programmeurs de par le monde.

Cette notion d'innovation collective et de propriété collective des logiciels séparaient le hacker informatique des années soixante du hacker téléphonique ou du spéléo-hacker des années cinquante, à l'époque où il s'agissait davantage d'un acte individuel ou de petit groupe. Les hackers spéléos et téléphoniques utilisaient l'équipement du campus comme matière première, mais la nature illégale de ces activités décourageait la divulgation de nouvelles découvertes. Par contre, les hackers informatiques travaillaient dans un environnement scientifique basé sur la collaboration et la récompense pour les innovations. Avec les informaticiens scientifiques "officiels", ils n'étaient pas toujours les meilleurs amis qu'il soit, mais avec l'évolution rapide du domaine informatique, les deux espèces de programmeurs évoluèrent vers une relation de coopération, certains disent même symbiotique.

C'est en hommage au talent prodigieux des hackers informatiques originels que leurs successeurs, dont Richard M. Stallman, cherchaient à conserver la même identité de hacker. Vers la moitié des années soixante-dix, le terme hacker avait acquis une connotation élitiste. En général, un hacker informatique était quiconque écrivant du code pour le code. Il s'agissait en particulier d'une reconnaissance de virtuosité technique. Tout comme le terme "artiste", la signification comportait un arrière-goût tribal. Qualifier un autre programmeur de hacker était un signe de respect. Se décrire en tant que hacker faisait montre d'une très grande confiance en soi. En tous cas, la flexibilité originale du mot "hacker" diminua à mesure que les ordinateurs se répandaient.

Durant ce changement de définition, le hacking informatique acquis d'autres sens. Pour être un hacker, une personne devait faire davantage que d'écrire d'intéressants programmes; il fallait appartenir à la "culture hacker" et en honorer les traditions tout comme un viticulteur médiéval aurait juré fidélité à une guilde des vins. La structure sociale n'était toutefois pas aussi rigide, mais les hackers des institutions d'élite comme MIT, Stanford et Carnegie Mellon commencèrent à parler ouvertement de l'"éthique des hackers" : les règles non écrites qui réglaient le comportement quotidien d'un hacker. En 1984, le livre "Hackers", écrit par Steven Levy après beaucoup de recherches et consultations, codifiait l'éthique des hackers en cinq principes fondamentaux.

De bien des manières, les cinq principes de Levy définissent toujours la culture du hacking informatique. Malgré tout, l'apparence guildienne de la communauté fut sapée par l'influence populiste de l'industrie informatique. Au début des années quatre-vingt, les ordinateurs se répandaient partout, et les programmeurs, qui autrefois avaient dû se déplacer dans de prestigieuses institutions ou entreprises juste pour accéder à une machine, avaient soudain la possibilité de coudoyer les plus grands hackers via ARPAnet. Plus ces programmeurs se côtoyaient, plus ils s'approprièrent la philosophie anarchiste des endroits comme le MIT. Par contre, le tabou de la méchanceté fut perdu au milieu du transfert culturel. Tandis que de jeunes programmeurs commencèrent à employer leurs capacités à des fins nuisibles (créer et disséminer des virus, s'introduire dans des systèmes militaires, faire délibérément planter des machines comme Oz du MIT, qui était alors un relais ARPAnet populaire), le terme "hacker" acquit une signification punk, nihiliste. Quand la

police et les entreprises commencèrent à remonter les pistes de crimes informatiques jusqu'à une bande de programmeurs renégats qui citaient de bonnes portions d'éthique hacker pour justifier leurs activités, le mot "hacker" apparut dans les magazines avec une connotation négative. Bien que des ouvrages tels que "Hackers" fissent beaucoup pour documenter l'esprit originel d'exploration qui donna vie à la culture hacker, pour la plupart des journalistes, "hacker informatique" devînt synonyme de "cambrioleur électronique".

Quoique les hackers aient lutté contre cette méprise depuis presque vingt ans, il est difficile, considérant l'origine rebelle des années cinquante, de faire la différence entre un gamin de quinze ans écrivant des programmes pour casser des systèmes de cryptage modernes et un étudiant des années soixante crochetant une serrure pour accéder à un unique terminal. La subversion créative d'une personne est le cauchemar sécuritaire d'une autre, après tout. Malgré cela, le tabou principal à l'encontre de la méchanceté ou d'un comportement délibérément nuisible reste suffisamment fort pour que les hackers préfèrent employer le terme "cracker" pour appeler ces hackers utilisant leurs capacités à mauvais escient, par exemple pour quelqu'un qui briserait volontairement la sécurité d'un système afin de voler ou saccager des informations.

Le tabou essentiel contre la méchanceté reste le lien principal entre la notion de hacker des années cinquante et du début du vingt-et-unième siècle. Il est important de noter que malgré son évolution durant les quarante dernières années, l'idée originelle du hacking, consistant à faire des blagues ou explorer des tunnels, a survécu. Fin 2000, le musée du MIT à révéré la tradition des hackers de l'institut en leur dédiant une exposition, le "Hall of Hacks". Cette dernière montre de nombreuses photographies remontant jusqu'aux années vingt, l'une d'elles montrant une fausse voiture de police. En 1993, les étudiants rendirent hommage à l'esprit originel des hackers en plaçant le même véhicule, gyrophares en marche, sur le toit du dôme principal de l'Institut. La plaque d'immatriculation de la voiture indiquait IHTFP, un acronyme populaire du MIT ayant plusieurs sens. La version la plus intéressante, remontant sans doute à la vie sous pression des étudiants dans les années cinquante, est : "I Hate This Fucking Place" (Je Hais Ce Putain d'Endroit). En 1990, le musée utilisa cet acronyme comme base pour une parution sur l'histoire des hackers, alors intitulée "Institute for Hacks Tomfoolery and Pranks" (l'Institut des Hacks, Niaiseries et Farces). Le journal offre un aperçu pertinent de ce qu'est le hacking :

"Dans la culture du hacking, une création simple et élégante est autant estimée qu'en science pure", écrit Randolph Ryan, reporter du Boston Globe dans un article de 1993 mentionnant la farce de la voiture de police. "Un hack diffère d'une farce estudiantine habituelle car il nécessite une planification élaborée, de l'ingéniosité, de la finesse, et possède à la fois esprit et inventivité", écrit Ryan. "La règle implicite est que le hacking doit être de bonne nature, non destructif et sûr. En fait, les hackers aident même parfois à défaire leur propre ouvrage".

Le besoin de délimiter la culture du hacking informatique par de telles frontières éthiques est plein de bon sens mais impossible. Bien que la plupart des hacks informatiques recherchent le même esprit, la même élégance et la même simplicité, la nature des systèmes informatiques donne peu de possibilités pour faire marche arrière. Démonter un véhicule de police est plus facile que défaire une idée. D'où la distinction grandissante entre "chapeau noir" et "chapeau blanc" : d'un côté, les hackers qui créent de nouvelles idées à des fins destructives, malignes et, de l'autre côté, les hackers dont les créations visent des fins positives ou, du moins, informatives.

Autrefois un terme obscur du jargon étudiant, le mot "hacker" est devenu une boule de billard linguistique sujette à des tournures politiques et des nuances éthiques. C'est peut être pourquoi tant de hackers et journalistes aiment l'utiliser. Où est-ce que la boule rebondira ensuite, par contre, nul ne le sait.

Annexe C — Licence de documentation libre GNU

Licence de documentation libre GNU

Disclaimer

This is an unofficial translation of the GNU Free Documentation License into French. It was not published by the Free Software Foundation, and does not legally state the distribution terms for documentation that uses the GNU FDL--only the original English text of the GNU FDL does that. However, we hope that this translation will help French speakers understand the GNU FDL better.

Ceci est une traduction française non officielle de la Licence de documentation libre GNU. Elle n'a pas été publiée par la Free Software Foundation, et ne fixe pas légalement les conditions de redistribution des documents qui l'utilisent -- seul le texte original en anglais le fait. Nous espérons toutefois que cette traduction aidera les francophones à mieux comprendre la FDL GNU.

Traduction française *non officielle* de la GFDL Version 1.1 (Mars 2000)

Pour la traduction :
Version 1.0 FR (Jean-Luc Fortin, juillet 2000)
Version 1.1 FR (Christian Casteyde, mars 2001)
Version 1.1.1 FR (César Alexanian, mars 2001)
Version 1.1.2r2 FR (Christian Casteyde et César Alexanian, juin 2001)

0 - PRÉAMBULE

L'objet de cette Licence est de rendre tout manuel, livre ou autre document écrit « libre » au sens de la liberté d'utilisation, à savoir : assurer à chacun la liberté effective de le copier ou de le redistribuer, avec ou sans modifications, commercialement ou non. En outre, cette Licence garantit à l'auteur et à l'éditeur la reconnaissance de leur travail, sans qu'ils soient pour autant considérés comme responsables des modifications réalisées par des tiers.

Cette Licence est une sorte de « copyleft », ce qui signifie que les travaux dérivés du document d'origine sont eux-mêmes « libres » selon les mêmes termes. Elle complète la Licence Publique Générale GNU, qui est également une Licence copyleft, conçue pour les logiciels libres.

Nous avons conçu cette Licence pour la documentation des logiciels libres, car les logiciels libres ont besoin d'une documentation elle-même libre : un logiciel libre doit être accompagné d'un manuel garantissant les mêmes libertés que celles accordées par le logiciel lui-même. Mais cette Licence n'est pas limitée aux seuls manuels des logiciels ; elle peut être utilisée pour tous les documents écrits, sans distinction particulière relative au sujet traité ou au mode de publication. Nous recommandons l'usage de cette Licence principalement pour les travaux destinés à des fins d'enseignement ou devant servir de documents de référence.

1 - APPLICABILITÉ ET DÉFINITIONS

Cette Licence couvre tout manuel ou tout autre travail écrit contenant une notice de copyright autorisant la redistribution selon les termes de cette Licence. Le mot « **Document** » se réfère ci-après à un tel manuel ou

travail. Toute personne en est par définition concessionnaire et est référencée ci-après par le terme « **Vous** ».

Une « **Version modifiée** » du Document désigne tout travail en contenant la totalité ou seulement une portion de celui-ci, copiée mot pour mot, modifiée et/ou traduite dans une autre langue.

Une « **Section secondaire** » désigne une annexe au Document, ou toute information indiquant les rapports entre l'auteur ou l'éditeur et le sujet (ou tout autre sujet connexe) du document, sans toutefois être en rapport direct avec le sujet lui-même (par exemple, si le Document est un manuel de mathématiques, une **Section secondaire** ne traitera d'aucune notion mathématique). Cette section peut contenir des informations relatives à l'historique du Document, des sources documentaires, des dispositions légales, commerciales, philosophiques, ou des positions éthiques ou politiques susceptibles de concerner le sujet traité.

Les « **Sections inaltérables** » sont des sections secondaires considérées comme ne pouvant être modifiées et citées comme telles dans la notice légale qui place le Document sous cette Licence.

Les « **Textes de couverture** » sont les textes courts situés sur les pages de couverture avant et arrière du Document, et cités comme tels dans la mention légale de ce Document.

Le terme « **Copie transparente** » désigne une version numérique du Document représentée dans un format dont les spécifications sont publiquement disponibles et dont le contenu peut être visualisé et édité directement et immédiatement par un éditeur de texte quelconque, ou (pour les images composées de pixels) par un programme de traitement d'images quelconque, ou (pour les dessins) par un éditeur de dessins courant. Ce format doit pouvoir être accepté directement ou être convertible facilement dans des formats utilisables directement par des logiciels de formatage de texte. Une copie publiée dans un quelconque format numérique ouvert mais dont la structure a été conçue dans le but exprès de prévenir les modifications ultérieures du Document ou dans le but d'en décourager les lecteurs n'est pas considérée comme une Copie Transparente. Une copie qui n'est pas « **Transparente** » est considérée, par opposition, comme « **Opaque** ».

Le format de fichier texte codé en ASCII générique et n'utilisant pas de balises, les formats de fichiers Texinfo ou LaTeX, les formats de fichiers SGML ou XML utilisant une DTD publiquement accessible, ainsi que les formats de fichiers HTML simple et standard, écrits de telle sorte qu'ils sont modifiables sans outil spécifique, sont des exemples de formats acceptables pour la réalisation de Copies Transparentes. Les formats suivants sont opaques : PostScript, PDF, formats de fichiers propriétaires qui ne peuvent être visualisés ou édités que par des traitements de textes propriétaires, SGML et XML utilisant des DTD et/ou des outils de formatage qui ne sont pas disponibles publiquement, et du code HTML généré par une machine à l'aide d'un traitement de texte quelconque et dans le seul but de la génération d'un format de sortie.

La « **Page de titre** » désigne, pour les ouvrages imprimés, la page de titre elle-même, ainsi que les pages supplémentaires nécessaires pour fournir clairement les informations dont cette Licence impose la présence sur la page de titre. Pour les travaux n'ayant pas de Page de titre comme décrit ci-dessus, la « **Page de titre** » désigne le texte qui s'apparente le plus au titre du document et situé avant le texte principal.

2 - COPIES CONFORMES

Vous pouvez copier et distribuer le **Document** sur tout type de support, commercialement ou non, à condition que cette Licence, la notice de copyright et la notice de la Licence indiquant que cette Licence s'applique à ce **Document** soient reproduits dans toutes les copies, et que vous n'y ajoutiez aucune condition restrictive supplémentaire. Vous ne pouvez pas utiliser un quelconque moyen technique visant à empêcher ou à contrôler la lecture ou la reproduction ultérieure des copies que vous avez créées ou distribuées. Toutefois, vous pouvez solliciter une rétribution en échange des copies. Si vous distribuez une grande quantité de copies, référez-vous aux dispositions de la section 3.

Vous pouvez également prêter des copies, sous les mêmes conditions que celles suscitées, et vous pouvez afficher publiquement des copies de ce **Document**.

3 - COPIES EN NOMBRE

Si vous publiez des copies imprimées de ce **Document** à plus de 100 exemplaires et que la Licence du

Document indique la présence de **Textes de couverture**, vous devez fournir une couverture pour chaque copie, qui présente les **Textes de couverture** des première et dernière pages de couverture du **Document**. Les première et dernière pages de couverture doivent également vous identifier clairement et sans ambiguïté comme étant l'éditeur de ces copies. La première page de couverture doit comporter le titre du **Document** en mots d'importance et de visibilité égales. Vous pouvez ajouter des informations complémentaires sur les pages de couverture. Les copies du **Document** dont seule la couverture a été modifiée peuvent être considérées comme des copies conformes, à condition que le titre du **Document** soit préservé et que les conditions indiquées précédemment soient respectées.

Si les textes devant se trouver sur la couverture sont trop importants pour y tenir de manière claire, vous pouvez ne placer que les premiers sur la première page et placer les suivants sur les pages consécutives.

Si vous publiez plus de 100 **Copies opaques** du **Document**, vous devez soit fournir une **Copie transparente** pour chaque **Copie opaque**, soit préciser ou fournir avec chaque **Copie opaque** une adresse réseau publiquement accessible d'une **Copie transparente** et complète du **Document**, sans aucun ajout ou modification, et à laquelle tout le monde peut accéder en téléchargement anonyme et sans frais, selon des protocoles réseau communs et standards. Si vous choisissez cette dernière option, vous devez prendre les dispositions nécessaires, dans la limite du raisonnable, afin de garantir l'accès non restrictif à la **Copie transparente** durant une année pleine après la diffusion publique de la dernière **Copie opaque** (directement ou *via* vos revendeurs).

Nous recommandons, mais ce n'est pas obligatoire, que vous contactiez l'auteur du **Document** suffisamment tôt avant toute publication d'un grand nombre de copies, afin de lui permettre de vous donner une version à jour du **Document**.

4 - MODIFICATIONS

Vous pouvez copier et distribuer une **Version modifiée** du **Document** en respectant les conditions des sections 2 et 3 précédentes, à condition de placer cette **Version modifiée** sous la présente Licence, dans laquelle le terme « **Document** » doit être remplacé par les termes « **Version modifiée** », donnant ainsi l'autorisation de redistribuer et de modifier cette **Version modifiée** à quiconque en possède une copie. De plus, vous devez effectuer les actions suivantes dans la **Version modifiée** :

1. Utiliser sur la **Page de titre** (et sur la page de couverture éventuellement présente) un titre distinct de celui du **Document** d'origine et de toutes ses versions antérieures (qui, si elles existent, doivent être mentionnées dans la section « **Historique »** du **Document**). Vous pouvez utiliser le même titre si l'éditeur d'origine vous en a donné expressément la permission.
2. Mentionner sur la **Page de titre** en tant qu'auteurs une ou plusieurs des personnes ou entités responsables des modifications de la **Version modifiée**, avec au moins les cinq principaux auteurs du **Document** (ou tous les auteurs s'il y en a moins de cinq).
3. Préciser sur la **Page de titre** le nom de l'éditeur de la **Version modifiée**, en tant qu'éditeur du **Document**.
4. Préserver intégralement toutes les notices de copyright du **Document**.
5. Ajouter une notice de copyright adjacente aux autres notices pour vos propres modifications.
6. Inclure immédiatement après les notices de copyright une notice donnant à quiconque l'autorisation d'utiliser la **Version modifiée** selon les termes de cette Licence, sous la forme présentée dans l'annexe indiquée ci-dessous.
7. Préserver dans cette notice la liste complète des **Sections inaltérables** et les **Textes de couverture** donnés avec la notice de la Licence du Document.
8. Inclure une copie non modifiée de cette Licence.
9. Préserver la section nommée « **Historique** » et son titre, et y ajouter une nouvelle entrée décrivant le titre, l'année, les nouveaux auteurs et l'éditeur de la **Version modifiée**, tels que décrits sur la **Page de titre**, ainsi qu'un descriptif des modifications apportées depuis la précédente version.
10. Conserver l'adresse réseau éventuellement indiquée dans le **Document** permettant à quiconque

d'accéder à une **Copie transparente** du **Document**, ainsi que les adresses réseau indiquées dans le **Document** pour les versions précédentes sur lesquelles le **Document** se base. Ces liens peuvent être placés dans la section « **Historique** ». Vous pouvez ne pas conserver les liens pour un travail datant de plus de quatre ans avant la version courante ou si l'éditeur d'origine vous en accorde la permission.

11. Si une section « **Dédicaces** » ou une section « **Remerciements** » sont présentes, les informations et les appréciations concernant les contributeurs et les personnes auxquelles s'adressent ces remerciements doivent être conservées, ainsi que le titre de ces sections.
12. Conserver sans modification les **Sections inaltérables** du **Document**, ni dans leurs textes, ni dans leurs titres. Les numéros de sections ne sont pas considérés comme faisant partie du texte des sections.
13. Effacer toute section intitulée « **Approbations** ». Une telle section ne peut pas être incluse dans une **Version modifiée**.
14. Ne pas renommer une section existante sous le titre « **Approbations** » ou sous un autre titre entrant en conflit avec le titre d'une **Section inaltérable**.

Si la **Version modifiée** contient de nouvelles sections préliminaires ou de nouvelles annexes considérées comme des **Sections secondaires** et que celles-ci ne contiennent aucun élément copié à partir du Document, vous pouvez à votre convenance en désigner une ou plusieurs comme étant des **Sections inaltérables**. Pour ce faire, ajoutez leurs titres dans la liste des **Sections inaltérables** au sein de la notice de Licence de la version Modifiée. Ces titres doivent êtres distincts des titres des autres sections.

Vous pouvez ajouter une section nommée « **Approbations** » à condition que ces approbations ne concernent que les modifications ayant donné naissance à la **Version modifiée** (par exemple, comptes rendus de revue du document ou acceptation du texte par une organisation le reconnaissant comme étant la définition d'un standard).

Vous pouvez ajouter un passage comprenant jusqu'à cinq mots en première page de couverture, et jusqu'à vingt-cinq mots en dernière page de couverture, à la liste des **Textes de couverture** de la **Version modifiée**. Il n'est autorisé d'ajouter qu'un seul passage en première et en dernière pages de couverture par personne ou groupe de personnes ou organisation ayant contribué à la modification du **Document**. Si le Document comporte déjà un passage sur la même couverture, ajouté en votre nom ou au nom de l'organisation au nom de laquelle vous agissez, vous ne pouvez pas ajouter de passage supplémentaire ; mais vous pouvez remplacer un ancien passage si vous avez expressément obtenu l'autorisation de l'éditeur de celui-ci.

Cette Licence ne vous donne pas le droit d'utiliser le nom des auteurs et des éditeurs de ce **Document** à des fins publicitaires ou pour prétendre à l'approbation d'une **Version modifiée**.

5 - FUSION DE DOCUMENTS

Vous pouvez fusionner le **Document** avec d'autres documents soumis à cette Licence, suivant les spécifications de la section 4 pour les **Versions modifiées**, à condition d'inclure dans le document résultant toutes les **Sections inaltérables** des documents originaux sans modification, et de toutes les lister dans la liste des **Sections inaltérables** de la notice de Licence du document résultant de la fusion.

Le document résultant de la fusion n'a besoin que d'une seule copie de cette Licence, et les **Sections inaltérables** existant en multiples exemplaires peuvent être remplacées par une copie unique. S'il existe plusieurs **Sections inaltérables** portant le même nom mais de contenu différent, rendez unique le titre de chaque section en ajoutant, à la fin de celui-ci, entre parenthèses, le nom de l'auteur ou de l'éditeur d'origine, ou, à défaut, un numéro unique. Les mêmes modifications doivent être réalisées dans la liste des **Sections inaltérables** de la notice de Licence du document final.

Dans le document résultant de la fusion, vous devez rassembler en une seule toutes les sections « Historique » des documents d'origine. De même, vous devez rassembler les sections « Remerciements » et « Dédicaces ». Vous devez supprimer toutes les sections « Approbations ».

6 - REGROUPEMENTS DE DOCUMENTS

Vous pouvez créer un regroupement de documents comprenant le **Document** et d'autres documents soumis à cette Licence, et remplacer les copies individuelles de cette Licence des différents documents par une unique copie incluse dans le regroupement de documents, à condition de respecter pour chacun de ces documents l'ensemble des règles de cette Licence concernant les copies conformes.

Vous pouvez extraire un document d'un tel regroupement et le distribuer individuellement sous couvert de cette Licence, à condition d'y inclure une copie de cette Licence et d'en respecter l'ensemble des règles concernant les copies conformes.

7 - AGRÉGATION AVEC DES TRAVAUX INDÉPENDANTS

La compilation du **Document** ou de ses dérivés avec d'autres documents ou travaux séparés et indépendants sur un support de stockage ou sur un média de distribution quelconque ne représente pas une **Version modifiée** du **Document** tant qu'aucun copyright n'est déposé pour cette compilation. Une telle compilation est appelée « agrégat » et cette Licence ne s'applique pas aux autres travaux indépendants compilés avec le **Document** s'ils ne sont pas eux-mêmes des travaux dérivés du **Document**.

Si les exigences de la section 3 concernant les **Textes de couverture** sont applicables à ces copies du **Document**, et si le **Document** représente un volume inférieur à un quart du volume total de l'agrégat, les **Textes de couverture** du **Document** peuvent être placés sur des pages de couverture qui n'encadrent que le **Document** au sein de l'agrégat. Dans le cas contraire, ils doivent apparaître sur les pages de couverture de l'agrégat complet.

8 - TRADUCTION

La traduction est considérée comme une forme de modification, vous pouvez donc distribuer les traductions du **Document** selon les termes de la section 4. Vous devez obtenir l'autorisation spéciale des auteurs des **Sections inaltérables** pour les remplacer par des traductions, mais vous pouvez inclure les traductions des **Sections inaltérables** en plus des textes originaux. Vous pouvez inclure une traduction de cette Licence à condition d'inclure également la version originale en anglais. En cas de contradiction entre la traduction et la version originale en anglais, c'est cette dernière qui prévaut.

9 - RÉVOCATION

Vous ne pouvez pas copier, modifier, sous-licencier ou distribuer le **Document** autrement que selon les termes de cette Licence. Tout autre acte de copie, modification, sous-Licence ou distribution du **Document** est sans objet et vous prive automatiquement des droits que cette Licence vous accorde. En revanche, les personnes qui ont reçu de votre part des copies ou les droits sur le document sous couvert de cette Licence ne voient pas leurs droits révoqués tant qu'elles en respectent les principes.

10 - RÉVISIONS FUTURES DE CETTE LICENCE

La Free Software Foundation peut publier de temps en temps de nouvelles versions révisées de cette Licence. Ces nouvelles versions seront semblables à la présente version dans l'esprit, mais pourront différer sur des points particuliers en fonction de nouvelles questions ou nouveaux problèmes. Voyez <A HREF="http://www.gnu.org/copyleft/">http://www.gnu.org/copyleft/</A> pour plus de détails.

Chaque version de cette Licence est dotée d'un numéro de version distinct. Si un **Document** spécifie un numéro de version particulier de cette Licence, et porte la mention « ou toute autre version ultérieure », vous pouvez choisir de suivre les termes de la version spécifiée ou ceux de n'importe quelle version ultérieure publiée par la Free Software Foundation. Si aucun numéro de version n'est spécifié, vous pouvez choisir n'importe quelle version officielle publiée par la Free Sofware Foundation.

Comment utiliser cette Licence pour vos documents

Pour utiliser cette Licence avec un document que vous avez écrit, incorporez une copie du texte de cette Licence en anglais et placez le texte ci-dessous juste après la page de titre :

Copyright (c) **ANNÉE VOTRE NOM**

Permission vous est donnée de copier, distribuer et/ou modifier ce document selon les termes de la Licence GNU Free Documentation License, Version 1.1 ou ultérieure publiée par la Free Software Foundation ; avec les sections inaltérables suivantes :

LISTE DES TITRES DES SECTIONS INALTÉRABLES

Avec le texte de première page de couverture suivant :

TEXTE DE PREMIÈRE PAGE DE COUVERTURE

Avec le texte de dernière page de couverture suivant :

TEXTE DE DERNIÈRE PAGE DE COUVERTURE

Une copie de cette Licence est incluse dans la section appelée **GNU Free Documentation License** de ce document.

Si votre Document ne comporte pas de section inaltérable, de textes de première et dernière pages de couverture, veuillez insérer les mentions suivantes dans les sections adéquates :

- pas de section inaltérable -

- pas de texte de première page de couverture -

- pas de texte de dernière page de couverture -

Vous pouvez également fournir une traduction de la Licence GNU FDL dans votre document, mais celle-ci ne doit pas remplacer la version anglaise. La section intitulée **GNU Free Documentation License** doit contenir la version anglaise de la Licence GNU FDL, c'est la seule qui fait foi.

Si votre Document contient des exemples non triviaux de code programme, nous recommandons de distribuer ces exemples en parallèle sous Licence GNU General Public License, qui permet leur usage dans les logiciels libres.

Annexe D — Texte original de la licence GNU FDL

Version 1.2, November 2002

SOMMAIRE

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively

with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it

was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

External links

- Official GNU FDL webpage

Pour la Liberté