Richard Stallman et la révolution du logiciel libre

Une biographie autorisée

Sam Williams

Eyrolles

Sam Williams, Richard Stallman &
Christophe Masutti

Richard Stallman et la révolution du logiciel libre.

Une biographie autorisée

Framabook
le pari du livre libre

Mention légale

Historique des modifications

Œuvre originale écrite par Sam Williams.

Mars 2007

Projet de traduction en français, initié par Framasoft (http ://www.framasoft.net/), Alexis Kauffmann, puis sous la direction de Christophe Masutti (Framasoft).

Décembre 2008 - novembre 2009

Modifications majeures apportées au texte original par Richard M. Stallman. Traduction agrémentée de notes et d'un index.

Septembre 2009 - 21 janvier 2010

Éditions Eyrolles : récriture, retraduction, demandes d'ajouts, relecture : Muriel Shan Sei Fan, François-Pierre Dubos, Daniel Garance, Karine Joly, Sophie Hincelin, Pascale Sztajnbok.

Richard Stallman : relecture finale, dernières corrections.

Mise en page : Chloé Girard et David Dauvergne (La Poule ou l'Œuf).

Illustrations

Titre : *PDP-10 with KL10 processor. Stanford Artificial Intelligence Laboratory*. 1979.

Auteur : Rodney Brooks

Licence : GNU Free Documentation Licence

Source : Rodney Brooks

Titre : *Happy hacking* (couverture)

Auteur : Nadège Dauvergne

Licence : GNU Free Documentation Licence

Source : La Poule ou l'Œuf

Préface

de Richard Stallman

J'ai tenté d'apporter à cette édition revue et augmentée mes connaissances, sans pour autant sacrifier le point de vue externe et les interviews de Sam Williams. Le lecteur sera juge du succès de mon entreprise.

J'ai lu pour la première fois le texte de l'édition anglaise en 2009 lorsque j'ai été sollicité pour aider à l'adaptation française de *Free as in Freedom*. Il lui fallait plus que des modifications mineures.

Outre que des faits étaient rapportés de façon erronée, il fallait des changements plus profonds. Sam Williams n'étant pas un

programmeur, il a brouillé certaines frontières techniques et juridiques fondamentales. Ainsi ne faisait-il pas de distinction claire entre, d'une part, modifier le code d'un programme existant et, d'autre part, implémenter certaines des idées de ce code dans un nouveau programme. Par exemple, il était dit dans la première édition qu'à la fois Gosmacs et GNU Emacs avaient été développés en modifiant le programme original PDP-10 Emacs, ce qui n'était le cas ni pour l'un, ni pour l'autre. De même, le texte désignait à tort Linux comme une « version de Minix » — SCO a d'ailleurs prétendu la même chose dans leur tristement célèbre procès contre IBM, et tant Torvalds que Tanenbaum ont dû réfuter l'affirmation.

La première édition dramatisait exagérément de nombreux événements en y projetant des émotions. Par exemple, il y était dit qu'en 1992 je « fuyais Linux » avant de faire en 1993 « une spectaculaire volte-face » en décidant de financer Debian GNU/Linux. Tant mon intérêt en 1993 que mon désintérêt en 1992 n'étaient que des actions au service d'un même but : achever le système GNU. Il en allait de même du lancement du noyau GNU Hurd en 1990.

Pour toutes ces raisons, de nombreuses affirmations dans l'édition originale étaient erronées ou incohérentes. Il était nécessaire de les rectifier, mais le faire honnêtement était délicat sans une récriture totale qui n'était pas désirable. L'utilisation de notes explicites rectificatives a été suggérée, mais l'ampleur des modifications l'aurait rendue impossible dans la plupart des chapitres. Certaines erreurs étaient trop profondément ancrées dans le texte pour être corrigées par des notes. Quant aux autres, des notes au fil ou en bas de page auraient alourdi considérablement et rendu l'ensemble illisible — les notes de bas de page étant évidemment ignorées par un lecteur lassé. C'est pourquoi j'ai directement corrigé le texte.

Je précise que je n'ai pas vérifié tous les faits et citations inconnus de moi ; je les ai laissés pour la plupart en m'en remettant à l'autorité de Sam Williams.

Par ailleurs, la version de Sam Williams contenait de nombreuses citations critiques envers moi. Je les ai laissées intactes, me contentant d'y répondre lorsque cela était approprié. Aucune citation n'a été supprimée, excepté certaines au chapitre 11, qui traitaient d'*open source* et n'avaient pas de rapport avec ma vie ou mon action. De même, j'ai conservé certaines interprétations personnelles de Sam Williams qui étaient critiques envers moi, lorsqu'elles ne présentaient pas de malentendus factuels ni techniques, mais j'ai librement corrigé des affirmations fausses concernant mon œuvre, mes pensées ou mes sentiments. J'ai préservé toutes ses impressions personnelles lorsqu'elles étaient présentées comme telles, et « je » désigne toujours Sam Williams, sauf dans les notes précédées de mon nom, ou dans celles en italique qui ponctuent certains chapitres et l'épilogue sous forme d'encarts.

Dans cette édition, le système complet qui combine GNU et Linux est appelé « GNU/Linux », tandis que « Linux » seul désigne toujours le noyau de Torvalds, excepté dans les citations où l'autre acception est signalée par *[sic]*. (Pour comprendre en quoi il est erroné et abusif de désigner l'ensemble par « Linux », voir http://www.gnu.org/gnu/gnu-linux-faq.html).

Je tiens à remercier John Sullivan pour ses nombreuses critiques et suggestions utiles.

Richard Stallman

Avant-propos à l'édition française

de Sam Williams

Sept ans après avoir mis la dernière touche à la première édition de *Free as in Freedom*, voilà que je me trouve dans la même situation en écrivant la préface à cette première version française du livre.

Avec deux dates butoirs qui me guettent depuis le calendrier et une masse impressionnante de brouillons qui emplissent mon disque dur, c'est un véritable combat que d'arriver à aligner des phrases correctes en anglais, et plus encore en français.

Je n'écris pas ceci pour attirer la sympathie ou implorer un quelconque pardon, mais pour souligner mon extrême gratitude envers l'association *Framasoft* qui a pris sur elle de traduire cet

ouvrage (y compris cette préface) et permet ainsi à d'autres de pouvoir contribuer à son amélioration. Si l'expérience de l'écriture d'un livre m'a bien appris une chose, c'est que l'aide généreuse de centaines de collaborateurs atténue largement la faiblesse d'un auteur.

J'en avais eu l'intuition avant même de commencer l'écriture de *Free as in Freedom*. Ayant étudié l'histoire de l'*open source* et des logiciels libres avant de me mettre à la tâche, j'avais déjà engrangé un grand nombre de critiques constructives au moment où la version papier du livre commençait à faire son apparition dans les librairies. De plus, je savais que Richard Stallman allait relever chaque inexactitude dès qu'il aurait en main sa première copie. L'une des principales raisons pour laquelle j'ai voulu mettre ce livre sous licence GNU FDL[1] était mon envie de pouvoir y revenir pour le corriger, voire l'améliorer, sans pour autant devoir le faire par l'intermédiaire d'une maison d'édition.

J'avais bien sûr d'autres raisons de choisir la GNU FDL. Comme je le dis dans l'épilogue intitulé « Écrasante solitude », l'ouvrage a été conçu dès le départ comme un livre électronique. Or, avant même que nous sachions à quoi il allait être utilisé, le matériel issu des nombreuses entrevues avec Richard M. Stallman devait selon moi être exploité en suivant un pré-requis éthique qui revenait à trouver une licence qui donnerait aux lecteurs des libertés au moins égales à celles dont jouissent déjà les lecteurs traditionnels — à savoir la liberté de partager et de copier, la liberté de lire dans l'environnement de son choix, la liberté d'en revendre des copies. Après avoir épuisé les premières tentatives de bricolage d'une telle licence, j'ai choisi la GNU FDL pour la bonne raison que Stallman avait participé à son écriture, et que je savais que

1. Voir le site *gnu.org* (http://www.gnu.org/licenses/licenses.fr.html) — NdT.

l'ensemble de la communauté des *hackers* aurait approuvé cette décision.

En fait, il me semble me souvenir que Stallman lui-même se demandait si le choix la GNU FDL n'était pas excessif sachant que l'objectif principal de cette licence est d'encourager la collaboration pour l'amélioration des manuels utilisateurs des logiciels, un sous-genre d'œuvre non romanesque où l'utilité de l'information et la réciprocité sociale prennent le pas sur la reconnaissance de « l'auteur unique ».

Malgré tout, Tim O'Reilly, l'homme dont la société a finalement décidé de publier le livre, n'a même pas tiqué sur la demande de cette licence (en tout cas, pas d'après mon agent, Henning Guttman, chargé des négociations). Après avoir publié quelques livres sous GNU FDL, O'Reilly s'y connaissait mieux que Richard et moi-même sur la meilleure façon d'en assurer la commercialisation. Une fois de plus, un bon écrivain travaille rarement seul.

Nous étions en 2001-2002, alors que Wikipédia commençait à voir le jour. Toutefois, nous étions encore dans la période qui précédait sa soudaine apparition sur le Web, telle le volcan du Paricutín. D'après moi, ce projet en ligne a fait à ce jour la meilleure démonstration de l'étonnante puissance de la GNU FDL. J'aimerais pouvoir dire que je partageais à l'époque la même vision que le créateur de Wikipédia, Jimmy Wales, qui considérait la GNU FDL comme un catalyseur de liberté de parole sur Internet. Mais en vérité, comme de nombreux autres observateurs de l'époque, je dédaignais Wikipédia, n'y voyant qu'un futur gigantesque fiasco. Bien qu'ayant trouvé dans le modèle de publication wiki un étonnant prolongement technique de l'éthique « hacker », j'étais dubitatif quant à son extensibilité et sa cohérence, sans parler des risques de plagiat et de diffamation. Tout ceci m'a finalement incité à me détourner du modèle de Wikipédia lors de la création

de mon site personnel, *faifzilla.org*, qui allait me servir à gérer l'évolution des textes constituant *Free as in Freedom*.

En fait, au moment de choisir la GNU FDL, c'est l'exemple de Linus Torvalds qui me servit de modèle. Tout en énonçant une série de raisons égoïstes et d'autres pas si égoïstes que cela, ce hacker finlandais a choisi de mettre sa création, Linux, sous la protection juridique de la GNU GPL. Un choix paradoxal, peut-être, mais que j'estime être en harmonie avec les grands thèmes de ce livre.

Égoïstement, je voulais que me revienne tout bénéfice du livre en termes de crédibilité ou de renommée. Et pas si égoïstement que cela, je voulais laisser la liberté à d'autres, y compris Stallman, de créer leurs propres versions parallèles, même si nos points de vue pouvaient diverger. Enfin, je souhaitais que ce travail soit adaptable. Dans l'épilogue, je cite le projet Xanadu de Ted Nelson comme un élément capital à cet égard. Je croyais alors et continue de croire à la nature talmudique d'Internet, c'est-à-dire que sa capacité non seulement à rapporter la naissance des idées mais aussi à garder les traces de leurs évolutions intellectuelles, est sa marque de fabrique. Je n'ai pas beaucoup entendu parler de Xanadu depuis que j'ai écrit ces lignes, mais je pense que les phénomènes « Web 2.0 » et « blog » sont des exemples suffisants pour cette partie du livre.

Malheureusement, je ne suis pas Linus Torvalds. Tout comme le sujet de cette biographie, ma personnalité est solitaire, quoi que je fasse pour lutter contre cela. Je n'ai pas de difficulté à travailler avec d'autres personnes, mais mon fonctionnement de base est de tout regrouper au même endroit et de me casser la tête sur un problème jusqu'à ce que ce problème (ou ma tête) s'avoue vaincu(e). Il m'est aussi arrivé d'être particulièrement conservateur quant à l'adoption de nouvelles technologies. En gérant faifzilla.org, j'étais devenu quelque peu expert en feuilles de style, en systèmes de gestion de versions et en logiciels de création de pages web. Mais en fin

de journée, ma réponse instinctive à un rapport d'erreur était de faire les modifications directement dans le code source HTML puis de mettre à jour une demi-douzaine de fois la page afin d'éliminer les nouvelles erreurs que j'avais peut-être moi-même induites. Ajoutez à cela la principale pathologie de tout journaliste — l'incapacité totale à finir quelque chose sans une date butoir ferme ou une menace de non-paiement — et vous êtes sûr qu'il va y avoir de la bagarre.

Même si je reste fier d'honorer mon engagement à intégrer les corrections et des modifications de mes lecteurs, l'élaboration d'une version améliorée 2.0 de *Free as in Freedom*, avec un chapitre sur les derniers rebondissements de la saga Stallman, et qui aurait tenu compte de ses propres corrections et commentaires, a été stoppée vers mi-2003. Tout comme un charpentier qui laisserait sa maison tomber en ruine, j'ai laissé le livre passer du statut d'investissement à long terme à celui d'un projet de vacances toujours remis à plus tard.

Arrive alors Framasoft, une association qui, en plus de publier le livre, a accepté de mettre en place et d'héberger une version wiki des textes (http://framabook.org/faif). Christophe Masutti, le membre du groupe qui a pris le temps de me retrouver et d'attendre patiemment ma réponse, m'a de plus informé que Framasoft avait été jusqu'à travailler avec Richard Stallman pour inclure ses propres changements au texte original du livre, effort auquel je souscris complètement même si cela éclipse mes vains efforts à faire exactement la même chose.

Ma suspicion initiale à propos de Wikipédia ayant depuis longtemps fait place à une jalousie aiguë, je me sens un peu comme le personnage de George Bailey dans la scène finale de *It's a Wonderful Life*. Je ne sais pas trop ce que j'ai fait pour mériter tant de générosité. Peut-être qu'ils étaient vrais, après tout, ces discours de la fin des années 1990 qui parlaient d'économie de partage, de

karma en ligne et d'informations qui souhaitaient juste être vraiment libres. Si l'on revient sur la difficile dernière décennie, il y a eu des moments ou j'ai méjugé de tout cela avec la désespérante naïveté d'un gamin.

Quoi qu'il en soit, assez parlé de moi. Le temps est venu de focaliser notre attention sur le véritable sujet, la vie et l'œuvre de Richard M. Stallman, hacker émérite. Je voudrais terminer cette préface de la version française du livre de la même façon dont j'ai fini la préface de la version anglaise : avec une invitation ouverte à toutes et à tous, contributeurs potentiels. Si vous découvrez une erreur ou un passage qui nécessite un remaniement, n'hésitez pas à corriger et à ajouter votre nom à la longue liste des co-auteurs. La seule différence, cette fois-ci, c'est que vous n'aurez pas à faire les modifications en passant par moi.

Sam Williams
Staten Island, USA

Préambule

par Alexis Kauffmann et Christophe Masutti de Framasoft

« Chaque génération a son philosophe, écrivain ou artiste qui saisit et incarne l'imaginaire du moment. Il arrive que ces philosophes soient reconnus de leur vivant, mais le plus souvent il faut attendre que la patine du temps fasse son effet. Que cette reconnaissance soit immédiate ou différée, une époque est marquée par ces hommes qui expriment leurs idéaux, dans les murmures d'un poème ou dans le grondement d'un mouvement politique. Notre génération a un philosophe. Ce n'est ni un artiste ni un écrivain. C'est un informaticien » — Lawrence Lessig[1].

1. L. Lessig, *Introduction to Free Software, Free Society : The Selected Essays of Richard M. Stallman.*

Qu'on l'apprécie, qu'on l'admire ou qu'il nous irrite, Richard Stallman est un personnage fascinant et incontournable de l'histoire de l'informatique en général et du logiciel libre en particulier. Une histoire toujours en marche dont l'influence dépasse désormais de beaucoup le seul champ technologique. Et pourtant, même si certains commencent à désormais bien connaître le conférencier parcourant inlassablement la planète pour prêcher la bonne parole, nombreux sont ceux qui ignorent tout ou partie des détails de sa vie qui l'ont justement amené à en arriver là.

Lorsqu'en 2002 est sorti le livre *Free as in Freedom : Richard Stallman's Crusade for Free Software* de Sam Williams[2], nous fûmes ravis de constater que pour la première fois un ouvrage lui avait été consacré. Nous fûmes également ravis du choix de sa licence, la *GNU Free Documentation License*[3], autorisant les modifications dont les plus naturelles sont les traductions.

Ce livre est le résultat d'un travail collectif initié en mars 2007 par le réseau Framasoft. Le projet s'était donné pour objectif de coopérer à la traduction de l'ensemble de l'ouvrage et de le publier dans la collection de livres libres Framabook. À mesure de l'avancement du projet, il nous a semblé opportun de contacter Richard Stallman afin de l'informer de notre intention de publier sa biographie. C'est alors que le projet prit une tournure et une ampleur inattendues : la contribution de Richard Stallman apporta des modifications si profondes du texte d'origine qu'une nouvelle version complète vit le jour : *Free as in Freedom (2.0)*[4].

2. Le texte original du livre (2002), sous licence GNU FDL, est accessible chez l'éditeur O'Reilly : http://oreilly.com/openbook/freedom/

3. Voir annexe B de cet ouvrage.

4. Le texte en anglais est publié par la Free Software Foundation. Le contenu du présent ouvrage fut travaillé en étroite collaboration avec Richard Stallman. Au fur et à mesure, les modifications de ce dernier furent implémentées dans la version française.

C'est ce nouveau texte que nous vous proposons en français. Ajoutons que la présente traduction se trouve bonifiée par une mise à jour des liens, un index, une bibliographie enrichie, des encarts, ainsi que de nombreuses notes afin de rendre le texte plus accessible.

Nous tenons enfin à témoigner notre reconnaissance envers Richard Stallman lui-même. Si le mouvement du logiciel libre est aujourd'hui une source d'espoir et de progrès, il le doit beaucoup à ce personnage hors normes.

Nous vous souhaitons une agréable lecture.

Remerciements

Nous tenons vivement à remercier l'équipe des éditions Eyrolles, Muriel Shan Sei Fan, François-Pierre Dubos et Daniel Garance, pour avoir osé s'embarquer avec nous sur ce projet original, qui plus est sous licence libre. Ils ont récupéré une première version de la traduction, plus que perfectible, et ont travaillé d'arrache-pied pour en faire l'ouvrage de qualité que vous avez aujourd'hui sous les yeux.

Qu'il nous soit également permis de rendre hommage aux personnes ayant collaboré en ligne aux premières versions de la traduction, nombreuses fourmis bénévoles parmi lesquelles Morgan Berger, Johann Bulteau, François Dusolle, Claude Le Paih, Guillaume Pasquet, Yonnel Poivre-le-Lohé, Léo Studer.

Enfin, nous souhaitons également mentionner Chloé Girard et David Dauvergne pour avoir mis à notre disposition leur outil libre La Poule ou l'Œuf[5] tout au long du processus d'édition du livre.

Alexis Kauffmann et Christophe Masutti

5. La Poule ou l'Œuf est une chaîne éditoriale web dédiée document long (http ://www.pouleouoeuf.org).

Note de l'éditeur [6]

Nous sommes fiers d'accueillir cet ouvrage de référence en partenariat avec le projet Framasoft, qui en est à l'initiative. Nous avons tenté, sur ce projet inhabituel mais ô combien gratifiant, de faire au mieux de ce que nous savons faire : rendre justice aux auteurs et aux contenus.

C'est avec un enthousiasme mêlé d'effroi que nous nous sommes lancés dans l'aventure, après avoir entamé des discussions avec Alexis Kauffmann et Christophe Masutti, au printemps 2009. L'ouvrage a représenté une somme de travail inattendue ; nul doute, hélas, qu'y subsistent des imperfections et inélégances qu'il reviendra à d'autres de corriger.

La libre redistribution d'œuvres n'est pas une première aux éditions Eyrolles. C'est pourtant la première fois qu'est mis d'emblée sous licence libre un ouvrage qui a représenté tant de travail.

Un bref historique chronologique pour que soient remerciées celles et ceux qui ont participé à sa récriture, retraduction, relecture et finalisation :

- François-Pierre Dubos, pour une première récriture,
- Daniel Garance pour une première passe sur le (contre-)sens,
- Karine Joly pour ses révisions intensives, et Fabrice Le Fessant pour ses précisions,
- Sophie Hincelin et Pascale Sztajnbok pour leurs relectures méticuleuses,
- Richard Stallman pour ses réponses parfois pleines d'humour et sa relecture finale,
- David Dauvergne pour son efficacité aux dernières étapes.

Muriel Shan Sei Fan

6. L'éditeur de l'ouvrage dans sa première version publiée le 21 janvier 2010 est Eyrolles.

Table des matières

Chapitre

1

Une histoire d'imprimante

« Je crains les Grecs. Même lorsqu'ils offrent des présents. »
Virgile, *L'Énéide*.

La nouvelle imprimante était encore bloquée.

Richard M. Stallman, vingt-sept ans, programmeur au laboratoire d'intelligence artificielle (AI Lab) du Massachussetts Institute of Technology (MIT), le constata à ses dépens. Une heure après l'envoi d'un fichier d'une cinquantaine de pages à l'imprimante laser du bureau, il devait interrompre une séance productive de travail pour aller récupérer ses documents. À l'arrivée, il ne trouvait dans le bac que quatre pages, qui ne lui appartenaient pas.

Son travail d'impression, ainsi que celui, inachevé, d'un autre utilisateur, étaient coincés quelque part dans les mailles électriques du réseau informatique du laboratoire.

Être tributaire du bon vouloir d'une machine fait partie des risques du métier de programmeur. Stallman devait donc prendre son mal en patience… À cette différence de taille près qu'il lui fallait rester planté devant la machine comme un valet au chevet de son maître. Ce n'était pas la première fois qu'il se voyait réduit à regarder les pages sortir une à une.

Consacrant le plus clair de son temps à améliorer l'efficacité des appareils et des logiciels qui les contrôlent, il ressentit tout naturellement le besoin d'ouvrir la bête, de regarder dans ses entrailles et de mettre à jour la faille. Malheureusement, ses talents de programmeur ne s'étendaient pas au monde de l'ingénierie mécanique. Alors que les documents sortaient fraîchement imprimés de la machine, il réfléchissait à un moyen de remédier aux problèmes de bourrage papier.

Combien de temps s'était-il écoulé depuis la réception enthousiaste 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 photocopieur rapide Xerox. Au lieu de faire de simples photocopies, elle transformait les données issues du réseau en documents d'aspect professionnel. Conçu par les ingénieurs du célèbre centre de recherches de Xerox à Palo Alto, le prototype annonçait déjà la révolution de l'impression bureautique qui allait marquer l'industrie informatique à la fin des années 1980.

Poussés par le besoin instinctif de jouer avec l'équipement dernier cri, les programmeurs du AI Lab avaient rapidement intégré la nouvelle machine au sein de l'infrastructure sophistiquée du laboratoire. Les résultats avaient immédiatement plu. Comparée à

la vieille imprimante, celle-ci était rapide. Les pages défilaient au rythme d'une par seconde : un travail d'impression de vingt minutes n'en durait plus que deux. Et l'ensemble était plus précis : les cercles ressemblaient à des cercles, pas à des ovales ; les lignes droites étaient véritablement droites, fort éloignées des anciennes sinusoïdes de faible amplitude.

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

C'est à l'usage que les défauts de la machine firent surface, en particulier une certaine propension aux bourrages papier. Les programmeurs, ingénieurs dans l'âme, devaient rapidement en comprendre la raison : comme tout photocopieur, la machine réclamait une surveillance humaine. Stipulant qu'un opérateur humain serait toujours disponible pour réparer ces incidents s'ils se produisaient, les ingénieurs de Xerox avaient investi temps et énergie à résoudre d'autres problèmes. En théorie, la diligence des utilisateurs faisait partie intégrante du système.

La transformation d'un photocopieur en imprimante avait subtilement mais profondément changé le rapport entre l'utilisateur et la machine. Autrefois dépendant d'un seul opérateur humain, le bon fonctionnement de l'appareil relevait désormais de tout un ensemble d'opérateurs en réseau. Ce n'était plus un seul utilisateur posté à proximité qui envoyait sa commande d'impression, mais un utilisateur situé à l'autre extrémité du réseau. À lui de se déplacer pour constater le peu de pages finalement imprimées.

Stallman n'était bien sûr pas le seul locataire du AI Lab à connaître le problème, mais c'est lui qui trouva un remède. Des années plus tôt, il avait résolu un problème similaire en modifiant le logiciel qui pilotait l'ancienne imprimante depuis un petit PDP-11, ainsi que le système ITS (*Incompatible Timesharing System*)

qui tournait sur le PDP-10 [1] — l'ordinateur central du laboratoire. Stallman ne pouvait rien aux bourrages mécaniques, mais il put programmer un morceau de code sur le PDP-11, vérifiant périodiquement l'imprimante et envoyant les rapports de bourrage à l'ordinateur central. Sur ce dernier, il ajouta aussi un programme pour qu'en cas de blocage, toute personne en attente d'un tirage soit informée. Le message de notification était simple et tenait en quelques mots : « L'imprimante est bloquée, merci de vérifier. » Les utilisateurs pressés d'obtenir leur impression étaient susceptibles d'accourir rapidement.

La solution de Stallman contournait certes le problème, mais avec élégance. Elle ne résolvait pas la question mécanique du bourrage, mais offrait un retour indispensable à l'utilisateur dans le dialogue avec la machine. Ces quelques lignes de code épargnaient chaque semaine aux employés du AI Lab plusieurs allers-et-retours à l'imprimante. En termes de programmation, cette solution tirait parti de l'interconnexion des opérateurs. Stallman se rappelle la logique du procédé : « La personne qui recevait ce message ignorait si elle était seule ou pas à l'avoir reçu. Elle n'avait d'autre choix que de se rendre jusqu'à l'imprimante. En quelques minutes seulement, deux ou trois personnes arrivaient ; l'une d'entre elles au moins savait résoudre le problème. »

De tels stratagèmes constituaient l'une des marques de fabrique du laboratoire et de sa population résidente de programmeurs. En fait, les meilleurs d'entre eux dédaignaient ce dernier terme, lui préférant celui, plus argotique, de *hacker* (bidouilleur). Un titre

1. Les ordinateurs de la gamme PDP (Programmable Data Processor) furent commercialisés dès le début des années 1960 par Digital Equipment Corporation (DEC), le PDP-8 marquant en 1965 une étape importante dans la miniaturisation et dans le rapport qualité/prix. On peut noter que ce fut sur un PDP-1 que le jeu SpaceWar fut créé en 1962. — Voir au chapitre 4 l'explication de l'écriture de l'ITS en réponse au système CTSS (Compatible Time Sharing System).

couvrant une foule d'activités — tout, de la plaisanterie créative à l'optimisation de programmes et de systèmes existants. Une appellation au parfum légèrement suranné qui évoquait aussi le bon vieux système D à l'américaine [2].

Des sociétés comme Xerox avaient pour politique d'offrir machines et logiciels sur les lieux fréquentés par des hackers. Si ces derniers utilisaient (et souvent amélioraient) le logiciel, ils pouvaient par la suite travailler pour ces compagnies. Des années 1960 au milieu des années 1970, celles-ci redistribuaient même parfois directement à leurs clients des programmes élaborés par des hackers.

Face aux bourrages fréquents de l'imprimante, Stallman pensait recourir au même vieux remède — ou *hack*. Or, en cherchant le logiciel pilote de l'imprimante Xerox, il fit une découverte troublante : rien de tel n'était présent, du moins aucun code intelligible ni pour lui-même ni pour d'autres programmeurs.

Jusqu'à présent, la plupart des sociétés avaient la courtoisie de publier le code source de leurs logiciels sous la forme de fichiers texte lisibles, qui tenaient lieu de documentation détaillant chaque commande. Or cette fois, Xerox n'avait fourni les fichiers du logiciel que sous une forme binaire (compilée). Si les programmeurs tentaient de l'ouvrir, ils ne pouvaient voir qu'une incompréhensible suite sans fin de 0 et de 1.

Ils auraient certes pu faire appel à des programmes appelés « désassembleurs », capables de convertir ces suites de 0 et de 1 en instructions machine de bas niveau. Cependant, se représenter ce que ces instructions sont censées *faire* est un travail long et

2. Pour un hacker (cf. annexe A), écrire un logiciel fonctionnel n'est qu'un début. Il faut ensuite afficher son ingéniosité et impressionner ses confrères en relevant un autre défi : faire briller le programme par sa rapidité, sa puissance et son élégance.

fastidieux, plus connu sous le terme de *rétro-ingénierie*. Un travail qui, pour le code de l'imprimante Xerox, aurait pris au moins le temps perdu en bourrages papier sur cinq années. Stallman, qui n'était pas encore assez désespéré pour s'y coller, mit le problème de côté.

La politique de rétention de Xerox contrastait vivement avec le mode coopératif en cours au sein de la communauté des hackers. Ainsi, pour développer le pilote de l'ancienne imprimante ainsi que celui permettant l'affichage des terminaux depuis deux PDP-11 distincts, le AI Lab avait eu besoin d'un assembleur croisé, compilant les codes du PDP-11 sur le PDP-10 principal.

Les hackers du laboratoire auraient pu écrire un tel programme. Stallman en avait cependant déniché un similaire auprès du département de sciences informatiques de son université, Harvard. Ce programme avait bien été conçu pour tourner sur un PDP-10, mais avec un autre système d'exploitation. Stallman ne sut jamais qui en était l'auteur — aucun nom n'étant mentionné. Il en rapporta malgré tout une copie au laboratoire, et l'adapta au système ITS en place. Le AI Lab acquit ainsi sans peine une composante de son infrastructure logicielle. Mieux, Stallman y ajouta des fonctions puissantes. « Nous l'avons quand même utilisé plusieurs années », indique-t-il.

Pour un programmeur des années 1970, ces emprunts de code étaient aussi anodins que la visite d'un voisin venu emprunter un peu de sucre ou un appareil ménager. À ceci près qu'en effectuant une copie du logiciel pour le AI Lab, Stallman ne privait personne de la possibilité d'utiliser le programme. Au contraire, tous étaient même invités à utiliser à leur tour les fonctions nouvellement intégrées. Ainsi Stallman se rappelle qu'un programmeur de chez Bolt, Beranek & Newman — une société d'ingénierie privée — réutilisa le programme sur un système nommé Twenex, y ajoutant des fonctions que Stallman put à son tour intégrer dans l'archive

du code source du AI Lab. Ils décidèrent même de maintenir une version commune dont le code s'exécutait tant sur Twenex que sur le système ITS.

« Un programme évoluait comme une ville », dit Stallman, évoquant l'infrastructure logicielle du AI Lab. « Certains quartiers étaient remplacés, reconstruits ; de nouveaux éléments étaient ajoutés. Mais il était toujours possible d'en regarder un bout et de dire : 'Bon, d'après le style, cette partie a été écrite dans les années 1960 et cette autre au milieu des années 1970'. »

Ce système simple de capitalisation intellectuelle mis en place par les hackers au AI Lab donna naissance à bien des programmes robustes. Même si ceux qui baignaient dans cette culture ne se disaient pas « hackers », la plupart convenaient du fait qu'un programme — ou l'un de ses correctifs — assez bon pour résoudre un problème pouvait servir à d'autres. Y avait-il une quelconque raison de ne pas le partager, ne serait-ce que par le plus élémentaire altruisme ?

Cet esprit de coopération fut peu à peu sapé par le secret commercial et l'appât du gain, ce qui conduisit parfois à des situations bancales, entre partage et cloisonnement. Ainsi les chercheurs de l'université de Californie à Berkeley avaient-ils élaboré un puissant système d'exploitation nommé BSD (*Berkeley Software Distribution*) à partir du système Unix qu'ils avaient obtenu d'AT&T. L'université distribuait BSD pour le simple coût de la copie sur bande, mais uniquement aux écoles justifiant de 50 000 dollars de licence chez AT&T. Les hackers de Berkeley s'accommodaient de cette contrainte tant qu'AT&T les laissait faire, sans percevoir alors de conflit entre les deux pratiques.

⁂

Stallman fut donc ennuyé que Xerox ne fournisse pas les fichiers du code source, sans pour autant y trouver matière à se mettre en colère. Il ne prit même pas la peine de contacter Xerox. « Ils nous avaient déjà donné l'imprimante laser, dit-il. Je ne pouvais pas prétendre qu'ils nous devaient quoi que ce soit. De plus, j'avais bien conscience que le fait de ne pas livrer le code source était une décision calculée, et qu'il était inutile d'essayer de leur faire changer d'avis. »

Un jour, une nouvelle se mit à circuler : un chercheur du département informatique de l'université de Carnegie Mellon possédait une copie du code source de l'imprimante laser.

Hélas, l'évocation de l'université de Carnegie Mellon n'augurait rien de bon. En 1979, le doctorant Brian Reid y avait provoqué de vifs remous en refusant de partager avec ses confrères son programme de formatage de texte nommé Scribe. Ce logiciel était le premier à interpréter des commandes de balisage vraiment sémantiques (« mettre ce mot en évidence » ou « ce paragraphe est une citation »), au lieu de commandes de pure mise en forme de bas niveau (« mettre ce mot en italique » ou « réduire les marges de ce paragraphe »). Plutôt que d'en faire profiter la communauté, Reid vendit *Scribe* à une société informatique de la région de Pittsburgh appelée Unilogic et déclara qu'il avait cherché à la fin de ses études à « confier » le programme à une équipe de développeurs soucieux de le conserver hors du domaine public — quoique l'on puisse se demander en quoi une telle perspective était si indésirable.

En guise de bonus commercial, Reid convint également d'intégrer un ensemble de fonctions programmées dans le temps : des « bombes à retardement », dans le langage des programmeurs, désactivant au bout de quatre-vingt-dix jours les versions du programme copiées gratuitement. Pour empêcher la désactivation du logiciel, les utilisateurs devaient payer une somme à la société informatique, laquelle fournissait alors un patch pour désamorcer « l'anti-fonction » à retardement.

Pour Stallman, c'était là une trahison pure et simple de l'éthique du programmeur. Au lieu d'honorer l'idéal de partage entre pairs, Reid initiait une pratique contraire : celle, pour les entreprises, de forcer les programmeurs à payer l'accès à l'information. Stallman, qui n'utilisait guère Scribe, n'en fit pas grand cas sur le moment. Unilogic en avait donné un exemplaire gratuit au AI Lab sans en retirer la « bombe à retardement », ni en faire mention. Le programme avait donc fonctionné un moment, puis s'était bloqué du jour au lendemain. Un autre hacker, Howard Cannon, passa des heures à le déboguer avant de trouver la « bombe » et de la supprimer en corrigeant le logiciel. Révolté, il ne manqua pas de se plaindre auprès de ses collègues de la manière dont Unilogic lui avait fait perdre son temps avec une erreur intentionnelle.

Quelques mois après, c'est avec l'épisode de Scribe en tête que Stallman, à l'occasion d'une visite professionnelle au campus de Carnegie Mellon, rendit visite à la personne censée détenir le code source du pilote de l'imprimante. Par chance, l'homme était à son bureau. Comme toute conversation entre ingénieurs, celle-ci fut cordiale mais directe. Après s'être brièvement présenté comme venant du MIT, Stallman demanda une copie du code source du pilote de l'imprimante afin de le modifier. À sa grande déception, le chercheur refusa. « Il m'a expliqué qu'il s'était engagé à ne pas en donner de copie », précise Stallman.

La mémoire humaine est étrange. Vingt ans après les faits, l'enregistrement mental de Stallman comporte quelques blancs. Non seulement ne se souvient-il ni des raisons motivant ce voyage, ni de la période de l'année, mais encore a-t-il oublié le nom de son interlocuteur. Selon Reid, la personne la plus susceptible d'avoir reçu Stallman est Robert F. Sproull, ancien chercheur au Xerox PARC et actuel directeur de Sun Laboratories, une division de recherche et développement du conglomérat informatique Sun Microsystems. Dans les années 1970, ce chercheur avait été au Xerox

PARC le principal développeur du logiciel en question. Au début des années 1980, il avait intégré le département de recherche de Carnegie Mellon pour y poursuivre ses travaux sur les imprimantes laser, entre autres projets.

Toutefois, interrogé directement par courriel, Sproull ne s'en rappelle rien. « Je ne peux commenter les faits », écrit-il en retour. « Je n'ai aucun souvenir de cet incident. »

« Le code source 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 Brian Reid. Peut-être y eut-il un malentendu, puisque Stallman ne désirait que la version ancienne du code source, utilisée au MIT depuis un certain temps — et non celle développée récemment ? La conversation fut si brève que la question de la version ne fut pas abordée.

En public, Stallman fait souvent référence à cet incident. Il souhaite faire comprendre que le refus du chercheur de partager son code source était directement lié à la signature de l'accord de non-divulgation. Xerox consentait à lui donner un accès au code source, contre l'assurance de sa discrétion.

⁂

Aujourd'hui fort répandue dans l'industrie logicielle, cette clause de confidentialité (*non-disclosure agreement* — NDA) n'en était alors qu'à ses balbutiements. Elle résultait d'une réflexion de la part de Xerox : la valeur commerciale de l'imprimante résidait aussi dans les informations nécessaires à son 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 cependant, cette clause de non-divulgation avait une tout autre signification. Elle signifiait le refus, de la part d'un chercheur de Carnegie Mellon, de prendre part à une société où tout programme était considéré comme une ressource collective. Tel le paysan qui voit le ruisseau irriguant ses champs depuis des siècles se tarir brutalement, Stallman était remonté jusqu'à la source. Il n'y avait trouvé qu'un barrage hydroélectrique flambant neuf, orné d'un beau logo Xerox.

Il ne prit pas conscience de suite de tout ce que cela impliquait, et qu'ainsi tout un système pervers allait se mettre en place. Dans un premier temps, il ne vit au refus qu'un caractère personnel. « J'étais tellement en colère que je ne pouvais pas l'exprimer. J'ai fait demi-tour, et suis sorti sans un mot, se souvient Stallman. J'ai même peut-être claqué la porte, qui sait ? Je ne me rappelle qu'une chose : je voulais sortir de là. En venant, pas un instant je n'avais imaginé que ce chercheur pourrait me refuser son aide, et je ne m'y étais pas préparé. Sa réponse m'a laissé sans voix, déçu et furieux. »

Vingt ans après les faits, la colère est toujours là, et Stallman présente cet événement comme l'un des plus décisifs parmi d'autres - dans sa réflexion sur les questions d'éthique autour du logiciel libre. Les mois suivants, la série d'événements impliquant la communauté des hackers du AI Lab aurait pourtant dû, en comparaison, renvoyer au rang de simple détail ces trente secondes de tension dans un bureau obscur de Carnegie Mellon. Pourtant c'est cette rencontre que Stallman invoque pour expliquer comment lui, hacker isolé, instinctivement méfiant vis-à-vis des autorités centralisées, est devenu un activiste à la tête d'une croisade informatique se réclamant des traditionnels principes de liberté, d'égalité et de fraternité.

« C'était la première fois que j'étais confronté à une clause de confidentialité ; j'ai tout de suite compris que ces clauses font des

victimes, déclare-t-il. En l'occurrence, j'en étais la victime ; [mon labo et moi] nous en étions les victimes. » Stallman poursuit : « S'il avait refusé sa collaboration pour des raisons personnelles, j'en serais resté là. Je l'aurais sans doute considéré comme un imbécile, mais sans plus. Ce qui rendait l'enjeu important était le caractère systématique et impersonnel de son refus, le fait qu'il s'était engagé d'avance à ne coopérer ni avec moi ni avec aucune autre personne. C'est cela qui a rendu l'enjeu important. »

Si la rencontre de Carnegie Mellon n'était pas le premier événement à provoquer ses foudres, elle fit réaliser à Stallman la menace qui pesait sur une culture qu'il tenait pour sacro-sainte. « J'avais déjà le sentiment que les logiciels devaient être partagés, mais sans l'avoir jamais clairement formulé. Mes idées sur la question n'étaient alors pas assez claires ni organisées pour pouvoir les exprimer au reste du monde de manière concise. C'est cet incident qui m'a fait prendre conscience de l'importance de l'enjeu. »

Appartenant à l'élite des programmeurs dans l'une des meilleures institutions au monde, Stallman se souciait peu des compromis et marchandages de ses collègues, tant qu'ils n'interféraient pas avec son travail. Il s'en était tenu à considérer avec dédain ces machines et ces programmes que d'autres toléraient en maugréant…

Ce, jusqu'à l'arrivée de l'imprimante laser de Xerox, qui provoqua un changement imperceptible mais profond : si la machine fonctionnait très bien, malgré des bourrages épisodiques, la possibilité d'en modifier le programme à son goût ou selon des besoins communs, avait disparu.

Pour l'industrie logicielle toute entière, la rétention d'information autour des pilotes d'impression annonçait un changement radical de stratégie commerciale. Le logiciel était devenu un actif d'une valeur telle qu'il n'était plus question d'en

publier les secrets de fabrication (le code source). Surtout si cette publication offrait aux concurrents potentiels la possibilité de le reproduire à meilleur marché.

Pour Stallman, l'imprimante était un cheval de Troie. Après dix ans de mise en échec, la « propriétarisation » des logiciels qui allait instaurer l'impossibilité pour l'utilisateur de modifier ou de partager ses logiciels — les futurs hackers parleront de « logiciels propriétaires » ou, aujourd'hui, de « logiciels privateurs », — avait établi une tête de pont à l'intérieur du AI Lab par la méthode la plus sournoise qui soit : déguisée en cadeau.

Vocabulaire Logiciel « privateur » — plutôt que « propriétaire » [1]

L'emploi de l'expression « logiciel privateur » (en anglais *proprietary software*) nécessite quelques explications. Pour traduire l'expression anglaise, la plupart des francophones utilisaient le mot « propriétaire » en procédant à une dérivation impropre du nom vers l'adjectif. Or, depuis quelque temps, dans ses discours en français, Richard Stallman préfère qualifier ces logiciels de « privateurs » pour en souligner les effets nocifs. Notons que ce terme n'est pas un néologisme. D'après le *Trésor de la Langue Française* : « **Privateur**, **-trice**, adj., rare. Qui prive. »

Plusieurs raisons justifient le choix de ce qualificatif. D'abord, l'expression « logiciel propriétaire » laisse croire à tort qu'il faudrait renoncer à ses droits d'auteurs (assimilés aux droits de propriété) pour créer un logiciel libre. Or, comme il en sera notamment question au chapitre 9, la GNU General Public Licence créée par Richard Stallman emprunte à la logique du droit d'auteur en stipulant les conditions d'exercice de ce droit sous la forme de quatre libertés accordées à l'utilisateur du logiciel : utiliser le programme, en étudier le code source, le modifier et distribuer des copies de la version originale ou modifiée.

Enfin, l'adjectif « privateur » exprime bien la privation de liberté — celle induite par le fait de ne pas rendre libre un programme. Quant à l'adjectif « privatif », il accentuerait encore la confusion évoquée plus haut.

1. Voir également la définition donnée par l'Association pour la promotion et la recherche en informatique libre (APRIL) : http://www.april.org/articles/intro/privateur.html.

Le fait que Xerox faisait preuve de générosité envers certains programmeurs, en échange de leur discrétion était également exaspérant, — Stallman reconnaît toutefois qu'il aurait peut-être accepté l'offre dans sa jeunesse — fût-ce à ce prix. Cela dit, l'entretien de Carnegie Mellon avait éveillé en lui une colère qui allait le forcer à abandonner son indolence morale et à considérer désormais avec suspicion tous les marchandages du même acabit. Cela l'avait aussi amené à pousser le raisonnement jusqu'au bout : et s'il arrivait qu'un ami hacker surgisse dans son bureau pour lui demander le code source et que, du jour au lendemain le travail de Stallman consiste à le lui refuser ?

« Je fus à mon tour invité à trahir mes collègues de la même façon, et je me suis alors souvenu de la colère que j'avais ressentie lorsque c'était nous qui étions trahis, 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'. »

⁂

Stallman n'oublierait pas la leçon au cours des tumultueuses années 1980, une décennie durant laquelle de nombreux collègues

du MIT quittèrent le AI Lab et signèrent eux-mêmes des clauses de confidentialité. Certains se dirent sans doute que c'était un mal nécessaire leur permettant de travailler sur les meilleurs projets. Pour Stallman au contraire, la simple signature d'une telle clause annulait toute légitimité d'un point de vue moral. À quoi bon un projet techniquement passionnant s'il ne peut bénéficier à la communauté ?

Comme Stallman devait très vite l'apprendre, refuser de telles offres représentait davantage qu'un sacrifice matériel. Cela allait le conduire à s'isoler des autres hackers qui, partageant la même aversion pour la rétention d'information, tendaient à l'exprimer avec plus de flexibilité morale.

Pour Stallman, refuser de partager un code source avec un confrère revenait non seulement à trahir la mission scientifique sous-tendant le développement logiciel depuis la fin de la Seconde Guerre mondiale, mais aussi à violer la règle d'or fondant toute morale : « ne faites pas aux autres ce que vous ne voudriez pas qu'on vous fît ».

Ainsi l'épisode de l'imprimante laser et l'entrevue qui en découla eurent-ils leur importance. Sans cela, nous dit Stallman, sa vie aurait peut-être suivi un chemin plus ordinaire, alliant le confort matériel d'un programmeur de logiciels commerciaux et la frustration suprême d'une vie passée à écrire des codes invisibles. Une vie dénuée de ce sentiment d'évidence, de cette urgence à résoudre un problème auquel personne d'autre ne s'attaquait. Plus important encore, il n'y aurait pas eu cette grande et juste colère qui, comme nous le verrons bientôt, devait animer son action aussi sûrement qu'une idéologie politique ou une croyance morale.

« J'ai décidé de ne jamais me rendre complice de ce système », dit Stallman, faisant référence à la fois au mécanisme de la clause de confidentialité (qui à ses yeux revient à « brader sa liberté

pour des raisons de commodité »), et à l'esprit qui encourageait ce marchandage moralement douteux. « J'ai décidé de ne jamais faire d'autres victimes comme moi. »

Chapitre

2

2001, l'odyssée d'un hacker

Le département d'informatique de l'université de New York se trouve dans l'enceinte du Warren Weaver Hall, forteresse s'élevant deux pâtés de maisons à l'est du Washington Square Park. L'air conditionné souffle à l'entrée du bâtiment une vague d'air chaud et moite, destinée à décourager vagabonds égarés et avocats en embuscade. Les visiteurs osant s'aventurer au-delà doivent faire face à une autre barrière : la sécurité campée juste derrière l'unique entrée de l'immeuble.

Passé le poste de contrôle, l'atmosphère se fait plus détendue. De nombreuses affichettes disséminées dans tout le rez-de-chaussée mettent toutefois en garde contre les portes coupe-feu laissées ouvertes. Ensemble, ces signes rappellent qu'à New York,

même dans les temps relativement tranquilles de l'avant-11 septembre, on n'est encore jamais assez prudent ni prévoyant.

Ces affichettes offrent un intéressant contrepoint thématique au nombre croissant de visiteurs réunis dans l'atrium du hall. Quelques-uns font penser à des étudiants de l'université de New-York. La plupart ressemblent à ces habitués des concerts, chevelus ébouriffés s'agitant devant la scène avant le grand spectacle. Le temps d'une matinée, les masses auront ainsi pris d'assaut le Warren Weaver Hall, ne laissant au vigile rien de mieux à faire que de regarder Ricky Lake à la télé et de diriger d'un signe de l'épaule vers l'amphithéâtre voisin les nombreux visiteurs demandant « la conf' ».

Une fois à l'intérieur de l'auditorium, le visiteur découvre celui qui a forcé l'arrêt temporaire des procédures de sécurité de l'immeuble. Il s'agit de Richard M. Stallman, fondateur du projet GNU, lauréat en 1990 de la bourse MacArthur [1] et du prix Grace Murray Hopper de l'ACM (*Association for Computing Machinery*), co-bénéficiaire du prix 2001 de la Fondation Takeda pour l'amélioration sociale et économique, et ex-hacker au laboratoire d'intelligence artificielle du MIT. Tous les sites web destinés aux hackers — y compris celui du projet GNU (*gnu.org*) — se sont passé le mot : Stallman est à Manhattan, sa ville d'origine, pour un discours récusant la récente campagne de Microsoft contre la licence GNU GPL (GNU *General Public License*).

Le discours porte sur l'histoire et l'avenir du mouvement pour le logiciel libre. L'endroit où il est prononcé est significatif. Moins d'un mois auparavant, le 3 mai 2001, le vice-président de Microsoft, Craig Mundie prononçait dans un lieu tout proche, la Stern School

1. Également surnommé « bourse des génies » (*genius grant*), le prix MacArthur récompense depuis 1978 aux États-Unis les travaux les plus créatifs et ingénieux tous domaines confondus.

of Business de l'université de New York, un discours dénonçant la GPL, dispositif juridique conçu par Stallman seize ans auparavant.

Pensée pour contrer le principe du secret dans l'industrie logicielle (*software secrecy*) — tendance remarquée pour la première fois par Stallman dès 1980 lors de ses difficultés avec l'imprimante laser Xerox —, la GPL est devenue un outil central de la communauté du logiciel libre.

Pour simplifier, en s'adossant à la puissance juridique du copyright, la GPL maintient de manière irrévocable les logiciels en une forme de propriété commune, que les juristes contemporains nomment désormais *digital commons* (biens communs numériques). Une fois sous GPL, un logiciel ne peut plus être rendu privateur par personne. Dès que, par ce moyen, un auteur fait don d'un code à la communauté, la liberté qu'il confère aux utilisateurs de ce code est inaliénable. De plus, les versions dérivées doivent être couvertes par la même licence si elles comportent une partie substantielle du code source original. C'est pour cette raison que les détracteurs de la GPL l'ont qualifiée de « virale », suggérant à tort qu'elle se propage d'elle-même à tous les logiciels qu'elle touche.

Note

Dans les faits, la GPL n'a pas tout à fait ce pouvoir : si, sur votre machine personnelle, votre code côtoie un programme couvert par la GPL, votre code n'est pas pour autant soumis à la GPL. Pour plus d'informations sur la licence publique générale GNU : http://www.gnu.org.

« La comparaison avec un virus est trop sévère, dit Stallman. Il serait plus approprié de la comparer avec une plante grimpante,

qui se met à pousser là où l'on en place des boutures. » Dans une économie de l'information de plus en plus dépendante des logiciels, et toujours davantage liée aux standards, la GPL est devenue un outil stratégique — tel le *Big Stick* de Roosevelt, appliqué au mouvement du logiciel libre [2]. Même les sociétés ayant d'abord tourné en dérision le « socialisme logiciel » s'accordent finalement à en reconnaître les bénéfices. Ainsi Linux, le noyau [3] de système d'exploitation développé par l'étudiant finlandais Linus Torvalds en 1991, est-il publié sous la licence GPL, comme la plus grande partie du système GNU : GNU Emacs, le débogueur GNU, le compilateur C GCC [4], etc. Ensemble, ces outils forment les composants du système d'exploitation libre GNU/Linux, développé, nourri et détenu par la communauté mondiale des hackers.

Loin d'être vue comme une menace, la GPL a même été un appui pour des compagnies spécialisées dans la haute technologie comme IBM, Hewlett Packard, et Sun Microsystems, qui vendent des applications et des services adaptés à l'infrastructure toujours plus évoluée des logiciels libres.

2. Allusion à la pensée de Théodore Roosevelt qui aimait citer ce proverbe africain : « Parlez doucement et portez un gros bâton, vous irez loin ». Tout au long du vingtième siècle, la stratégie du *Big Stick*, portée par la droite américaine, consistait pour les États-Unis à profiter du poids considérable que leur conférait leur puissance industrielle, afin d'affirmer leur politique de la zone d'influence, garantie par la force. La doctrine du *Big Stick* est néanmoins plus ancienne. Elle berça notamment la fin du dix-neuvième siècle et justifia par exemple l'annexion de l'île de Guam, des Philippines, de Porto Rico ou de Panama (plusieurs fois). On retrouve des éléments de cette stratégie centenaire lors de la récente guerre du Golfe — NdT.

3. Comme nous le verrons plus tard, Linux n'est que le composant du système d'exploitation chargé de fournir les ressources de la machine aux programes.

4. *GNU Compiler Collection*. Il est issu du projet GNU qui comporte une collection de logiciels capables de compiler divers langages comme C, C++, Fortran, etc. Le projet GCC fut initié par Richard Stallman en 1985 et, depuis, il a été largement enrichi suivant l'évolution des langages utilisés — NdT.

Si ces applications s'exécutent sur des systèmes GNU/Linux, elles n'en sont pas toujours pour autant elles-mêmes des logiciels libres, bien au contraire. La plupart sont des logiciels privateurs qui respectent aussi peu la liberté de l'utilisateur qu'un système Windows. Non libres, elles peuvent participer au succès de GNU/Linux mais sans contribuer à la finalité libératrice qui motiva la création de ce système.

La GPL est également considérée par ces entreprises comme une arme stratégique dans la longue guerre que livre la communauté des hackers à la compagnie de Redmond, État de Washington. De par l'hégémonie de son système d'exploitation Windows, cette dernière est en situation de quasi-monopole sur le marché des logiciels PC depuis la fin des années 1980. Microsoft serait ainsi le premier à pâtir d'un basculement massif de l'industrie logicielle vers la licence GPL.

Chacun des programmes du colosse logiciel qu'est Windows est couvert par des copyrights et des contrats de licence (*End User License Agreements* — EULAs, « Contrat de licence utilisateur final » ou CLUF). Ces dispositifs stipulent le statut privateur non seulement des fichiers exécutables, mais aussi des codes source sous-jacents, auxquels l'utilisateur n'a de toute façon pas accès. Incorporer à l'un de ces programmes un code protégé par la virale GPL est interdit ; pour satisfaire aux obligations de cette dernière, c'est tout ledit programme que Microsoft serait légalement contraint de rendre libre. Les concurrents pourraient alors le copier, le modifier et en envoyer des versions améliorées, sapant de ce fait les bases du verrouillage imposé par Microsoft à ses utilisateurs.

D'où l'inquiétude grandissante de la compagnie quant au taux d'adoption de la GPL… D'où, aussi, le discours de Craig Mundie en 2001, démontant systématiquement la GPL, attaquant la

conception « open source » [5] du développement et de la vente de logiciels. D'où, enfin, la décision de Stallman de réfuter ce jour-là publiquement, sur ce même campus, les arguments de ce discours.

Deux décennies constituent une longue période pour l'industrie logicielle. Pensons que lorsque Stallman maudissait l'imprimante Xerox du AI Lab en 1980, Microsoft n'était encore qu'une start-up détenue par quelques personnes. IBM, considérée comme la société la plus puissante du secteur industriel du matériel informatique, n'avait pas encore introduit son premier ordinateur personnel, initiateur du boom des PC bon marché. La plupart des technologies qui nous entourent au quotidien — de la grande toile mondiale du World Wide Web à la télévision par satellite, en passant par les consoles de jeux vidéo 32 et 64 bits — n'existaient pas. Pas plus que les entreprises qui dominaient le début des années 2000 (AOL, Sun Microsystems, Amazon.com, Compaq, et Dell, notamment). La liste est longue.

Pour certains, qui raisonnent avant tout en termes de progrès plutôt que de liberté, la croissance fulgurante du marché des hautes technologies est un argument à la fois pour et contre la GNU GPL.

Certains y voient un recours nécessaire, en raison de la durée de vie toujours plus courte des plates-formes matérielles informatiques. Le risque d'acheter un produit obsolète pousse les consommateurs à se tourner massivement vers les sociétés les plus pérennes, avec pour résultat un oligopole où seuls quelques gros acteurs se partagent le marché. Le modèle économique du logiciel privateur mène selon eux à des situations de monopole et à

5. Microsoft préfère ignorer l'expression « logiciel libre » et attaquer directement le camp apolitique de « l'open source » pour détourner l'attention du mouvement du logiciel libre (cf. chap. 11).

des abus de position dominante, ainsi qu'à la stagnation du marché. Les puissants y accaparent tout l'oxygène, au détriment des concurrents et des start-ups innovantes. [6]

D'autres considèrent au contraire que la GPL précipitera la perte de ceux qui l'utilisent. Selon eux, vendre un logiciel est au moins aussi risqué que l'acheter. Sans la protection juridique de licences logicielles restrictives, et sans la perspective alléchante d'être seul détenteur d'un logiciel révolutionnaire initiant tout un nouveau marché (*killer app* [7]), plus aucune société n'aurait d'intérêt à se lancer. Là aussi, on verrait le marché stagner et l'innovation se tarir. Pour Craig Mundie, dans son discours du 3 mai sur ce même campus, la nature « virale » de la GPL « menace » toute entreprise dont l'actif principal est l'exclusivité qu'elle prétend détenir sur son logiciel. Mundie ajoutait : « Cela minerait l'avènement d'acteurs indépendants sur le marché logiciel, car il deviendrait alors impossible de distribuer des logiciels en faisant payer le produit lui-même plutôt que son seul coût de distribution. » [8]

Le succès tant de GNU/Linux que de Windows au cours des vingt dernières années tend à montrer qu'il y a du vrai dans ces deux approches à la fois. Toutefois, les activistes du logiciel libre, comme Stallman, pensent que le véritable enjeu du débat est ailleurs. La vraie question, à leurs yeux, n'est pas de savoir

6. Voir Shubba Ghosh, *Revealing the Microsoft Windows Source Code*, Gigalaw.com (janvier 2000) [Ghosh, 2000].

7. Un logiciel révolutionnaire (*killer app*) peut très bien être libre. C'est le cas du légendaire navigateur *Mosaic*, créé en 1993 et à l'origine de Netscape puis de Firefox. Sa licence permettait des dérivés non commerciaux sous certaines restrictions. De plus, le lecteur l'aura compris : le marché du logiciel est une sorte de loterie où le nombre de participants augmente avec le potentiel de gains. Pour un bon résumé sur le phénomène du logiciel révolutionnaire, voyez [Ben-David, 2000].

8. Voir le discours de Craig Mundie (vice-président senior, Microsoft Corp.), prononcé le 3 mai 2001 à la New York University Stern School of Business : [Mundie, 2001].

lequel des deux modèles aura le plus de succès, mais lequel est le plus éthique.

Quoi qu'il en soit, se battre pour conserver une masse critique reste un enjeu majeur dans l'industrie du logiciel. Ainsi, même les plus gros vendeurs, tel Microsoft, n'hésitent pas à prendre appui sur des entreprises tierces qui, par leurs outils, leurs logiciels ou leurs jeux, contribuent à renforcer l'attrait de la plate-forme pour le consommateur moyen. Se référant à l'évolution rapide du marché technologique des vingt dernières années, et l'impressionnant parcours de sa propre entreprise, Mundie conseille de ne pas se laisser emporter par la vogue récente du logiciel libre : « Les deux dernières décennies ont démontré qu'un schéma économique protégeant la propriété intellectuelle, associé à un modèle commercial permettant de récupérer les coûts de recherche et développement, peut générer d'impressionnants bénéfices économiques et les redistribuer largement. »

⁂

C'est à ces critiques que répond aujourd'hui le discours de Stallman. Moins d'un mois après ces déclarations, il se tient devant l'un des tableaux noirs du fond de la salle, impatient de riposter.

Si les vingt dernières années ont changé la face du marché logiciel, elles ont transformé plus encore Richard Stallman lui-même. Il n'est plus le hacker mince et 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. Vêtu d'un t-shirt couleur eau et d'un pantalon en

polyester marron, Stallman a l'allure d'un ermite du désert sortant d'un vestiaire de l'Armée du Salut.

La foule est constituée de visiteurs partageant ses goûts vestimentaires. Beaucoup sont venus avec leur portable et modem cellulaire : quoi de mieux pour enregistrer et transmettre les paroles de Stallman, via Internet, à un auditoire distant dans l'expectative. Le ratio des genres est d'environ quinze hommes pour une femme, et l'une des sept ou huit présentes tient un manchot en peluche, la mascotte officielle de Linux, alors qu'une autre porte un nounours.

Agité, Stallman quitte la scène et rejoint une chaise du premier rang, tapant quelques commandes sur un portable déjà ouvert. Durant les dix minutes suivantes, il reste concentré, oublieux du nombre croissant d'étudiants, de professeurs et d'admirateurs qui passent devant lui.

En préliminaire, le rituel baroque des formalités académiques doit être observé. La présence de Stallman ne mérite pas un, mais deux discours d'introduction. Mike Uretsky, co-directeur du Stern School's *Center for Advanced Technology*, prononce le premier. « Le rôle d'une université est de favoriser le débat et d'être le lieu de discussions intéressantes. Cette présentation particulière, ce séminaire, entrent parfaitement dans ce cadre. Je trouve la question de l'open source tout particulièrement intéressante. »

Avant qu'Uretsky ne puisse prononcer un autre mot, Stallman est déjà debout, faisant de grands signes de bras tel un automobiliste en panne : « Je fais du *logiciel libre*!, lance-t-il sous les rires croissants. L'open source est un tout autre mouvement. » Les rires laissent place aux applaudissements. La salle est pleine de partisans de Stallman, des gens connaissant de réputation son exactitude verbale, mais aussi son conflit très médiatisé en 1998 avec les défenseurs de l'open source. Beaucoup sont venus pour

assister à de tels éclats, tels des amateurs des émissions de radio de Jack Benny attendant sa traditionnelle réplique : « Maiiiis arrêêête çaaa ! ».

Uretsky se hâte de clore son introduction et cède la scène à Edmond Schonberg, professeur au département de 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 liés à l'inaccessibilité des codes source. Il a développé une philosophie cohérente qui nous contraint tous à réexaminer nos idées sur la manière dont les programmes sont produits, sur la signification de la propriété intellectuelle[9] 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 puis 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 de m'avoir donné l'opportunité d'être présent sur cette estrade, ironise-t-il. Depuis ces dernières semaines, je me suis senti comme un auteur dont les livres ont été fortuitement interdits quelque part. »

⁂

9. Si ce discours avait eu lieu de nos jours, Stallman s'opposerait à l'emploi de l'expression « propriété intellectuelle », qu'il considère source de confusion ([Stallman, 2008]).

Pour les néophytes, Stallman se lance, en guise d'échauffement, dans une rapide analogie. Il compare un logiciel à une recette de cuisine : les deux donnent d'utiles instructions, pas à pas, pour accomplir une tâche souhaitée. Ils peuvent être aisément modifiés en fonction des désirs spécifiques de l'usager, ou de 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 en raffolez. Mettre moins de sel car votre médecin vous le conseille — peu importe. »

« De surcroît, poursuit-il, 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 il l'inscrit. Partager un logiciel nécessite encore moins, habituellement quelques clics de souris et un minimum d'électricité. Dans tous les cas, la personne qui donne l'information y gagne deux choses : davantage d'amitié et la possibilité de récupérer en retour d'autres recettes intéressantes. »

Stallman monte d'un cran : « Imaginez que les recettes soient emballées dans des boîtes noires. Vous ne pourriez pas connaître les ingrédients utilisés, encore moins les changer. Et si vous en faisiez une copie pour un ami, ils vous qualifieraient de pirate et essaieraient de vous faire emprisonner des années durant. Un tel changement susciterait un énorme scandale chez les gens ayant l'habitude de partager des recettes. Mais c'est exactement ce que nous impose le monde du logiciel propriétaire. Un monde dans lequel la bienséance commune envers les autres est prohibée ou empêchée. »

Après avoir posé cette analogie en introduction, Stallman se lance une nouvelle fois dans le récit de l'épisode de l'imprimante laser Xerox. À l'instar de l'analogie culinaire, l'histoire de l'imprimante est un procédé rhétorique fort utile. Structurée comme une parabole, elle illustre la volatilité des choses dans le

monde du logiciel. L'auditoire, ramené à une ère antérieure à celle d'Amazon.com-achetez-en-un-clic, de Microsoft Windows et des bases de données Oracle, doit reconsidérer la notion de propriété logicielle sans les principaux logos corporatifs qui l'ornent alors.

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

« Il nous avait trahi, dit-il. Mais pas seulement nous. Sans doute *vous* a-t-il trahi aussi. » Sur le mot « vous », Stallman pointe un doigt accusateur vers un membre insouciant de l'auditoire. La cible a à peine le temps de sourciller que les yeux de Stallman sont déjà ailleurs. Lentement et délibérément, il désigne un second auditeur, générant une vague de gloussements nerveux dans la foule « Et il vous a probablement trahi, vous aussi », dit-il désignant un spectateur trois rangées derrière le premier.

Au moment où Stallman désigne une troisième personne, un rire général succède à la fébrilité. Le geste semble mis en scène, et c'est le cas. Puis, lorsque vient le temps de clore son récit, Stallman embrasse d'un geste théâtral toute l'assemblée, avec l'art consommé d'un homme de scène. « Il a probablement trahi l'ensemble des personnes présentes dans cette salle — à l'exception, peut-être, de celles qui n'étaient pas encore nées en 1980, clame-t-il, provoquant de nouveaux rires. « Parce qu'il s'est engagé à refuser de coopérer avec pratiquement toute la population de la planète Terre. »

Stallman laisse ce commentaire faire son effet, avant d'ajouter : « Il a signé un accord de non-divulgation. »

⁂

Au cours des vingt dernières années, du chercheur académique déçu au leader politique, l'ascension de Richard Matthew Stallman est éloquente. Elle témoigne de sa nature opiniâtre et de sa volonté prodigieuse, de la clarté avec laquelle s'articule sa vision, et des valeurs du mouvement du logiciel libre qu'il a aidé à construire. Elle souligne la haute qualité des logiciels qu'il a créés, et qui ont assis sa réputation de programmeur légendaire. Elle met en relief la dynamique qu'a fait naître la licence GPL, innovation juridique que de nombreux observateurs voient comme son accomplissement le plus capital.

Plus important encore, cette ascension illustre la versatilité du 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 cette raison, qu'à une époque où les célébrités du monde de la haute technologie tendent à décliner, l'étoile de Stallman brille davantage. Depuis le lancement du projet GNU en 1984, il fut tour à tour ignoré, caricaturé, diffamé et attaqué tant à l'extérieur du mouvement du logiciel libre qu'à l'intérieur. Le projet GNU n'en a pas moins réussi à atteindre les objectifs qu'il s'était fixés — non sans quelques retards notoires — et est ainsi demeuré pertinent dans un marché mille fois plus complexe que dix-huit ans plus tôt. Il en va de même de l'idéologie du logiciel libre, idéologie soigneusement mûrie par Stallman lui-même.

Culture L'acronyme récursif GNU's Not Unix

GNU est l'acronyme de « GNU's Not Unix » (GNU N'est pas Unix). Dans ce même discours du 29 mai 2001 à l'université de New-York, Stallman en résume l'origine : « Nous, hackers, cherchons toujours un nom drôle ou coquin pour un logiciel, parce que cela fait partie intégrante du plaisir de l'écrire. Nous avions

cette tradition des acronymes récursifs, pour signifier que tel programme que vous écrivez est similaire à un autre déjà existant. Je cherchais un acronyme récursif pour *Something Is Not Unix*. J'ai essayé les vingt-six lettres, avant de découvrir qu'aucune combinaison ne composait un mot. J'ai décidé de recourir à la contraction du *Is* qui me permettrait d'obtenir un acronyme de trois lettres *Something's Not Unix*. J'ai réessayé des lettres, et suis tombé sur 'GNU'. C'est tout. »

Bien qu'amateur de jeux de mots, Stallman recommande aux programmeurs de prononcer le 'g' au début de l'acronyme (c'est-à-dire 'gue-niou'). Cela aide à éviter la confusion non seulement avec le mot *gnu* (gnou), cette antilope africaine *connochaetes gnou*, mais aussi avec l'adjectif *new* (nouveau). « Nous y travaillons depuis dix-sept ans maintenant, alors ce n'est plus réellement nouveau », dit Stallman[1].

1. D'après les notes de l'auteur et la transcription du discours de Richard Stallman du 29 mai 2001 à l'université de New York, « *Free Software : Freedom and Cooperation* » : http://www.gnu.org/events/rms-nyu-2001-transcript.txt

***

Pour comprendre les raisons de cette adhésion massive, il n'est pas inutile d'examiner Richard Stallman à travers les témoignages de ceux qui collaborèrent ou luttèrent avec lui tout au long de ce parcours. Le personnage n'est pas complexe ; il est bien de ceux qui sont exactement ce qu'ils paraissent.

« Je crois que si vous voulez comprendre l'être humain qu'est Richard Stallman, vous devez vraiment considérer toutes ses facettes comme un ensemble cohérent », conseille Eben Moglen, ju-

riste à la *Free Software Foundation*[10] et professeur de droit à la Columbia University Law School. « Toutes les excentricités personnelles que beaucoup voient comme un obstacle à la compréhension de Stallman font réellement *partie* de lui : son fort sentiment de contrariété personnelle, son sens aigu de l'engagement éthique et son incapacité à faire des compromis, surtout face aux problèmes qu'il juge fondamentaux, sont autant de raisons qui expliquent ce qu'il a fait, et pourquoi. »

Il n'est pas aisé de retracer le voyage qui a commencé avec une imprimante laser, puis a conduit à une confrontation radicale avec l'entreprise à l'époque la plus riche du monde. Il faut prendre en compte de manière approfondie les forces qui ont consacré l'appropriation des logiciels dans notre société. Examiner de façon réfléchie le parcours d'un homme qui, comme bien des leaders politiques avant lui, connaît la malléabilité de la mémoire humaine. Il faut encore pouvoir interpréter les mythes et les mots chargés de sens politique qui se sont accumulés avec le temps autour de Stallman. Et comprendre, enfin, son génie en tant que programmeur, comme 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, reconnaît-il. La plupart des gens qui essaient de faire quelque chose de très

10. La Fondation pour le logiciel libre est une fondation américaine à but non lucratif basée à Boston, Massachusetts. À l'initiative de Richard Stallman, elle fut créée en 1985 afin de lever des fonds pour supporter le projet GNU. Actuellement la FSF promeut le logiciel libre en général, œuvre à la défense de ses intérêts et développe le projet GNU ainsi que toutes ses ramifications. C'est une organisation politique et lobbyiste jouant un rôle incontournable dans la définition du logiciel libre aujourd'hui. Pour plus d'information voyez le site de la FSF : http://www.fsf.org, et sur l'histoire du projet GNU : http://gnu.org/gnu/thegnuproject.html — NdT.

difficile finissent par se décourager et abandonnent. Je n'ai jamais abandonné. »

Il reconnaît aussi une part de hasard. Sans la 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 s'imagine très facilement prenant un autre chemin. Même si, aujourd'hui, il remercie les forces et les circonstances qui l'ont mis en position de changer le monde.

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

Chapitre

3

Portrait de Richard en jeune homme

La mère de Richard Stallman, Alice Lippman, se souvient encore du moment où elle s'est rendu compte du don particulier de son fils. « Il devait avoir huit ans », se souvient-elle. C'était en 1961, un après-midi de fin de semaine.

Mme Lippman, jeune mère célibataire récemment divorcée, passait le temps dans le minuscule deux-pièces familial, dans l'ouest de Manhattan. Feuilletant un exemplaire du magazine *Scientific American*, elle tomba sur sa rubrique préférée, les « Jeux mathématiques » de Martin Gardner. Alors professeur d'arts intérimaire, elle aimait la gymnastique intellectuelle que procurait cette lecture.

Voyant son fils à ses côtés, déjà absorbé dans un livre, elle décida de s'essayer au problème de la semaine. « Je n'étais pas vraiment douée pour les casse-têtes, avoue-t-elle. Mais en tant qu'artiste, ce genre d'activité m'aidait à franchir certaines barrières conceptuelles ». Or cette fois-là, Alice Lippman eut l'impression de se heurter à un mur de briques. Dépitée, elle s'apprêtait à jeter le magazine, lorsqu'elle sentit qu'on la tirait gentiment par la manche.

« C'était Richard, se souvient-elle. Il voulait savoir si j'avais besoin d'aide. » Regardant tour à tour son fils et le magazine, elle accueillit tout d'abord l'offre avec un certain scepticisme. « J'ai demandé à Richard s'il avait lu le magazine, dit-elle, et il m'a répondu que non seulement il l'avait lu, mais qu'il avait même résolu le problème. Et il a enchaîné en m'expliquant la solution. »

Face à la logique mise en œuvre par son fils, Mme Lippman passe rapidement du scepticisme à l'incrédulité. « Certes, j'avais toujours su que c'était un garçon brillant. Mais c'était la première fois que j'avais un indice clair de son degré d'avancement intellectuel. »

Trente ans plus tard, Mme Lippman s'amuse encore d'un tel souvenir : « Pour vous dire la vérité, je ne crois pas que j'aie jamais compris comment résoudre le casse-tête, dit-elle. Je me rappelle juste que je n'en revenais pas qu'il ait trouvé la solution ».

Assise dans la salle à manger de son deuxième et spacieux appartement à Manhattan — dans lequel elle a emménagé avec son fils lors de son deuxième mariage (avec feu Maurice Lippman, en 1967), Alice Lippman mêle à la fierté typique d'une mère juive un certain étonnement, lorsqu'elle se remémore les premières années de son fils. Bien en évidence sur un buffet trône une très grande photo montrant un Stallman barbu, rayonnant dans sa toge doctorale. Le portrait éclipse toutes les autres photographies de nièces

et de neveux, mais avant que le visiteur n'ait pu en faire trop grand cas, Mme Lippman se hâte de relativiser, légèrement ironique.

« Richard a insisté pour me la donner quand il a obtenu 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'aie assisté !' »

Des commentaires imprégnés de cet humour qu'on retrouve chez tous ceux qui ont la difficile tâche d'éduquer un enfant prodige. Soyez-en sûrs, pour chaque anecdote qu'Alice Lippman lit ou entend concernant l'obstination de son fils et son étrange comportement, elle pourrait en raconter une douzaine d'autres.

« Il était si conservateur ! s'exclame-t-elle, levant les mains au ciel dans un accès de fausse exaspération. C'est à cette table que nous avions nos pires disputes. Lorsque j'ai fait grève avec d'autres pour former le premier syndicat d'instituteurs de l'école publique, Richard a été furieux contre moi. Pour lui, les syndicats étaient corrompus. Il était aussi très opposé à un système de sécurité sociale. Il pensait que les gens pouvaient gagner beaucoup plus d'argent en investissant eux-mêmes, de leur côté… Qui aurait pensé que dix ans plus tard il serait devenu aussi idéaliste ? Je me souviens encore de sa demi-sœur me demandant : 'Mais qu'est-ce qu'il va devenir ? Un fasciste ?' ».

L'avis de Stallman a bien sûr radicalement changé et il dit aujourd'hui : « Lorsque j'étais adolescent, je ne comprenais pas la plupart des difficultés auxquelles les gens étaient confrontés dans leur vie car mes problèmes étaient différents. Je n'appréciais pas la façon dont les plus riches réduisaient les autres à la pauvreté, le seul rempart étant de nous organiser à tous les niveaux pour les en empêcher. Je ne comprenais pas à quel point il est difficile, pour la plupart des gens, de résister à la pression sociale — celle

qui incite à faire des choses stupides comme dépenser tout son argent au lieu de l'épargner — parce que j'étais à peine conscient de cette pression. En outre, les syndicats des années 1960, à l'apogée de leur puissance, étaient parfois arrogants ou corrompus. Ils sont fort affaiblis aujourd'hui, avec pour résultat que la croissance économique, lorsqu'elle a lieu, ne bénéficie qu'aux riches. » D'ailleurs, au sujet de la réforme du système de santé américain, sa position est au moins progressiste — voir son appel à soutenir un système de sécurité sociale : http://stallman.org/archives/2009-jul-oct.html#25%20October%202009.

Mariée en 1948 avec Daniel Stallman, le père de Richard, Alice Lippman divorça en 1958 — la décision de justice entérinant une garde partagée. Mère célibataire pendant près de dix ans, elle peut témoigner de l'aversion de Richard pour toute forme d'autorité, mais aussi de son inextinguible soif de connaissance. C'est lorsque ces deux aspects se combinaient que Richard et sa mère connaissaient leurs pires affrontements.

« On aurait dit qu'il ne voulait jamais manger, dit-elle en se remémorant son comportement, de l'âge de huit ans jusqu'à son baccalauréat en 1970. Il ne m'entendait pas quand je l'appelais pour le dîner. Je devais crier neuf ou dix fois avant qu'il me prête attention. Il était complètement absorbé. »

Stallman, quant à lui, se remémore à peu près les mêmes choses, mais en y ajoutant une connotation politique : « J'adorais lire. Si je voulais lire et que ma mère m'ordonnait d'aller manger ou dormir, je n'obéissais pas. Je ne voyais aucune raison pour qu'elle m'empêche de lire, ou qui l'autorise à me dire ce que j'avais à faire, point. La majeure partie de ce que j'avais lu à propos de la démocratie et des libertés individuelles, je l'appliquais à moi-même. Car je ne voyais aucune raison d'en exclure les enfants. »

Cette conviction que la liberté individuelle doit primer sur l'autorité arbitraire se manifestait aussi à l'école. À onze ans, avec deux années d'avance par rapport à ses camarades de classe, Richard Stallman subit toutes les frustrations habituelles réservées à un écolier surdoué dans le système scolaire public américain. Peu après l'épisode de l'énigme mathématique, sa mère participa à ce qui devait être la première étape d'une longue série de rencontres avec les professeurs.

« Il refusait résolument d'écrire, dit-elle, se rappelant un ancien sujet de conflits. Je crois que le dernier travail qu'il a rendu, avant sa dernière année de lycée, était un essai sur l'histoire du système numérique occidental, pour un professeur de quatrième. » Pour Stallman, être obligé de choisir un thème alors que rien ne lui inspirait de réelle envie d'écrire était tout bonnement impossible, si bien qu'il faisait tout pour l'éviter.

Doué pour tout ce qui exigeait un raisonnement analytique, Richard Stallman était attiré par les mathématiques et les sciences, au détriment des autres matières. Là où certains professeurs fustigeaient l'obstination d'un esprit borné, sa mère ne voyait que de l'impatience. Selon elle, les mathématiques et les sciences offraient tant de matière à apprendre qu'elles en occultaient totalement les autres disciplines, pour lesquelles son fils avait moins de dispositions.

Quand il eut dix ou onze ans, les garçons de sa classe commencèrent à s'entraîner au football américain. Alice Lippman se souvient que son fils rentra un jour en rage à la maison. « Il aurait tant voulu jouer ! Mais il manquait de coordination, dit-elle. Cela le mettait hors de lui… »

Cette frustration devait amener Stallman à se focaliser plus encore sur les mathématiques et les sciences. Or, même dans ces matières, son intransigeance lui jouait des tours. Versé dans l'analyse

mathématique depuis ses sept ans, Stallman ne voyait pas la nécessité de simplifier son discours avec les adultes. Un jour, sa mère engagea un étudiant de l'université de Columbia toute proche pour jouer le rôle de grand frère. L'étudiant quitta l'appartement après la première session, et ne revint jamais. « Je pense que Richard lui parlait de choses qui lui passaient au-dessus de la tête », dit-elle.

Une des anecdotes favorites de Mme Lippman remonte au début des années 1960, peu après l'épisode du casse-tête. Vers ses sept ans, soit deux années après le divorce et le déménagement, Richard se prit d'intérêt pour le lancement de fusées miniatures, dans le parc de Riverside Drive. Ce qui avait commencé comme un jeu anodin prit bientôt un tout autre aspect quand le garçon commença à consigner les résultats de chaque lancement. Tout comme son intérêt pour les jeux mathématiques, ce passe-temps n'attira guère l'attention de Mme Lippman, jusqu'au jour où elle demanda à son fils s'il voulait regarder la retransmission d'un lancement important de la NASA.

« Il enrageait littéralement, dit-elle. Tout ce qu'il put dire fut : 'Mais, je n'ai encore rien publié'. Apparemment, il avait rédigé quelque chose qu'il voulait réellement montrer à la NASA ». Stallman, quant à lui, ne se souvient pas de l'incident et croit plus probable que son angoisse était due au fait qu'il n'avait précisément rien à montrer.

Ces anecdotes constituent un premier témoignage de l'intensité intellectuelle qui allait devenir la marque de Stallman tout au long de sa vie. Quand les autres enfants venaient à table, Stallman restait lire dans sa chambre. Quand ils se prenaient pour Johnny Unitas, ce célèbre footballeur américain des années 1950, Stallman jouait, lui, à l'astronaute. « J'étais bizarre, dit Stallman, résumant brièvement sa jeunesse, lors de l'interview de Michael Gross en

1999 [1]. Passé un certain âge, je n'ai plus eu que des enseignants pour amis ». Si Stallman n'avait pas honte de ces particularités, il considérait en revanche son inaptitude sociale comme un échec. En fait, tous deux contribuèrent à son exclusion sociale.

Quoique cela ouvrît la porte à de nouvelles prises de bec à l'école, Alice Lippman décida d'encourager la passion de son fils. À douze ans, Richard fréquentait des camps scientifiques l'été, et des écoles privées au cours de l'année scolaire. Lorsqu'un enseignant lui recommanda de s'inscrire au Columbia Science Honors Program (SHP), un programme post-Spoutnik créé pour les étudiants et collégiens surdoués de la ville de New York, Stallman élargit ses activités parascolaires et se mit à faire régulièrement la navette en métro, le samedi, jusqu'au campus de la Columbia University.

Dan Chess, un ancien de ce programme, se rappelle que Stallman semblait un peu étrange, même au milieu d'une foule d'étudiants partageant la même fascination 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 aussi incroyablement intelligent. J'ai rencontré bien des gens intelligents dans ma vie, mais je crois que c'est le plus intelligent de tous. »

Seth Breidbart, qui suivait lui aussi ce programme, abonde dans ce sens. Programmeur informatique resté en contact avec

1. L'une des principales sources documentaires de ce chapitre est l'interview « Richard Stallman : lycéen marginal, symbole du logiciel libre, génie certifié MacArthur » (« *Richard Stallman : High School Misfit, Symbol of Free Software, MacArthur-Certified Genius* ») de Michael Gross, auteur du livre *Talking About My Generation* (*Discussions sur ma génération*), un recueil d'entrevues avec plusieurs personnalités de la génération « Baby Boom ». Bien que Stallman n'apparaisse pas dans l'ouvrage, Gross a publié le compte rendu de la rencontre sur le site Internet du livre en tant que supplément virtuel. L'URL de cette entrevue a changé plusieurs fois depuis que je l'ai consultée, mais plusieurs lecteurs en ont rapporté la disponibilité (voir la référence bibliographique) : [Gross, 2000].

Stallman, grâce à une passion commune pour la science-fiction et les conventions afférentes, il se souvient de lui à quinze ans, la boule à zéro, « effrayant », surtout pour un compagnon du même âge.

« Je ne pourrais pas l'expliquer, dit Breidbart. Ce n'est pas qu'il était inabordable. Il était tout simplement très sérieux. Très cultivé, mais aussi très borné, d'une certaine manière. »

De telles descriptions incitent à se poser des questions : des adjectifs comme « sérieux » et « borné » sont-ils simplement une manière de décrire des traits de personnalité qui, aujourd'hui, pourraient être vus comme des troubles comportementaux juvéniles ? Un article de la revue *Wired* de décembre 2001, intitulé « *The Geek Syndrome* », par Steve Silberman, dresse le portrait de plusieurs enfants doués pour les sciences et atteints d'autisme de haut niveau ou du syndrome d'Asperger. À bien des égards, les souvenirs des parents cités dans cet article présentent une similitude troublante avec ceux de Mme Lippman.

Stallman lui-même s'est renseigné à ce sujet. Lors d'une entrevue pour un portrait au *Toronto Star* en 2000, il a déclaré qu'il se demandait s'il n'était pas « atteint d'une forme légère d'autisme ». L'article déforma ses propos, transformant son questionnement en certitude. Néanmoins Stallman défend une vision du logiciel libre et de la coopération sociale qui contraste vivement avec l'isolement de sa vie personnelle. Son excentricité le rapproche de Glenn Gould, pianiste canadien brillant et grand orateur, mais terriblement seul [2]. Stallman se considère affecté, jusqu'à un certain point, par l'autisme qui, dit-il, complique ses relations avec les gens

Cette hypothèse est bien sûr facilitée par le caractère imprécis des fameux « troubles du comportement », comme on les nomme

2. Steed, 2000

aujourd'hui. Steve Silberman observe que les psychiatres américains n'ont que récemment accepté le syndrome d'Asperger comme terme générique, regroupant nombre de ces traits comportementaux.

Ces caractéristiques vont du manque d'habileté motrice et d'une sociabilité limitée, à une intelligence fortement développée et à un goût prononcé voire obsessionnel pour les chiffres, les ordinateurs et autres systèmes ordonnés [3].

« Il n'est pas impossible que j'aie été atteint d'une chose de ce genre, dit-il. D'un autre côté, un des aspects de ce syndrome est la difficulté à suivre un rythme. Or je sais danser et j'adore même suivre les rythmes les plus complexes. Toutes ces choses sont de toute façon un peu trop floues pour se faire une idée claire ». Stallman a également pu souffrir d'une ombre de syndrome d'Asperger, tout en restant cependant dans les limites de la normalité [4].

Dan Chess, pour sa part, rejette toute tentative de post-diagnostic. « Je ne l'ai jamais cru atteint d'une telle chose, dit-il. Il était juste très asocial, comme nous tous là-bas, d'ailleurs. »

Mme Lippman, pour sa part, n'écarte pas une telle possibilité. Elle se rappelle en effet de quelques histoires offrant matière à spéculation. L'un des symptômes les plus frappants de l'autisme est l'hypersensibilité aux bruits et aux couleurs. Mme Lippman se rappelle 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 qu'au bout de la troisième fois que nous avons compris ce qui se passait : le bruit du ressac lui faisait mal aux oreilles. » Elle se remémore aussi une réaction similaire, concernant la couleur :

3. Silberman, 2001

4. Voir à ce propos John Ratey et Catherine Johnson, *Shadow Syndromes*, New York : Pantheon Books, 1997.

« Ma mère avait de brillants cheveux roux, et chaque fois qu'elle se penchait pour prendre Richard, il poussait un cri. »

Mme Lippman dit s'être documentée ces dernières années sur l'autisme. Elle croit que ces épisodes sont plus qu'une coïncidence. « Je sens bien que Richard possède certains traits d'un enfant autiste, dit-elle. Je regrette que l'autisme ait été si mal connu à l'époque. »

Avec le temps, cependant, elle reconnaît que son fils a su s'adapter. À sept ans, raconte-t-elle, il adorait prendre place dans le wagon de tête du métro, face à la fenêtre, traçant et mémorisant le plan labyrinthique du réseau de chemins de fer sous la ville. Un passe-temps qui exigeait de s'accoutumer aux bruits intenses accompagnant chaque trajet en train. « Seul le bruit initial semblait l'incommoder, raconte-t-elle. C'était comme s'il était choqué par le son, puis que ses nerfs parvenaient à s'adapter. »

***

Le souvenir de Madame Lippman dépeint un garçon animé du même enthousiasme, de la même énergie et de la même sociabilité que n'importe quel autre du même âge. Ce n'est qu'après une série d'événements difficiles ayant ébranlé le foyer familial que son fils devint, selon elle, introverti et distant.

Le premier de ces événements traumatiques fut le divorce d'Alice et de Daniel Stallman. Mme Lippman raconte que, bien qu'elle et son ex-mari aient tenté d'y préparer leur fils, ce choc n'en fut pas moins destructeur. « Il semblait ne pas prêter attention à ce que nous tentions de lui expliquer, se souvient-elle. Mais la réalité l'a brutalement rattrapé lorsque nous avons emménagé, lui et moi, dans un nouvel appartement. La première chose qu'il a dite fut : 'Où sont les meubles de papa ?' »

La décennie suivante, Stallman allait passer la semaine chez sa mère à Manhattan, et le week-end au domicile de son père dans le quartier du Queens. Ces allers-retours lui permirent d'observer deux styles d'éducation parentale très différents. Une expérience qui, jusqu'à présent, l'a fermement convaincu de ne pas élever d'enfant lui-même.

Sur son père, vétéran de la Deuxième Guerre mondiale, décédé début 2001, Stallman oscille entre respect et rancœur. D'un côté, il y avait l'homme moralement engagé qui allait apprendre le français pour servir les alliés lorsqu'ils combattraient les nazis en France. De l'autre, il y avait ce père toujours inventif voire cruel pour mieux dénigrer [5]. « Mon père avait un horrible tempérament, dit Stallman. Il ne hurlait jamais, mais trouvait toujours le mot apte à faire mouche, à vous démolir. »

Concernant la vie chez sa mère, Stallman est moins équivoque. « C'était la guerre, dit-il. Dans ma désolation, je disais souvent : 'Je veux rentrer chez moi' en faisant référence à ce refuge inexistant que je n'aurais jamais. »

Les premières années après le divorce, Stallman trouva calme et tranquillité chez ses grands-parents paternels. Mais l'un décéda lorsque Stallman avait huit ans, et l'autre lorsqu'il en avait dix. La perte fut dévastatrice pour lui. « Je leur rendais visite, et je me sentais aimé, choyé, se souvient-il. C'est le seul endroit où j'ai ressenti ça, jusqu'à mon départ pour l'université. »

Mme Lippman considère le décès des grands-parents de Richard comme le deuxième événement traumatisant. « Il en a été

5. Malheureusement, je n'ai pu interviewer Daniel Stallman pour cet ouvrage. Pendant mes recherches préliminaires, 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 qu'il était décédé plus tôt cette année là.

bouleversé, dit-elle. Il était très proche d'eux. Avant leur disparition, il était très extraverti, presqu'un meneur pour les autres enfants. Après leur mort, il est devenu très renfermé. »

Du point de vue de Stallman, ce repli sur soi était une tentative pour affronter l'angoisse de l'adolescence. Qualifiant ces années de « pure horreur », Stallman raconte s'être senti comme un sourd perdu au milieu d'une bruyante foule de mélomanes.

« J'avais souvent le sentiment de ne pas comprendre ce que les autres disaient, poursuit-il, se souvenant de son sentiment d'exclusion. Je comprenais les mots, mais quelque chose se passait en sous-main, qui m'échappait. Je n'arrivais pas à saisir pourquoi les gens s'intéressaient à ces choses que d'autres racontaient. »

Malgré toute la souffrance engendrée, l'adolescence aura stimulé Stallman dans l'affirmation de son individualité. À une époque où ses collègues de classe laissaient pousser leurs cheveux, Stallman préférait les siens courts. Quand tout le monde se mettait au rock'n roll, Stallman écoutait de la musique classique. Fervent amateur de science-fiction, de la revue *Mad* et des programmes de nuit, Stallman avait une personnalité atypique, qui plongeait dans l'incompréhension tant ses camarades que ses parents.

« Ah les jeux de mots ! s'exclame Mme Lippman, exaspérée au souvenir de l'âge ingrat de son fils. À table, vous ne pouviez pas dire un mot sans qu'il vous le retourne sous forme de calembour. »

⁂

Hors du foyer familial, Stallman réserve ses blagues aux adultes encourageant son don naturel. L'un d'eux était moniteur au camp d'été. Il prêta à Stallman un manuel pour l'IBM 7094, au cours

de sa huitième ou neuvième année. Pour un gamin fasciné par les sciences et les chiffres, c'était un don du ciel[6]. Bientôt, Stallman se lança dans la rédaction, sur papier, de programmes basés sur instructions du 7094. Il n'avait pas d'ordinateur à portée de main pour les exécuter, ni la possibilité de demander à en utiliser un, mais il brûlait d'envie d'écrire un programme — n'importe quel programme. Il demanda même au moniteur de lui suggérer au hasard quelque chose à coder.

Le premier ordinateur personnel ne devait apparaître qu'une décennie plus tard, Stallman avait encore quelques années à patienter avant d'accéder à sa première machine. La chance lui sourit lors de sa dernière année de lycée. Le Centre scientifique IBM de New York, un pôle de recherche situé en plein Manhattan, et aujourd'hui disparu, allait lui offrir la chance d'écrire son premier vrai programme. L'idée qu'il avait en tête était d'écrire un préprocesseur rajoutant la convention de sommation en algèbre tensoriel, aux fonctionnalités du langage de programmation PL/I[7]. « Je l'avais d'abord écrit en PL/I, puis j'ai recommencé en langage d'assembleur, après m'être aperçu que le programme PL/I compilé était trop lourd pour l'ordinateur », se souvient-il.

Après l'obtention de son diplôme de dernière année, il fut engagé par le New York Scientific Center. Chargé d'écrire un programme d'analyse numérique en Fortran, il le boucla en quelques semaines, acquérant au passage un tel dégoût pour ce langage qu'il jura de ne plus jamais l'utiliser. Il passa le reste de l'été à concevoir un éditeur de texte en APL (A *Programming Language*).

Simultanément, Stallman obtint un poste d'assistant de laboratoire au département de biologie de l'université Rockefeller.

6. Athée, Stallman ergoterait probablement sur cette expression (*godsend* en anglais). En tout cas il l'accueillit avec enthousiasme. « Aussitôt que j'ai entendu parler d'ordinateurs, j'ai voulu en voir un, et pouvoir jouer avec. »

7. Le PL/I (*Programming Language number 1*) est un langage de programmation développé par IBM au début des années 1960.

Bien qu'il s'acheminât vers une carrière liée aux mathématiques ou à la physique, son esprit analytique impressionna le directeur du laboratoire au point qu'il appela Mme Lippman quelques années plus tard. Stallman était alors déjà parti pour l'université. « 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 aurait un grand avenir en tant que biologiste. »

Des aptitudes qui impressionnèrent tout autant les membres de la faculté de Columbia, même lorsque Stallman devint la cible de leur colère. « Généralement, deux ou trois fois par heure, Stallman relevait une erreur dans le cours, raconte Breidbart. Et il ne se gênait pas pour le faire savoir au professeur. Cela lui valut beaucoup de respect, mais une faible popularité. »

Entendant une nouvelle fois l'anecdote de Breidbart, Stallman esquisse un sourire désabusé. « J'ai dû être un peu crétin parfois, admet-il. Mais j'ai trouvé des amis en certains professeurs qui, eux aussi, aimaient apprendre. Contrairement à la plupart des jeunes. Ou tout du moins pas de la même façon. »

⁂

La fréquentation de jeunes surdoués le samedi devait néanmoins le pousser à reconsidérer les mérites d'une sociabilité accrue. L'université approchant à grands pas, Stallman, comme beaucoup au Columbia Science Honors Program, réduisit sa liste de demandes d'admission à deux institutions : Harvard et le MIT.

Face au désir de son fils d'entrer dans une université prestigieuse du nord-est, Mme Lippman se montra inquiète. Le lycéen de quinze ans avait encore de vives prises de bec avec les professeurs et l'encadrement. L'année d'avant, il avait obtenu des A en

histoire américaine, en chimie, en français et en algèbre, mais un F retentissant en anglais, conséquence de son boycott continu des devoirs écrits. De tels couacs donneraient lieu à quelques sourires entendus au MIT, mais ne seraient pas tolérés à Harvard.

Lorsque son fils fut en première, Mme Lippman prit rendez-vous avec un thérapeute. Ce dernier fit tout de suite part de son inquiétude face au refus de Stallman d'écrire et à ses démêlés avec les professeurs. Le jeune homme avait certainement les aptitudes intellectuelles pour réussir à Harvard, mais avait-il la patience de suivre des cours qui exigeaient la remise de devoirs réguliers ? Le thérapeute suggéra un galop d'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 faire de même à Harvard.

Sa classe de première achevée, Stallman alla s'inscrire à un cours public d'été, dans le centre ville, et se mit au rattrapage des cours en sciences humaines qu'il avait boudés au lycée. À l'automne, il avait rejoint le gros des lycéens new-yorkais, à l'école Louis D. Brandeis, située dans la 84e Rue. Il ne lui fut pas facile de suivre des cours faisant pâle figure face aux études du samedi à Columbia, mais Mme Lippman se souvient fièrement de la manière dont son fils sut tenir le cap.

« Il a dû 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 tenait du miracle. C'était le professeur de mathématiques. Richard avait interrompu sa leçon. Comme je lui demandais comment il s'y était pris, il me répondit que mon fils l'avait accusé d'utiliser une fausse preuve. 'D'accord, mais avait-il raison ?' ai-je demandé. L'enseignant me répondit : 'Oui, mais je ne peux pas le dire en classe. Les autres élèves ne comprendraient pas'. »

À la fin de son premier semestre au lycée, les choses se mirent en place. Un score de 96 en anglais fit oublier les stigmates du 60

obtenu deux ans auparavant. Pour faire bonne mesure, Stallman confirma avec d'excellentes notes en histoire américaine, mathématiques avancées, et microbiologie. La cerise sur le gâteau fut une note de 100 en physique. Toujours exclu socialement, Stallman termina ses dix mois à Brandeis à la 4e place d'un cours de 789 élèves.

***

En dehors des cours, Stallman poursuivait ses études avec encore plus de diligence, courant remplir ses fonctions d'assistant laborantin à l'université Rockefeller la semaine, et esquivant les manifestations contre la guerre du Vietnam pour aller à Columbia le samedi. C'est là-bas, alors que les autres étudiants du Science Honors Program étaient assis pour discuter de leurs choix d'université juste avant le début de la classe, que Stallman énonça le sien.

Breidbart s'en souvient : « La plupart des étudiants choisissaient Harvard ou le MIT, bien sûr, mais quelques-uns se tournaient vers les autres universités du nord-est des États-Unis. 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 a eu le courage de lui demander ce qu'il pensait faire. »

Trente ans plus tard, Breidbart garde un clair souvenir de cet instant. Dès que Stallman annonça qu'il irait lui aussi à Harvard à l'automne, un silence embarrassé envahit la salle. Pour toute réplique, Stallman esquissa un sourire d'autosatisfaction.

Breidbart de conclure : « C'était sa façon silencieuse de dire : 'Eh non. Vous n'êtes pas encore débarrassés de moi.' »

Chapitre 4

Destituer Dieu

Malgré leur relation tendue, Richard Stallman hérita de sa mère une même passion pour la politique progressiste. Ce trait de caractère marquant mit toutefois quelques décennies à émerger et au cours des premières années de sa vie, il admet avoir vécu dans un « vide politique » [1].

Comme la plupart des Américains sous Eisenhower, la famille Stallman avait passé les années 1950 à tenter de retrouver une normalité perdue lors de la Deuxième Guerre mondiale.

1. Voir l'interview par Michael Gross, *« Richard Stallman : High School Misfit, Symbol of Free Software, MacArthur-certified Genius »* (1999).

« Le père de Richard et moi-même étions des démocrates, mais ça s'arrêtait là, raconte madame Lippman, se rappelant leurs années dans le Queens. Nous n'étions guère impliqués dans la politique locale ou nationale. »

Tout changea à la fin des années 1950 lorsqu'Alice divorça de Daniel Stallman. Le retour à Manhattan fut plus qu'un simple changement d'adresse : ce fut une nouvelle identité, une nouvelle indépendance, et la fin brutale de la tranquillité.

« C'est là que mon goût pour l'activisme est né. Lorsque je me suis rendue à la bibliothèque municipale, alors que nous étions encore dans le Queens, j'ai constaté qu'il n'y avait qu'un seul livre sur le divorce, se rappelle-t-elle. La bibliothèque était très encadré par l'Église catholique, du moins à Elmhurst, où nous vivions. C'est sans doute la première fois que j'entrevoyais les forces qui contrôlaient nos vies subrepticement. »

De retour dans les quartiers de son enfance, au nord-ouest de Manhattan, Mme Lippman fut surprise des changements survenus depuis son départ vers le Hunter College, quinze ans plus tôt. Après la guerre, l'explosion de la demande en logements avait transformé le quartier en champ de bataille politique. D'un bord se tenaient les politiciens municipaux pro-développement et les affairistes prêts à démolir et transformer en bureaux de nombreux immeubles du quartier pour faire face à l'afflux d'employés en col blanc. De l'autre, les locataires irlandais et porto-ricains pauvres qui s'étaient trouvé un refuge abordable dans le quartier.

Au début, Mme Lippman ne savait quel camp choisir. En tant que nouvelle arrivante, elle se sentait attirée par un logement neuf ; mais en tant que mère célibataire aux faibles revenus, elle partageait les mêmes difficultés que les plus pauvres des locataires laissés pour compte d'un nombre croissant de projets destinés aux

résidents les plus riches. Indignée, elle chercha un moyen de combattre cette machine politique qui essayait de transformer son voisinage en clone du quartier chic de New York Upper East Side.

Son premier contact avec le siège local du parti démocrate eut lieu en 1958, raconte-t-elle. À la recherche d'une garderie pour son fils, elle fut scandalisée par l'état de l'un des centres municipaux destiné aux citoyens à faibles revenus. « Tout ce dont je me souviens, c'est de l'odeur de lait tourné, des couloirs sombres, et de la pénurie d'équipements. J'avais déjà été enseignante dans une école maternelle privée. Le contraste était énorme. Nous n'avons jeté qu'un regard dans la salle, et nous sommes partis. J'étais bouleversée. »

Sa visite à l'antenne du parti devait se révéler tout aussi décevante. La décrivant comme « l'éternelle salle enfumée », Mme Lippman raconte y avoir, pour la première fois, pris conscience que la corruption au sein du parti était sans doute à l'origine de l'hostilité à peine voilée de la ville envers les citoyens les moins nantis. Au lieu de retourner à l'antenne du parti, elle décida de rejoindre l'un des multiples clubs dont l'objectif était de réformer le parti démocrate et de renverser les derniers vestiges de la machinerie clientéliste du Tammany Hall. Mme Lippman et son club, le Woodrow Wilson / *FDR Reform Democratic Club*, commencèrent à se présenter aux réunions de planification ainsi qu'au conseil municipal, comptant bien y être entendus.

« Notre but principal était de combattre le Tammany Hall, Carmine DeSapio et son acolyte [2], poursuit Mme Lippman. J'étais la représentante au conseil municipal et je me suis impliquée dans la création d'un plan de renouvellement urbain viable, qui ne se

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 [Davenport, 1975].

contentait pas de construire des demeures luxueuses dans le quartier. »

Ce premier engagement allait s'épanouir et mener à une plus grande activité politique durant les années 1960. En 1965, Mme Lippman affichait son soutien à des 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'oppose à l'intervention des États-Unis en Indochine. « J'étais contre la guerre du Vietnam, déjà au moment de l'envoi des troupes par Kennedy, dit-elle. J'avais lu les reportages de journalistes qui racontaient les débuts du conflit. J'étais sûre qu'ils avaient raison de prédire l'enlisement. »

L'opposition à la guerre du Vietnam fit bientôt partie du quotidien du foyer Stallman-Lippman. En 1967, la mère de Richard s'était remariée. Son nouvel époux, Maurice Lippman, alors commandant de l'Air National Guard, démissionna de ses fonctions pour marquer son opposition à la guerre. Son fils, Andrew Lippman, était au MIT et donc encore admissible au sursis étudiant. Mais cette mesure allait bientôt disparaître, et rendait l'aggravation possible du conflit plus menaçante que jamais. Richard, pourtant plus jeune, risquait lui aussi d'être appelé alors que la guerre perdurait jusque dans les années 1970.

« La guerre du Vietnam était pour nous un sujet d'inquiétude constante, raconte Mme Lippman. Nous en parlions tout le temps : que ferions-nous si la guerre continuait ? Que feraient Richard et son demi-frère s'ils étaient appelés ? Nous étions tous opposés à la guerre et à l'incorporation. Nous pensions que cela était tout à fait amoral. »

Chez Richard, la guerre du Vietnam éveillait un mélange complexe d'émotions : de la confusion, de l'horreur et, en fin de compte, un sentiment profond d'impuissance politique. Enfant, il avait déjà eu du mal à tenir le coup dans l'univers modérément autoritaire de l'école privée. Il ne supportait pas l'idée du camp d'entraînement militaire ; il ne pensait pas pouvoir en ressortir sain d'esprit.

« J'étais tétanisé par la peur, j'aurais voulu faire quelque chose mais je n'avais pas le courage d'aller manifester », se souvient Stallman, dont la date d'anniversaire, le 16 mars, était dans les premières tirées au sort à la loterie tant redoutée de l'incorporation. Cela ne l'affecta pas immédiatement, car il bénéficiait d'un report automatique pour sa scolarité à l'université — l'un des derniers accordés par le gouvernement — mais le répit ne serait que temporaire. « Je ne m'imaginais pas déménager au Canada ou en Suède. L'idée de me lever comme ça et de partir seul me paraissait irréaliste ! J'étais incapable de vivre seul. Je n'étais pas de ceux qui se sentaient à l'aise face à ce genre de chose. »

Stallman dit avoir été impressionné par ceux de ses proches qui osaient s'exprimer. Il se souvient de la fierté qu'il avait éprouvée quand son père imprima et distribua un autocollant qui comparait le massacre de My Lai avec les atrocités commises par les nazis durant la Deuxième Guerre mondiale. « Je l'admirais de l'avoir fait, dit-il, mais je ne m'imaginais pas moi-même capable de quoi que ce soit de cet ordre. J'avais peur de ressortir détruit de l'armée. »

Toutefois, Stallman fut vite déçu par le ton et l'orientation que prit le mouvement de contestation. Comme pour d'autres membres du Science Honors Program [3], il voyait surtout les manifestations de fin de semaine à Columbia comme un spectacle distrayant [4].

3. Cf. chapitre 3 — NdT.

4. Chess, un autre élève du programme SHP, décrit ces manifestations comme un « bruit de fond ». « Nous étions tous politisés », dit-il, « mais les cours du SHP étaient bien plus importants. Jamais nous n'aurions séché pour aller à une manifestation. »

Au bout du compte, il y avait pour lui autant d'irrationnel dans les forces qui menaient le mouvement contre la guerre que dans la culture des jeunes, et ces deux mouvements en devenaient indissociables. 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 homme comme Stallman, qui peinait à comprendre ses pairs adolescents, il y avait de l'ironie dans des slogans tels que « faites l'amour, pas la guerre ». Stallman ne voulait pas faire la guerre, en tout cas pas en Asie du Sud-Est et pour autant, personne ne lui demandait de faire l'amour…

« Je n'aimais pas la contre-culture, se rappelle-t-il. Je n'aimais ni leur musique ni leurs drogues — dont j'avais peur. En particulier, je n'aimais pas leur anti-intellectualisme, ni leurs préjugés contre la technologie. Après tout, j'aimais les ordinateurs. Je n'aimais pas non plus l'anti-américanisme primaire que je rencontrais souvent. Il y avait des gens dont la pensée était si simpliste que s'ils désapprouvaient la conduite des États-Unis dans la guerre du Vietnam, ils devaient forcément soutenir le Nord-Vietnam. J'imagine qu'il leur était impossible d'imaginer une position plus subtile. »

De tels commentaires soulignent un trait essentiel pour la maturation politique de Stallman. Pour lui, l'assurance politique était directement liée à sa confiance en soi. Jusqu'en 1970, il se sentait peu sûr de lui hors du domaine des mathématiques et des sciences. Mais c'est précisément cette assurance toute mathématique qui lui permit d'analyser, en termes purement logiques, l'extrémisme du mouvement contre la guerre, et de lui opposer une faille de raisonnement. Bien qu'opposé à celle du Vietnam, Stallman ne voyait en effet aucune raison de désavouer la guerre elle-même comme moyen de défense de la liberté ou de lutte contre les injustices.

Dans les années 1980, un Stallman plus confiant décida de rattraper sa passivité d'autrefois en participant à des rassemblements

en faveur du droit à l'avortement, à Washington DC. « J'étais mécontent d'avoir manqué à mon devoir de protestation contre la guerre du Vietnam », explique-t-il.

⁂

En 1970, lorsqu'il partit pour Harvard, Stallman laissa derrière lui les conversations quotidiennes à la table du dîner sur la politique ou la guerre. Rétrospectivement, il décrit le passage de l'appartement maternel de Manhattan au dortoir de Cambridge comme une « évasion ». Il pouvait désormais se réfugier dans sa chambre à tout moment et avoir la paix. Une vision fort différente de celle de certains de ses camarades qui, l'ayant observé, avaient vu peu de signes suggérant une expérience libératrice.

Dan Chess, l'ancien camarade de classe du SHP également admis à Harvard, s'en souvient : « Il semblait plutôt malheureux les premiers temps. On voyait bien que l'interaction humaine lui posait de réelles difficultés, or elle était inévitable à Harvard. C'était un endroit intensément social. »

Pour faciliter la transition, Stallman se repliait sur ses domaines d'excellence : les mathématiques et les sciences. Comme les autres anciens élèves du SHP, il réussit aisément l'examen de qualification à Math 55, le légendaire cours de type « camp d'entraînement » pour les nouveaux étudiants en majeure de mathématiques à Harvard.

Dans cette classe, ceux qui venaient du SHP formaient une équipe soudée. « Nous étions la mafia des maths, relate Chess en riant. Harvard n'était rien, comparé au SHP ». Et pour obtenir le droit de se vanter, Stallman, Chess et les autres devaient triompher de ce cours qui promettait l'équivalent de quatre ans

de mathématiques en deux semestres, et qui favorisait les vrais passionnés.

« C'était un cours extraordinaire », raconte David Harbater, ancien membre de la « mafia des maths », aujourd'hui professeur de mathématiques à l'université de Pennsylvanie. « On peut affirmer sans crainte qu'il n'y a jamais eu de cours d'entrée d'université aussi intensif et avancé. Pour que les gens s'en rendent compte, je précise d'habitude que dès le deuxième semestre, entre autres choses, nous discutions la géométrie différentielle des espaces de Banach. C'est là que les yeux s'écarquillent, car la plupart des gens ne commencent à en parler qu'en troisième cycle. »

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. « Le dernier, conclut Harbater, était Richard Stallman. »

Seth Breidbart, lui aussi vétéran du SHP et de Math 55, se souvient que même alors, Stallman se distinguait de ses collègues : « Il était étrangement pointilleux, poursuit Breidbart. En mathématiques, il y a une technique standard que tout le monde fait de travers. C'est un abus de notation où vous devez définir une fonction, et ce que vous faites, c'est la définir et ensuite prouver qu'elle est bien définie. Sauf que la première fois qu'il l'a faite et présentée, il a défini une relation et prouvé ensuite que c'était une fonction. C'est exactement la même preuve, mais il a utilisé la bonne terminologie, ce que personne d'autre ne faisait. Voilà, c'était Richard tout craché. »

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, mit du temps à l'accepter. Il dit n'avoir

réalisé l'éventualité que Stallman soit le meilleur mathématicien de la classe que l'année suivante. Chess, aujourd'hui professeur de mathématiques à Hunter College, s'en souvient : « C'était pendant le cours d'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 d'une manière originale et brillante à la fois. »

Ce fut pour Chess un moment troublant. Tel un oiseau heurtant une fenêtre en plein vol, il allait lui falloir quelque temps pour réaliser que certains niveaux d'intuition étaient tout simplement hors de sa portée.

« C'est ainsi avec les mathématiques, reprend Chess. Vous n'avez pas besoin d'être un mathématicien de haut niveau pour reconnaître un grand talent mathématique. Je savais que j'étais un bon mathématicien, mais je pouvais aussi voir que je n'occupais pas le premier rang. Si Richard l'avait voulu, il serait devenu un mathématicien hors pair. » [5]

***

Le succès académique de Stallman était à la mesure de son échec dans l'arène sociale. Même lorsque les autres membres de la mafia des maths se rassemblaient pour s'attaquer aux problèmes, il préférait travailler seul. Il en allait de même pour son quotidien.

5. Stallman émet des doutes quant à cette affirmation : « L'une des raisons qui m'ont fait quitter les mathématiques et la physique pour la programmation, est que je n'ai jamais réussi à découvrir quelque chose de nouveau dans les deux premières, je ne faisais qu'apprendre ce que d'autres avaient fait. En programmation, je pouvais créer quelque chose d'utile chaque jour. »

Ainsi dans sa demande d'hébergement à Harvard, Stallman avait-il été clair sur ses préférences. « J'avais écrit que je souhaitais un camarade de chambre invisible, inaudible et intangible », dit-il. Par un rare accès de lucidité bureaucratique, l'administration de Harvard accepta sa demande en lui octroyant une chambre individuelle au cours de sa première année.

Breidbart, le seul de la mafia des maths à partager le même bâtiment que Stallman cette année-là, raconte comment ce dernier apprit lentement mais sûrement à interagir avec les autres. Il se souvient que les autres étudiants du dortoir, impressionnés par sa logique imparable, commençaient à solliciter ses interventions lorsqu'un débat intellectuel faisait rage dans la salle à manger ou les pièces communes.

« Nous avions ces discussions à bâtons rompus où nous refaisions le monde et imaginions les conséquences d'un événement, 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 est vite 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 créez une inégalité de classe. Dans ce genre de discussions, Richard était simplement le meilleur pour voir les conséquences insoupçonnées de toute décision. »

Stallman se rappelle fort bien ces discussions. « J'étais toujours en faveur de l'immortalité, dit-il. De quelle autre manière pourrions-nous voir ce que sera le monde dans deux cents ans ? » Il commença d'ailleurs par curiosité à demander autour de lui ce que chacun répondrait si l'immortalité lui était offerte. « J'ai été choqué de constater que la plupart considéraient l'immortalité comme une mauvaise chose, dit-il. Beaucoup disaient que la mort était une bonne chose car il n'y a pas d'intérêt à vivre en état de décrépitude mais, d'un autre côté, les mêmes considéraient que le

vieillissement est positif car il prépare à la mort. Aucun pourtant ne reconnaissait le caractère circulaire de ce raisonnement. »

⁂

Malgré sa réputation de mathématicien et de rhéteur de premier ordre, Stallman se tenait à l'écart des concours qui auraient définitivement confirmé son génie. Vers la fin de la première année à Harvard, Breidbart se souvient comment son condisciple évita ostensiblement l'examen Putnam, une prestigieuse épreuve de mathématiques ouverte aux étudiants américains et canadiens. En plus de leur donner une occasion de se mesurer à leurs pairs, Putnam était le premier instrument de recrutement dans les départements académiques de mathématiques. Selon une légende qui courait sur le campus, le meilleur score vous qualifiait automatiquement pour l'obtention d'une bourse universitaire à l'école de votre choix, Harvard inclus.

Comme le cours de Math 55, l'épreuve Putnam était impitoyable. D'une durée de six heures, en deux volets, elle semblait clairement conçue pour séparer le bon grain de l'ivraie. Breidbart la décrit comme étant de loin la plus difficile à laquelle il ait participé. « Pour vous donner une idée de la difficulté, commence Breidbart, la note maximale était de 120, et ma note la première année était dans les 30. Et encore, cette note était suffisamment bonne pour me classer 101^{e} à l'échelle du pays. »

Étonné que Stallman, le meilleur étudiant de la classe, ait évité ce test, Breidbart raconte comment, avec un collègue de classe, ils le coincèrent dans la salle à manger commune pour lui demander des explications. « Il disait qu'il avait peur de ne pas bien réussir », se souvient-il. Breidbart et son ami recopièrent de mémoire sur un bout de papier quelques-uns des problèmes posés. « Il les a tous

résolus, rapporte Breidbart, et j'en étais arrivé à la conclusion que 'ne pas bien réussir' signifiait pour lui finir deuxième ou se tromper quelque part. »

Stallman se souvient de cet épisode un peu différemment. « Je me rappelle 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 confirme ce que disait Breidbart : c'est bien par peur qu'il n'avait pas passé le test. Malgré un empressement notoire à signaler les faiblesses intellectuelles de ses pairs et professeurs en classe, Stallman haïssait et craignait la compétition directe — pourquoi alors ne pas tout simplement l'éviter ?

« C'est pour cette 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 (et de perdre) que j'en faisais des bêtises très tôt dans la partie. La crainte devenait une prophétie se réalisant d'elle-même ». Stallman se tenait donc à l'écart des échecs.

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, d'autres centres d'intérêt allaient l'éloigner de ce domaine. La programmation informatique, objet d'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 se ressourçait dans le laboratoire de sciences informatiques.

Son premier contact réel avec la programmation informatique au centre scientifique d'IBM à New York avait éveillé en lui le 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 demandai s'ils avaient des copies supplémentaires de manuels que je pourrais lire. » Emportant ces manuels chez lui, Stallman examina les spécifications des machines afin d'en apprendre davantage sur les différents modèles d'ordinateurs.

***

Un jour, vers la fin de sa première année universitaire, il entendit parler d'un laboratoire spécialisé près du MIT. Celui-ci était situé au neuvième étage d'un immeuble du Tech Square : un ensemble de bureaux essentiellement commercial que le MIT avait construit en face du campus. Selon les rumeurs, le laboratoire se consacrait à l'intelligence artificielle, une science de pointe, et s'enorgueillissait de son lot de programmes informatiques et de machines ultramodernes. Intrigué, Stallman décida de s'y rendre.

Le trajet était court, environ trois kilomètres à pied, dix minutes en train, mais comme il allait bientôt le découvrir, le MIT et Harvard peuvent donner l'impression d'être les pôles opposés d'une même planète. Avec ses connexions labyrinthiques entre édifices, le campus de l'institut présentait une architecture complexe contrastant avec le spacieux village colonial de Harvard.

Des deux, les méandres du MIT étaient plus du goût de Stallman, comme l'était d'ailleurs le corps étudiant : une collection de « geeks » et d'anciens lycéens inadaptés, plus connus pour leur prédilection pour les canulars.

Le AI Lab, laboratoire d'intelligence artificielle du MIT, contrastait tout autant avec les laboratoires informatiques de Harvard. Nul gardien ni liste d'attente pour l'accès aux terminaux, et

nulle atmosphère feutrée semblant dire : « 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. Lorsqu'il rencontra un employé du laboratoire, il demanda 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 », ajoute-t-il amusé, en référence à cette tendance qu'ont les hackers de passer à de nouveaux projets sans prendre le temps de documenter les anciens.

Stallman repartit avec bien plus qu'un manuel : un emploi. Son premier projet consistait à écrire un simulateur de PDP-11 pouvant tourner sur un PDP-10. Il revint au AI Lab la semaine suivante, s'accapara un terminal disponible et commença à écrire le code.

Rétrospectivement, Stallman ne note rien d'inhabituel à la bonne volonté dont fit preuve le AI Lab en acceptant 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 volontiers quelqu'un quand je le rencontre et que 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 avant-goût de ce qu'il appelait la « bureaucratie étouffante », Stallman n'avait eu qu'à visiter les laboratoires informatiques de Harvard. L'accès aux terminaux y était attribué au compte-gouttes selon le rang académique. En tant qu'étudiant de premier cycle, il devait parfois attendre jusqu'à quatre heures.

Une formalité supportable en soi, mais frustrante dans son principe. Faire le pied de grue pour 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'absurde. Même si Stallman continuait à se rendre occasionnellement dans les labos informatiques de Harvard, il préférait la politique plus égalitaire du AI Lab au MIT. « C'était une bouffée d'oxygène, 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 due aux efforts de quelques personnes vigilantes. La plupart d'entre elles venaient du projet MAC [6], le programme de recherche subventionné par le Département de la Défense qui avait donné naissance au tout premier système d'exploitation à temps partagé. D'autres étaient déjà des légendes dans le monde de l'informatique, à commencer par 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 aussi Gerald Sussman, l'auteur du programme robotique *HACKER* pour la résolution de problèmes et la planification. Et il y avait Bill Gosper, le

6. En 1964, l'équipe du projet MAC, ayant déjà développé CTSS, donna naissance à Multics (Multiplexed Information and Computing Service). La grande leçon de ce dernier fut de démontrer qu'il était possible de sécuriser des données et d'optimiser l'utilisation du temps partagé par un système révolutionnaire de mémoire segmentée. Ken Thompson et Dennis M. Ritchie, qui travaillaient pour Bell sur le projet Multics, développèrent alors une version nommée Unics, qui deviendra Unix. Elle reprenait les mêmes principes : une programmation en langage de haut niveau (PL/I pour Multics, C pour Unix), la même hiérarchie de fichiers, un interpréteur de commande (terminal), ainsi que l'exécution des processus distincts entre eux et hiérarchisés — NdT.

génie de la maison en mathématiques, alors plongé dans un marathon de programmation qui dura dix-huit mois, motivé par les implications philosophiques du jeu *LIFE* (« jeu de la vie »)[7].

Les membres de ce groupe soudé se disaient « hackers ». Avec le temps, ils étendirent cet attribut à Stallman et lui inculquèrent la déontologie traditionnelle de « l'éthique hacker ». Dans leur soif d'explorer les limites de ce qu'ils pouvaient faire faire à un ordinateur, les hackers pouvaient rester assis devant un terminal trente-six heures durant si le défi leur semblait en valoir la chandelle. Cela signifiait avoir accès à tout moment à un ordinateur (si personne d'autre ne l'utilisait) ainsi qu'aux informations qui permettaient de l'utiliser.

Les hackers parlaient ouvertement de changer le monde par les logiciels, et Stallman hérita ce dédain instinctif du hacker pour tout obstacle empêchant la réalisation de cette noble cause. Les trois principales barrières identifiées étaient les logiciels de piètre qualité, la bureaucratie universitaire et l'égoïsme.

Stallman devint familier du folklore des hackers et des histoires où s'illustrait leur créativité, notamment pour contourner les obstacles. Cela incluait les nombreuses techniques de « libération » des terminaux séquestrés dans les bureaux des professeurs. Contrairement à leurs homologues gâtés de Harvard, les professeurs du MIT disposaient d'un nombre limité de terminaux et avaient la sagesse de ne pas en disposer comme d'une propriété privée. Si quelqu'un enfermait par erreur un terminal la nuit, les hackers étaient prompts à en « libérer » l'accès — et à faire quelque remontrance à celui qui avait ainsi lésé la communauté. Certains se livraient ainsi au crochetage de serrure (le « hack de verrou »),

7. Voir [Levy, 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 vivement ce livre comme complément, voire pré-requis, à celui-ci.

d'autres en ôtaient quelques dalles du plafond et escaladaient le mur. Au neuvième étage, certains faisaient de la spéléologie dans les faux plafonds où passait le câblage informatique. « On m'avait même montré un petit chariot muni d'un lourd cylindre de métal qui avait servi à enfoncer la porte d'un des bureaux des professeurs », raconte Stallman. Cela dit, Gerald Sussman, membre du MIT et hacker qui travailla au AI Lab avant Stallman, conteste cette histoire. Selon lui, les hackers n'enfonçaient pas de portes.

Avec obstination, ils veillaient à ce que le travail au laboratoire ne soit pas gêné par les demandes des professeurs. Ils tenaient compte des besoins de chacun en insistant pour que cela ne gêne personne d'autre. Par exemple, si un professeur souhaitait protéger du vol l'un de ses objets, il s'entendait répondre : « Personne ne fera d'objection à ce que vous fermiez votre bureau à clef — quoique cela manque de convivialité — tant que vous n'y enfermez pas le terminal du laboratoire. »

Même si, au AI Lab, la population universitaire surpassait largement en nombre celle des hackers, ces derniers faisaient prévaloir leur éthique. Et pour cause : n'étaient-ils pas en effet ces employés et étudiants qui avaient conçu et construit les pièces d'ordinateurs, et écrit quasi tous les programmes utilisés au laboratoire ? N'étaient-ils pas ceux qui faisaient tourner l'ensemble ?

Leur travail était essentiel et ils refusaient la moindre pression. Travaillant aussi bien sur des projets personnels qu'à la demande des utilisateurs, ils passaient le plus clair de leur temps à optimiser toujours plus les machines et les logiciels — y compris dans le cadre de leurs propres projets. Comme ces jeunes fous de mécanique automobile (*hot-rodders*), ils voyaient dans le bricolage de ces machines une fin en soi.

***

Le système d'exploitation pilotant l'ordinateur central PDP-10 du labo était l'expression la plus manifeste de cette passion. Conçu et nommé en réaction au CTSS [8], le système originel du projet MAC, l'ITS, abréviation de *Incompatible Time Sharing System*, intégrait l'éthique hacker dans sa conception même.

À cette époque, le CTSS était jugé trop restrictif dans sa conception, car il limitait la possibilité pour le programmeur de modifier et d'améliorer, si nécessaire, l'architecture interne du logiciel.

Selon la légende, la création de l'ITS avait aussi une visée politique. Alors que le CTSS était conçu pour l'IBM 7094, l'ITS était compilé spécifiquement pour le PDP-6. En laissant les hackers programmer le système, l'administration du AI Lab garantissait qu'ils seraient seuls à pouvoir l'utiliser aisément. Dans le monde de la recherche universitaire, féodal sous certains aspects, la manœuvre fut une réussite. Bien que le PDP-6 fût la propriété commune de plusieurs départements, les chercheurs du AI Lab furent bientôt les seuls à l'exploiter. Avec l'ITS sur le PDP-6, le laboratoire put afficher son indépendance vis-à-vis du projet MAC juste avant l'arrivée de Stallman.

En 1971, l'ITS fut migré sur le PDP-10, nouveau mais compatible, laissant au PDP-6 certaines applications autonomes annexes. Le PDP-10 avait une très grande capacité de mémoire pour l'époque, l'équivalent d'un méga-octet, ce qui fut doublé à la fin des années 1970. Du côté du projet MAC, on avait acheté deux autres PDP-10 et installé ces derniers au neuvième étage. L'ITS tournait sur les trois machines.

Ces machines furent en outre améliorées par les hackers spécialisés dans le matériel informatique. Ils implémentèrent notamment

8. Abréviation de *Compatible Timesharing System*.

le système de mémoire virtuelle paginée — qui allait permettre au PDP-10 de s'affranchir de certaines limitations matérielles de mémoire [9].

En tant qu'apprenti hacker, Stallman s'enticha rapidement de l'ITS. Bien qu'inaccessible au commun des non-hackers, le système affichait insolemment des caractéristiques que la plupart de ses équivalents commerciaux n'offriraient pas avant au moins une décennie : le multitâche, le lancement du débogueur dès l'exécution des programmes, ou encore l'édition en mode multiligne plein-écran.

« L'ITS avait un mécanisme interne très élégant permettant à un programme d'en examiner un autre, se souvient Stallman. Vous pouviez analyser tous les états d'un autre programme d'une manière propre et bien détaillée ». Des possibilités non seulement très pratiques pour le débogage, mais qui permettaient aussi à des programmes d'en contrôler, démarrer ou stopper d'autres.

Autre fonction appréciée, la possibilité pour un programme de geler, entre deux instructions, la tâche d'un autre. Sur d'autres systèmes d'exploitation, une opération semblable aurait pu provoquer un arrêt du programme au beau milieu d'un appel système, avec un état interne impossible à connaître pour l'utilisateur et dénué d'une signification déterminée. Grâce à l'ITS, le contrôle des opérations étape par étape devenait fiable et cohérent.

« 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 'reprends le travail', il continuait

9. Il faut excuser la brièveté de ce résumé de la genèse de l'ITS, un système d'exploitation que bien des hackers considèrent encore comme le parangon de l'éthique hacker. Pour en savoir plus sur la signification politique du logiciel, voyez [Garfinkel and Abelson, 1999].

proprement. En plus, si vous aviez à changer l'état (explicitement visible) de la tâche avant de la relancer, tout restait cohérent. Il n'y avait d'état caché nulle part. »

⁂

Dès septembre 1971, le *hacking* au AI Lab du MIT était devenu une activité régulière dans l'agenda hebdomadaire de Stallman. Du dimanche au vendredi, il était à Harvard ; dès le vendredi après-midi, il prenait le métro vers le MIT pour y passer le week-end. Il s'arrangeait pour être sûr d'arriver avant le départ de la traditionnelle excursion du dîner. Rejoignant cinq ou six autres hackers dans leur quête nocturne de nourriture chinoise, il sautait à bord d'une vieille voiture pour passer le pont de Harvard en direction de Boston, non loin 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 de son système d'idéogrammes. Après le dîner, le groupe s'en retournait au MIT et programmait jusqu'à l'aurore, ou décidait de retourner à Chinatown vers trois heures du matin.

Stallman passait parfois toute la matinée du samedi à programmer ou dormir sur un canapé. Au réveil, il se remettait au travail ou retournait à un restaurant chinois, avant de rentrer à Harvard. Il lui arrivait de rester jusqu'au dimanche. Ces délicieux repas lui permettaient de pallier la médiocrité de ceux servis aux cantines de Harvard, où il ne trouvait en moyenne comestible qu'un repas par jour — il n'y appréciait pas les petits déjeuners, de surcroît à un moment de la journée où il sommeillait.

Pour l'excentrique marginal fréquentant peu ses condisciples, c'était une expérience grisante que de soudainement flâner avec des

gens partageant sa prédilection pour les ordinateurs, la science-fiction et la nourriture chinoise. Quinze ans après, lors d'un discours à l'Institut Technique Royal de Suède, Stallman se remémore l'époque avec nostalgie : « Je me souviens des levers de soleil, qu'on admirait depuis la voiture en revenant de Chinatown. C'était magnifique à voir, c'est un moment si calme dans la journée. L'instant est merveilleux pour se préparer au coucher. C'est si bon de rentrer chez soi au petit matin et d'entendre le chant des oiseaux. On se sent profondément serein, content du travail de la nuit. » [10]

***

Plus Stallman fréquentait les hackers, plus il faisait sienne leur vision du monde. Lui-même grand défenseur de la liberté individuelle, il commença à y mêler le sens d'une responsabilité envers la communauté. Lorsque les autres violaient le code commun, Stallman n'hésitait pas à le faire observer. Moins d'un an après sa première visite, il faisait déjà partie de ceux qui « libéraient » les terminaux appartenant à l'ensemble du laboratoire.

Stallman ajouta même sa touche personnelle à cet art. La méthode classique lui paraissait trop difficile [11]. Une autre méthode consistait à faire coulisser les dalles du plafond afin de franchir le mur. Cela fonctionnait parfaitement (tant qu'il y avait un bureau

10. Voir la transcription de la conférence de Richard Stallman au KTH en Suède (30 octobre 1986). http://www.gnu.org/philosophy/stallman-kth.html.

11. L'une des manières les plus sophistiquées d'ouvrir les portes, communément attribuée à Greenblatt, consistait à plier une tige de fer en formant plusieurs angles droits puis à attacher à son extrémité un morceau de bande adhésive. Glissant la tige sous la porte, un hacker pouvait ainsi la tourner de manière à ce que la bande adhésive atteignît la poignée. Si l'adhésif ajouté sur la bande tenait bon, le hacker pouvait ouvrir la porte en tirant sur la tige.

à l'intérieur sur lequel prendre appui), avec toutefois le très désagréable inconvénient de se trouver rapidement couvert de fibres de verre irritantes.

Stallman y remédia. Au lieu de passer une tige sous la porte, pourquoi ne pas la glisser entre les deux panneaux au-dessus du montant de la porte ? Il essaya et, à la place du fil de fer, il laissa pendre une longue boucle de ruban magnétique en « U » terminée par un morceau de ruban adhésif, face collante vers le haut. En la glissant par-dessus la porte et en l'agitant jusqu'à ceinturer la poignée, il n'avait plus qu'à remonter le ruban jusqu'à ce que l'adhésif prenne. Il tirait ensuite sur l'une des extrémités pour faire tourner la poignée. Comme on pouvait s'y attendre, la porte s'ouvrait. Il avait ainsi donné sa touche personnelle à l'art d'ouvrir des portes closes. « Parfois vous deviez frapper la porte du pied après avoir tourné la poignée, ajoute-t-il, se souvenant du petit défaut de la nouvelle méthode. Il fallait un bon sens de l'équilibre pour réussir ce geste tout en se tenant debout sur une chaise posée sur un bureau. »

Ces exemples montrent simplement que Stallman se sentait prêt à parler et agir pour défendre ses convictions politiques. La culture de l'action directe qui régnait au AI Lab l'avait suffisamment inspiré pour effacer en lui toute trace de la passivité de son adolescence. Ouvrir un bureau pour libérer un terminal n'était pas la même chose que de participer à une manifestation, mais d'une certaine manière, c'était bien plus efficace : cela résolvait les problèmes dans l'instant.

***

Pendant ses dernières années à Harvard, Stallman commença à y appliquer les leçons fantasques et irrévérencieuses apprises au AI

Lab. « Vous a-t-il raconté l'histoire du serpent ? demande sa mère. Lui et ses compères de dortoir avaient soumis la candidature d'un serpent pour des élections étudiantes. Apparemment, le serpent a obtenu un nombre considérable de votes. » [12]

Stallman se rappelle que le serpent avait attiré un nombre significatif de votes, principalement parce qu'il portait le même nom de famille que son propriétaire. « Les gens croyaient sans doute voter pour son maître, dit Stallman. Nos affiches de campagne disaient que le serpent 'rampait vers' son siège. Nous disions aussi que c'était un candidat *at large* (c'est-à-dire sans district, mais aussi, pour un criminel, 'en fuite') puisqu'il avait grimpé dans le mur par une bouche de ventilation quelques semaines auparavant, et que personne ne savait où il se trouvait. » Stallman et ses amis nominèrent aussi le fils du concierge, alors âgé de trois ans. « Son principal slogan de campagne électorale était la retraite obligatoire à l'âge de sept ans », se souvient Stallman.

Les canulars de Harvard n'étaient rien en comparaison des 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 grand nombre de votes nuls suggérait que le chat avait réellement gagné. Deux ou trois ans plus tard, une voiture a écrasé Woodstock de manière suspecte. Personne ne sait si le conducteur travaillait pour l'administration du MIT ». Stallman précise ne pas avoir participé à la candidature de Woodstock, « mais j'en étais admiratif », ajoute-t-il. [13]

12. En fait, ce serpent était candidat à l'élection au sein de la Currier House, le dortoir de Stallman, et non pas au conseil des étudiants du campus.

13. Dans un courriel reçu peu après le dernier cycle d'édition de la première version de cet ouvrage, Stallman dit avoir trouvé également une partie de son

***

Au AI Lab, les activités politiques de Stallman avaient une tonalité plus tranchante. Pendant les années 1970, les hackers devaient affronter les mises en cause régulières des membres de la faculté et de ses administrateurs. Ces derniers voulaient mettre fin à l'ITS et à son fonctionnement à la mode hacker. L'ITS permettait en effet à quiconque de s'installer derrière un terminal et de faire ce qu'il voulait, y compris ordonner l'arrêt du système cinq minutes plus tard. Et si l'arrêt du système était demandé sans bonne raison, un autre hacker pouvait en annuler la commande.

Or, au milieu de la décennie, de plus en plus d'enseignants (surtout ceux qui n'avaient pas baigné dans la culture du MIT) demandèrent un système de fichiers sécurisé pour protéger l'accès à leurs données. Comme d'autres systèmes d'exploitation présentaient cette fonctionnalité, ces utilisateurs avaient pris l'habitude de vivre avec le sentiment qu'ils étaient protégés contre quelque chose de dangereux.

Malgré tout, sur l'insistance de Stallman et des autres hackers, le AI Lab demeura provisoirement une zone franche. Stallman opposait des arguments d'ordre à la fois éthique et pratique à ces exigences sécuritaires. Sur le plan éthique, il en appelait à la tradition d'ouverture et de confiance intellectuelle qui étaient la marque du AI Lab. Sur le plan pratique, il faisait valoir que la structure interne de l'ITS était construite de manière à encourager le hacking

inspiration politique sur le 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 Qin », écrit-il. [Il s'agit de la dynastie dont le cruel fondateur brûla tous les livres et se fit enterrer avec son armée en terre cuite] « Même si sa véracité est loin d'être établie d'un point de vue historique, j'en avais été ému. »

et la coopération, plutôt que sur la volonté de contrôler les utilisateurs. Toute tentative d'inverser cette conception aurait demandé une restructuration majeure du système.

Pour rendre quasi impossible une éventuelle refonte de l'ITS, Stallman utilisa le dernier champ vide de chaque descripteur de fichier pour y implémenter une fonction enregistrant le dernier utilisateur ayant modifié le fichier. Cette fonction nouvelle ne laissait plus de place pour stocker d'autres informations pour la sécurisation, et elle était si utile que personne ne pouvait sérieusement en proposer la suppression.

« Les hackers qui avaient écrit l'ITS avaient décidé qu'après tout, la protection des fichiers n'était habituellement utilisée que par un soi-disant administrateur système désireux de s'arroger du pouvoir sur tous les autres, expliqua Stallman par la suite. Or les hackers ne voulaient pas que quelqu'un ait un tel pouvoir sur eux. Ils n'ont donc pas implémenté de fonctions de ce type. Il en résultait que, peu importe ce qui pouvait se casser dans le système, vous pouviez toujours le réparer puisqu'aucun contrôle d'accès n'était en place. »

Grâce à leur vigilance, les hackers parvinrent à préserver les machines du AI Lab des artifices de la sécurisation. Non loin de là, dans un groupe du laboratoire de sciences informatiques (*Laboratory for Computer Science* — LCS) du MIT, l'esprit sécuritaire des enseignants gagna cependant la partie. Le groupe de recherche sur la modélisation dynamique y installa son premier système basé sur mot de passe en 1977.

Là encore, 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 un programme qui déchiffrait les mots de passe chiffrés que le système enregistrait. Il se lança ensuite dans une campagne par e-mail enjoignant

les utilisateurs de n'enregistrer qu'un retour chariot en guise de mot de passe. Ainsi, si un utilisateur utilisait *starfish*, il recevait ce message : « Je vois que vous avez choisi le mot de passe 'starfish'. Je vous suggère d'utiliser le mot de passe 'Entrée'. Plus facile à taper, il réfute le caractère illusoire des mots de passe et de la sécurité. »

Les usagers qui se servaient du retour chariot — c'est-à-dire ceux qui appuyaient sur la touche « Entrée » et saisissaient en fait une chaîne de caractères vide — laissaient leur compte accessible à quiconque, exactement comme cela avait été le cas peu auparavant. Et c'était bien là la logique : en refusant de fermer le verrou flambant neuf de leur compte, ces utilisateurs discréditaient l'idée même de verrou. Ils savaient que la faiblesse de cette sécurisation n'aurait pas arrêté de vrais intrus. En outre, peu importait : pourquoi s'inquiéter d'éventuels intrus ? Qui aurait voulu faire intrusion de toute façon ? Tous ceux qui venaient ne voulaient que visiter.

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 [14]. Cette campagne finit pourtant par être mise en échec et au début des années 1980, même les machines du AI Lab avaient été dotées de systèmes de sécurité.

Cet épisode constitua une étape importante pour Stallman. Dans cette opposition à la sécurité informatique, il faisait appel aux idées qui avaient façonné son passé : la soif de connaissances, le mépris de l'autorité et des préjugés, et la frustration devant l'existence de règles secrètes excluant certaines personnes. Mais il exprimait aussi les principes éthiques qui allaient façonner sa vie : la responsabilité envers la communauté, la confiance, et la

14. Voir [Levy, 1984, *op. cit.*], p.417

tendance du hacker à agir directement. Dit en des termes communs aux programmeurs, le mot de passe vide est la version 1.0 de la vision politique de Richard Stallman — incomplète par endroits mais mûre en grande partie.

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 tend à prouver que beaucoup pensaient que c'était la meilleure chose à faire. J'étais simplement enclin à militer sur ce point. »

Quoi qu'il en soit, Stallman dit que c'est au AI Lab que son esprit activiste s'est éveillé. Adolescent, il avait observé les événements politiques sans vraiment savoir ce qu'il pouvait faire ou en dire d'important. Jeune adulte, il s'exprimait sur des domaines où il se sentait très sûr de lui, tels que la conception logicielle, la responsabilité envers la communauté, et la liberté individuelle.

« J'ai rejoint cette communauté dont le style de vie impliquait le respect de la liberté d'autrui, dit-il. J'ai compris assez vite que c'était une bonne chose. Il m'aura fallu plus de temps pour réaliser qu'il y avait là un enjeu moral. »

⁂

Faire du *hacking* au AI Lab ne fut pas la seule activité qui lui permit de gagner en confiance. Au début de sa première année à Harvard, Stallman avait rejoint les rangs d'un groupe spécialisé dans les danses folkloriques internationales, au sein de Currier House. Il ne comptait aucunement danser jusqu'à ce qu'un ami lui fasse remarquer : « Comment peux-tu savoir que tu ne peux pas si

tu n'as jamais essayé ? » À son grand étonnement, Stallman était assez doué pour la danse et y prit plaisir. Ce qui avait commencé comme une expérience finit par devenir une autre passion, avec le *hacking* et les études. Ce fut aussi, occasionnellement, un moyen de rencontrer des filles, même s'il n'engagea pas de relation au cours de sa carrière universitaire.

En dansant, Stallman ne se sentait plus cet enfant de dix ans maladroit dont le manque de coordination motrice avait rendu frustrants ses essais au football américain. Il se sentait confiant, agile et vivant. Au début des années 1980, il alla même plus loin et rejoignit le *MIT Folk Dance Performing Group*. Revêtu de l'habit traditionnel des paysans des Balkans, il prit plaisir à danser devant les spectateurs et se découvrit une aptitude à la scène qui l'aida plus tard à s'exprimer en public.

La danse et la programmation (*hacking*), sans réellement améliorer son aisance sociale, devaient malgré tout l'aider à surmonter le sentiment d'exclusion qui avait assombri sa vie pré-Harvardienne.

En 1977, alors qu'il assistait pour la première fois à une convention sur la science-fiction, il rencontra Nancy, une femme qui faisait de la calligraphie à la demande sur des badges. Fébrile, Stallman en commanda un avec l'inscription « Destituons Dieu » (*Impeach God*).

Pour lui, le message était à plusieurs niveaux. Athée depuis toujours, Stallman y vit l'ouverture d'un « deuxième front » dans le débat religieux en cours. « À l'époque, la question qui préoccupait tout le monde était de savoir si un dieu existait vraiment, se souvient-il. 'Destituons Dieu' abordait le problème sous un angle complètement différent. Si un dieu était si puissant qu'il ait créé le monde, mais qu'ensuite il ne fasse rien pour y corriger les problèmes, à quoi bon l'adorer ? Ne serait-il pas plus juste de le juger ? »

À un autre niveau, Stallman faisait référence au scandale du Watergate des années 1970, qui comparait effectivement Nixon à une divinité tyrannique[15]. Ce scandale affectait profondément Stallman.

Enfant, il avait grandi en détestant l'autorité. Adulte, il voyait sa défiance consolidée par la culture du AI Lab. Pour eux, le Watergate n'était qu'une illustration shakespearienne des luttes de pouvoir quotidiennes qui rendaient la vie si difficile à ceux qui ne bénéficiaient d'aucun privilège. Une parabole démesurée illustrant ce qui arrivait quand le peuple échangeait liberté et ouverture contre sécurité et confort.

Porté par une confiance en lui plus affirmée, Stallman arborait le badge avec fierté. Les gens assez curieux pour le questionner sur le sujet recevaient un laïus bien préparé : « Mon nom est Jéhovah, disait-il. J'ai un plan secret pour mettre fin à l'injustice et à la souffrance, mais pour des raisons de sécurité divine, je ne peux rien vous en dévoiler. J'en ai une vision globale, mais pas vous. Vous savez que je suis bon parce que je vous l'ai dit. Donc placez votre foi en moi et obéissez-moi sans vous poser de question. Sinon, cela veut dire que vous êtes mauvais, aussi vous mettrai-je sur la liste de mes ennemis et vous jetterai-je dans une fosse où l'*Infernal Revenue Service*[16] vous fera subir un contrôle fiscal tous les ans pour l'éternité. »

15. La destitution (*impeachement*) aux États-Unis est une procédure législative consistant en la mise en accusation d'un haut fonctionnaire (président, vice-président, chef de cabinet, juge fédéral, etc.). Cette procédure permet d'engager des poursuites judiciaires vis-à-vis de ces hauts fonctionnaires que l'on destitue alors de leur poste. La procédure est votée par la Chambre des Représentants et se tient devant le Sénat, c'est-à-dire que seul le Congrès peut en entamer la procédure. Cette perspective força par exemple le président Nixon à démissionner suite au scandale du Watergate en 1974 — voir plus loin dans le texte — NdT.

16. Jeu de mot de Stallman sur l'*Internal Revenue Service*, le fisc américain. Nixon demandait au fisc de contrôler les membres de sa liste d'ennemis.

Ceux qui interprétaient ce petit discours comme une parodie littérale des audiences du Watergate ne comprenaient que la moitié du message. Pour Stallman, seuls ses camarades hackers semblaient comprendre l'autre moitié. Cent ans après que Lord Acton [17] eut averti que le pouvoir absolu corrompait absolument, les Américains semblaient avoir oublié le fondement de son truisme : le pouvoir, lui-même, corrompt. Plutôt que souligner de multiples exemples de corruption mineurs, Stallman préférait s'indigner d'un système entier qui, de façon inhérente, faisait confiance au pouvoir.

« Je me suis demandé : pourquoi m'arrêter au menu fretin ?, dit-il, se souvenant du badge et de son message. Si nous y allons contre Nixon, pourquoi ne pas y aller contre Mr. Big [18] ? Telles que je vois les choses, tout être ayant du pouvoir et qui en abuse, mérite qu'il lui soit retiré. »

17. Lord Acton (John Emerich Edward Dalberg, 1834-1890), historien et homme politique britannique. Penseur libéral, il concevait l'histoire humaine comme un progrès général vers toujours davantage de liberté. Selon lui, la défense de cette liberté est un devoir moral. Il voyait en revanche le pouvoir politique comme un pouvoir de commandement qui n'engage pas la responsabilité de celui qui commande — cette responsabilité, en démocratie, revenant au peuple et à la justice. Ainsi, logiquement, place est faite à la corruption dans la sphère du pouvoir politique et, dans le cas de l'absolutisme (plus aucune responsabilité nulle part puisque le pouvoir est concentré sur un seul) la corruption est d'autant plus grande — NdT.

18. En anglais, l'appellation *Mr Big* désigne le chef d'une mafia.

Chapitre

5

Une oasis de liberté

Notes de Richard Stallman sur ce chapitre

J'ai corrigé dans ce chapitre certaines erreurs, y compris concernant mes pensées ou sentiments supposés. J'ai gommé dans la description de certains événements une hostilité injustifiée. Les impressions de Sam Williams sont bien sûr demeurées lorsque présentées comme telles.

Toute personne ayant passé plus d'une minute en présence de Richard Stallman vous dira la même chose : oubliez les cheveux longs, l'apparente excentricité. La première chose que vous noterez

est son regard. Plongez dans ses yeux verts et vous saurez que vous êtes en présence d'un vrai croyant.

Qualifier d'intense le regard de Stallman est un euphémisme. Il fait plus que vous voir, il vous transperce. Même quand, momentanément, vos yeux s'égarent ailleurs par une simple politesse primitive, ceux de Stallman restent fixés, comme deux faisceaux de photons crépitant sur votre visage.

C'est sans doute pour cela que les comparaisons religieuses abondent à son sujet. En 1998, dans un article de *Salon.com* intitulé « Le saint du logiciel libre », Andrew Leonard voit les yeux verts de Stallman comme « rayonnants du pouvoir d'un prophète de l'Ancien Testament » [1]. En 1999, un article du magazine *Wired* déclare que Stallman et sa barbe font « penser à Raspoutine » [2], tandis que le journal londonien *The Guardian* fait de son sourire celui « d'un disciple voyant Jésus » [3].

Et ce ne sont là que quelques extraits des nombreuses comparaisons religieuses qui ont été publiées, la plus extrême à cette date étant attribuée à Linus Torvalds qui, dans son autobiographie, écrit : « Richard Stallman est le dieu du logiciel libre » [4]. Quant à Lawrence Lessig, il compare Stallman à Moïse dans une note de bas de page de son ouvrage [5].

1. Voir l'article d'Andrew Leonard, *The Saint of Free Software*, sur Salon.com, en août 1998 : http://www.salon.com/21st/feature/1998/08/cov_31feature.html

2. Voir l'article de Leander Kahney, *Linux's Forgotten Man*, dans Wired News le 5 mars 1999 : http://www.wired.com/science/discoveries/news/1999/03/18291

3. Voir l'article du London Guardian du 6 november 1999, *Programmer on moral high ground ; Free software is a moral issue for Richard Stallman believes in freedom and free software.*

4. Voir Linus Torvalds et David Diamond, *Just For Fun : The Story of an Accidental Revolutionary* (HarperCollins Publishers, Inc., 2001), p. 58.

5. Larry Lessig écrit dans son livre *The Future of Ideas* (Random House, 2001), page 270 : « […] comme avec Moïse, c'est un autre leader nommé Linus

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. Jamais ils n'auraient transigé sur des questions d'ordre moral. Ils ne pouvaient être achetés, et ils étaient habituellement méprisés. »

***

Ces comparaisons sonnent vrai sans toutefois rendre compte de la part vulnérable du personnage. Observez son regard assez longtemps, et vous commencerez à noter des changements subtils. Ce qui ne ressemble au début qu'à une tentative d'intimidation ou d'hypnose se révèle à mesure n'être qu'une tentative frustrée d'établir et de maintenir le contact. Si sa personnalité revêt quelques traits d'autisme ou du syndrome d'Asperger (hypothèse que Stallman envisagea parfois), ses yeux constituent à eux seuls une confirmation du diagnostic. Même à leur plus forte intensité, ils se font nuageux et distants, comme ceux d'un animal blessé se préparant à rendre l'âme.

Ma première rencontre personnelle avec ce regard légendaire remonte à mars 1999, au salon LinuxWorld de San Jose, en Californie. Étiquetée comme le *coming out* de la communauté Linux,

Torvalds qui a finalement emmené [le mouvement] jusqu'à la Terre Promise, en aidant à compléter le puzzle du système d'exploitation. Comme Moïse, Stallman est à la fois respecté et rejeté par ses alliés au sein du mouvement. C'est un leader intransigeant, et donc pour certains d'autant plus convaincant, dans un domaine crucial de la culture moderne. J'ai un profond respect pour les principes et l'engagement de cet homme extraordinaire, mais j'ai le même respect pour ceux qui sont assez courageux pour remettre ses idées en question et encourir sa colère. »

cette convention fut aussi l'événement qui réintroduisit Stallman dans la presse spécialisée. Déterminé à faire valoir sa véritable part de mérite, Stallman s'y était rendu pour instruire les spectateurs et les journalistes sur l'histoire du projet GNU et ses objectifs ouvertement politiques.

En tant que journaliste couvrant l'évènement, j'avais reçu par ailleurs une documentation sur Stallman lors d'une conférence de presse annonçant la sortie de Gnome 1.0, une interface utilisateur graphique libre.

Involontairement, je déclenchai en masse tous les signaux d'alerte 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 ? »

Ses yeux se fixant immédiatement sur les miens, il répondit : « Je vous demande d'avoir l'obligeance d'arrêter d'appeler ce système d'exploitation 'Linux'. Le noyau Linux n'est qu'une petite partie du système d'exploitation. De nombreux programmes qui font partie du système 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 tel que nous le connaissons aujourd'hui. Ne pas reconnaître la contribution 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 bien d'utiliser l'expression « GNU/Linux », et non « 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. À cette seconde, je sentis un léger frisson me parcourir . 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'est pas en train de me regarder », pensai-je.

Pour Stallman, ce genre de confrontation avait son utilité. 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, désireux de minimiser l'importance des objectifs trop politisés de son projet[6]. D'autres articles suivirent ; peu de journalistes utilisaient l'appellation GNU/Linux dans leurs écrits, mais la plupart étaient prompts à reconnaître Stallman comme l'instigateur du mouvement pour créer un système d'exploitation libre, quinze ans plus tôt.

***

Je ne devais rencontrer à nouveau Stallman que dix-sept mois plus tard. Entre temps, il visita la Silicon Valley une fois de plus pour la convention « LinuxWorld Show », en août 1999. Bien qu'il ne fût pas invité à y donner de discours, Stallman s'y fit tout de même l'auteur de la plus belle phrase de l'événement. Alors qu'il

6. Voir Leander Kahney, *op. cit.*

acceptait, au nom de la Fondation pour le logiciel libre (FSF — *Free Software Foundation*), le prix Linus Torvalds pour services rendus à la communauté (un prix qui empruntait son nom à celui du développeur 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., distributeur fort connu de GNU/Linux, entra en bourse. La nouvelle confirma ce que nous, journalistes, soupçonnions déjà : le mot « Linux » était sur toutes les lèvres de Wall Street, tout comme l'étaient auparavant *e-commerce* et *.com*. Alors que le marché boursier abordait le passage à l'an 2000 comme une hyperbole approche son asymptote verticale, toute discussion abordant la question des logiciels libres ou de l'open source sous l'angle politique tomba rapidement dans l'oubli.

D'où, peut-être, l'absence remarquée de Stallman, lorsque le LinuxWorld enchaîna sur une troisième convention en août 2000.

⁂

Ma deuxième rencontre avec Stallman et son regard légendaire eut lieu peu après ce troisième LinuxWorld. Apprenant que Stallman allait passer dans la Silicon Valley, j'organisai une entrevue à l'heure du déjeuner à Palo Alto, en Californie. Le choix du lieu semblait ironique, et pas seulement à cause de l'absence de Stallman à la convention. À l'exception de Redmond dans l'état de Washington, peu de villes offraient un tel témoignage de la richesse économique induite par le commerce des logiciels privateurs. Je m'y rendis en voiture depuis Oakland, curieux de savoir comment, ayant passé le plus clair de sa vie à vilipender notre

penchant culturel pour la cupidité et l'égoïsme, Stallman se débrouillait dans une ville où la moindre maisonnette valait autour d'un demi-million de dollars.

Je suivis les indications fournies par Stallman et arrivai au siège de Art.net, « Association d'artistes virtuels » à but non lucratif. Situées dans une maison bordée de haies dans un coin du nord de la ville, les installations de Art.net avaient un petit côté plaisamment déglingué. Que Stallman soit venu se tapir au fin fond de la Silicon Valley sembla soudain une idée moins surprenante.

Je le trouvai assis dans une pièce sombre, pianotant sur son ordinateur portable gris. Dès que j'entrai, il me fixa avec toute l'intensité des deux cents watts de son regard. Il m'offrit un « bonjour » réconfortant mais avant même que j'aie pu terminer de lui retourner la politesse, il avait déjà retrouvé l'écran du portable.

« Je termine un article sur l'esprit du hacking, dit-il sans cesser de taper. Regardez. »

Je lus. La pièce était faiblement éclairée, et il me fallut un moment pour m'habituer au texte d'un vert blanchi sur fond noir, les couleurs étant inversées par rapport à l'affichage de rigueur dans la plupart des traitements de texte. Il s'agissait du compte rendu d'un dîner récent, dans un restaurant coréen. Avant le repas, Stallman avait remarqué que la personne ayant dressé la table avait laissé devant la place de Stallman six baguettes, au lieu des deux habituelles. La plupart des gens auraient fait fi des deux paires supplémentaires, mais Stallman avait préféré trouver une façon d'utiliser les six baguettes en même temps. Une solution qui, comme bien des programmes de hackers, était tout aussi ridicule qu'originale et illustrait son propos.

Alors que je lisais l'extrait, je sentis qu'il me regardait, absorbé. Je relevai la tête et lus sur son visage un demi-sourire, fier

et enfantin. Mon compliment ne suscita guère que la levée d'un sourcil. « Je serai prêt dans un instant », dit-il.

Stallman se remit au clavier de son portable. Rectangle gris et massif, l'appareil était l'antithèse des portables élancés qui avaient la faveur des programmeurs, à la récente convention LinuxWorld. Au-dessus du clavier s'en trouvait un plus léger et plus petit, rappel des mains vieillissantes de Stallman. En effet, au milieu des années 1990, les douleurs étaient devenues si insoutenables qu'elles l'avaient conduit à engager un dactylographe [7].

Stallman a tendance à ignorer tout stimulus externe lorsqu'il travaille. À l'observer, les yeux rivés sur l'écran alors que ses doigts dansent sur le clavier, on songe à deux amis de longue date pris dans une intense conversation.

La séance se termina avec quelques coups forts assénés sur les touches, puis le rangement du portable. « Prêt pour le déjeuner ? », demanda Stallman.

***

Nous marchâmes jusqu'à ma voiture. Se plaignant d'une entorse, Stallman avançait lentement, en boitant. La faute à une blessure au tendon du pied gauche, survenue trois ans avant, qui avait empiré à un point tel que Stallman, grand amateur de danse folklorique, avait été obligé d'abandonner toute pratique. « J'aime énormément la danse folklorique, se lamentait-il. Ne plus pouvoir danser est une vraie tragédie pour moi. »

Le corps de Stallman en témoignait aussi. Le manque d'exercice l'avait laissé joufflu et ventru, une bedaine qui se voyait

7. Il s'en remet de nos jours à un clavier nécessitant moins de pression qu'un clavier classique.

moins l'année précédente. Une prise de poids probablement sérieuse, car en marchant, Stallman cambrait le dos, telle une femme enceinte s'accommodant d'un poids inhabituel.

Stallman s'arrêta pour sentir le parfum des roses, ralentissant un peu plus le rythme de notre marche. Visant une floraison particulièrement belle, il approcha son nez des pétales centraux, inspira profondément et recula avec un soupir de satisfaction.

« Hmmm, rhinophytophilie ! » [8], s'exclama-t-il en se frottant le dos.

Le trajet jusqu'au restaurant dura moins de trois minutes. Suivant la recommandation de Tim Ney, ancien directeur exécutif de la FSF, j'avais laissé à Stallman le choix du restaurant. Alors que certains journalistes étaient prompts à grossir le trait concernant le style de vie supposément monacal de Stallman, je faisais quant à moi face à un véritable épicurien s'agissant de nourriture. L'un des privilèges du missionnaire parcourant le globe pour y délivrer la bonne parole du logiciel libre est d'avoir le droit partout de goûter les meilleurs mets. « Prenez n'importe quelle grande ville dans le monde, il y a des chances que Richard y connaisse le meilleur restaurant, rapporte Ney. Il est très fier d'en connaître le menu et de pouvoir commander pour tous les convives » (avec leur accord, bien entendu).

Pour notre repas, Stallman avait choisi un restaurant cantonnais de vapeurs *dimsum*, à deux rues de University Avenue, la principale artère de Palo Alto. Un choix influencé en grande partie par son récent voyage en Chine — qui incluait un arrêt à Hong

8. 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 de mettre son nez dans une fleur et de savourer l'instant, présentée comme une pratique sexuelle douteuse entre le nez et la plante. Le site personnel de Richard Stallman raconte une autre histoire de fleurs : http ://www.stallman.org/articles/texas.html.

Kong — mais aussi par son peu de goût pour les cuisines du Hunan et du Sichuan, plus relevées. « Je ne suis pas emballé par la cuisine épicée », dit Stallman.

Nous arrivâmes quelques minutes après onze heures et dûmes patienter vingt minutes. Connaissant l'aversion des hackers pour le temps perdu, je retenais mon souffle, craignant une saute d'humeur. Contre toute attente, Stallman accueillit la nouvelle avec philosophie.

« Dommage que nous n'ayons pas trouvé quelqu'un pour se joindre à nous, me dit-il. C'est toujours plus agréable de manger en groupe. » Pendant l'attente, Stallman effectua quelques pas de danse. Ses gestes étaient tout juste esquissés mais habiles.

***

Nous discutâmes de l'actualité. La seule chose qu'il regrettait, en ayant raté la conférence LinuxWorld, était d'avoir manqué le lancement de la *GNOME Foundation*. Soutenue par Sun Microsystems et IBM, la fondation constitue à bien des égards une preuve pour Stallman que logiciel libre et économie de marché ne s'excluent pas nécessairement l'un l'autre. Mais au-delà, il n'approuvait pas le message livré par cet événement.

« De la manière dont les choses ont été présentées, 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, notait Stallman. Un survol rapide de ses voyages en l'an 2000 témoignait bien de la popularité grandissante du message porté par le mouvement

du logiciel libre. Entre ses visites en Inde, en Chine et au Brésil, Stallman n'avait passé que douze des cent quinze derniers jours sur le sol américain. Ses voyages lui avaient donné 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 une fortune. 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. Mais quand même, l'intérêt accordé au logiciel libre était sérieux, 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[9]. La compagnie avait mis sur le marché un outil informatique controversé, qui laissait les mordus de musique échantillonner et télécharger les fichiers d'autres mordus de musique. Grâce au pouvoir attractif de l'Internet, ce logiciel soi-disant pair-à-pair (*peer-to-peer* ou P2P) était devenu *de facto* un véritable juke-box en ligne, donnant le moyen à tout amateur 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 non libre, le système Napster tirait son inspiration d'une thèse longtemps soutenue par Stallman, à savoir qu'une fois une œuvre passée sous forme numérique (en d'autres termes, lorsqu'il n'était plus question de dupliquer des sons ou des atomes mais de dupliquer de

9. En 2002 — NdT.

l'information), la naturelle propension humaine à partager cette œuvre devenait plus difficile à restreindre.

Au lieu d'imposer des restrictions supplémentaires, les dirigeants de Napster avaient décidé de tirer avantage de cette propension. En proposant aux amateurs de musique un lieu central pour échanger des fichiers musicaux, la compagnie avait misé sur la transformation du trafic des usagers en opportunité commerciale.

Le succès soudain du modèle de Napster suscita d'importantes craintes 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 l'U.S. District Court avait émis une ordonnance contre le service d'échange de fichiers, à la demande de la puissante Association américaine de l'industrie du disque (RIAA — *Recording Industry Association of America*). Cette ordonnance fut par la suite suspendue par la Neuvième cour d'appel fédérale américaine (*U.S. Ninth District Court of Appeals*). Début 2001, toutefois, la cour d'appel avait également considéré que la compagnie de San Mateo enfreignait la loi sur le copyright [10]. Une décision que la porte-parole de la RIAA, Hillary Rosen, avait plus tard accueillie comme une « victoire pour la communauté des créateurs et le commerce électronique licite. » [11]

⁂

Pour les hackers, tel Stallman, le modèle commercial de Napster était embarrassant, à bien des égards. La volonté de la compagnie de s'approprier des principes hackers éprouvés, comme le

10. Voir l'article de Cecily Barnes et Scott Ard, "*Court Grants Stay of Napster Injunction*", News.com (28 juillet 2000) : http ://news.cnet.com/2100-1023-243817.html

11. Voir le communiqué de presse de la RIAA du 12 février 2001 "*A Clear Victory for Recording Industry in Napster Case*" : http://www.riaa.com/PR_story.cfm?id=372

partage de fichiers et la propriété commune de l'information, alors que l'on vendait un service utilisant un logiciel non libre, envoyait un signal confus et plutôt pénible. Ayant déjà de la difficulté à faire passer son propre message de façon bien articulée dans les médias, Stallman se montra plutôt réticent quand vint le moment de parler de cette compagnie. Il admit tout de même avoir tiré quelques enseignements du phénomène social Napster.

« Avant Napster, je pensais qu'il suffisait de pouvoir distribuer en privé des œuvres du domaine du divertissement. Le nombre de gens qui trouvent Napster utile me semble montrer au contraire que le droit de redistribuer des copies non seulement de voisin à voisin mais aussi 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. L'hôtesse nous invita à entrer. En quelques secondes, nous fumes assis à côté d'un grand mur paré de miroirs, dans un coin du restaurant.

Le menu servait de formulaire de commande, et Stallman en était déjà à cocher les cases avant même que notre eau ne soit servie. « Rouleaux de crevettes frites enveloppés dans des feuilles de tofu séchées, disait-il à voix haute. La texture de la feuille de tofu séchée est très 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 se chargeant d'une pointe d'émotion, pour la première fois de la matinée. « 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 au point que je tenais un journal pour détailler chacun de ces merveilleux repas ! »

La conversation se tourna vers la cuisine coréenne. Durant sa tournée asiatique en juin 2000, Stallman était allé visiter la Corée du Sud. Son arrivée avait enflammé les médias locaux. En effet, la même semaine, Bill Gates, le fondateur et PDG de Microsoft, participait là-bas à une conférence sur les logiciels. Outre sa photo juste au-dessus de celle de Gates à la Une du plus grand journal de Séoul, Stallman considérait la nourriture comme le summum du voyage : « On m'a servi un bol de *naeng myun*, des nouilles froides. Elles procurent une sensation intéressante. Selon les 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 une authentique louange de la part de Stallman. Je devais vite m'en rendre compte, car quelques instants après avoir entendu sa rhapsodie en l'honneur des *naeng myun*, je sentis le faisceau laser de ses yeux brûler par dessus mon épaule droite. « Il y a une femme des plus exquises assise juste derrière vous », indiqua-t-il.

Je me tournai pour regarder, apercevant du coin de l'oeil le dos d'une femme. Elle était jeune, dans la vingtaine, et portait une robe blanche cousue de paillettes. Elle et son compagnon de table réglaient la note. Je n'eus pas à les regarder pour savoir qu'ils avaient quitté le restaurant car les yeux de mon interlocuteur 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 ressaisit.

***

Cet intermède m'offrait l'occasion d'aborder la question de sa réputation, parfois contradictoire, vis-à-vis du sexe opposé. Nombre de hackers rapportaient que Stallman accueillait les dames d'un baisemain [12]. Pourtant, dans un article de *Salon.com* en date du 26 mai 2000, il est dépeint comme un hacker à la Lothario [13]. La journaliste, Annalee Newitz, pour documenter la relation entre amour libre et logiciel libre, y présente un Stallman rejetant les valeurs familiales traditionnelles, et le cite : « Je crois en l'amour, mais pas en la monogamie. » [14] Le menu de Stallman s'abaissa un brin lorsque j'amenai le sujet.

« Bon. La plupart des hommes semblent ne vouloir que le sexe, et ont une attitude plutôt méprisante envers les femmes, dit-il. Même celles avec lesquelles ils sont liés. C'est une chose que je n'ai jamais réussi à comprendre. »

Je lui mentionnai le passage du livre de 1999, *Open Sources*, dans lequel il confessait avoir voulu nommer l'infortuné noyau GNU en l'honneur d'une petite amie de l'époque. Elle s'appelait Alix, prénom parfait pour la convention des développeurs d'Unix, voulant que les noms de systèmes d'exploitation ou de noyaux (« Linux » par exemple) se terminent par un « x ». Alix, qui administrait des systèmes Unix, avait dit un jour à ses amis qu'elle souhaitait qu'un noyau porte son nom. Stallman décida alors de

12. Voir l'article de Mae Ling Mak du 17 décembre 1998, « *A Mae Ling Story* » : http://www.crackmonkey.org/pipermail/crackmonkey/1998q4/003006.html. Elle est la seule à avoir accepté de parler publiquement de cette pratique, même si j'ai entendu d'autres femmes en parler. En tout cas elle a surmonté son appréhension et dansé avec Stallman lors d'un show à LinuxWorld en 1999 : http://www.linux.com/interact/potd.phtml?potd_id=44

13. Lothario, séducteur et libertin, est un personnage célèbre de la tragédie *The Fair Penitent* (1703), de Nicholas Rowe — NdT.

14. Voir l'article d'Annalee Newitz du 26 mai 2000 sur Salon.com : « Si le code est libre, pourquoi pas moi ? » : http://www.salon.com/tech/feature/2000/05/26/free_love/print.html

lui faire la surprise de nommer « Alix » le noyau GNU. Le développeur principal le renomma Hurd [15], tout en gardant « Alix » pour une partie du noyau. Lorsqu'un des amis d'Alix le lui apprit, cette dernière en fut touchée — même si une refonte de Hurd entraîna plus tard l'élimination de cette fraction du noyau [16].

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 dames apprécient. C'est l'occasion d'offrir un peu d'affection et d'être apprécié en retour. »

L'affection n'était qu'un fil ténu parcourant l'existence de Richard Stallman, et il se montrait douloureusement 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 lapidaires, il finit par lever son menu, coupant court à l'interrogatoire.

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

15. Voir Richard Stallman, *The GNU Operating System and the Free Software Movement*, coll. Open Sources, O'Reilly & Associates, Inc., 1999, p. 65.

16. Note de Richard Stallman — Sam Williams a interprété cet épisode en suggérant que j'étais un incorrigible romantique désespéré, et que mes efforts n'avaient d'autre but que d'impressionner une jeune femme, encore non identifiée. Aucun hacker du MIT ne pourrait le croire, car nous avons appris très jeunes que les femmes ne nous remarquaient pas, et pouvaient encore moins nous aimer — fût-ce pour notre art de programmer. Nous programmions parce que nous trouvions cela passionnant. Si ces événements ont été possibles, c'est parce que j'avais une petite amie clairement identifiée à l'époque. Si j'étais romantique, ce n'était ni par espoir ni par désespoir, mais plutôt parce qu'à ce moment j'avais rencontré un peu de succès ! Avec cette interprétation naïve, Sam Williams me comparait à une sorte de Don Quichotte. Pour être exact, dans la première édition, voici ce qu'il rapportait de mes propos — que je reconnais confus, d'ailleurs : « Je n'essayais pas vraiment d'être romantique. C'était plutôt une plaisanterie. Je veux dire… c'était romantique, mais c'était aussi une plaisanterie, voyez-vous ? Ça aurait été une belle surprise. »

⁂

Le repas arrivé, la conversation rebondit entre les différents services. Nous discutions de l'inclination notoire du hacker pour la nourriture chinoise, des excursions hebdomadaires dans le district de Chinatown à Boston pendant ses années de programmeur au AI Lab, et de la logique sous-jacente de la langue chinoise et de son système d'écriture propre.

Chacune de mes questions rencontrait une réponse érudite de la part de Stallman : « J'ai entendu des gens parler shanghaïen la dernière fois que j'étais en Chine. C'était intéressant à entendre, et 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 c'est exact, et on sait que de tels changements arrivent au cours de l'histoire, les dialectes doivent avoir divergé avant la perte de ces syllabes finales. »

Le premier plat, une assiette de gâteaux aux navets sautés, arriva. Nous prîmes un moment 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 revins sur la question de son exclusion sociale, en lui demandant si son adolescence avait conditionné sa propension à prendre des positions impopulaires, surtout l'âpre 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 [mon exclusion] m'a aidé, répondit-il en mâchant une boulette de pâte. 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. Il n'y aurait eu 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, lui préférait la musique classique. Le souvenir lui rappela un des rares épisodes comiques de ses années en tant que collégien.

Après le passage des Beatles à l'émission *Ed Sullivan Show* en 1964, la plupart de ses camarades s'étaient précipités pour acheter les derniers singles et albums du groupe. À ce moment précis, Stallman décida de boycotter les Fab Four : « J'appréciais quelques chansons populaires pré-Beatles. Mais je n'aimais pas les Beatles. Je détestais particulièrement la manière folle dont réagissaient les gens à leur sujet. C'était à qui aurait le groupe qui les adulerait le plus ! » Son boycott n'ayant prise sur personne, Stallman chercha d'autres moyens pour décrier l'esprit grégaire de ses pairs. Il raconta avoir brièvement imaginé de créer une formation musicale dans le but de satiriser le groupe de Liverpool : « Je voulais appeler ça 'Tokyo Rose and the Japanese Beetles'. »

Connaissant sa prédilection pour la musique folklorique internationale, je lui demandai s'il avait une affinité similaire pour Bob Dylan ou d'autres musiciens du début des années 1960. Stallman hocha la tête. « J'aimais Peter, Paul et Mary, dit-il. Ça me rappelle un grand *filk*. »

Lorsque je lui demandai une définition de « filk », Stallman m'expliqua que le terme était originellement utilisé par les mordus de science-fiction pour désigner la parodie des paroles d'une

chanson (ces dernières décennies, certains sont même allés jusqu'à écrire des mélodies). Parmi les *filks* classiques, on retrouvait « *On Top of Spaghetti* », une révision de « *On Top of Old Smokey* », et « Yoda » du maître du genre Al Yankovic, sa chanson reprenant « Lola » des Kinks, revue à la sauce *Guerre des Étoiles*.

Stallman me demanda si je désirais l'entendre. Sitôt que j'eus acquiescé, il entonna d'une voix étonnamment claire l'air de la chanson « *Blowin' in the Wind* » de Bob Dylan, avec ces paroles :

How much wood could a woodchuck chuck,
If a woodchuck could chuck wood ?
How many poles could a pollack lock,
If a pollack 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 « stick it in your ear »

Combien de bois pourrait chucker une marmotte,
Si une marmotte pouvait chucker du bois ?
Combien de mâts pourrait bloquer un Polonais,
Si un Polonais pouvait bloquer des mâts ?
Combien d'genoux pourrait faire pousser un nègre,
Si un nègre pouvait faire pousser des g'noux ?
La réponse, très cher,
c'est mets-le toi dans l'oreille.
La réponse, c'est de se le mettre dans l'oreille… [17]

17. Non sans gaieté, Richard Stallman donne quelques explications au lecteur francophone sur les paroles de ce filk : « Les trois premiers vers sont de vieux jeux de mots, qui existaient avant ma naissance. L'humour vient de ce qu'on réinterprète les mots *woodchuck* (marmotte), *pollack* et *negro* comme des mots composés, un peu comme on le ferait avec 'tourne-vice'. Mais tout le

La chanson terminée, les lèvres de Stallman se courbèrent en un demi-sourire enfantin. Je regardai les tables environnantes. Des familles asiatiques se régalaient de leur déjeuner du dimanche, faisant peu de cas de cet alto barbu qui se trouvait parmi eux. 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 réagir au jeu de mots [18], il s'en empara de ses deux baguettes et la leva fièrement. « Sans doute devrais-je être celui qui prendra cette boulette », poursuivit-il. La nourriture disparue, notre conversation reprit le cours habituel d'une interview. Stallman se cala bien dans sa chaise et souleva délicatement sa tasse de thé dans le creux de ses mains.

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 tels que la publication musicale ? » demandai-je. « C'est une erreur que de transposer à un autre problème une réponse existante », dit-il, marquant la distinction entre musique et logiciel. « La bonne approche consiste à examiner chaque type d'œuvre et à en tirer les conclusions que vous pouvez. »

⁂

monde sait que ce sont de faux mots composés. Dans le cas de *woodchuck*, *wood* rappelle *would* (car ils se prononcent de même) et *would* est un verbe auxiliaire parallèle à *could*. C'est plus tard, probablement dans les années 1970, que quelqu'un a remarqué qu'ils pouvaient être assemblés pour former une chanson. Quant à *Stick it in your ear*, c'est une façon un peu ancienne et un peu argotique de dire qu'on n'en a rien à faire de quelque chose. » Pour d'autres *filks* de Stallman, consultez son site personnel. Pour l'entendre chanter la chanson *The Free Software Song*, rendez-vous sur http ://www.gnu.org.

18. En anglais, cornball désigne aussi un incorrigible romantique— NdT.

Stallman propose de classer les œuvres soumises au copyright en trois catégories. La première, fonctionnelle, comprend les logiciels informatiques, les dictionnaires, les manuels. La deuxième comprend les œuvres ayant rôle de témoignage — par exemple des documents scientifiques ou historiques. Leur fonction pourrait être mise à mal si les auteurs comme les lecteurs étaient libres de les modifier à volonté. Cette catégorie inclut aussi les œuvres d'expression personnelle — journaux intimes, autobiographies… — dont la modification reviendrait à falsifier les souvenirs d'une personne ou ses opinions, ce que Stallman considère comme injustifiable d'un point de vue éthique. Enfin, la troisième catégorie concerne les travaux artistiques et de divertissement [19].

Les droits accordés aux utilisateurs de chaque œuvre doivent, pour Stallman, être adaptés au type d'œuvre. Ainsi pour la première catégorie des œuvres fonctionnelles, les utilisateurs devraient-ils se voir conférer le droit illimité d'en faire des versions modifiées. Pour les deuxième et troisième catégories, les droits de l'utilisateur devraient être modulés selon le souhait de l'auteur.

Cependant, Stallman insiste sur le fait que, quelle que soit la catégorie de l'œuvre, la liberté de copier et de redistribuer de manière non commerciale devrait s'appliquer intégralement et en tout temps. Si cela signifie de laisser les internautes imprimer une centaine de copies d'un article, d'une image, d'une chanson ou d'un livre et ensuite d'en distribuer par courriel les copies à une centaine d'étrangers, alors qu'il en soit ainsi.

« Il est évident que la redistribution privée occasionnelle 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

19. Stallman précise que l'appartenance d'une œuvre à la catégorie fonctionnelle ne signifie bien sûr pas qu'elle est dépourvue de valeur esthétique.

également besoin de permettre — nous devons permettre — même 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 très utile ! »

⁂

Lorsque je demandai si les cours de justice pourraient jamais accepter d'être aussi permissives, Stallman m'interrompit : « Vous ne posez pas la bonne question, et vous changez complètement de sujet. Vous mélangez la question de l'éthique, et celle de l'interprétation de la loi. Et ce sont deux questions totalement indépendantes, qu'il est inutile de relier. Les juges continueront de toute façon à interpréter les lois existantes avec sévérité, puisque ces lois ont été commandées par les grosses compagnies d'édition dans ce seul but. »

Ce commentaire illustre bien la philosophie politique de Stallman. Que l'arsenal juridique soit le soutien des entreprises qui veulent assimiler le copyright à une réplique logicielle de la traditionnelle propriété foncière n'oblige en rien les utilisateurs de logiciels à jouer un tel jeu. La liberté est en effet une question éthique, et non juridique.

« Au-delà des lois telles qu'elles existent, je vois ce qu'elles *devraient* être, dit-il. Je n'essaie pas de proposer une législation, je me demande plutôt à quoi elle devrait servir. Pour moi, une loi qui interdit de partager avec des amis a aussi peu de fondement moral que les lois Jim Crow [20] ; elle ne mérite pas qu'on la respecte. »

20. Jim Crow est un personnage de café-théatre, né en 1828 et interprété par Thomas Dartmouth « Daddy » Rice dans la chanson *Jump Jim Crow*, archétype de l'Afro-Américain du Deep South vu par les Blancs. Ce nom fut repris pour un ensemble de lois sur la ségrégation raciale, promulguées par

L'évocation des lois de ségrégation raciale amène une autre question, celle de l'inspiration qu'ont donnée à Stallman les leaders politiques du passé. En effet, comme le mouvement des droits civils des années 1950 et 1960, ses tentatives pour amener un changement social en appellent à des valeurs intemporelles : la liberté, la justice et le respect. Stallman partageait son attention entre mon analogie et une mèche de cheveux particulièrement emmêlée. Quand je prolongeai la figure de style en le comparant au Dr. Martin Luther King Jr., il m'interrompit, détachant une fourche qu'il mit dans sa bouche. « Je ne joue pas au même niveau, mais je joue au même jeu », dit-il en mâchouillant sa mèche.

Je suggérai Malcolm X pour un autre point de comparaison. Comme l'ancien porte-parole de *Nation of Islam*, Stallman avait cultivé une réputation d'amateur de controverses, s'aliénant des alliés potentiels et prônant un message donnant la priorité à l'autosuffisance plutôt qu'à l'intégration culturelle. S'attaquant à une autre mèche de cheveux, Stallman rejeta la comparaison.

« Mon message est plus proche de celui de Martin Luther King, dit-il. C'est un message universel. La condamnation ferme de certaines pratiques visant à maltraiter les gens. Ce n'est pas un message de haine envers qui que ce soit. Et ça ne vise pas un petit groupe de personnes. J'invite tout le monde à mieux apprécier la valeur de la liberté et à se battre pour l'obtenir. »

Or, nombreux sont ceux qui critiquent Stallman pour avoir rejeté des alliances politiques pourtant commodes. Certains se lancent dans des explications psychologiques en invoquant un trait de caractère. Mais concernant sa répugnance notoire à utiliser le

différents états du sud des États-Unis après la Guerre de Sécession. Ces lois, abolies successivement dans les années 1950 et 1960, entrèrent dans le langage courant, au point que l'on utilise le nom Jim Crow pour parler de ségrégation raciale — NdT.

terme « open source » [21], on comprend sa réticence à participer à de récents projets de coalition. Stallman a en effet passé les dernières décennies à se battre au nom du logiciel libre, et ce terme est porteur d'une bonne part de son capital politique.

Des saillies telles que la boutade sur Han Solo au LinuxWord de 1999 n'ont fait que renforcer, pour les conformistes voyant le grégarisme comme une vertu, sa réputation de conservateur aigri résistant à toute mode politique ou économique.

« J'admire et respecte Richard pour tout le travail qu'il a accompli », confia Robert Young, président de la compagnie Red Hat, à propos de la conduite politique paradoxale de Stallman. « Ma seule critique tiendrait au fait que parfois Richard traite plus mal ses amis que ses ennemis. »

Le terme d'amis ne conviendrait qu'en partie à des gens comme Young ou à des entreprises comme Red Hat. Il serait justifié si l'on ne prenait en compte que certaines de leurs actions, comme la contribution qu'ils apportent au développement des logiciels libres, y compris de certains programmes GNU. Or, Red Hat freine par ailleurs ce mouvement, notamment en incluant des logiciels non libres dans les versions de GNU/Linux qu'ils distribuent. Et si l'on se place sur le simple plan de la communication, le fait qu'ils désignent l'ensemble du système par le seul mot « Linux » au lieu de « GNU/Linux » est tout sauf amical vis-à-vis du projet GNU ; de même, promouvoir l'open source plutôt que le logiciel libre trahit nos valeurs. Je peux travailler avec Young et Red Hat quand nous allons dans la même direction, mais cela arrive trop peu souvent pour les considérer comme des alliés.

⁂

21. Sur l'expression « open source » et ses enjeux, voir le chapitre 8 — NdT.

Si Stallman n'embrasse pas d'autres causes politiques que celle du logiciel libre, ce n'est pas par manque d'intérêt. En visitant ses bureaux au MIT, vous tombez nez-à-nez avec un vaste stock d'articles de journaux de gauche, répertoriant les atteintes aux droits civils à travers le globe. Visitez son site web personnel et vous y trouverez des attaques en règle contre le Digital Millenium Copyright Act, contre la guerre aux drogues, et contre l'OMC (Organisation mondiale du commerce).

« Nous devons faire attention à ne pas entraîner le mouvement pour le logiciel libre vers d'autres causes politiques auxquelles un nombre substantiel de sympathisants risque de ne pas adhérer. Ainsi, nous évitons de nous lier à des partis politiques parce que nous ne voulons pas exclure de notre cause les militants ou les représentants d'autres partis », explique Stallman.

Considérant son penchant pour l'activisme, je lui demandai pourquoi il n'avait pas recherché un écho plus large. Pourquoi n'avait-il pas tiré profit de son rayonnement chez les hackers pour se lancer dans la politique ?

Je fais de la politique dès que s'en présente une bonne occasion ; pour preuve mon site personnel http://stallman.org.

Stallman considéra la question un moment : « J'hésite à exagérer l'importance de cette modeste oasis de liberté parce que les autres domaines plus connus et conventionnels de la lutte pour la liberté et l'amélioration de la société sont d'une importance extrême. Je ne peux dire que le logiciel libre est aussi important que ces domaines mais c'est une responsabilité que j'ai assumée parce qu'elle m'est tombée dessus, et que j'ai vu que je pouvais y faire quelque chose. Cela dit, mettre fin à la brutalité des polices, à la

guerre aux drogues, mettre un terme aux racismes de tout ordre qui ont encore cours, aider les gens à vivre une vie plus confortable, protéger le droit à l'avortement, nous protéger de la théocratie… ce sont autant de causes importantes qui dépassent largement ce que je fais. J'aimerais juste savoir que faire pour y remédier. »

Ma réponse ne semble pas répondre à la question de Williams telle que formulée, mais plutôt à celle-ci : « Pourquoi vous être concentré sur les logiciels libres et non sur les autres causes en lesquelles vous croyez ? »

⁂

Là encore, Stallman dose son activité politique à la mesure de son assurance. Il lui a fallu longtemps pour asseoir les fondamentaux du mouvement du logiciel libre ; il hésite à croire qu'il peut faire avancer les autres causes qu'il soutient.

« J'aurais aimé savoir comment jouer un rôle majeur dans toutes ces causes importantes, j'aurais été très fier de pouvoir le faire. Mais bien des gens probablement meilleurs que moi y ont travaillé et n'ont pu accomplir que les avancées actuelles, dit-il. D'autres se battent contre ces grandes menaces visibles ; j'en ai vu une qui restait délaissée et je suis allé me battre sur ce terrain. La cause n'est peut-être pas aussi grande, mais j'étais le seul à l'avoir identifiée. »

Mâchant une dernière mèche de cheveux, Stallman proposa de payer l'addition. Avant que le serveur ne la retirât, il sortit un billet de couleur blanche et le jeta sur la pile. De toute évidence, le billet ne provenait pas des banques américaines, et je ne pus m'empêcher de l'examiner.

Assurément, c'était un billet contrefait [22]. Au lieu d'arborer l'image de George Washington ou d'Abraham Lincoln, l'une des faces était celle d'un cochon de dessin animé. Au lieu de la mention « *United States of America* », la bannière au-dessus du cochon se lisait « *Untied Status of Avarice* ». Le billet était de zéro dollar, et lorsque le serveur prit l'argent, Stallman lui tira la manche : « J'ai ajouté un zéro à votre pourboire » dit-il, encore ce demi-sourire sur ses lèvres. Le serveur perplexe, ou leurré par l'allure du billet, sourit et repartit.

Stallman conclut : « Je crois que cela signifie que nous sommes libres de sortir. »

22. Selon Richard Stallman, Sam Williams a qualifié à tort ce billet de « contrefait ». Il est tout à fait légal d'utiliser un billet de zéro dollar pour le paiement d'une dette. Tout bureau du gouvernement américain pourra le convertir en équivalent zéro dollar-or.

Chapitre

6

La commune Emacs

Le AI Lab des années 1970 était un endroit unique à tout point de vue. Réunissant les meilleurs chercheurs autour de projets d'avant-garde, le laboratoire était, dans le domaine des sciences de l'information, une institution mondialement reconnue. La culture hacker qui y régnait et sa politique d'anarchie allaient conférer au lieu l'aura d'éternel rebelle. Ce n'est que plus tard, quand les scientifiques et les superstars du logiciel eurent quitté l'endroit, que les hackers prirent pleinement conscience du caractère unique et éphémère de l'endroit où ils avaient vécu.

« C'était un peu le jardin d'Éden », raconte Stallman en 1998 dans un article de Forbes[1] décrivant ce qu'était le laboratoire et

1. McHugh, 1998

son éthique de partage logiciel. « L'idée de ne pas coopérer ne nous avait jamais effleurés. »

Si la description du laboratoire comme d'un lieu mythique peut sembler exagérée, elle souligne que 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 terme de « chez-soi » est bien pesé dans le lexique de Stallman. Dans un reproche à peine voilé à ses parents, il se refuse encore à nommer comme tel tout autre domicile qu'il aurait occupé avant celui de Currier House, le dortoir de Harvard où il vécut pendant ses études. Sa façon de décrire son départ de l'endroit, en termes tragi-comiques, est d'ailleurs révélatrice.

Retraçant le cours de ses années là-bas, Stallman raconte que son seul regret est d'en avoir été chassé. Ce n'est qu'en lui demandant ce qui avait précipité ce départ, que je réalisai que j'étais tombé dans un de ses pièges typiques : « À Harvard, ils ont une politique étrange : ceux qui réussissent trop d'examens sont invités à quitter les lieux. »

N'ayant aucune intention de retourner à New York, Stallman suivit le chemin tracé par Greenblatt, Gosper, Sussman et bien d'autres avant lui. S'inscrivant au MIT en tant qu'étudiant diplômé, il loua une chambre dans un appartement près de Cambridge mais finit par considérer rapidement le AI Lab comme son véritable domicile.

Lors d'un discours en 1986, Stallman évoquait ses souvenirs de l'époque : « J'ai peut-être habité au laboratoire plus que d'autres, parce que tous les un ou deux ans, pour une raison ou une autre, je me retrouvais sans appartement, et je passais donc quelques mois à y vivre. J'ai toujours trouvé l'endroit très confortable, agréable et frais durant l'été. Il n'était pas inhabituel d'y voir

des gens s'endormir, épuisés par leur propre enthousiasme. Vous restez éveillé aussi longtemps que possible à programmer, parce que vous n'avez pas envie de vous arrêter. Et lorsque vous êtes complètement lessivé, vous rampez jusqu'à la plus proche surface molle horizontale. C'était très décontracté [2]. »

L'atmosphère accueillante du laboratoire pouvait cependant soulever la réprobation. Ce que certains voyaient comme un dortoir, d'autres le considéraient comme une fumerie d'opium électronique. Dans son livre de 1976, *Computer Power and Human Reason*, le chercheur Joseph Weizenbaum fait une critique cinglante du *computer bum*, ou informaticien-clochard, terme qu'il utilise pour désigner les hackers peuplant les salles d'informatique comme celle du AI Lab : « Les vêtements défraîchis, les joues mal rasées, les cheveux sales et hirsutes montrent à quel point ils sont oublieux de leur corps et du monde dans lequel ils évoluent. [Les informaticiens-clochards] n'existent, du moins pour ceux engagés à ce point, que pour et par l'ordinateur. » [3]

Un quart de siècle après cette publication, Stallman s'agace encore de cette description. Il en parle au présent, comme si Weizenbaum lui-même était dans la pièce. « Il veut condamner les gens à n'être que 'professionnels', à ne travailler que pour une rémunération et à fuir leur travail au plus vite pour l'oublier aussitôt, dit-il. Ce qu'il voit comme un état normal des choses, je le vois comme une tragédie. »

Pour Stallman, la vie de hacker eut pourtant son lot de tragédies. Son évolution du statut de programmeur du dimanche à celui de résident permanent du AI Lab s'accompagna d'une série d'infortunes que seule l'euphorie de la programmation pouvait soulager, à commencer par ce qu'il voit comme un événement malheureux, l'obtention de son diplôme de Harvard.

2. Stallman, 1986
3. Weizenbaum, 1976, p.116

Déterminé malgré cela à poursuivre ses études de physique, il s'inscrivit en second cycle au MIT, choix tout naturel. Non seulement cela lui fournissait l'opportunité de suivre les pas d'étudiants renommés de l'école tels que William Shockley (1936), Richard P. Feynman (1939), et Murray Gell-Mann (1951), mais cela le rapprochait aussi — de trois kilomètres — du AI Lab et de son nouvel ordinateur, le PDP-10. « Je m'orientais de façon certaine vers la programmation, mais tout en me disant que, peut-être, j'arriverais à faire les deux », confie-t-il.

Travaillant dur les matières scientifiques du second cycle universitaire la journée, et programmant le soir dans les confins monastiques du AI Lab, Stallman essayait de maintenir le parfait équilibre. Ce jeu de bascule n'était mis entre parenthèses que lors de sa séance hebdomadaire avec le Folk-Dance Club, seule sortie lui garantissant un minimum d'interaction sociale avec le sexe opposé.

C'est à la fin de cette première année au MIT que le désastre se produisit : une blessure au genou le contraignit à cesser la danse. Stallman n'y vit d'abord qu'un problème temporaire : il continua à se rendre au Club pour parler avec ses amis, tout en écoutant la musique qu'il appréciait. Mais à la fin de l'été, la rentrée approchant, il commença à s'inquiéter. « L'état de mon genou ne s'améliorait pas, se souvient-il, ce qui voulait dire qu'il y avait un fort risque que je ne puisse plus jamais danser. J'en avais le cœur brisé. »

Sans dortoir ni danse, l'univers social de Stallman venait d'imploser. Danser était pour lui le seul moment où il pouvait s'attendre à un certain succès avec la gent féminine. Ne plus pouvoir danser, déjà pénible en soi, pouvait signifier la fin des sorties avec le sexe opposé.

« J'avais l'impression que toute énergie m'avait quitté, se souvient Stallman. Je n'avais plus de force pour rien, hormis les choses

qui présentaient un attrait immédiat. L'énergie de faire autre chose était annihilée. J'étais totalement désespéré. »

⁂

Stallman concentra alors toute son énergie dans son travail au AI Lab, se mettant plus encore en retrait du monde. En octobre 1975, il abandonna ses études au MIT et ses études de physique, pour ne plus jamais 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. « En physique et en mathématiques, je ne trouvais pas de moyen d'apporter ma contribution, dit-il au souvenir des efforts précédant sa blessure au genou. J'aurais été fier d'apporter ma pierre à l'un ou l'autre de ces domaines, mais je ne voyais pas comment. Je ne savais par où commencer. En programmation au contraire, je voyais immédiatement comment écrire des programmes qui fonctionneraient et seraient utiles. C'était grisant et cela m'amenait à toujours vouloir en faire plus. »

Stallman n'était pas le seul à assimiler programmation et plaisir. Bien des hackers du AI Lab exhibaient fièrement des cursus universitaires incomplets. La plupart venait poursuivre des études de mathématiques ou de génie électronique et finissaient par troquer carrière universitaire et ambition professionnelle pour l'exaltation pure et simple qui accompagnait 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 la sensation d'euphorie récompensant une session de programmation de vingt heures.

Au-delà de ces sensations physiques, c'est encore le sentiment de l'accomplissement personnel qui était le plus appréciable, surtout en programmation, l'élément naturel de Stallman. Une enfance passée à étudier le soir l'avait habitué à travailler de longues heures sans dormir. Socialement inadapté depuis l'âge de dix ans, il avait peu de difficulté à œuvrer seul. De plus, le mathématicien naturellement doué pour la logique, clairvoyant sur les enchaînements de causes et de conséquences, savait dépasser les écueils de conception qui laissaient la majorité des autres hackers tourner en rond.

« Il était particulier », se souvient Gerald Sussman, chercheur au AI lab et, depuis 1985, membre du bureau de la FSF. Le décrivant comme « un penseur clair et un concepteur clair », Sussman invita Stallman à le rejoindre sur deux projets de recherche au AI Lab en 1973 et 1975, projets visant à créer des programmes d'intelligence artificielle capables d'analyser des circuits à la manière d'un ingénieur humain. Il fallait pour cela une maîtrise experte de Lisp, langage de programmation construit spécifiquement pour de telles applications, ainsi que la compréhension (fournie par Sussman) de la façon dont un être humain aborderait la même tâche. Le projet de 1975 fut novateur : il créa la technique dite du « recul non chronologique » ou « conservation de la cohérence », consistant à poser des hypothèses, à rechercher les éventuelles contradictions qui en découlent et, le cas échéant, à reconsidérer les hypothèses de départ.

***

Lorsqu'il ne planchait pas sur ces travaux officiels, Stallman consacrait son temps à des projets plus personnels. Voyant comme

tout hacker l'intérêt d'améliorer l'infrastructure logicielle du laboratoire, Stallman investit des efforts particuliers dans Teco, l'éditeur de texte du laboratoire.

Le travail de Stallman sur Teco dans les années 1970 est intimement lié à l'histoire de son leadership futur au sein du mouvement du logiciel libre. C'est également un stade important de l'histoire de l'informatique, au point qu'en retracer un bref résumé s'impose.

Durant les années 1950-1960, alors que les ordinateurs faisaient leur première apparition dans les universités, la programmation informatique était une chose incroyablement abstraite. Pour communiquer avec la machine, les programmeurs créaient une série de cartes perforées, dont chacune représentait une commande logicielle unique. Ils 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 les programmeurs déchiffraient alors comme données de sortie.

Cette procédure, appelée « traitement par lots » (*batch processing*) était lourde et très longue. Elle offrait aussi un terrain pour les abus de pouvoir. L'une des raisons de l'aversion spontanée des hackers pour la centralisation était d'ailleurs le pouvoir détenu par les premiers opérateurs système, qui décidaient quels lots seraient exécutés en priorité.

En 1962, les informaticiens et hackers participant au projet MAC, un précurseur du AI Lab, prirent des mesures pour alléger cette frustration. Le partage de temps (*time-sharing*), originellement appelé « vol de temps » (*time stealing*), 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 grâce à une série 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 1960, la conception des interfaces connut d'autres progrès. Lors d'une célèbre conférence en 1968, Doug Engelbart, un scientifique du Stanford Research Institute, dévoila un prototype d'interface graphique moderne. Branchant à l'ordinateur un téléviseur, ainsi qu'un dispositif de pointage qu'il 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 permettait à l'utilisateur de déplacer le curseur à l'écran et de voir sa position mise à jour par l'ordinateur en temps réel. Il était à présent possible de placer du texte n'importe où à l'écran.

De telles innovations mettraient deux décennies supplémentaires avant de se retrouver sur le marché. Sous peu, au cours des années 1970, les écrans vidéo allaient remplacer les télétypes comme terminaux d'affichage, créant ainsi la possibilité d'une édition en plein écran (par opposition au ligne par ligne).

L'un des premiers programmes à profiter de l'édition plein écran se trouvait au AI Lab, et était baptisé Teco, acronyme de *Text Editor and Corrector*[4] pour « éditeur et correcteur de texte ». Le programme résultait d'une mise à niveau, réalisée par des hackers, du vieil éditeur télétype de la machine PDP-6. S'il représentait une amélioration substantielle par rapport aux vieux éditeurs de texte, Teco présentait encore des inconvénients. Pour créer et éditer un document, un programmeur devait entrer une série de commandes spécifiant chaque édition. Le processus était abstrait. Contrairement aux traitements de texte modernes, qui mettent à jour le texte à chaque frappe sur le 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

4. D'après le Jargon File : http://www.catb.org/jargon/html/T/TECO.html

le texte. Avec le temps, un hacker devenait suffisamment habile pour procéder à d'importantes modifications en une seule et élégante chaîne de commandes. Mais comme Stallman le soulignerait plus tard, la procédure nécessitait « une faculté mentale équivalente à celle requise pour jouer aux échecs les yeux bandés. » [5]

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 à jour l'affichage à chaque frappe d'une commande. Dans le jargon informatique des années 1970, E était l'un des premiers logiciels rudimentaires d'édition wysiwyg. Contraction de *What You See Is What You Get* (« vous voyez ce que vous obtenez »), wysiwyg signifiait que l'usager pouvait manipuler le fichier en naviguant dans le texte à l'écran, contrairement à ce qui se passait avec un logiciel d'édition en arrière-plan [6].

Impressionné par cette trouvaille, Stallman chercha une manière similaire d'améliorer Teco à son retour au MIT. Il trouva une fonction dénommée *Control-R*, écrite par Carl Mikkelson et nommée d'après la combinaison de touches 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. Il avait comme défauts de n'utiliser que cinq lignes à l'écran, et d'être trop peu efficace pour une utilisation réelle.

5. Voir [Stallman, 1981]. Une version mise à jour de ce mémo (originellement : *AI Lab Memo*, 1979) au format HTML, dont j'ai extrait cette citation, est disponible sur http ://www.gnu.org. Pour faciliter la procédure, les hackers du AI Lab avaient élaboré un système qui affichait le texte et les lignes de commande sur un écran divisé. En dépit de cette innovation, utiliser *TECO* nécessitait talent et organisation.

6. Voir [Stallman, 1987]. http ://www.lysator.liu.se/history/garb/txt/87-1-emacs.txt

Stallman réimplémenta cette fonction pour exploiter l'écran en entier et de manière efficace. Il étendit ensuite cette fonction d'une manière subtile mais fondamentale, en associant des chaînes de commandes Teco (ou « macro ») à des combinaisons de touches. Les utilisateurs avancés de Teco, qui savaient sauvegarder leurs macros sous forme de fichier, pouvaient ainsi appeler leurs macros très facilement.

Alors qu'auparavant les utilisateurs tapaient une série de commandes qui s'évaporaient aussitôt, le hack de Stallman donnait la possibilité de sauvegarder les macros dans un fichier, pour les solliciter à nouveau, à volonté. Le résultat éleva Teco au rang d'éditeur wysiwyg. « C'était là la réelle avancée », relate Guy Steele, hacker au AI Lab durant cette période.

De ce que s'en rappelle lui-même Stallman, ce hack déclencha une explosion d'autres innovations. « Chacun y allait de sa propre collection de commandes personnelles d'édition, à raison d'une pour chaque chose qu'il aimait faire habituellement. Les gens se les passaient, les amélioraient, les rendant plus puissantes et plus génériques. Ces séries de commandes redéfinies devinrent elles-mêmes peu à peu des utilitaires du système à part entière. »

Tant de gens trouvèrent cette innovation utile et l'incorporèrent à leurs propres programmes Teco que l'éditeur lui-même devint secondaire à la macro-mania qu'il avait lancée. « Nous commencions à le considérer mentalement comme langage de programmation plutôt qu'éditeur de texte, raconte Stallman. Les utilisateurs prenaient un grand plaisir à bidouiller eux-mêmes le logiciel et à s'échanger de nouvelles idées[7]. »

Deux ans après cette explosion, le rythme des innovations se mit à avoir des effets secondaires indésirables. S'il avait certes validé, de façon excitante, le mode de fonctionnement collaboratif

7. *Ibid.*

des hackers, le nombre excessif de fonction avait fini par conduire à des incompatibilités.

« Nous nous trouvions face à une Tour de Babel, raconte Guy Steele. Voilà qui menaçait de tuer l'esprit qui l'avait fait naître. » Les hackers avaient en effet conçu l'ITS pour faciliter le partage de leurs connaissances et améliorer le travail de chacun. Cela signifiait pouvoir s'asseoir au bureau d'un autre programmeur, ouvrir un de ses projets et faire des commentaires et des 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.

Dès sa deuxième année d'existence, la possibilité d'étendre Teco par macro mettait à mal cette souplesse. Dans leur envie d'utiliser à fond les nouvelles fonctionnalités du mode plein écran, les hackers avaient tellement transformé leurs versions respectives de Teco, qu'il était devenu impossible d'utiliser le terminal d'un autre sans passer la première heure à chercher ce que faisait telle ou telle macro.

Frustré, Steele prit sur lui de régler le problème. Il se pencha sur les quatre packages de macros existantes pour en déduire un diagramme des commandes les plus utiles. Alors qu'il commençait à l'implémenter, l'attention de Stallman fut attirée.

« Il a commencé à regarder par-dessus mon épaule et m'a demandé ce que je faisais », se souvient-il. Pour ce hacker discret, qui échangeait rarement avec Stallman, le souvenir reste vif. Au AI Lab, il arrivait souvent qu'un hacker regarde par-dessus l'épaule d'un autre pendant qu'il travaillait. Mais Stallman était responsable de la maintenance de Teco au sein du laboratoire.

Il jugea le travail de Steele « intéressant » et se mit rapidement à l'ouvrage pour l'achever. « J'aime dire que j'ai implémenté le

premier 0,001% de la mise en œuvre et que Stallman a fait le reste », dit Steele en riant.

***

Le projet fut rebaptisé *Emacs* avec la bénédiction de Stallman. Acronyme de *editing macros* (« éditer des macros »), il exprimait le cap évolutionniste qui avait été franchi deux ans plus tôt, lors de l'explosion du nombre de macros personnelles. Il venait aussi remplir un trou alphabétique dans le lexique de la programmation. Notant dans l'ITS l'absence de nom de programme commençant par la lettre « E », Stallman choisit Emacs, pour pouvoir y faire référence grâce à une seule lettre. Là encore une marque du goût démesuré des hackers pour l'efficience [8].

Bien sûr, tous n'allaient pas d'emblée passer à Emacs. Les utilisateurs étaient libres de continuer à employer leurs propres éditeurs, basés sur Teco. La plupart décida pourtant de faire le saut, notamment parce qu'Emacs permettait d'ajouter ou de supprimer des fonctions sans toucher au reste.

« D'un côté, nous tentions de recréer un ensemble uniforme de commandes. De l'autre, nous voulions le garder entièrement ouvert, afin que les possibilités de programmation restent le plus vastes possible », se souvient Steele.

Stallman se trouvait désormais confronté à un autre problème : si les utilisateurs opéraient de nouveaux changements sans les communiquer au reste de la communauté, l'effet Tour de Babel sévirait de plus belle. S'en remettant à la doctrine hacker du partage de l'innovation, il incorpora un message dans le code source, définissant des conditions d'utilisation : les utilisateurs étaient libres de

8. *Ibid.*

modifier et de redistribuer le code, à la condition de reverser en retour à la communauté les extensions qu'ils écrivaient. Stallman disaient de ceux qui adoptaient ce système qu'ils « rejoignaient la commune Emacs ». Tout comme Teco était devenu plus qu'un simple éditeur de texte, Emacs était devenu plus qu'un simple logiciel.

Pour Stallman, il s'agissait d'un contrat social. En 1981, dans une première note du projet, il décrivait ainsi 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 afin d'être incorporée et redistribuée. » [9]

À l'origine, Emacs ne fonctionnait que sur le PDP-10, mais rapidement les utilisateurs des autres machines voulurent eux aussi en faire leur éditeur. Le rythme des innovations se maintint durant toute la décennie, amenant une foule de programmes *à la* Emacs, mais dotés de différents degrés d'interopérabilité. Ils n'étaient pas soumis aux règles de la commune Emacs car leur code en était séparé. 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*). Cependant, pour mériter d'être qualifié de véritable Emacs, un éditeur se devait de fournir à l'utilisateur les mêmes possibilités de programmation que l'Emacs original. À défaut, même avec des raccourcis clavier similaires, il n'était considéré que comme un « ersatz d'Emacs » — tel était le cas de Mince (*Mince is Not Complete Emacs*).

Alors que Stallman développait Emacs au AI Lab, un autre événement était en passe de semer la confusion dans la communauté hacker. En 1979, Brian Reid décida en effet d'implémenter

9. *Ibid.* note : *Emacs Distribution.*

des « bombes à retardement » (*time bomb*) dans Scribe, permettant à l'entreprise Unilogic de limiter l'utilisation gratuite du logiciel. Un sombre présage pour Stallman. « Il considérait ça comme la chose la plus 'nazie' qu'il ait jamais vue au cours de sa vie », se rappelle Reid.

Bien que devenu aujourd'hui une grande figure de l'Internet, en tant que co-créateur de la hiérarchie Usenet *alt*, Reid assure qu'il porte encore le fardeau de cette décision de 1979, tout au moins aux yeux de Stallman. « Selon lui tous les logiciels devaient être libres et la simple volonté de faire payer un programme constituait un crime contre l'humanité. » [10]

Quoique impuissant à influer sur le marchandage entrepris par Reid, Stallman était en mesure de contrer les autres formes de comportement qu'il jugeait contraires à l'éthique. En tant que principal mainteneur du code source d'Emacs, il mit son pouvoir au service de ses desseins politiques. Dans les derniers temps du conflit l'opposant aux administrateurs du laboratoire d'informatique, au sujet des mots de passe, il entreprit une « grève » [11] du logiciel. Stallman refusa d'envoyer la dernière version d'Emacs aux membres tant qu'ils ne rejetaient pas le système de sécurité présent sur les ordinateurs. Il s'agissait davantage d'agitation politique que d'une sanction, car dans les faits rien n'empêchait les administrateurs d'installer ledit système. Malgré tout, le geste de Stallman donnait corps à une réprobation.

« Beaucoup de gens étaient en colère contre moi, disant que je tentais de les prendre en otage ou de les faire chanter. Ce que

10. Dans une interview du magazine online *MEME*, Stallman qualifia de rebutante la commercialisation de Scribe, mais il refusa de mentionner Reid nominativement. « Le problème est que personne n'a censuré ni puni cet étudiant pour ce qu'il a commis, affirma Stallman. Au final, d'autres furent tentés de suivre son exemple ». Voir [Bennahum and Stallman, 1996]

11. Levy, 1984, p.419

je faisais dans un sens, dira 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. »

⁂

Avec le temps, Emacs devint littéralement un argument de vente de l'éthique hacker. Non seulement la flexibilité insufflée par Stallman, au cœur même du logiciel, encourageait la collaboration, mais elle l'exigeait. Les usagers qui ne suivaient pas les derniers développements d'Emacs, ou qui ne renvoyaient pas leurs contributions à Stallman, couraient le risque de manquer les derniers aboutissements — et ils étaient nombreux. Vingt ans plus tard, les utilisateurs avaient modifié GNU Emacs (une seconde implémentation commencée en 1984) pour tant d'usages différents (tableur, calculateur, base de donnée et navigateur web) que ses derniers développeurs adoptèrent l'image d'un lavabo plein à ras bord pour représenter son inépuisable polyvalence. « C'est l'idée que nous voulions faire passer, dit Stallman, la quantité de trucs qu'il contient est à la fois merveilleuse et monstrueuse. »

Les contemporains de Stallman au AI Lab se montrent plus charitables. Hal Abelson, un étudiant du MIT qui travailla avec Sussman durant les années 1970 et assista plus tard Stallman au bureau de la 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 de collaboration à grande échelle. « Sa structure était si robuste que des gens du monde entier pouvaient y collaborer ou y

contribuer sans concertation préalable, dit Abelson. Je ne sais pas si cela avait jamais été réalisé auparavant. » [12].

Guy Steele est tout aussi admiratif. Actuellement chercheur pour Sun Microsystems, il se souvient surtout de Stallman comme « [d']un programmeur génial qui possède le talent de générer du code relativement dénué de bogues ». Bien que dotés de personnalités assez différentes, tous deux collaborèrent assez longtemps pour que Steele se fasse une idée du style de code très pointilleux que produisait Stallman. Il se souvient à ce titre d'un épisode marquant, à la fin des années 1970. Les deux programmeurs faisaient équipe pour rédiger la fonction *pretty print* (« belle impression ») de l'éditeur. À l'origine conçue par Steele, cette nouvelle propriété de frappe reformatait le code source d'Emacs de façon à ce qu'il soit plus lisible et moins volumineux, tout en mettant en évidence les qualités wysiwyg du programme. Une fonction suffisamment stratégique pour susciter un vif intérêt chez Stallman. Il fallut peu de temps pour que Steele annonçât que tous deux en préparaient une version améliorée.

« Nous nous sommes assis un matin, se rappelle Steele. J'étais au clavier et lui à mes côtés. Il était tout à fait d'accord pour me

12. En écrivant ce chapitre, j'ai été amené à insister davantage sur la dimension sociale d'Emacs que sur son aspect logiciel. Pour en savoir plus sur la partie logicielle, je recommande la lecture du mémo de 1979, réécrit en 1981, et tout particulièrement la section intitulée « *Research Through Development of Installed Tools* ». Non seulement cette partie est intelligible pour un non spécialiste, mais elle illustre aussi la manière dont pensée politique et conception de logiciels se mêlent intimement, chez Stallman. En voici un exemple : « On ne serait pas parvenu à créer Emacs à partir d'un plan initial, soigneusement élaboré. Un tel plan ne peut être défini qu'à partir des souhaits formulés avant que le projet ne soit entamé, des souhaits qui pourront à terme ne recouvrir que très peu les besoins auxquels on fera finalement face. Ni moi ni personne n'avions imaginé un éditeur extensible jusqu'à ce que j'en crée un, et personne 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. »

laisser taper, mais il me disait quoi. » La session de programmation dura dix heures. Pendant tout ce temps, Steele raconte que ni lui ni Stallman ne firent de pause ou n'échangèrent un mot. À la fin de la séance, ils avaient réussi à réduire le code source de la fonction de *pretty print* à un peu moins de cent lignes. « Mes doigts étaient tout le temps sur le clavier, se remémore 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. » Steele ne se rendit compte de la durée de la session qu'en quittant le AI Lab. À l'extérieur, il fut surpris de trouver l'obscurité de la nuit. En tant que programmeur, il était habitué aux sessions marathon mais celle-ci avait été différente. Travailler avec Stallman l'avait contraint à refouler tous les stimuli externes et à focaliser la totalité de son énergie intellectuelle sur leur tâche.

Avec le recul, il dit avoir trouvé la force mentale de Stallman aussi exaltante que terrifiante : « J'ai pensé, juste après coup, que ça avait été une grande expérience, très intense… mais que je ne voulais plus jamais revivre. »

Chapitre 7

Une morale à lépreuve

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 du jour, à minuit et demi exactement, et signé *rms@mit-oz*, l'objet du message était laconique mais attirait l'attention. « Nouvelle implémentation d'Unix », pouvait-on lire.

Pourtant, au lieu de présenter une version fraîchement disponible d'Unix, le premier paragraphe du message était en fait un appel à contribution [1] :

```
Dès le Thanksgiving prochain, je commencerai à écrire un système logiciel
complet, compatible Unix, appelé GNU (pour GNU N'est pas Unix), et le
```

1. Stallman, 1983 ; McKusick, 1999, p.38

```
distribuer librement à tous ceux qui souhaitent l'utiliser. Je fais appel
à toute contribution en temps, en argent, en programmes et en matériel pour
faire avancer ce projet.
```

Pour un développeur Unix expérimenté, le message traduisait un mélange d'idéalisme et d'orgueil démesuré. Non content de s'engager à repartir de zéro dans la reconstruction du système d'exploitation Unix déjà abouti, l'auteur proposait en plus de l'améliorer par endroits. Le nouveau système GNU, prédisait-il, intégrerait tous les composants essentiels : un éditeur de texte, un *shell* pour lancer des applications compatibles Unix, un compilateur, « et diverses autres choses » [2]. À cela s'ajouteraient de nombreuses fonctions particulièrement séduisantes, pas encore disponibles dans les autres systèmes Unix : une interface graphique basée sur le langage de programmation Lisp, un système de fichiers résistant aux pannes et des protocoles réseaux prenant modèle sur ceux 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, d'après notre expérience au contact d'autres systèmes d'exploitation. »

Prévoyant une réaction sceptique de la part de certains lecteurs, l'auteur poursuivait l'exposé de son ébauche de système d'exploitation avec une brève note biographique intitulée « Qui suis-je ? » [3] :

```
Je suis Richard Stallman, l'inventeur de l'éditeur Emacs si souvent imité,
et je travaille actuellement au Laboratoire d'intelligence artificielle
du MIT. J'ai beaucoup travaillé sur des compilateurs, des éditeurs, des
débogueurs, des interpréteurs de commandes, ainsi que sur l'ITS et le
système d'exploitation des machines Lisp. J'ai été le premier à mettre au
point un affichage indépendant du terminal pour ITS. De plus, j'ai mis
```

2. Stallman, 1983, *loc. cit.*
3. Stallman, 1983, *loc. cit.*

```
en place un système de fichiers résistant aux pannes et deux systèmes de
fenêtrage pour machines Lisp.
```

Le destin a finalement voulu que le projet fou de Stallman, le projet GNU, ne soit pas lancé en ce jour de Thanksgiving. Ce ne fut qu'en janvier 1984 que Stallman, tenant sa promesse, s'immergea entièrement dans le développement de programmes Unix. Pour un architecte logiciel nourri au lait de l'ITS, c'était comme concevoir un petit centre commercial en banlieue après avoir rêvé de palais mauresques. Néanmoins, bâtir un système d'exploitation de type Unix avait certains avantages insoupçonnés. Car malgré sa puissance, l'ITS avait également un talon d'Achille : les développeurs du MIT l'avaient écrit spécifiquement pour la puissante architecture du PDP-10. Quand les administrateurs du AI Lab choisirent de remplacer cette machine au début des années 1980, le système d'exploitation, comparé jusqu'alors par les hackers à une cité grouillante d'activité, ne fut subitement plus qu'une ville fantôme.

Unix, à l'inverse, avait été conçu pour le portage, ce qui l'immunisait contre un tel danger. À l'origine développé par les jeunes chercheurs d'AT&T, le programme avait échappé au contrôle des dirigeants de l'entreprise et trouvé la terre promise dans le monde des systèmes informatiques universitaires, moins soucieux des contingences matérielles. Disposant de ressources plus limitées que celles de leurs frères du MIT, les développeurs d'Unix avaient adapté le logiciel pour qu'il puisse piloter un assortiment de matériels des plus divers : le PDP-11 16 bits, tout d'abord, considéré par les hackers du AI Lab comme une machine adaptée uniquement aux petits travaux, mais aussi, plus tard, les unités centrales 32 bits telles le VAX 11/780. En 1983, quelques sociétés, parmi lesquelles Sun Microsystems, notamment, avaient lancé le développement d'une génération d'ordinateurs de bureau plus puissants baptisés « stations de travail », afin de tirer profit de ce

système d'exploitation toujours plus présent sur les machines de capacités équivalentes à l'ancien PDP-10.

Pour faciliter le portage, les développeurs d'Unix ajoutèrent une couche d'abstraction entre le logiciel et la machine. Plutôt que d'écrire du code spécifiquement pour une machine donnée — comme cela avait été le cas avec l'ITS et le PDP-10 — ils utilisèrent un langage de haut niveau appelé C. Ils se concentrèrent sur les interfaces et les spécifications liant entre eux les nombreux sous-composants du système d'exploitation, au lieu de s'attacher exclusivement aux composants eux-mêmes. Ainsi créèrent-ils un système qui pouvait facilement être adapté pour s'exécuter sur n'importe quel type de machine. Si un utilisateur n'était pas satisfait d'un composant, il lui était alors possible, via ces interfaces, d'en retirer une partie spécifique pour le corriger, ou de le remplacer. En bref, l'approche d'Unix mettait en avant flexibilité et économie, d'où son adoption rapide [4].

***

La décision d'amorcer le développement du système GNU avait été provoquée par l'abandon de l'ITS, système que les hackers du AI Lab avaient passé tant de temps à perfectionner. La mort du programme, qui signifait aussi celle de la communauté qui le soutenait, fut un coup dur pour Stallman. Si l'épisode de l'imprimante laser Xerox lui avait ouvert les yeux sur l'iniquité de la privatisation des logiciels, l'anéantissement de la communauté le mettait face à un choix cornélien : capituler et rejoindre le camp privateur, ou s'y opposer.

À l'instar de son code source, les causes de la mort de l'ITS étaient profondément enracinées dans le passé. En 1980, la plupart

4. Stallman, 1983, *loc. cit.* ; *Ibid.*, p.38

des hackers du AI Lab travaillaient déjà sur le développement de la machine Lisp et son système d'exploitation.

Inventé par John McCarthy, chercheur pionnier en intelligence artificielle au MIT à la fin des années 1950, Lisp était un langage élégant, bien adapté à l'écriture de programmes complexes pour le traitement de données à structure irrégulière. 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 perfectionnèrent le langage en créant un dialecte local dénommé MacLisp. Ce nom faisait référence au projet MAC, projet de recherche de la DARPA [5] qui avait donné naissance au AI Lab et au Laboratoire de sciences informatiques. Menés par Richard Greenblatt, ils fabriquèrent à la fin des années 1970 un ordinateur spécialement optimisé pour Lisp, la *machine Lisp*, avant de lui dédier un système d'exploitation complet, lui aussi basé sur Lisp.

Toujours en 1980, deux compagnies, formées par deux groupes rivaux de hackers, avaient vu le jour dans le but de fabriquer et de vendre des copies de la machine Lisp. Greenblatt avait lancé Lisp Machines Inc. (LMI), comptant se passer d'investissements extérieurs pour former une pure « compagnie de hackers ». Mais la plupart d'entre eux rejoignirent Symbolics Inc., start-up plus conventionnelle dirigée par Russell Noftsker, un ancien administrateur du AI Lab. Dès 1982, toutes ces recrues cessèrent de travailler au MIT.

Seuls quelques hackers étant restés pour tenir la boutique, les délais de maintenance des programmes et des machines

5. La DARPA (*Defense Advanced Research Projects Agency*) est l'agence du Département de la Défense des États-Unis chargée de la recherche et du développement des nouvelles technologies. Elle est réputée être à l'origine d'avancées technologiques majeures telles que le réseau Arpanet, ancêtre de l'Internet.

s'allongèrent — si tant est qu'ils fussent mis à jour. Pire encore, selon Stallman, le labo amorçait un « changement démographique ». Les hackers qui formaient autrefois une minorité active au sein du AI Lab étaient presque tous partis, tandis que « les professeurs et les étudiants qui n'aimaient pas vraiment le PDP-10 étaient tout aussi nombreux qu'auparavant. » [6]

C'est aussi en 1982 que le AI Lab reçut la machine qui devait remplacer son principal ordinateur, le PDP-10, alors âgé de plus de douze ans. Or si le modèle de Digital (DEC) alors en cours, le Decsystem 20, était compatible avec les programmes des utilisateurs, il aurait fallu récrire en profondeur l'ITS, ou du moins le « porter » à nouveau, pour que les hackers puissent continuer à l'utiliser. Craignant que le labo n'ait perdu un nombre trop important de ses programmeurs de talent, les membres de la faculté firent pression pour obtenir Twenex, un système d'exploitation commercial développé par Digital. Minoritaires, les hackers n'eurent d'autre choix que de s'incliner.

« Sans hackers pour maintenir le système, nous allons droit au désastre ; il nous faut un logiciel commercial. » Stallman se souviendrait de ces mots prononcés par les membres de la faculté quelques années plus tard. « 'Nous pouvons compter sur la société [Digital] pour en assurer la maintenance', affirmaient-ils. La suite a prouvé qu'ils se trompaient lourdement, mais c'est pourtant ça qu'ils ont fait. » [7]

Au début, les hackers considérèrent le système Twenex comme un nouveau symbole d'autorité ne demandant qu'à être détourné. Le nom même du système était une provocation. Officiellement dénommé TOPS-20 par DEC, il se présentait comme le successeur du TOPS-10, le système d'exploitation non libre distribué pour

6. Stallman, 1986
7. *Ibid.*

PDP-10 avec processeur KL-10 (un PDP-10 similaire à celui du AI Lab) Stanford Artificial Intelligence Laboratory, 1979.

le PDP-10. Or, en réalité, le TOPS-20 n'avait pas grand chose à voir avec le TOPS-10. C'était un dérivé du système Tenex que Bolt Beranek Newman avait développé pour le PDP-10[8]. Stallman, qui était à l'origine du nom « Twenex », explique qu'il l'a inventé pour éviter d'utiliser celui de TOPS-20. « Ce système était loin d'être le meilleur, il était donc hors de question de lui donner un nom aussi flatteur, se souvient-il. C'est pourquoi j'ai décidé de glisser un W dans son nom et de l'appeler Twenex. »

La machine sur laquelle tournait le système Twenex/TOPS-20 fut elle aussi dotée d'un sobriquet : Oz. Elle avait hérité ce surnom car elle avait besoin d'un petit PDP-11 pour faire fonctionner son terminal, dit la légende. Un hacker, voyant fonctionner pour la première fois le tandem KL-10—PDP-11 aurait fait référence à l'entrée en scène de l'orgueilleux magicien du film *Le Magicien*

8. Sources multiples : une interview de Richard Stallman, un e-mail de Gerald Sussmann, et le Jargon File 3.0.0. http://catb.org/jargon/html/T/TWENEX.html

d'Oz : « Je suis le grand, le puissant Oz, entonna-t-il, ne faites pas attention au PDP-11 qui se cache derrière la console ! » [9]

***

Si la découverte du KL-10 ne manqua pas de provoquer les rires des hackers, leur premier contact avec Twenex leur fit rapidement perdre tout sens de l'humour. Non content de se prévaloir d'une sécurité intégrée, le système de sécurité était devenu une obsession pour les ingénieurs qui avaient conçu les utilitaires et les applications. Avec Twenex, ce qui revenait autrefois à jouer au chat et à la souris avec les mots de passe dans le cadre du dispositif de sécurité en vigueur au laboratoire de sciences informatiques s'apparentait désormais à 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 refonte du code source. Malheureusement, le nombre d'entre eux ayant le temps et l'envie de procéder à ce type de vérification avait diminué, à tel point que c'est l'argument des administrateurs système qui eut le dessus.

La politique initiale voulait que tout membre du AI Lab ait les droits pour contourner les restrictions de sécurité. Or toute personne se dotant de ces privilèges était en mesure d'en priver n'importe quelle autre, qui se trouvait alors dans l'impossibilité de les restaurer. Naturellement, une poignée de hackers fut tentée de prendre le contrôle total du système en gardant pour eux seuls ces privilèges…

Détournant des mots de passe et à l'aide du débogueur lancé dès le démarrage, Stallman réussit à déjouer ces tentatives. Après

9. Voir http://www.as.cmu.edu/~geek/humor/See_Figure_1.txt

avoir mis en échec un second « coup d'état » de ce type, il envoya un message d'alerte à tout le personnel du AI Lab : « Une nouvelle tentative de prise de pouvoir a eu lieu, écrivit-il. Jusqu'ici, les forces aristocratiques ont été défaites. » Afin de protéger son identité, il signa son message « Radio Free OZ ».

Mais 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 des utilisateurs externes au AI Lab employaient son compte comme passerelle pour accéder à l'Arpanet — le réseau informatique financé par la recherche, ancêtre de l'Internet actuel. Parmi ces « touristes » du début des années 1980, on compte Don Hopkins, un programmeur de Californie. Grâce au téléphone arabe des réseaux de hackers, il avait appris que tout ce qu'un intrus avait à faire pour accéder au célèbre système ITS du MIT était d'ouvrir une session sous les initiales RMS et de répéter ce même monogramme de trois lettres en guise de mot de passe.

« Je serai éternellement reconnaissant au MIT de m'avoir, comme de nombreux autres, laissé utiliser librement ses ordinateurs, déclare Hopkins. Cela a beaucoup compté pour un grand nombre d'entre nous. »

Cependant, cette politique « touristique », ouvertement tolérée par les responsables du MIT au cours des années ITS [10], tourna court quand Oz devint le premier maillon de raccordement du labo à l'Arpanet. Les premiers temps, Stallman poursuivit sa stratégie qui consistait à reprendre son identifiant de connexion comme mot de passe, permettant ainsi aux utilisateurs extérieurs d'avoir un accès via son propre compte. Mais la fragilité d'Oz incita par la suite les administrateurs à interdire l'accès aux intrus qui, par pure maladresse ou par malveillance délibérée, auraient

10. Voir « MIT AI Lab Tourist Policy » http://www.art.net/~hopkins/Don/text/tourist-policy.html

pu endommager le système. Lorsque ces mêmes administrateurs exigèrent par la suite de Stallman qu'il cesse de diffuser son mot de passe, ce dernier, invoquant son éthique personnelle, préféra cesser complètement d'utiliser Oz.

« [Lorsque] les mots de passe ont fait leur première apparition au AI Lab, j'ai [décidé] de suivre ma conviction selon laquelle il ne devrait pas y avoir de mots de passe. Dans la mesure où je ne crois pas qu'il soit vraiment souhaitable d'avoir une quelconque sécurité sur un ordinateur, je ne vois pas pourquoi j'aurais dû collaborer avec ce régime sécuritaire. » [11]

⁂

Le refus de Stallman de s'incliner devant le grand et puissant Oz était représentatif de la tension croissante entre hackers et responsables du AI Lab au début des années 1980. Une tension qui allait malgré tout passer au second plan, occultée par le conflit qui faisait rage au sein même de la communauté des hackers… À l'arrivée du Decsystem 20, celle-ci était bel et bien divisée en deux camps, fédérés chacun autour d'une des sociétés LMI ou Symbolics.

Symbolics, grâce à ses investissements externes, avait recruté des développeurs du AI Lab et les faisait travailler à l'amélioration de divers composants du système d'exploitation de la machine Lisp, hors du giron du laboratoire. Fin 1980, la société avait recruté quatorze hackers du laboratoire comme conseillers à temps partiel pour le développement de sa version de la machine Lisp. Les quelques autres qui restaient se mirent à travailler pour LMI [12].

11. *Ibid.*
12. Levy, 1984, p.423

Quant à Stallman, préférant la vie sans contraintes du laboratoire et non enclin à prendre parti, il fit le choix de ne rejoindre aucune des deux sociétés.

Les premiers temps, les hackers continuèrent à passer une partie de leur temps au MIT, contribuant au développement de la machine Lisp. Les deux sociétés, Symbolics et LMI, avaient acquis du MIT la licence du code source. Cette licence les obligeait à retourner leurs modifications au MIT, sans toutefois les contraindre à laisser l'institut redistribuer leurs modifications. Malgré cela, au milieu de l'année 1981, elles donnèrent leur accord de principe autorisant cette redistribution. Ainsi, toutes leurs améliorations du système furent intégrées à la version du MIT et partagées entre tous les utilisateurs de machines Lisp. C'est ce qui permit à ceux restés au centre de conserver leur neutralité.

Or, le 16 mars 1982, date mémorable pour Stallman car c'était son anniversaire, les cadres de Symbolics mirent fin à cet accord. La raison de cette décision était simple : porter atteinte à LMI. En effet, du fait qu'elle employait un nombre plus restreint de programmeurs et de personnel en général, c'est LMI qui aurait tiré les principaux bénéfices du partage des améliorations, selon Symbolics. En mettant fin au partage du code, les cadres de Symbolics espéraient ainsi liquider leur rivale. C'est pourquoi ils décidèrent d'appliquer la licence à la lettre.

Désormais, au lieu de contribuer directement aux améliorations de la version du MIT, ce dont LMI pouvait profiter, ils se contentèrent de fournir une copie de leur version complète du système à l'usage des employés du MIT. Tout utilisateur se retrouvait alors à tester cette version dans l'unique intérêt de Symbolics et, s'il y apportait des améliorations, il était plus que probable qu'elles soient utiles avant tout à Symbolics.

***

En tant qu'administrateur responsable de la machine Lisp du laboratoire (assisté de Greenblatt, dans les premiers mois), Stallman était fou de rage. Les programmeurs de Symbolics avaient en effet laissé dans le code des centaines de modifications inachevées, à l'origine de nombreuses erreurs. Considérant cette annonce comme un ultimatum, il riposta en coupant la liaison sans fil entre la société et le laboratoire. Il se jura ensuite de ne jamais plus travailler sur la machine de Symbolics et s'engagea à poursuivre le développement du système du MIT, de manière à défendre LMI contre sa rivale. « De mon point de vue, le AI Lab était un pays neutre, tout comme la Belgique durant la Première Guerre mondiale, explique 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. »

Lorsque les cadres de Symbolics prirent conscience que leurs dernières fonctionnalités étaient encore implémentées systématiquement dans la machine Lisp du AI Lab et, par extension, dans celle de LMI, cela ne les enchanta guère. Mais Stallman, qui connaissait les implications de la loi sur le copyright, repartait de zéro dans la récriture des fonctionnalités. Tirant parti de la lecture du code source fourni au MIT par Symbolics, il faisait en sorte de comprendre les erreurs et leurs corrections de manière à produire un code source aussi différent que possible.

Inutile de dire que les cadres de Symbolics n'en croyaient pas un mot. Ils installèrent un programme espion sur la machine de Stallman afin d'obtenir des preuves contre lui. Pourtant, lorsqu'ils plaidèrent leur cause auprès de l'administration du MIT, au début de l'année 1983, les preuves avancées étaient bien minces : à peine une douzaine d'occurrences dans le code source où les deux versions apparaissaient modifiées de manière similaire.

Lorsque les administrateurs du AI Lab présentèrent ces preuves à Stallman, ce dernier les réfuta en démontrant que les similarités

dataient en fait d'avant la séparation. Il en profita pour retourner l'argument contre ses accusateurs : si, après les milliers de lignes de code qu'il avait réécrites, Symbolics n'avait pas de preuves plus convaincantes, cela démontrait qu'il s'était véritablement appliqué à éviter toute copie de code. Son travail ayant été approuvé par l'administration du AI Lab, il continua à intégrer les changements effectués par Symbolics jusqu'à la fin de l'année 1983 [13].

Stallman modifia néanmoins sa manière de procéder : « Pour plus de sûreté, j'ai cessé de lire leur code source [pour y chercher les nouvelles fonctionnalités ou changements majeurs]. Je ne me suis plus référé qu'à la documentation, en partant d'elle pour écrire le code. » Quant aux améliorations les plus importantes, il les concevait seul, sans attendre la publication de la nouvelle documentation. Une fois celle-ci disponible, il modifiait alors ses ajouts de façon à les rendre compatibles avec l'interface de Symbolics. Enfin, la lecture du code source fourni par la compagnie lui permettait d'y trouver la correction des bogues mineurs, et ainsi de les corriger à son tour, de manière différente.

Cette expérience conforta Stallman dans sa détermination. Tout à la création des nouvelles fonctions destinées à remplacer celles de Symbolics, il engagea les membres du AI Lab à poursuivre leur travail sur le sytème du MIT, de manière à fournir un flux soutenu de rapports de bogues. Il s'assura par la même occasion que les programmeurs de LMI disposent toujours d'un accès direct à ces modifications. « Je devais punir Symbolics coûte que coûte, serait-ce la dernière chose que je dusse faire », affirme Stallman.

De telles déclarations sont riches d'enseignement. Non seulement mettent-elles en lumière le tempérament quelque peu belli-

13. Le Livre de H. P. Newquist, *The Brain Makers*, affirme de manière erronée que le AI Lab avait demandé à Stallman de se tenir éloigné du projet de Machine Lisp.

ciste de Stallman, mais elles reflètent également l'intensité émotionnelle du conflit.

⁂

Si le désespoir de Stallman atteignait un tel degré, c'est qu'il ressentait comme la « destruction » de son « foyer », la disparition de la sous-culture très unie des hackers au AI Lab. Au cours d'un entretien par courrier électronique avec Levy, il allait par la suite se comparer à la figure historique d'Ishi, le dernier survivant des Yahis, une tribu du nord-ouest de la côte Pacifique qui fut décimée lors des guerres indiennes des années 1860 et 1870. Une analogie qui élève sa lutte pour la survie à un niveau épique, aux frontières du mythe [14].

Du côté de Symbolics, on envisageait les choses différemment. En effet, plutôt que de considérer leur entreprise comme une force exterminatrice, nombre des collègues de Stallman y voyaient l'occasion tardive de trouver enfin leur place. En commercialisant la machine Lisp, la compagnie exportait les principes d'ingénierie logicielle des hackers hors de la tour d'ivoire du AI Lab, pour les faire pénétrer le domaine commercial, où les principes de conception étaient dictés par des gestionnaires. Au lieu de voir en Stallman un résistant, beaucoup le considéraient comme un rétrograde.

À tout cela s'additionnaient des conflits personnels. Bien avant que Symbolics n'ait débauché la majeure partie des hackers du AI

14. Dans son livre *Hackers*, Steven Levy se réfère à cette période lorsqu'il écrit que Stallman est « le dernier des vrais hackers ». Mais ne nous y trompons pas : si Levy utilise l'expression « vrais hackers », c'est pour distinguer la communauté du MIT de deux autres communautés hackers qu'il décrit et nomme plus loin dans son livre. C'est lorsque celle du MIT fut dissoute, laissant Stallman isolé, qu'il devint le dernier des « vrais hackers », une expression que d'ailleurs jamais Stallman n'a employée pour lui-même.

Lab, Stallman indique que plusieurs d'entre eux l'évitaient déjà. « On ne m'invitait plus à Chinatown, se souvient-il. La coutume inaugurée par Greenblatt voulait que si vous partiez dîner en ville, vous fassiez le tour ou envoyiez un message pour proposer à tous ceux du laboratoire de vous accompagner. C'est vers 1980-1981 que l'on a cessé de m'inviter. Comme si cela ne suffisait pas, quelqu'un m'avoua même plus tard qu'on l'avait poussé à me mentir pour tenir leurs dîners sans moi secrets. »

Stallman avait été blessé par cet ostracisme mesquin, mais il ne pouvait rien y faire. L'ultimatum de Symbolics fit passer le problème d'un rejet personnel à une plus grande injustice. Lorsque la société cessa de redistribuer ses améliorations de code source à seule fin de mettre en faillite sa concurrente, Stallman décida de contre-attaquer. En se terrant dans les bureaux du MIT pour y réécrire chaque nouvelle fonctionnalité, chaque nouvel outil, il donnait à tous les utilisateurs de machines Lisp, y compris les clients de LMI, un accès aux mêmes fonctions que celles fournies par Symbolics.

⁂

Son statut de légende vivante n'en fut que renforcé au sein de la communauté. Déjà consacrée lors de 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), constitue l'un des plus grands exploits humains de l'ère de l'information.

Y voyant le « coup de maître [d'un] John Henry virtuel de la programmation », l'auteur Steven Levy remarque que nombre des rivaux de Stallman, chez Symbolics, ne pouvaient faire autrement que de tenir en estime, même à contrecœur, leur ancien camarade

idéaliste. Levy rapporte les propos de Bill Gosper, qui finira par travailler pour Symbolics dans les bureaux de Palo Alto, exprimant son émerveillement pour le travail fourni par son ex-collègue durant cette période :

« S'il aurait pu m'arriver de trouver mauvais quelque chose écrit par Stallman (cela est peu probable, mais quelqu'un aurait pu m'en convaincre…), eh bien même alors, j'aurais continué à dire : 'Attendez une minute ! Stallman n'avait personne avec qui débattre des nuits entières, là-bas. Il a travaillé seul ! C'est tout bonnement incroyable que quelqu'un ait pu réaliser tout cela tout seul !' » [15]

Les mois passés à lutter contre Symbolics laissèrent à Stallman un mélange de fierté et de profonde tristesse. Libéral convaincu dont le père avait servi pendant la Deuxième Guerre mondiale, il n'avait rien d'un pacifiste. De bien des manières, la guerre contre Symbolics représentait le rite de passage auquel il s'était préparé depuis qu'il avait rejoint le personnel du AI Lab, dix ans plus tôt.

Mais en même temps, ce conflit coïncidait avec la traumatisante destruction de la culture hacker, au sein de laquelle Stallman avait évolué depuis son adolescence. Un jour, raconte-t-il, alors qu'il faisait une pause dans l'écriture du code, il éprouva un sentiment bouleversant au moment de traverser la salle des machines du laboratoire. Il y faisait face à la carcasse massive du PDP-10 abandonné. Saisi par les voyants éteints, voyants qui autrefois clignotaient silencieusement pour indiquer l'état du programme en cours, il ressentit une émotion similaire à celle que lui aurait inspirée la vision d'un être cher défunt.

« Je me mis à pleurer d'un coup, là en plein milieu de la salle des machines. Voir cette machine-là, morte, sans plus personne

15. *Ibid.*, p.426

pour s'en occuper, me remémora ô combien ma communauté avait été détruite. »

Mais Stallman n'aurait guère le temps de pleurer. En dépit de toutes les passions qu'elle avait provoquées et de tout le travail consacré à son élaboration, la machine Lisp n'avait donné lieu qu'à une escarmouche sur le champ des grandes batailles en cours sur le marché des nouvelles technologies. La course implacable vers la miniaturisation des ordinateurs apportait avec elle de nouveaux microprocesseurs plus puissants, qui n'allaient pas tarder à contenir toutes les capacités matérielles et logicielles de la machine, à l'image d'une métropole moderne dévorant les ruines d'un vieux village fantôme.

***

Surfant sur cette vague du microprocesseur, des centaines, pour ne pas dire des milliers, de logiciels non libres, chacun protégé par un patchwork de licences utilisateur et de clauses de confidentialité, interdisaient aux développeurs l'examen ou le partage du code source. Les licences furent d'abord rudimentaires et mal taillées, mais en 1983, elles étaient devenues assez solides pour satisfaire les tribunaux et dissuader tout contrevenant potentiel. Le logiciel, autrefois considéré comme une forme de garniture offerte par les fabricants pour donner plus de saveur à leurs coûteux systèmes informatiques, devint rapidement le plat principal. Toujours plus affamés de jeux et de nouveautés, les utilisateurs en omettaient de demander la recette, comme on le fait traditionnellement après chaque repas.

Nulle part cet état de fait n'était plus évident qu'au royaume de l'ordinateur individuel. Des compagnies comme Apple ou Commodore faisaient la fortune de jeunes entrepreneurs grâce à la vente

de machines dotées de systèmes d'exploitation intégrés. Ignorant tout de la culture hacker et de son aversion pour le logiciel précompilé, la plupart des utilisateurs n'éprouvèrent pas le besoin de protester lorsque ces compagnies cessèrent de fournir leurs programmes accompagnés des fichiers contenant le code source. Et si quelques adeptes rebelles de l'éthique hacker réussirent à lui donner une petite place sur le nouveau marché, le fait est que ce dernier ne récompensait, pour l'essentiel, que les programmeurs suffisamment prompts pour créer de nouveaux logiciels et assez futés pour écrire des licences (CLUF) et en verrouiller l'usage.

⁂

Parmi les plus célèbres d'entre eux figurait Bill Gates. Ayant rapidement abandonné ses études à Harvard, Gates était un jeune chef d'entreprise, co-fondateur de la société de logiciels Micro-Soft (qui s'écrirait plus tard Microsoft), établie à Albuquerque. Bien que Stallman, de deux ans son aîné, ne l'ait pas su alors, Gates avait, sept ans avant l'envoi du message de *rms* au groupe *net.unix-wizards*, adressé lui aussi une lettre ouverte à la communauté des développeurs. Sa « lettre ouverte aux amateurs », du 3 février 1976, destinée aux utilisateurs de PC copiant les programmes de Micro-Soft, condamnait sans appel la notion de développement communautaire des logiciels.

« Qui peut se permettre d'effectuer un travail professionnel pour rien ? Quel amateur peut-il investir trois années de sa vie à développer, trouver tous les bogues et documenter son produit, pour ensuite le distribuer gratuitement ? » demandait Bill Gates [16].

16. Gates, 1976, Une copie de cette lettre *An open letter to hobbyists* figure sur Wikipedia (en anglais) : http://en.wikipedia.org/wiki/Open_Letter_to_Hobbyists

Bien que peu de hackers du AI Lab l'aient lue, cette lettre illustre néanmoins le changement d'attitude qui s'est opéré à l'égard des logiciels, tant de la part des compagnies qui les vendent que de celle des développeurs commerciaux. Pourquoi traiter le logiciel comme un produit sans valeur alors que le marché en a décidé autrement ?

Au tournant des années 1980, la vente de programmes ne constituait plus seulement un moyen d'amortir les coûts : c'était devenu un enjeu politique. À l'heure où l'administration Reagan s'empressait de démanteler nombre de règlements fédéraux et suspendait les projets publics 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-patriotique. Au mieux, c'était un retour aux attitudes anti-corporatistes de la fin des années 1960 et du début des années 1970. Tel un banquier de Wall Street découvrant un de ses vieux T-shirts hippies caché entre ses chemises Cardin et ses costumes trois-pièces, nombre d'informaticiens ne regardaient plus l'éthique des hackers que comme le rappel embarrassant d'une époque idéaliste.

Pour un homme qui avait vécu toutes les années 1960 tourné vers les années 1950, Stallman n'avait que faire de vivre en décalage par rapport à ses pairs. Cependant, programmeur habitué à travailler avec les meilleures machines et les meilleurs logiciels, il se retrouvait face à ce qu'il ne pouvait qu'appeler un « choix moral cornélien » : soit il ravalait ses objections contre les logiciels « propriétaires » — terme alors employé par Stallman et ses pairs pour qualifier les programmes dont les termes de copyright ou de licence en restreignent la copie ou la modification — soit il consacrait son existence à l'élaboration d'un autre système libre de logiciels.

Après deux années de lutte contre Symbolics, Stallman avait suffisamment gagné en confiance pour choisir la seconde solution.

« J'aurais pu tout aussi bien arrêter toute activité informatique, indique-t-il. Je ne suis pas spécialement doué, mais je suis sûr que j'aurais pu faire serveur. Probablement pas dans un restaurant chic, mais j'aurais bien trouvé une place quelque part. »

Mais laisser tomber complètement la programmation pour devenir serveur, cela aurait voulu dire abandonner une activité qui lui tenait tant à cœur. Stallman devait admettre que, depuis son entrée à Cambridge, le développement logiciel avait bien souvent été sa seule source de plaisir. Plutôt que de tout lâcher, il décida de s'accrocher.

***

Athée, Stallman refuse les notions de destin, de karma ou autre vocation divine, qui influeraient sur le cours de notre existence. Il a le réel sentiment, néanmoins, que sa décision de combattre les logiciels non libres en créant un système d'exploitation qui aiderait les autres à faire de même, est toute naturelle. Après tout, c'était bien la combinaison de son obstination, de son intuition et de sa virtuosité dans l'écriture du code, qui lui avait permis d'envisager cette alternative que nombre d'autres n'auraient même pas pu imaginer. Dans son article intitulé « Le projet GNU », il affirme son adhésion aux idéaux contenus dans les mots du sage juif Hillel : « Si je ne suis pas pour moi, qui le sera ? Et si je ne suis que pour moi, qui suis-je ? Et si pas maintenant, quand ? » [17]

En public, Stallman évite toute référence religieuse et relate sa décision en termes pragmatiques.

17. Stallman, 1999, p.56, Stallman ajoute une note de bas de page à ce propos : « Je suis athée et je n'ai d'obédience pour aucun chef religieux, mais j'ai parfois de l'admiration pour ce que l'un d'entre eux peut dire. »

« Je me suis demandé, que pourrais-je faire, moi, développeur de systèmes d'exploitation, pour améliorer la situation ? Ce n'est qu'après avoir longuement examiné la question que j'ai pris conscience qu'un développeur de systèmes d'exploitation était précisément ce qu'il fallait pour résoudre le problème. »

Dès lors, tout le reste allait, selon ses propres mots, « se mettre en place ». En 1983, le MIT fit l'acquisition, auprès de Symbolics, de machines Lisp de seconde génération. Sur ces modèles, le système interne n'avait aucune chance de fonctionner. Et une fois la plupart des machines remplacées, Stallman aurait dû tout arrêter, dans l'incapacité de maintenir le système de manière efficace, par manque de rapports de bogues de la part des utilisateurs. Quoi qu'il en fût, il ne souhaitait pas poursuivre. Le système du MIT n'était pas libre, et même si les utilisateurs avaient accès au code source, ils ne pouvaient pas le redistribuer. Sans compter qu'il avait réussi son pari : LMI avait survécu et développait désormais seule ses logiciels.

Plutôt que de passer sa vie à châtier ceux qui avaient détruit son ancienne communauté, Stallman préféra en fonder une nouvelle. Ainsi décida-t-il de refuser tout programme qui le forcerait à compromettre ses convictions morales, et de consacrer sa vie à la création de logiciels permettant d'éviter ce genre d'écueil, à lui comme aux autres. S'engageant à bâtir un système d'exploitation libre « ou [à] mourir à la tâche… de vieillesse, bien entendu », plaisantait-il, Stallman démissionna du MIT en janvier 1984, afin de se lancer dans le projet GNU.

Sa démission le soustrayait du giron légal du MIT. Mais il avait encore suffisamment d'amis et d'alliés au AI Lab pour conserver un accès gracieux à son bureau. Il parvint également à s'assurer des missions en tant que consultant externe pour soutenir les phases initiales du projet GNU.

En démissionnant, Stallman n'en avait pas moins tué dans l'œuf tout débat sur un éventuel conflit d'intérêt ou sur l'appartenance au MIT du logiciel en question. Lui que la crainte de l'isolement social avait conduit à se réfugier tant et plus au sein du AI Lab en tant que jeune adulte, construisait désormais un pare-feu légal entre cet environnement et lui.

Les premiers mois, il travailla également à l'écart de la communauté Unix. Bien que son annonce au groupe *net.unix-wizards* ait attiré des messages de sympathie, peu de volontaires s'engagèrent à rejoindre la croisade dès la première heure.

« La réaction de la communauté fut assez unanime, se rappelle Rich Morin, alors à la tête d'un groupe d'utilisateurs Unix. 'Quelle superbe idée ! disait-on. Maintenant, nous voulons voir votre code. Montrez-nous qu'on peut le faire'. »

⁂

Conscient de l'ampleur du travail, Stallman décida d'essayer de réutiliser des logiciels libres existants partout où cela était possible. Il commença par rechercher les programmes et outils libres susceptibles d'être convertis en programmes et outils GNU. L'un des premiers candidats fut un compilateur appelé Vuck, qui convertissait des programmes écrits en langage C, très populaire à l'époque, en code compréhensible pour une machine. Traduit du néerlandais, l'acronyme du programme signifiait « Kit de compilation de l'université libre ». Plein d'optimisme, Stallman s'enquit de savoir si le programme était libre. Quelle ne fut pas sa déception lorsque l'auteur l'informa que l'expression « université libre » faisait référence à la Vrije Universiteit Amsterdam (Université Libre d'Amsterdam) !

« Il m'a répondu, moqueur, que si l'université était libre, le compilateur, lui, ne l'était pas », se souvient Stallman. L'auteur de Vuck ne se contenta d'ailleurs pas d'un refus. Il suggéra même à son interlocuteur d'abandonner le projet GNU pour développer des modules complémentaires qui amélioreraient les ventes de Vuck, en prenant part aux bénéfices. « C'est pourquoi j'ai décidé que mon premier programme pour le projet GNU serait un compilateur multi-langage et multi-plate-forme » [18], ajoute Stallman.

À défaut de Vuck, il trouva un compilateur Pastel [19], écrit par des programmeurs du Lawrence Livermore National Lab. Ils lui en donnèrent une copie et, selon leurs dires, le compilateur était libre d'être copié et modifié. Malheureusement, le programme ne convenait pas, nécessitant des ressources bien trop importantes en mémoire. Il analysait en effet la totalité d'un fichier en mémoire centrale, ce qui avait pour effet d'empêcher les autres données de circuler tant que la compilation n'était pas terminée. Sur de gros ordinateurs, ce défaut de conception était pardonnable. Sur les systèmes Unix, en revanche, il constituait une barrière infranchissable, les machines 32 bits étant alors elles-mêmes souvent trop peu puissantes pour fournir autant de mémoire à un seul programme.

Dans un premier temps, Stallman fit faire au programme des progrès considérables. Il ajouta au compilateur un frontal compatible C et le testa sur le gros Vax, dont le système pouvait dédier davantage de place à la mémoire. Mais lorsqu'il tenta de le porter sur le processeur 68010 et investiga la raison du plantage, il comprit que le problème de taille de mémoire l'obligerait à récrire entièrement un nouveau compilateur à partir de zéro. C'est d'ailleurs ce qu'il finirait par faire plus tard, en créant le *GNU C Compiler*, plus connu sous le nom de GCC. Mais en 1984, il était

18. *Ibid.*, p.56

19. Le langage Pastel fut ainsi nommé ainsi car considéré comme une version altérée du langage Pascal (« Off-color Pascal »).

encore trop tôt pour aviser quoi faire avec le compilateur ; Stallman mit donc cette tâche en attente et préféra concentrer toute son attention sur le reste du projet GNU.

⁂

Ainsi commença-t-il, dès septembre 1984, le développement d'une version GNU d'Emacs, le programme qu'il avait lui-même maintenu une décennie durant. Au sein du monde Unix, les deux éditeurs de texte alors disponibles étaient *vi*, écrit par Bill Joy, cofondateur de Sun Microsystems, et *ed*, de Ken Thompson, des laboratoires Bell, par ailleurs co-créateur d'Unix. Tout aussi utiles que populaires, aucun de ces deux programmes n'offrait pourtant les possibilités d'extension sans limites d'Emacs.

En y repensant, Stallman déclare qu'il n'y avait rien de stratégique dans sa décision : « Je voulais un Emacs et c'était l'occasion d'en développer un. »

Une fois de plus, il avait trouvé un code préexistant grâce auquel il espérait gagner du temps. En écrivant une version Unix d'Emacs, il se retrouva vite sur les pas de James Gosling, un diplômé de Carnegie Mellon, auteur d'une version d'Emacs en C baptisée *Gosling Emacs* ou *Gosmacs*. La version de Gosling incluait un interpréteur de langage Lisp simplifié, appelé Mocklisp. Bien que son auteur ait placé Gosmacs sous copyright et vendu les droits à UniPress, une compagnie logicielle privée, Stallman fut conforté dans ses intentions par l'un de ses collègues qui avait participé aux premières phases de développement de Gosmacs. Celui-ci assurait en effet que Gosling, alors encore doctorant à Carnegie, lui avait donné par courriel la permission de distribuer sa propre version de Gosmacs, en échange de sa contribution au développement.

Stallman avait pensé en premier lieu ne modifier que les commandes utilisateur, afin d'offrir une compatibilité parfaite avec l'Emacs original tournant sur PDP-10. Pourtant, s'apercevant combien Mocklisp était faible en comparaison du vrai Lisp, il se sentit vite contraint d'en revenir à l'original. Tout naturellement, l'envie lui vint de récrire la plus grande partie du cœur de Gosmacs, mais d'une manière totalement différente, afin de tirer parti de la puissance et de la flexibilité de Lisp. Ainsi, dans le programme GNU Emacs disponible sur l'Internet au milieu de l'année 1985, seuls quelques fichiers contenaient encore du code de Gosmacs.

Cela n'empêcha pas que lorsqu'elle eut vent du projet, la firme UniPress nia les affirmations du développeur qui prétendait avoir reçu le droit de distribuer sa propre version du programme. Hélas, celui-ci ne put remettre la main sur l'ancien e-mail en question pour se défendre. Stallman coupa court en récrivant tout bonnement le code des quelques modules résiduels de Gosmacs.

Néanmoins, la simple idée que des développeurs puissent liquider leurs droits — et, au-delà, l'idée même qu'un développeur dispose de tels pouvoirs commerciaux — lui était intolérable. Dans le discours qu'il prononça en 1986 à l'Institut technique royal suédois, il montra du doigt l'incident d'UniPress comme un nouvel exemple des dangers liés à la privatisation des logiciels.

« Parfois, je me dis que l'une des meilleures choses que je pourrais faire dans ma vie serait de dégotter une quantité gigantesque de logiciels couverts par le secret commercial, et d'en distribuer des copies au coin de la rue. Ainsi, il n'y aurait plus de secret commercial, dit Stallman. Ce serait sans doute une manière beaucoup plus efficace d'offrir aux gens de nouveaux logiciels libres plutôt que de tout écrire moi-même ; mais les gens seraient encore trop lâches pour les prendre. »

⁂

En dépit de la tension qu'elle avait provoquée, la controverse sur le code de Gosling aida en fait Stallman et le mouvement pour le logiciel libre sur le long terme. Stallman fut ainsi forcé de trouver des solutions aux faiblesses de la « Commune Emacs » [20] et du contrat de confiance informel qui laissait la porte ouverte aux abus. Cela l'obligea aussi à préciser les objectifs politiques du mouvement du logiciel libre.

Suivant la sortie de GNU Emacs en 1985, Stallman publia « Le manifeste GNU », un complément à l'annonce initiale de septembre 1983. Une importante section de ce document était consacrée aux nombreux argumentaires déployés par les programmeurs des secteurs privés et universitaires pour justifier de la prolifération des logiciels privateurs. L'un des arguments avancés, en particulier, « Les programmeurs ne méritent-ils pas une récompense pour leur créativité ? », recevait une réponse qui contenait en elle toute la colère de Stallman à la suite de l'affaire Gosling : « Si quelque chose mérite une récompense, c'est bien la contribution à la société, écrivait Stallman. La créativité peut être un apport à la société, mais *uniquement dans le cas* 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 tout autant d'être punis s'ils en limitent l'utilisation. » [21]

Avec la publication de GNU Emacs, le projet GNU avait enfin du code à montrer. S'ensuivirent tous les soucis propres à n'importe quelle entreprise produisant du logiciel. En effet, de plus en plus de concepteurs Unix se frottaient au logiciel, et l'argent, les cadeaux ou les demandes de copies sur bande commencèrent à affluer. Pour gérer les aspects commerciaux, Stallman réunit quelques-uns de ses collègues pour former la *Free Software Foundation* (FSF), une organisation à but non lucratif pour aider le

20. Cf. chap. 6.
21. Stallman, 1985

projet GNU à atteindre son but le plus rapidement possible. Avec Stallman comme président, et quelques amis et collègues hackers comme membres du conseil, la FSF permit de donner une vitrine publique au projet.

Robert Chassell, à l'époque programmeur chez LMI, devint l'un des cinq membres du conseil de la FSF à la suite d'une conversation avec Stallman alors qu'ils dînaient ensemble. Il endossa également le rôle de trésorier de l'organisation, un rôle initialement modeste mais qui prit rapidement en importance.

« Je pense qu'en 1985, le total de nos transactions, recettes et dépenses comprises, s'élevait à environ 23 000 dollars, évoque Chassell. Richard avait son bureau et, pour le reste, nous occupions tout l'espace disponible. Tout le fourbi, et surtout les bandes, était rangé sous mon bureau. Cela dura jusqu'à ce que, un peu plus tard, LMI nous prête un local pour stocker les bandes et autres choses de ce genre. »

Au-delà de son rôle de vitrine, la FSF était devenue également le centre de gravitation de tous les programmeurs désabusés. Sur le marché d'Unix, d'apparence encore si collégiale, même lors de l'annonce initiale du projet GNU, la concurrence faisait rage. Tentant de resserrer leur emprise sur les clients, des compagnies commençaient à refuser l'accès au code source d'Unix, tendance qui ne fit qu'accroître le nombre de requêtes concernant les projets de logiciels GNU en cours. Les Unixiens de la première heure, qui avaient par le passé considéré Stallman comme un hurluberlu turbulent, commencèrent à voir en lui un prophète ou une Cassandre du logiciel, selon l'état d'esprit — espérance ou désespoir — avec lequel chacun d'entre eux abordait l'issue du problème.

« Beaucoup ne réalisent pas, jusqu'à ce que cela leur arrive, la frustration qu'il peut y avoir à travailler plusieurs années sur

un programme pour se le voir finalement enlever, explique Chassell, résumant par là les sentiments et opinions exprimés dans la correspondance reçue par la FSF pendant les premières années. Quand cela vous arrive une seconde fois, vous commencez à vous dire que cela commence à bien faire ! »

C'est l'expérience de sa propre spoliation qui décida Chassel à prendre part à la FSF. Avant de travailler pour LMI, il avait été engagé pour écrire un livre d'introduction à Unix, pour Cadmus, une compagnie de logiciels située dans la région de Cambridge. Mais lorsque l'entreprise ferma ses portes, emportant dans sa chute les droits du livre, Chassell tenta en vain de les racheter.

« Pour ce que j'en sais, raconte-t-il, ce livre repose toujours sur une étagère quelque part, inutilisable, incopiable, tout simplement hors système. C'était vraiment une bonne introduction, si je puis me permettre. Cela n'aurait pris que trois ou quatre mois, peut-être, pour la 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 le fruit de son labeur partir à vau-l'eau pendant que son employeur d'autrefois luttait contre la faillite, Chassell déclare avoir eu alors un avant-goût de la colère qui menait Stallman au bord de l'apoplexie.

« Le plus clair à mes yeux, c'est que si on veut avoir une vie convenable, on ne peut accepter de s'en voir interdire l'accès à certains fragments. Toute cette idée d'être libre d'entrer, de rectifier quelque chose et de le modifier, quoi que cela puisse être, voilà qui change tout ! On peut alors se réjouir du fait qu'après avoir vécu quelques années, ce que l'on a accompli reste valable. Parce que sans cette liberté, votre travail est juste bon à vous être retiré, pour être ensuite rejeté, abandonné ou tout du moins dénué de tout lien avec votre personne. C'est un peu comme perdre un bout de sa vie. »

Chapitre

8

Sur scène avec Saint Ignucius

Au beau milieu des collines poussiéreuses qui surplombent la ville de Kihei à Hawaï, un bâtiment de plain-pied abrite le centre de calcul de haute performance de Maui, ou MHPCC (*Maui High Performance Computing Center*). Entre ce paysage hors de prix et les habitations luxueuses du Silversword Golf Course, le centre fait d'abord penser à une énième gabegie scientifico-financière. Loin des confinements étroits de Tech Square, ou encore d'une vaste métropole de recherche comme Argonne dans l'Illinois, ou Los Alamos au Nouveau Mexique, le Centre de calcul semble accueillir des scientifiques plus soucieux de leur bronzage que de leurs recherches post-doctorales.

Une image partiellement fausse. Bien que les chercheurs du centre profitent effectivement des distractions locales, ils n'en

prennent pas moins leur travail très au sérieux. Selon *Top500.org*, un site web qui référence les plus puissants supercalculateurs de 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 ordinateurs les plus puissants du monde. Propriété détenue en commun par l'Université de 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.

Bref, le MHPCC est un lieu unique, où la sophistication du génie scientifique et la traditionnelle affabilité des îles hawaïennes cohabitent dans un calme équilibre culturel. En 2000, le site web du centre résume tout d'un seul slogan : « Calculer au paradis. »

Pas vraiment le type d'endroit où l'on s'attend à trouver Richard Stallman. Admirant la magnifique vue de la passe de Maui à travers la fenêtre du bureau d'un membre de l'équipe, l'homme murmure une brève critique : « Trop de soleil ». Reste que, en tant qu'émissaire faisant le voyage entre deux paradis de l'informatique, Stallman a un message à faire passer, même si cela implique d'exposer ses yeux de hacker aux douloureux rayons du soleil.

⁂

La salle de conférence est déjà bondée lorsque j'arrive pour assister au discours de Stallman. La proportion entre les deux sexes est sensiblement plus équilibrée qu'à la conférence de New York (85 % d'hommes et 15 % de femmes), mais elle reste globalement similaire. La moitié environ de l'assistance porte des pantalons de toile et des polos frappés de logos. L'autre moitié semble retournée aux sources hawaïennes. Vêtus des ostentatoires

chemises à fleurs si populaires dans ce coin du monde, les visages sont bronzés à l'extrême. Les seuls indices révélant leur statut de geeks sont les gadgets : téléphones Nokia, Palm Pilots, et ordinateurs portables Sony VAIO.

Il va sans dire que Stallman, qui se tient devant cette assemblée vêtu d'un simple tee-shirt bleu, d'un *baggy* marron et de chaussettes blanches, fait tache. L'éclairage au néon du lieu souligne la couleur maladive de cette peau qui ne voit le soleil que rarement [1]. Sa barbe et ses cheveux sont tels qu'ils inonderaient de sueur le plus frais des cous hawaïens. Le mot « métropolitain » quasiment tatoué sur son front, il pourrait difficilement paraître plus étranger, même s'il le souhaitait [2].

***

Alors que Stallman s'affaire du côté de la scène, quelques membres du public, vêtus de tee-shirts aux couleurs du Maui FreeBSD User Group (Groupe d'utilisateurs de FreeBSD de Maui) s'empressent d'installer l'équipement audio et vidéo. FreeBSD est un rejeton libre de la distribution logicielle de Berkeley (BSD), la vénérable et académique version Unix des années soixante-dix. Techniquement, FreeBSD représente un compétiteur du système d'exploitation GNU/Linux. Cela n'empêche que, dans le monde du logiciel libre, les discours de Stallman sont accueillis avec une ferveur presqu'égale à celle que rencontre le groupe Grateful Dead, et sa légendaire armée d'archivistes amateurs. En tant

1. Richard Stallman tient à préciser que « l'idée que la peau a soif de soleil ou que le teint pâle est signe de mauvaise santé relève de la désinformation dangereuse. Rester à l'abri du soleil n'est pas nuisible tant qu'on n'a pas de carence en vitamine D. »

2. Richard Stallman ajoute : « Qu'y a-t-il de mal à paraître différent des autres ? »

qu'organisateurs locaux, il échoit aux membres du MFUG de s'assurer que les collègues programmeurs à Hambourg, Mumbai, et Novosibirsk ne ratent rien des derniers trésors de sagesse de RMS.

La comparaison avec les Grateful Dead est appropriée. Stallman y recourt souvent pour décrire les possibilités commerciales inhérentes au logiciel libre. En autorisant les fans à enregistrer leurs concerts publics, le groupe est devenu plus qu'un groupe de rock ; il est devenu le centre d'une communauté tribale gravitant autour de leur musique. Au cours du temps, cette communauté est devenue si importante et dévouée, que le groupe a pu se passer de contrat avec une maison de disque, pour se financer avec les seuls tournées et concerts *live*. En 1994, pour sa dernière tournée, le groupe a levé 52 millions de dollars grâce aux seules entrées des représentations publiques [3].

Alors que peu d'entreprises informatiques privées ont réussi à égaler un tel succès financier, la structure tribale de la communauté du logiciel libre en a amené beaucoup, au cours des années 1990, à accepter la publication du code source comme une bonne chose. En espérant pouvoir se constituer de la même façon de fidèles suites de partisans, des entreprises comme IBM, Sun Microsystems et Hewlett Packard en sont venues à accepter la lettre, sinon l'esprit, du message de Stallman.

Le chroniqueur informatique de ZDNet, Evan Leibovitch, décrit la GPL comme la « Magna Carta » des industries œuvrant dans les technologies de l'information. Selon lui, cet intérêt croissant pour le projet GNU constitue plus qu'une simple tendance : « Ce changement sociétal laisse les utilisateurs prendre en main leur avenir. Tout comme la Magna Carta a donné des droits aux

3. Voir : « Grateful Dead Time Capsule : 1985-1995 North American Tour Grosses » http://www.accessplace.com/gdtc/1197.htm.

sujets de l'empire britannique, la GPL définit les droits et libertés des utilisateurs de logiciels informatiques. » [4]

L'organisation de la communauté du logiciel libre explique aussi pourquoi tous ces programmeurs dans la quarantaine, qui pourraient être en train de travailler sur des projets de physique ou encore parcourir l'Internet à la recherche d'informations météorologiques pour planche à voile, 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 se présente d'ailleurs pas lui-même. Quand le matériel des gens de FreeBSD est finalement mis en route, il se contente de s'avancer, commence à parler, sa voix couvrant bientôt toutes les autres.

« La plupart du temps, quand un débat a lieu sur les règles qu'une société devrait adopter pour encadrer l'utilisation des logiciels, les intervenants sont des éditeurs logiciels dont l'avis est tout sauf désintéressé, commence-t-il. La question qu'ils se posent est 'Quelles règles imposer aux masses pour les obliger à nous verser beaucoup d'argent ?' Mais j'ai eu la chance, dans les années 1970, de faire partie d'une communauté de programmeurs qui s'échangeaient leurs logiciels. Cela me fait considérer la question sous un angle inédit, et je préfère demander 'quelles sont les règles qui aboutiraient à une société bénéfique pour les gens qui la composent ?' Bien sûr, j'arrive à des conclusions complètement différentes. »

4. Voir [Leibovitch, 2000]. La Magna Carta est une charte de 63 articles arrachée en 1215 au roi Jean Sans Terre, qui limite l'arbitraire royal et empêche, entre autres, l'emprisonnement arbitraire, en établissant l'*habeas corpus*.

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, 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'. Cela serait un sage conseil si c'était le cas. Mais le but n'est *pas* de construire un système d'exploitation ; le but est de *libérer* les utilisateurs d'ordinateurs. Et pour cela, nous devons leur permettre de tout faire avec leur ordinateur, en toute liberté. » [5]

« Il reste énormément de travail à faire », ajoute-t-il.

Certains dans le public connaissent déjà tout cela. Pour d'autres, cela reste un peu obscur. Quand l'un des porteurs de polos commence à piquer du nez, Stallman arrête son discours et demande à quelqu'un de réveiller l'étourdi. « Quelqu'un m'a dit un jour que ma voix était si apaisante qu'il se demandait si je n'étais pas une sorte de guérisseur, dit-il, provoquant les rires du public. Je pense que ça veut dire que je peux vous aider à glisser doucement dans un sommeil délicieux et paisible. Et certains d'entre vous en ont peut-être besoin. Peut-être que je ne devrais

5. Pour des raisons narratives, j'ai hésité à entrer dans les détails concernant la définition complète que donne Stallman de la « liberté » logicielle. Le site Internet du projet GNU énumère quatre points fondamentaux : la liberté d'utiliser le programme comme on le souhaite, pour quelque but que ce soit (liberté 0) ; celle d'étudier le code source du programme et de le modifier pour qu'il fasse ce que l'on souhaite (liberté 1) ; celle de distribuer des copies du programme afin d'aider son prochain (liberté 2) ; celle de distribuer des copies des versions que l'on a modifiées, afin que la communauté entière en bénéficie (liberté 3). Pour plus d'informations, consultez la définition du logiciel libre : http ://www.gnu.org/philosophy/free-sw.html.

pas m'y opposer. Si vous avez besoin de dormir, alors faites-le, je vous en prie. »

⁂

Le discours s'achève par une discussion rapide sur les brevets logiciels, un problème de plus en plus préoccupant, qui touche à la fois l'industrie informatique et la communauté du logiciel libre. Comme dans le cas de Napster, les brevets logiciels démontrent la difficulté d'appliquer des concepts créés pour le monde physique au nouvel univers dématérialisé des technologies de l'information.

Le droit du copyright et le droit du brevet ne fonctionnent pas de la même façon, et ont des effets complètement différents dans le domaine du logiciel.

Le copyright d'un programme détermine qui a le droit d'en copier et d'en adapter le code, et il appartient à son développeur. Mais il ne couvre pas les idées. Autrement dit, le copyright n'empêche pas un programmeur de mettre en œuvre dans son propre code des fonctions et des commandes inspirées de programmes existants. Les fonctions et les commandes sont des idées, non des œuvres de l'esprit, et aucun copyright ne s'y applique donc.

Il est tout aussi légal — bien que laborieux — de décoder la manière dont un programme binaire fonctionne, et d'implémenter ensuite les mêmes idées et algorithmes dans un code différent. Cette pratique est plus connue sous le nom de rétro-ingénierie.

Les brevets logiciels fonctionnent différemment. Selon le bureau américain des brevets (*U.S. Patent Office*), les entreprises ou les individus peuvent enregistrer des brevets pour des idées novatrices dans le domaine informatique (ou, du moins, des idées encore

non enregistrées). En théorie, cela permet au possesseur du brevet de retarder la diffusion des informations techniques de son invention, en lui assurant un monopole limité à vingt ans à compter de la date d'enregistrement du brevet. En pratique, la divulgation de ces informations par le détenteur du brevet n'a qu'une valeur restreinte, puisque la façon de fonctionner du programme est souvent évidente et peut toujours être analysée par rétro-ingénierie. Mais contrairement au copyright, un brevet autorise son possesseur à empêcher le développement indépendant de tout logiciel doté de fonctionnalités utilisant l'idée brevetée.

Dans l'industrie du logiciel, où vingt années peuvent couvrir tout le cycle de vie d'un marché, les brevets ont une importance stratégique.

Alors que des entreprises comme Microsoft et Apple s'affrontaient sur le copyright et sur « l'apparence et l'ergonomie » de diverses technologies, les entreprises actuelles de l'Internet utilisent les brevets pour préempter les applications individuelles et les *business models* individuels — pour preuve la tentative d'Amazon, en 2000, de faire breveter son processus d'achat en ligne en « un-clic » (*one-click*).

Pour la plupart des entreprises, cependant, les brevets logiciels sont devenus des armes défensives, utilisées comme contrepartie dans des accords bilatéraux où les portefeuilles de brevets s'équilibrent l'un l'autre, sorte de forme tendue d'apaisement diplomatique interentreprises.

Malgré tout, dans quelques cas notables d'algorithmes de chiffrement ou de compression graphique, certains éditeurs logiciels ont réussi à étouffer l'innovation chez la concurrence. C'est ainsi que certaines fonctions de rendu de polices de caractères n'existent pas en logiciel libre en raison de la menace que fait peser Apple en invoquant ses brevets.

Pour Stallman, la problématique des brevets logiciels rend plus vitale que jamais la vigilance permanente des hackers. Cela souligne aussi l'importance de faire valoir les bénéfices des logiciels libres en termes politiques plutôt qu'en termes d'avantages concurrentiels. Il affirme que la performance et le prix, deux critères sur lesquels des systèmes d'exploitation libres comme GNU/Linux et FreeBSD détiennent déjà un avantage conséquent sur leurs alter ego commerciaux, sont secondaires au regard du problème plus large des entraves imposées aux utilisateurs et aux développeurs.

***

Et voilà justement un sujet à controverse au sein de la communauté : les partisans du mouvement open source insistent davantage sur les avantages fonctionnels des logiciels libres que sur leurs implications politiques. Ils ont choisi de mettre l'accent sur l'efficacité technique du modèle de développement hacker. Invoquant la puissance de la revue par les pairs, l'argumentaire de l'open source décrit des programmes comme GNU/Linux ou FreeBSD comme étant mieux conçus, mieux audités et, partant, plus fiables pour l'utilisateur lambda.

Cela ne signifie pas pour autant que l'expression open source n'a pas d'implications politiques. Pour ses défenseurs, elle sert deux objectifs. D'abord, elle élimine l'ambiguïté du mot *free* qui veut à la fois dire « libre » mais aussi « gratuit », les entreprises l'interprétant invariablement comme « à coût nul ». Deuxièmement, elle permet aux entreprises de considérer le phénomène du logiciel libre sur le seul plan technique, et non éthique.

Eric Raymond, cofondateur de l'*Open Source Initiative*, et l'un des plus importants hackers à avoir adopté le terme « open

source », a clairement expliqué son refus de suivre le chemin politique de Stallman dans son article de 1999 intitulé « Tais-toi et montre-leur le code ».

« La rhétorique de Richard Stallman est très séduisante pour des gens comme nous, les hackers. Nous sommes des penseurs et des idéalistes et quiconque en appelle aux « principes », à la « liberté » et aux « droits » trouve facilement un écho chez nous. Même si nous sommes en désaccord avec certains détails de son programme, nous voudrions que la rhétorique de Stallman fonctionne, nous pensons qu'elle devrait fonctionner, mais nous sommes perplexes et incrédules quand elle échoue sur 95% de la population, qui ne fonctionne pas comme nous. » [6]

Parmi ces 95%, écrit Raymond, on retrouve la majeure partie des managers, investisseurs et simples utilisateurs d'ordinateurs qui, du simple fait de leur nombre, définissent 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 Richard Stallman insiste pour que nous parlions des 'droits des utilisateurs d'ordinateurs', il nous soumet à la dangereuse tentation de répéter des erreurs du passé. Nous devrions rejeter cette idée — non parce qu'elle serait fausse dans son principe, mais parce que ce type de discours, appliqué au logiciel, ne convainc personne excepté nous. En réalité, il déroute et fait fuir la plupart des gens étrangers à notre culture. » [7]

Stallman, quant à lui, rejette cette interprétation : « La tentative de Raymond d'expliquer notre échec est trompeuse parce que nous n'avons pas échoué. Notre dessein est grand, et il reste

6. Raymond, 1999a
7. *Ibid.*

beaucoup de chemin pour l'atteindre, mais nous en avons déjà parcouru une grande partie. Il est exagérément pessimiste sur le sujet des valeurs que ne partageraient pas les non-hackers. Beaucoup d'utilisateurs se sentent davantage concernés par les enjeux politiques sur lesquels nous mettons l'accent que par les arguments techniques que met en avant l'open source. C'est souvent le cas des dirigeants politiques, bien que ce ne soit pas le cas dans tous les pays.

C'est bien l'éthique du logiciel libre, et non la recherche du 'meilleur logiciel', qui a persuadé les chefs d'état de l'Équateur et du Brésil de faire passer les institutions gouvernementales au logiciel libre. Ils ne sont pas des *geeks* et pourtant, ils comprennent ce que signifie la liberté. »

Le principal défaut de l'argumentaire open source, selon Stallman, est qu'il conduit à de bien plus faibles conclusions. Il convainc la plupart des utilisateurs d'exécuter quelques logiciels libres, sans toutefois leur donner de bonne raison de migrer entièrement vers le logiciel libre. Ce compromis ne leur donne de liberté que partiellement, mais ne leur permet pas de reconnaître et d'estimer en tant que telle cette liberté. Ils sont donc susceptibles d'y renoncer et de la perdre. Par exemple, que se passe-t-il lorsque l'amélioration d'un logiciel libre est bloquée par un brevet ?

Les partisans de l'open source sont en général au moins aussi véhéments que Stallman dans leur opposition aux brevets logiciels. Il en va de même, d'ailleurs, des développeurs de logiciels privateurs puisque la brevetabilité des logiciels menace également leurs projets. Stallman fait cependant remarquer que la logique des brevets restreint l'espace des fonctionnalités qui sont à la portée de ceux qui développent des logiciels, et montre ainsi en quoi le point de vue du logiciel libre diffère de celui de l'open source sur la question.

« Améliorer les logiciels n'est plus une question de talent, dit Stallman, mais d'interdiction ou d'autorisation. Quelqu'un nous interdit de servir l'intérêt général. Alors, que se passera-t-il quand les utilisateurs verront ces manques fonctionnels dans les logiciels libres ? Eh bien, si le mouvement open source les a persuadés que ces libertés sont souhaitables juste parce qu'elles permettent des logiciels plus puissants et plus fiables, ils diront : 'Vous ne tenez pas vos promesses. Tel logiciel n'est pas plus puissant. Il lui manque telle fonctionnalité. Vous m'avez menti'. Au contraire, s'ils en sont venus à comprendre, avec le mouvement du logiciel libre, que c'est la liberté qui est importante en elle-même, alors ils diront : 'Comment ces gens osent-ils m'empêcher d'avoir cette fonctionnalité et restreindre ma liberté ?' Or c'est ce type de réponse qui nous permettra de survivre aux coups qui nous viseront lorsque ces brevets se mettront à proliférer. »

En regardant Stallman dérouler en personne son argumentaire politique, il est difficile de voir quoi que ce soit de confus ou de déplaisant. Son apparence physique peut certes sembler peu engageante, mais son message est logique. Quand un membre du public demande si, en se détournant des logiciels privateurs, les partisans du logiciel libre ne renoncent pas aux plus récentes avancées de la technologie, Stallman répond selon ses convictions.

« Je pense que la liberté est plus importante que les avancées technologiques. Je choisirai toujours un logiciel libre moins avancé du point de vue technologique à un logiciel non libre plus avancé parce ça ne vaudrait pas que je renonce à ma liberté. Je m'en tiens à cette règle : je ne prends rien que je ne puisse partager. »

Ceux qui confondent éthique et religion voient dans ce discours une preuve de plus de la nature quasi-religieuse du message de Stallman. Mais contrairement aux pratiquants religieux, Stallman n'obéit pas à un commandement ; il refuse juste de céder sa liberté. Et son discours ne fait qu'en expliquer les conditions pratiques : les

programmes non libres privent de liberté ; ceux qui veulent rester libres doivent donc les rejeter.

⁂

Stallman présente son raisonnement comme un choix personnel, qu'il espère donner l'envie de partager. Pour se démarquer des évangélistes logiciels, il se refuse à imposer ses préceptes à son auditoire (quoique, encore une fois, le public de Stallman quitte rarement la salle sans se sentir éclairé sur ce qu'il serait juste de faire en matière de logiciel).

Comme pour bien le faire comprendre, Stallman achève son discours d'un rituel iconoclaste. Sortant d'un sac plastique une robe noire, il l'enfile. Puis il sort un disque informatique marron brillant et le place sur sa tête. Le public laisse échapper un rire interloqué.

« Je suis Saint IGNUcius de l'église d'Emacs » dit-il, levant sa main droite et simulant une bénédiction. « Je bénis ton ordinateur, mon fils. »

Les rires deviennent applaudissements à tout rompre après quelques secondes. Le disque sur la tête de Stallman accroche alors la lumière d'un spot de la scène, révélant une auréole parfaite. En un clin d'œil, Stallman se met à ressembler à une icône religieuse russe.

« Au début, Emacs était un éditeur de texte », dit Stallman, expliquant la panoplie. « Puis c'est devenu un style de vie pour beaucoup, et pour certains une religion, que nous appelons l'Église d'Emacs. »

Le sketch est un moment de légèreté et d'autodérision, une réponse pleine d'humour à ceux, nombreux, qui pourraient voir en l'ascétisme logiciel de Stallman une forme déguisée de fanatisme religieux. C'est aussi le signe qu'il se montre tel qu'il est. Comme si, en enfilant cette robe et cette auréole, il rassurait finalement le public et disait : « Riez, je sais que je suis bizarre. »

> *Rire de la bizarrerie de quelqu'un est grossier et il n'est pas dans mes intentions de cautionner cela. J'espère que les gens rient à cause de mon sempiternel numéro de St. IGNUcius.*

Évoquant plus tard le personnage de Saint 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 la lutte pour le *leadership* de la communauté hacker. Il cherchait à l'époque 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'est pas ce fanatique que certains dépeignaient. Ce n'est que plus tard, ajoute-t-il, que d'autres utilisèrent le personnage pour stigmatiser son image d'idéologue du logiciel.

C'est ce que fit Eric Raymond en 1999 dans une interview 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 un côté cabotin. L'avez-vous déjà vu dans son accoutrement de St. IGNUcius, bénissant le logiciel avec un disque en guise d'auréole ? C'est principalement inconscient ; il a simplement trouvé la manière d'agacer juste ce qu'il faut, et d'attirer l'attention du public sans (généralement) trop l'effrayer. » [8]

8. Raymond, 1999c

Stallman est en désaccord avec l'analyse de Raymond. « Ce n'est qu'une façon de rire de moi-même, dit-il. Si certains y voient plus que cela, cela reflète leurs intentions, pas les miennes. »

Cela dit, l'homme avoue aimer être en représentation. « Vous plaisantez ? dit-il à un moment. J'adore attirer l'attention ». Pour cela, il s'est inscrit un jour à *Toastmaster*, une organisation qui aide ses membres à améliorer la façon dont ils parlent en public — Stallman la recommande vivement. Il possède une présence scénique qui rendrait jaloux la plupart des acteurs et rappelle à sa façon un ancien acteur de variétés. Quelques jours après la présentation au MHPCC, je fis allusion à sa prestation à LinuxWorld 1999, et lui demandai s'il avait le complexe de Groucho Marx — c'est-à-dire la réticence à faire partie de tout club qui aimerait l'avoir comme l'un de ses membres. La réponse de Stallman est immédiate : « Non, mais j'admire Groucho Marx pour de nombreuses raisons. Il m'a sûrement inspiré quelque part. Mais j'ai aussi été influencé par Harpo sur d'autres points. »

Williams se fourvoie en donnant une interprétation psychologique à cette célèbre phrase de Groucho. Cette phrase vise les nombreux clubs ouvertement antisémites qui refusaient de l'admettre parmi leurs membres. Je l'ignorais aussi jusqu'à ce que ma mère me l'apprenne. Williams et moi avons grandi à une époque où la bigoterie était moins tolérée et où il n'était plus nécessaire de recourir à l'humour pour en faire la critique.

L'influence de Groucho Marx est évidente dans le goût qu'il a toujours eu pour les calembours — qu'il partage, encore une fois, avec la plupart des hackers.

L'aspect de la personnalité de Stallman le rapprochant le plus de Groucho Marx est sans doute son talent de pince-sans-rire, et le

sérieux avec lequel il lance ses blagues. La plupart de ses blagues arrivent furtivement, sans même l'indice d'un mouvement de sourcil ou l'esquisse d'un sourire, au point qu'on en vient presque à se demander si Stallman ne rit pas plus de son public que ce dernier ne rit de lui.

En voyant les membres du MHPCC s'esclaffer devant la parodie de Saint IGNUcius, ces considérations s'évaporent. Bien que n'étant pas à proprement parler une bête de scène, Stallman a de toute évidence ce qu'il faut pour tenir en haleine une salle pleine d'ingénieurs. « Être un saint dans l'église d'Emacs ne requiert pas le célibat, mais de s'engager à une vie de pureté morale, explique-t-il au public de Maui. Vous devez exorciser les systèmes d'exploitation diaboliques de tous vos ordinateurs et en installer un purement et saintement libre. Vous ne devrez ensuite installer que des logiciels libres sur cette première base. Si vous suivez cette discipline pour la vie, alors vous serez un saint de l'église d'Emacs, et vous pourrez peut-être même avoir une auréole. »

⁂

Le sketch de Saint IGNUcius s'achève sur une brève blague à l'attention des initiés. Sur la plupart des systèmes Unix et dérivés, le premier programme concurrent d'Emacs est *Vi*, que l'on prononce « vi-aïe », un éditeur de texte développé par l'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 d'Emacs d'utiliser *Vi*, dit-il. Utiliser une version libre de *Vi* n'est pas un péché, c'est une pénitence. Alors bon hack. » [9]

9. La liturgie de l'Église d'Emacs a évolué depuis 2001. Les utilisateurs peuvent rejoindre l'Église en récitant la Profession de foi : « Il n'y a pas

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

Cette remarque semble confirmer que, contrairement à d'autres personnages de scène ou hommes politiques, Stallman ne s'arrête jamais. En dehors du rôle de Saint 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, reprend le convive : « Vous voulez dire logiciel libre. C'est la façon correcte d'y référer. »

⁂

Au cours de la séance de questions-réponses, il admet faire parfois de la pédagogie. « 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 s'occupe d'inviter les 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 : « Il y a des millions de personnes qui arrivent et qui

d'autre système que GNU, et Linux est l'un de ses noyaux ». Stallman cite souvent la cérémonie religieuse de la « Foobar Mitzvah », le Grand Schisme entre les différentes versions d'Emacs, le Culte de la Vierge d'Emacs (toute personne n'ayant pas encore appris à utiliser *Emacs*). Enfin, « vi vi vi » (666) a été identifié comme étant l'Éditeur de la Bête.

construisent 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 qui travaillent sur la première. Il nous faut plus de volontaires pour la seconde. »

Travailler sur la « seconde étape » signifie faire comprendre que la liberté, et non la soumission, est le problème fondamental du mouvement du logiciel libre. Ceux qui espèrent réformer l'industrie des logiciels non libres depuis l'intérieur suivent 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 polos : « Sans brevet logiciel, comment pensez-vous pouvoir gérer l'espionnage industriel ?

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

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

Stallman recule comme touché par un jet mortel. « Attendez un instant… *Voler* ! ? Excusez-moi, il y a tant de présupposés faux dans cette affirmation que je ne peux rien dire sinon que je les rejette. » Stallman met alors le doigt sur le véritable enjeu de la question.

« Les sociétés qui développent du logiciel privatif, entre autres, conservent par devers elles énormément de secrets commerciaux, et ce n'est pas près de changer. Dans le bon vieux temps, 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 alors, c'est que les gens publiaient les idées intéressantes, et s'ils ne faisaient pas partie du mouvement du logiciel libre, ils gardaient secrets tous les petits détails. Aujourd'hui, ils gardent toujours secrets les petits détails, mais ils font breveter les idées générales. En conséquence, en ce qui concerne la situation que vous décrivez, les brevets ne font strictement aucune différence dans un sens ou dans l'autre.

— Mais si ça n'influence pas leur publication…, commence un autre membre du public, d'une voix tremblante.

— Mais ça l'influence, dit Stallman. La publication de ces brevets signifie que ces idées seront hors de portée du reste de la communauté durant vingt ans. Bon sang mais comment cela pourrait-il être bénéfique ? De plus, ces brevets sont écrits de manière si illisible — pour à la fois maquiller l'idée et rendre le brevet aussi général que possible — qu'il devient carrément inutile de chercher à apprendre quoi que ce soit des informations qui y sont publiées. La seule raison d'examiner un brevet est de savoir ce que vous n'avez pas le droit de faire. »

Le silence se fait dans le public. Le discours, qui a débuté à quinze heures quinze, approche de la sonnerie de dix-sept heures. La plupart des auditeurs remuent sur leurs sièges et se montrent impatients de débuter le week-end. Sentant cette fatigue, Stallman regarde le public et met fin précipitamment à la séance. « Il semble que nous en ayons fini », dit-il, avant d'enchaîner, à la manière d'un commissaire-priseur « une fois… deux fois… adjugé ! », décourageant ainsi toute question de dernière minute.

Personne ne lève la main, Stallman lance sa conclusion habituelle : « Hackez bien ! »

Chapitre

9

Genèse de la Licence publique générale de GNU (GNU GPL)

Au printemps 1985, Richard Stallman était parvenu au premier résultat du projet GNU : une version d'Emacs basée sur Lisp pour les systèmes d'exploitation de type Unix. Pour la rendre disponible en tant que logiciel libre, il lui fallait en définir les modalités de publication — ce qui serait en fait l'étape suivante de la charte Emacs [1].

1. Voir le chapitre 6 : Stallman surnomma « Commune Emacs » la charte inscrite dans le code source d'Emacs — et l'organisation communautaire qui en découla — prescrivant que les modifications soient reversées à la communauté des développeurs Emacs s'il y avait redistribution.

En réalité, le problème de l'arbitrage entre, d'une part, la liberté de modification et, d'autre part, les privilèges accordés aux auteurs existait bien avant que Gosmacs ne voie le jour. Le *Copyright Act* de 1976 avait étendu aux logiciels la législation américaine sur le copyright. Selon la section 102(b) de cette loi, les individus et les entreprises pouvaient exercer leur copyright sur « l'expression » d'un logiciel, mais pas sur les « processus ou méthodes eux-mêmes implémentés dans le programme » [2].

Tout programme s'en trouvait assimilable à un manuel d'algèbre : l'auteur pouvait prétendre à des droits sur le texte mais pas sur les idées mathématiques ou sur les techniques pédagogiques employées pour les expliquer. Ainsi, avant même que Stallman ne parle de l'utilisation du code de l'Emacs originel, les autres programmeurs étaient déjà légalement autorisés à rédiger leurs propres implémentations des idées ou des commandes d'Emacs. Ce qu'ils firent. Ainsi Gosmacs n'est-il qu'un exemple parmi une trentaine d'autres imitations d'Emacs développées pour de nombreux systèmes différents.

Les principes de la charte Emacs ne s'appliquaient qu'au code d'Emacs développé par Stallman lui-même. Même si cette charte avait eu force de loi, elle n'aurait pu s'appliquer à des imitations développées séparément comme Gosmacs. Certes, faire de Gosmacs un programme non libre n'était pas éthique, selon les principes du logiciel libre, car il priverait ainsi l'utilisateur de ses libertés ; mais cette question était sans rapport avec l'origine des idées implémentées dans Gosmacs.

Selon la législation du copyright, un programmeur désirant copier tout ou partie du code d'un programme existant (avec ou sans modification) devait en demander l'autorisation à son auteur. Bien

2. Voir Hal Abelson, Mike Fischer, and Joanne Costello, *Software and Copyright Law*, version mise à jour (1997).

que la loi sur le copyright de 1976 s'appliquât même en l'absence de mentions légales — chose que les hackers ignoraient pour la plupart — les mentions interdisant la copie se mirent à fleurir.

Stallman les considéra d'emblée comme les drapeaux d'une armée d'occupation. Rares étaient les logiciels qui n'empruntaient pas quelques bouts de code source issus de tel ou tel programme préexistant. Pourtant, d'une seule signature, le gouvernement américain avait donné aux programmeurs et aux compagnies le pouvoir légal d'interdire tout réemploi de ce type. Cette loi injectait aussi une dose de formalisme dans ce qui était jusque-là un système informel. Les litiges, auparavant réglés entre hackers, étaient dorénavant confiés aux avocats, ce qui conférait un avantage automatique aux entreprises plutôt qu'aux hackers. Certains pensaient que mentionner son nom dans une notice légale revenait à endosser la responsabilité de la qualité du code, et une notice légale incluait généralement le nom d'une société — les individus ayant d'autres moyens de se prévaloir de l'écriture de telle ou telle fraction de code.

Cependant, au cours des années précédant la mise en œuvre du projet GNU, Stallman remarqua que le copyright permettait à un auteur d'autoriser explicitement certains usages normalement interdits par le copyright, et d'assortir ces autorisations de conditions. « J'avais vu des courriels accompagnés de mentions de copyright assorties de simples licences autorisant la copie verbatim [3], se souvient-il. Ce fut une source d'inspiration certaine ». Ces licences comportaient notamment l'obligation de ne pas ôter la licence. Stallman voulait pousser la logique un peu plus loin. Par exemple, une notice légale aurait pu autoriser les utilisateurs à redistribuer des copies, mêmes modifiées, à condition d'accorder à leur tour ce même droit.

3. Copie conforme sans modification, mot à mot.

Stallman en conclut que recourir au copyright n'était pas forcément contraire à l'éthique ; plutôt, c'était l'usage qui en était fait habituellement — sans doute encouragé par l'esprit qui avait présidé sa création — qui déniait aux utilisateurs des libertés fondamentales. La plupart des auteurs n'imaginaient pas d'autre manière d'utiliser le copyright. Il en existait pourtant une : rendre le programme libre et s'assurer qu'il le reste.

Début 1985, pour GNU Emacs 16, Stallman avait préparé une licence fondée sur le droit d'auteur qui permettait à l'utilisateur de faire des copies et de les distribuer. Permission lui était aussi donnée d'opérer des modifications et de les diffuser, mais uniquement sous cette même licence. Sur ces versions modifiées, les utilisateurs étaient alors limités dans les droits a priori illimités que pouvaient leur conférer le copyright ; il devenait dès lors impossible d'en faire des versions privatrices à l'image de Gosmacs. Sans compter qu'ils devaient fournir le code source. Cette disposition vint combler le vide juridique qui aurait permis l'émergence de versions restreintes et privatrices de GNU Emacs.

Bien qu'utile pour codifier le contrat social de la Commune Emacs, cette première licence restait trop « informelle » pour ses objectifs. Aussi, peu après la création de la FSF, Stallman commença-t-il à travailler sur une version plus robuste. Il prit conseil auprès des avocats et des administrateurs qui l'avaient aidé à mettre en place la fondation.

Mark Fischer, un avocat de Boston spécialisé dans le droit d'auteur, se rappelle leurs échanges de l'époque au sujet de la licence. « Richard avait des opinions très arrêtées concernant la façon dont tout cela devait fonctionner, rapporte-t-il. Il tenait à deux principes : rendre le logiciel aussi ouvert que possible [il semble d'ailleurs qu'au moment où il prononçait ces mots, Fischer ait été influencé par les partisans de l'open source car Stallman n'a jamais cherché à rendre les logiciels 'ouverts'], et encourager à

utiliser le même mode de publication sous licence ». C'est pour ce second principe qu'étaient conçues les obligations exprimées dans la licence.

Il y avait là une révolution, mais il fallait du temps pour le comprendre. Fischer raconte qu'à l'époque, il considérait la licence GNU Emacs comme un simple contrat. Elle indiquait un coût d'utilisation, mais au lieu d'argent, réclamait aux utilisateurs l'accès à leurs propres modifications futures. Cela dit, Fischer se souvient que les termes du texte étaient uniques en leur genre : « Je crois que demander aux autres d'accepter un tel prix était, sinon inédit, du moins très inhabituel à l'époque. »

En façonnant la licence GNU Emacs, Stallman initia un changement majeur dans les principes informels qui avaient donné corps à la Commune Emacs. Là où il avait autrefois exigé que les membres lui envoient toutes les modifications qu'ils écrivaient, il demandait cette fois qu'ils transmettent le code source et la liberté de modification dès qu'ils choisissaient de redistribuer le programme.

En d'autres termes, ceux qui modifiaient Emacs pour leur usage personnel n'avaient plus l'obligation de renvoyer le code de leurs modifications. Dans ce qui deviendrait un rare effet de variation dans la doctrine du logiciel libre, Stallman baissait son « prix ». Les utilisateurs pouvaient innover sans être épiés par lui et distribuer leur version lorsqu'ils le souhaitaient, à condition que les copies distribuées soient assorties de la permission, pour les détenteurs, de les développer et de les redistribuer à leur tour.

Stallman explique cette évolution par sa propre insatisfaction face à l'aspect *Big Brother* de l'ancien contrat social de la Commune Emacs. Tout comme il avait un temps jugé utile que tous les utilisateurs lui fassent parvenir leurs modifications, il en vint à penser qu'il n'était pas juste d'exiger cela.

« C'était un tort que d'exiger des gens la publication de toutes leurs modifications, dit-il, et c'était un tort que d'exiger qu'ils en réfèrent à un seul développeur privilégié. Ce type de centralisation et de privilège n'est pas compatible avec une société dans laquelle tous sont égaux en droits. »

La Licence publique générale (*General Public License*) de GNU Emacs fit sa première apparition dans une version de GNU Emacs en 1985. Stallman invita les hackers à lui faire des retours pour en améliorer les termes. L'un de ceux qui répondirent était John Gilmore, futur activiste alors employé comme consultant chez Sun Microsystems. Dans le cadre de sa mission, il avait porté Emacs sur la version maison d'Unix, SunOS. À cette occasion, il avait publié sa version modifiée sous licence GNU Emacs.

Au lieu de considérer la licence comme une contrainte, Gilmore y voyait l'expression claire et concise de la philosophie des hackers. « Jusqu'alors, la plupart des licences étaient très informelles », se souvient-il. Gilmore cite à titre d'exemple une licence du milieu des années 1980 pour Trn, un lecteur de news, écrit par Larry Wall, autre hacker qui devint célèbre plus tard grâce à la création de l'utilitaire Patch pour Unix et du langage Perl. Dans l'espoir de trouver un équilibre entre la libre mise à disposition usuelle des hackers et le droit d'un auteur à dicter les modalités de commercialisation de son code, Wall utilisait une notice de copyright en guise de tribune éditoriale [4] :

```
Copyright (c) 1985, Larry Wall
Vous pouvez copier tout ou partie du kit trn tant que vous ne tentez pas
d'en faire commerce, ou de prétendre que vous l'avez écrit.
```

Ces déclarations, quoique fidèles à l'éthique des hackers, montraient la difficulté d'en traduire la nature informelle dans le langage juridique rigide du copyright.

4. Voir Trn Kit README, http://www.za.debian.org/doc/trn/trn-readme.

En écrivant la licence GNU Emacs, Stallman faisait plus que combler le vide juridique permettant les dérivés non libres ; il traduisait l'éthique hacker en des termes compréhensibles à la fois par les juristes et par les programmeurs.

Il fallut peu de temps pour que les hackers cherchent un moyen de transposer la licence de GNU Emacs pour leurs propres programmes. Lors d'une discussion sur *Usenet* en novembre 1986, Gilmore envoya un message à Stallman, lui proposant des modifications.

« Vous devriez probablement enlever 'EMACS' de la licence et le remplacer par 'LOGICIEL' ou quelque chose comme ça. Bientôt, on l'espère, Emacs ne sera pas la pièce principale du système GNU, et la licence s'appliquera à l'ensemble. » [5]

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, le débogueur de code source GDB. Pour le publier, il avait dû lui adapter la licence de GNU Emacs. C'était une tâche apparemment anodine, mais qui pouvait amener son lot d'erreurs.

Ce n'est qu'en 1989 que Stallman sut ôter les références spécifiques à Emacs et exprimer le lien entre le code source du programme et sa licence uniquement dans les fichiers sources. De cette manière, n'importe quel développeur pouvait placer son programme sous la licence sans modifier cette dernière.

La Licence publique générale de GNU, abrégée en GNU GPL (*General Public License*), était née. Le projet GNU en fit aussitôt la licence officielle de tous les programmes GNU.

***

5. John Gilmore, citation d'un courriel à Sam Williams.

Pour nommer les versions successives de la licence GPL, Stallman suivit la convention en cours dans le monde des programmeurs : les versions contenant des modifications mineures étaient indiquées par des chiffres décimaux, celles contenant des modifications majeures par des nombres entiers.

Ainsi en 1989, la première version fut-elle baptisée 1.0. Son préambule dévoilait les desseins politiques qui avaient présidé à sa création [6] :

```
La Licence publique générale est faite pour s'assurer que vous ayez la
liberté d'offrir ou de vendre des copies de logiciels libres, que vous en
receviez le code source ou puissiez y accéder si vous le souhaitez, que vous
puissiez le modifier ou en réutiliser des parties dans de nouveaux logiciels
libres, et que vous sachiez que vous avez le droit de faire tout cela. Pour
protéger vos droits, nous devons imposer des restrictions interdisant à
quiconque de vous disputer ces droits, ou de vous demander d'y renoncer.
Ces restrictions se traduisent par certaines obligations pour vous, si vous
distribuez des copies du logiciel ou le modifiez.
```

La GPL apparaît comme l'un des meilleurs *hacks* de Stallman. Elle a créé un système de propriété collective à l'intérieur même des habituels murs du copyright. Surtout, elle a mis en lumière la possibilité de traiter de façon similaire « code » juridique et code logiciel. Ainsi, dans le préambule de la GPL résidait le message, implicite pour les hackers, qu'au lieu de considérer la loi sur le copyright logiciel avec suspicion, il fallait plutôt y voir un système dangereux, ne demandant qu'à être hacké.

« La GPL s'est développée comme n'importe quel bout de logiciel libre, avec une vaste communauté qui en discutait la structure, le respect ou les transgressions qu'ils observaient, et la nécessité de faire les ajustements ou les légers compromis nécessaires à une plus grande acceptation, dit Jerry Cohen, un autre avocat ayant conseillé Stallman après le départ de Fischer. Le processus a été

6. Voir Richard Stallman *et al.*, « GNU General Public License : Version 1 », février 1989 : http://www.gnu.org/copyleft/copying-1.0.html.

très efficace et la GPL, dans ses diverses versions, est passée du scepticisme général, voire des réactions hostiles, à un large assentiment. »

En 1986, dans une interview pour le magasine *Byte*, Stallman résume la GPL en termes imagés. En plus de la proclamation des valeurs des hackers, dit-il, les lecteurs devraient aussi y voir « une forme de ju-jitsu intellectuel, destiné à retourner le système légal mis en place par ceux-là mêmes qui souhaitaient retenir pour eux seuls les biens logiciels. »

Cette interview de 1986[7] offre un aperçu intéressant, pour ne pas dire candide, des positions politiques de Stallman au cours des premiers jours du projet GNU. Cet élément se révèle par ailleurs très utile pour qui désire retracer l'évolution de sa rhétorique. Définissant l'objet de la GPL, Stallman explique : « J'essaie de changer la façon dont les gens conçoivent la connaissance et l'information en général. Je crois qu'essayer de *posséder* du savoir, tenter d'en contrôler l'utilisation par les autres, ou les empêcher de le partager, sont autant d'actes de sabotage ». Une déclaration à rapprocher d'une autre, en août 2000 : « Je vous conjure de ne plus utiliser l'expression 'propriété intellectuelle' dans vos réflexions. Elle prête à confusion en amalgamant copyrights, brevets et marques déposées. Ces choses ont des effets tellement différents qu'il devient stupide d'essayer d'en parler en les confondant. Si vous entendez quelqu'un parler de propriété intellectuelle sans utiliser les guillemets, alors c'est qu'il ne pense pas de façon claire, et vous ne devriez pas le suivre sur son terrain. »

7. Voir [Betz and Edwards, 1986]. http://www.gnu.org/gnu/byte-interview.html

Ce que cette citation montre, c'est que j'ai appris à être plus prudent dans la généralisation. Je ne parlerais pas aujourd'hui de « possession du savoir » car c'est une idée beaucoup trop large. Cependant, généraliser en parlant de « possession du savoir » n'est pas la même chose que généraliser en parlant de « propriété intellectuelle ». Ce dernier terme englobe trois législations qu'il est crucial de différencier pour comprendre toute question juridique concernant la « possession du savoir ». Les brevets accordent une exclusivité directe sur l'utilisation de savoirs spécifiques ; c'est bien une forme de « possession du savoir ». Le copyright est l'une des méthodes utilisées pour empêcher le partage d'œuvres qui peuvent recéler ou enseigner des savoirs, ce qui est tout à fait différent. Les marques déposées, elles, concernent d'autres aspects qui n'ont que peu de rapport avec le sujet du savoir.

Des années plus tard, il décrira la création de la GPL en des termes moins martiaux : « Je pensais à des problématiques à la fois éthiques, politiques et juridiques, explique-t-il. J'ai essayé de faire quelque chose de viable dans le système législatif actuel. Dans l'esprit, il s'agissait de concevoir les bases législatives d'une nouvelle société, mais ne gouvernant pas, je ne pouvais changer aucune loi. Il me fallait donc tenter de les bâtir au-dessus du système juridique existant, qui n'a en rien été conçu pour cela. »

⁂

Alors que Stallman en était à soupeser les problèmes éthiques, politiques et juridiques associés au logiciel libre, un hacker californien nommé Don Hopkins, hacker Unix lui aussi passionné de science-fiction, lui réexpédia le manuel du microprocesseur 68000 qu'il lui avait emprunté quelque temps auparavant. En guise de

remerciement, Hopkins avait décoré l'enveloppe de retour avec des autocollants trouvés lors d'une convention locale.

L'un d'eux, en particulier, attira l'attention de Stallman. On y lisait : « Copyleft (L) — Tous droits reversés ». Inspiré par cet autocollant, Stallman surnomma la technique juridique de la licence GNU Emacs, appelée à devenir la GNU GPL, « Copyleft », en la représentant avec humour par un « C » renversé dans un rcle.

Au cours du temps, ce terme de *copyleft*[8] fut consacré par la FSF comme qualifiant toute licence « rendant un programme libre et requérant que toute version modifiée ou étendue de ce programme soit de même un logiciel libre. »

Le sociologue allemand Max Weber avança un jour que toutes les grandes religions étaient construites autour d'une forme de « routinisation » ou d'institutionnalisation du charisme. Chaque religion établie, argumentait-il, avait réussi à convertir le charisme ou le message du leader religieux originel en tout un appareil social, politique et éthique plus aisé à transmettre à travers les cultures et les âges.

Bien que non religieuse en tant que telle, la GPL se qualifie sûrement comme un exemple d'institutionnalisation à l'œuvre dans le monde moderne et décentralisé du développement logiciel. Depuis sa mise en œuvre, des programmeurs et des entreprises qui, par ailleurs, ne manifestaient pas de loyauté particulière envers Stallman, ont d'emblée accepté le marché de la GPL au pied de la lettre. Et des dizaines de milliers de programmeurs l'ont intégrée comme mécanisme préemptif de protection pour leurs logiciels. Même ceux qui jugent ses conditions trop contraignantes en reconnaissent l'influence.

***

8. Copyleft est traduit en français par « gauche d'auteur ».

C'était le cas du hacker Keith Bostic, employé de l'université de Californie à l'époque de la sortie de la GPL 1.0. Le département où il travaillait, le *Computer Systems Research Group* (CSRG), s'était impliqué dans le développement d'Unix depuis la fin des années 1970 et était responsable de nombreuses parties clefs du système, dont l'implémentation du protocole réseau TCP/IP, pierre angulaire de l'Internet moderne. À la fin des années 1980, AT&T, propriétaire originel d'Unix, commença à songer à le commercialiser. Considérant son intérêt commercial, l'entreprise s'intéressa à la *Berkeley Software Distribution*, dite BSD, version académique d'Unix développée par Bostic et ses confrères à Berkeley.

L'accès au code rédigé par Bostic et son équipe, mélangé à du code propriétaire d'AT&T, fut de ce fait refusé à quasiment tout le monde. Les distributions Berkeley n'étaient disponibles que pour les institutions qui avaient déjà une licence Unix délivrée par AT&T. Or, à mesure de l'augmentation du prix des licences par AT&T, le compromis de prime abord inoffensif (du moins pour les universitaires cantonnés à leur tour d'ivoire) devint de plus en plus lourd à supporter.

Pour réutiliser du code de Berkeley dans le projet GNU, Stallman allait devoir convaincre ses auteurs de le séparer de celui d'AT&T. Aussi, vers 1984-85, il rencontra ceux qui étaient à la tête du projet BSD pour les convaincre de le publier en tant que logiciel libre. Stallman argumenta que AT&T n'était pas un organisme de bienfaisance et qu'il n'était pas convenable qu'une université fasse don de son code à une telle entreprise.

Embauché en 1986, Bostic s'était donné comme projet personnel de porter la dernière version de BSD sur l'ordinateur PDP-11. C'est durant cette période, raconte-t-il, qu'il entra en contact avec Stallman, à l'occasion d'une des incursions occasionnelles de ce dernier sur la côte Ouest. « Je me souviens de discussions agitées à propos du copyright, alors qu'il utilisait des stations de travail

empruntées au CSRG [9], dit Bostic. Nous allions dîner et continuions à discuter de copyright en mangeant. »

Ces arguments finirent par trouver une oreille réceptive, mais pas de la manière dont Stallman l'aurait voulu. En effet, en juin 1989, Berkeley sépara son code réseau du reste de la distribution, propriété d'AT&T, et le distribua sous une licence libre s'appuyant sur le droit d'auteur. Les termes de cette licence étaient assez libéraux. Tout ce que le destinataire avait à faire était de citer l'université dans les publicités du programme et de ses dérivés [10]. À la différence de la GPL, la licence permettait les dérivés non libres.

La diffusion de BSD Networking fut cependant limitée par un élément : il ne s'agissait pas d'un système d'exploitation à part entière, mais de ses seuls composants réseau. Alors que ce code aurait pu être l'une des plus importantes contributions aux systèmes d'exploitation libres, on ne pouvait l'exécuter à l'époque qu'en conjonction avec d'autres codes sous licence non libre.

Au cours des années suivantes, Bostic et d'autres permanents de l'université de Californie travaillèrent à combler les parties manquantes pour transformer BSD en un système d'exploitation complet et librement redistribuable. Bien que retardé par une bataille juridique avec *Unix Systems Laboratories* — l'émanation d'AT&T

9. Il s'agit du *Computer Systems Research Group*, le laboratoire de recherche en informatique de l'université de Californie où travaillait Bostic.

10. « L'odieuse clause de publicité » de l'université de Californie s'avéra par la suite très problématique. Cherchant une alternative aussi permissive que la GPL, certains hackers utilisèrent la licence BSD originelle, en remplaçant « université de Californie » par leur propre nom ou celui de leur institution. Résultat : les systèmes libres qui utilisaient plusieurs de ces programmes devaient citer des douzaines de noms différents dans leur notice. En 1999, après quelques années de lobbyisme de la part de Stallman, l'université de Californie accepta d'abandonner cette clause. Voir « *The BSD License Problem* » en français sur http://www.gnu.org/philosophy/bsd.fr.html.

possédant le code Unix —, la démarche porta finalement ses fruits au début des années 1990. Mais déjà bien avant cette date, de nombreux composants réseau de Berkeley se retrouvaient dans le système GNU de Stallman.

« Je pense que jamais nous n'aurions pu aller aussi vite sans l'influence de GNU, estime Bostic rétrospectivement. Ils étaient manifestement engagés à fond, 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 contenir la GPL pour être qualifié de libre — en témoignent les composants réseau de BSD —, mais mettre un programme sous GPL envoyait un message sans équivoque.

« Je crois que l'existence même de la GPL encourageait les gens à se demander s'ils faisaient vraiment du logiciel libre, et sous quelle licence ils allaient mettre leur travail », dit Bruce Perens, le créateur d'Electric Fence [11] 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 de celle de Stallman. « C'était en fait vraiment facile à faire », se souvient-il.

Rich Morin, un programmeur qui avait accueilli l'annonce de GNU par Stallman avec un certain scepticisme, se souvient avoir été impressionné par le nombre de logiciels qui commencèrent à s'assembler sous la bannière de la GPL. En tant que leader d'un groupe d'utilisateurs de SunOS, l'une des tâches principales de Morin dans les années 1980 avait été d'envoyer des distributions sur bande contenant une sélection des meilleurs outils gratuits

11. Electric Fence (ou eFence) est un utilitaire Unix (sous licence GPL) très populaire de débogage des accès mémoire. Voir le site de Bruce Perens : http://perens.com/FreeSoftware/.

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. Vers 1989, Morin remarqua que les meilleurs logiciels étaient généralement sous licence GPL. « En tant que distributeur de logiciels, dès que je voyais la mention GPL, je savais que c'était bon », se souvient-il.

Pour compenser les efforts engagés à compiler des enregistrements pour le groupe d'utilisateurs Sun, Morin demandait un peu d'argent aux destinataires. Or, avec les logiciels passant massivement sous GPL, il mettait subitement deux fois moins de temps à assembler les logiciels, faisant même un peu de profit au passage. Voyant une opportunité commerciale dans son hobby, il en fit une entreprise qu'il baptisa 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. Logique qu'il résumait, à la fin des années 1980, par un simple moyen mnémotechnique pour résoudre l'ambiguïté du mot anglais *free* : « *Don't think free as in free beer ; think free as in free speech.* » [12]

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

***

12. « Pensez à 'libre' comme dans 'liberté d'expression', pas comme dans 'bière gratuite' ». L'ambiguïté n'existe pas en français.

L'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 amateur de jazz suivait son artiste préféré. Ce n'est qu'avec la publication en 1987 du compilateur C GNU, dit GCC, qu'il commença toutefois à mesurer le plein potentiel du logiciel libre. Définissant GCC comme une « bombe », Tiemann voyait ce logiciel comme la preuve, dans le domaine de la programmation, de la détermination de Stallman.

« Tout comme chaque écrivain rêve d'écrire *le* grand roman américain, chaque programmeur des années 1980 parlait d'écrire *le* grand compilateur américain, se souvient Tiemman. Tout à coup, Stallman l'a fait. C'était une leçon d'humilité. »

« On pourrait parler de *single point of failure*[13] pour GCC, renchérit Bostic. Personne n'avait de compilateur à l'époque, avant l'arrivée de GCC. »

Plutôt que de concurrencer Stallman, Tiemann décida de construire à partir de son travail. La version originelle de GCC pesait cent dix mille lignes de code, mais Tiemann se souvient du code comme étant étonnamment facile à comprendre. Si facile en fait qu'il rapporte avoir eu besoin de moins de cinq jours pour le maîtriser, et une semaine supplémentaire pour le porter sur une nouvelle plate-forme matérielle, le microprocesseur National Semiconductor 32032. L'année suivante, il se mit à jouer avec le code source, créant le premier compilateur « natif » pour le langage C++ — en étendant GCC pour qu'il supporte aussi bien le C++ que le C[14]. Un jour, au cours d'un exposé aux laboratoires

13. (SPOF). En sécurité informatique, élément qui, s'il s'écroule, fait s'écrouler l'ensemble.

14. L'implémentation existante, et non libre, du langage C++ fonctionnait par conversion du code en langage C, et envoi du résultat vers un compilateur C.

Bell, il croisa des développeurs de chez AT&T, qui luttaient pour atteindre le même objectif.

« Il y avait quarante ou cinquante personnes dans la salle, et j'ai demandé combien travaillaient sur le compilateur natif, se rappelle Tiemann. Mon hôte m'a répondu que l'information était confidentielle, mais a ajouté que si je regardais dans la pièce, je pouvais m'en faire une bonne idée. »

Peu après, poursuit-il, il eut comme une révélation. « J'avais travaillé sur ce projet pendant six mois. Je me suis simplement dit : 'qu'il s'agisse de moi ou du code, voilà un niveau d'efficacité que le marché du logiciel libre devrait pouvoir récompenser'. »

Tiemann trouva une inspiration supplémentaire dans le Manifeste GNU, qui, bien que fustigeant l'avidité des entreprises de logiciels privateurs, encourageait les compagnies, tant qu'elles respectaient la liberté des utilisateurs, à utiliser et distribuer du logiciel libre dans leurs activités commerciales. En supprimant le pouvoir d'établir des monopoles dans le commerce des logiciels, la GPL offrait la possibilité, même aux petites entreprises, de se battre sur le marché des services, ce qui allait du simple support technique à la formation, en passant par l'adaptation des logiciels aux besoins spécifiques des clients.

Dans un essai daté de 1999, Tiemann rappelait 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. » [15]

Si ce modèle économique n'était pas nouveau (Stallman lui-même l'avait soutenu à la fin des années 1980), Tiemann avait l'intention de l'étendre à grande échelle. Faisant équipe avec John Gilmore et David Vinayak Wallace, il lança un service de conseil

15. Tiemann, 1999

logiciel dédié à la personnalisation des logiciels GNU. Nommée Cygnus Support (Cygnus étant l'acronyme récursif pour *Cygnus, Your GNU 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.

***

Le système d'exploitation GNU que projetait Stallman nécessitait davantage que des outils de développement de logiciel. Dans les années 1990, le projet GNU élabora également un interpréteur de lignes de commande (*shell*) écrit par Brian Fox lorsqu'il était employé à la FSF — cette version plus complète du Bourne Shell fut rebaptisée par Stallman le Bourne Again Shell, ou Bash. Furent aussi créés l'interpréteur PostScript nommé Ghostscript, ou encore la plate-forme hypertextuelle de création et de consultation de documentation Texinfo, la bibliothèque C requise par les programmes C pour s'exécuter et converser avec le noyau (glibc), le tableur Oleo (« bien mieux pour vous qu'un autre plus coûteux » [16]), et même un assez bon jeu d'échecs. Cependant, les programmeurs étaient plus intéressés par les outils de développement GNU.

GNU Emacs, GDB et GCC étaient les trois grands outils orientés développeurs, mais ils ne furent pas les seuls générés par le projet GNU dans les années 1980. Jusqu'en 1990, GNU avait aussi développé des versions du gestionnaire de compilation Make, le compilateur de parseur Yacc (*Yet Another Compiler Compiler*)

16. Oleo (désigne aussi l'oléo-margarine en anglais). Ce tableur (*spreadsheet*) donnait lieu à un jeu de mots : *Oleo, better for you than the more expensive spread.*

rebaptisé Bison ainsi que l'utilitaire de manipulation de fichiers textuels awk (rebaptisé gawk), et des dizaines d'autres encore.

Comme GCC, chaque logiciel GNU était conçu pour pouvoir fonctionner sur de multiples systèmes, et non sur la plate-forme d'un unique éditeur. Dans ce travail d'amélioration de la flexibilité des programmes, Stallman et ses collaborateurs les rendaient aussi plus utiles.

Rappelant l'ambition d'universalité du projet GNU, Rich Morin fait référence à un logiciel non destiné à être utilisé mais d'une importance pourtant vitale : le paquet GNU Hello, servant à montrer aux programmeurs comment empaqueter correctement un programme GNU. « Il s'agit du fameux programme qui affiche '*hello world*' : cinq lignes de C, empaquetées comme s'il s'agissait d'une distribution GNU, décrit Morin. Par conséquent, il contient toute l'information relative à Texinfo et à Configure [17]. Et tout le reste des rouages d'ingénierie logicielle que le projet GNU a inventés pour permettre le portage aisé vers les autres environnements. C'est un travail extrêmement important, et ça n'affecte pas seulement les logiciels de Stallman, mais aussi tous les autres logiciels du projet GNU. »

Selon Stallman, améliorer techniquement les composants Unix était un objectif moins important que de les remplacer par des logiciels libres. « Chaque composant peut donner lieu, ou non, à amélioration, dit Stallman lors de son interview pour *Byte*. D'un côté, le fait de réimplémenter permet d'améliorer nettement de nombreux systèmes. De l'autre, c'est possible parce que je suis moi-même depuis longtemps dans le milieu et que j'ai travaillé sur de nombreux autres systèmes. J'ai donc de nombreuses idées — venues en fréquentant ces autres systèmes — à mettre en œuvre. » [18]

17. La mission du fichier configure est de paramétrer la compilation pour générer un programme pour le système choisi.

18. Stallman, 1986

Malgré tout, alors que les outils GNU s'imposaient à la fin des années 1980, la réputation de Stallman, acquise au AI Lab pour sa minutie dans la conception, allait devenir légendaire au sein de la communauté toute entière des développeurs.

Jeremy Allison, utilisateur de Sun à la fin des années 1980 et programmeur qui allait co-créer le logiciel libre Samba [19], dans les années 1990, se souvient en riant qu'il en avait fait les frais. À la fin de la décennie, il commença à utiliser Emacs. Inspiré par le modèle de développement communautaire de ce programme, il envoya un petit bout de code source, qui fut rejeté par Stallman. « C'était comme le titre d'une mauvaise Une dans *The Onion* [20], plaisante Allison. 'Dieu répond aux prières d'un enfant : Non'. »

Au gré de son succès dans la création de bibliothèques et de logiciels, le projet GNU repoussait d'autant le développement du noyau de son système, cet « agent de circulation » logiciel contrôlant l'accès des autres programmes au processeur et aux ressources de la machine.

Comme pour plusieurs autres composants majeurs du système, Stallman voulait optimiser le développement du noyau en cherchant un programme existant à adapter. Une lecture des *GNUsletters* du projet GNU à la fin des années 1980 révèle que cette approche avait ses inconvénients, tout comme l'essai initial de construire GCC à partir de Pastel. Une GNUsletter de janvier 1987 rapporte que le projet GNU tentait de refondre Trix, un noyau Unix développé au MIT. Stallmann ne concrétisa pas, en réalité, ce projet, car il travaillait alors sur GCC, mais il conclut

19. Samba est une implémentation libre du protocole SMB/CIFS sous GNU/Linux qui permet de faire interopérer des serveurs/clients GNU/Linux et Windows.

20. Quotidien en ligne américain satirique et parodiant de fausses nouvelles créé en 1988 (http://www.theonion.com).

plus tard que Trix nécessitait trop de modifications pour constituer un bon point de départ. En février 1988, selon une autre lettre de diffusion, le projet GNU avait changé son fusil d'épaule pour Mach, un micro-noyau léger développé à Carnegie Mellon. Mach n'était pas, alors, un logiciel libre, mais ses développeurs confessaient en privé qu'ils allaient le libérer. Lorsque cela arriva, en 1990, le développement du noyau GNU put réellement commencer [21].

Les délais de développement du noyau n'étaient pas le seul souci de Stallman à cette époque. En 1989, la Lotus Development Corporation poursuivit en justice des compagnies rivales, Paperback Software International et Borland, pour avoir copié les commandes de menus du logiciel populaire de Lotus qu'était Tableur 1-2-3. L'action en justice de Lotus, ajoutée à la bataille Apple-Microsoft sur « l'apparence et l'ergonomie » (*look-and-feel*), mettaient indirectement en danger l'avenir du projet GNU. Elles menaçaient en effet le droit de développer des logiciels compatibles avec des programmes existants, ce qui était l'un des principes de GNU. Ce procès risquait de paralyser la culture toute entière du développement logiciel.

Déterminé à faire quelque chose, Stallman et quelques professeurs rédigèrent un communiqué dans *The Tech* (le journal étudiant du MIT), fustigeant le procès et appelant au boycott des entreprises à l'origine de la bataille judiciaire. Stallman organisa parallèlement un groupe de protestation. Baptisé *League for Programming Freedom*, le collectif manifesta devant les bureaux de Lotus.

Ces manifestations étaient remarquables [22]. Elles illustraient l'évolution de l'industrie du logiciel. Les applications avaient progressivement remplacé les systèmes d'exploitation comme front

21. Voir *HURD History* sur http://www.gnu.org.

22. Selon un communiqué de la Ligue pour la liberté de programmer, ces manifestations inaugurèrent le premier chant de protestation en hexadéci-

de bataille des entreprises. Mais dans sa quête inachevée pour construire une distribution logicielle libre, le projet GNU semblait désespérément en retard aux yeux de ceux pour qui la mode et le succès passaient en premier. En effet, le fait même que Stallman ait éprouvé le besoin de mettre en place une équipe pour contrer les batailles judiciaires liées à « l'apparence et l'ergonomie » amenait certains observateurs à conclure que la FSF n'était plus qu'en sursis.

Pourtant, Stallman avait une raison stratégique de lancer une organisation indépendante de lutte contre l'imposition de nouveaux monopoles dans le développement de logiciels : il avait l'espoir que les développeurs de logiciels privateurs s'y joindraient. L'extension du copyright aux interfaces les menaçait tout autant que les développeurs de logiciels libres. S'il était peu probable qu'ils soutiennent la FSF, rien dans la *League for Programming Freedom* n'était fait pour les rebuter. Pour cette même raison, Stallman confia au plus vite à d'autres que lui la direction de la ligue.

En 1990, la Fondation John D. et Catherine T. MacArthur certifia le statut de génie de Stallman en lui accordant la bourse MacArthur, aussi appelée « bourse des génies » — en l'occurrence un revenu de 240 000 dollars sur cinq ans. Bien que les raisons d'octroi de cette bourse n'aient jamais été données par la Fondation, cette bourse fut néanmoins considérée comme la récompense au lancement du projet GNU et comme un écho donné à la philosophie du logiciel libre.

mal (*hex chant*) : « 1-2-3-4, jeter 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, le *look-and-feel* doivent s'en aller » (http://progfree.org/Links/prep.ai.mit.edu/demo.final.release). On peut noter que, en anglais, *hex* signifie aussi « sort jeté par une sorcière ».

Cette bourse dissipait quelques préoccupations immédiates pour Stallman. Elle lui permit de cesser ses activités de consultant grâce auxquelles il gagnait sa vie dans les années 1980 et de dédier davantage de temps à la cause du logiciel libre. En effet, un ou deux ans après la remise du prix MacArthur, Stallman avait commencé à souffrir de douleurs chroniques au niveau des mains et dictait son travail à des sténographes employés par la FSF [23].

La « bourse des génies » l'autorisait aussi à s'inscrire normalement sur les listes électorales. En 1985, un feu dans son immeuble l'avait en effet privé de domicile officiel et avait couvert d'une cendre épaisse presque tous ses livres — les tentatives de nettoyage n'avaient pas donné de très bons résultats. Il avait alors vécu comme un « squatteur » [24] au 545 Technology Square [25], ce qui l'obligeait à voter en qualité de *sans domicile fixe*. « [Le bureau du registre des votants de Cambridge] ne voulait pas accepter cela comme une adresse, se souvient-il. Un article de journal à propos du prix MacArthur ayant évoqué cette situation, le bureau a alors accepté de m'inscrire. » [26]

Plus important encore, la « bourse des génies » lui attira l'attention de la presse. Invité à discourir, il pouvait y parler du projet GNU, des logiciels libres et des dangers qui les menaçaient comme les procès sur « l'apparence et l'ergonomie » ou encore les brevets logiciels.

⁂

23. Il s'agissait sans aucun doute de troubles musculo-squelettiques (TMS) mais pas de ceux dont on entendait souvent parler. « Ce n'était pas le syndrome du canal carpien, écrit-il. Mon problème était localisé dans les mains, pas dans les poignets. »
24. Lerner, 1990
25. L'adresse du AI Lab.
26. Gross, 2000

Chose intéressante, le bouclage du système GNU put avoir lieu grâce à l'un des voyages de Stallman. En avril 1991, il effectua une visite à l'université polytechnique d'Helsinki en Finlande. Au sein du public se trouvait Linus Torvalds. Alors âgé de vingt-et-un ans, le jeune homme commençait à travailler sur Linux — le noyau libre qui allait permettre de combler le principal manque du système GNU.

Étudiant à l'université 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-il dans son autobiographie. Nous n'en avions pas beaucoup à Helsinki. » [27]

Bien que peu en phase avec le côté « sociopolitique » de la stratégie de Stallman, Torvalds appréciait cependant l'un des aspects de la logique sous-jacente : aucun programmeur n'écrit de code sans commettre d'erreurs. Même si les utilisateurs ne souhaitent pas adapter un programme à leurs préférences spécifiques, tout programme peut être amélioré. En partageant le code source, les hackers plaçaient l'amélioration du programme avant tout, les protégeant de l'appât du gain ou des questions d'ego.

À l'instar 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, il avait découvert la programmation PC sur Unix, via le MicroVAX [28] de l'université.

27. Torvalds and Diamond, 2001a, pp.58—59. Bien qu'a priori fidèle quant à de Torvalds, ce livre présente quelques inexactitudes à propos de Stallman. Par exemple, il y est dit qu'il veut « tout rendre open source » et qu'il « se plaint de tous ceux qui n'utilisent pas la GPL ». En réalité, Stallman défend les logiciels libres et non l'open source. S'il demande aux auteurs d'utiliser la GPL chaque fois que c'est possible, il affirme néanmoins que toutes les licences libres respectent l'éthique du logiciel libre.

28. MicroVAX est le nom d'une famille d'ordinateurs fabriqués par Digital Equipment Corporation (DEC). Le MicroVAX I fut introduit sur le marché en 1984 — NdT.

Cette progression étape par étape, l'avait amené à considérer différemment le problème de l'accès aux machines.

Pour Stallman, les barrières majeures étaient la bureaucratie et les privilèges. Pour Torvalds, c'était la géographie et le rude hiver d'Helsinki. Obligé de cheminer péniblement à travers l'université d'Helsinki dans le seul but de se connecter à son compte Unix, il songea rapidement à trouver un moyen de se connecter depuis son appartement chauffé, situé hors du campus.

Torvalds utilisait Minix, un système à l'époque non libre développé à titre d'exemple didactique par le professeur d'université Andrew Tanenbaum aux Pays-Bas. Minix incluait l'outil de compilation non libre de l'Université Libre d'Amsterdam [29] ainsi que quelques autres utilitaires à l'écriture desquels Tanenbaum avait, non sans une certaine condescendance, proposé à Stallman de participer [30].

Minix était suffisamment léger pour pouvoir résider dans la mémoire du PC 386 de Torvalds, mais il avait été élaboré à des fins pédagogiques, et non pour une utilisation véritable. Il lui manquait

29. Contrairement à ce que pourrait laisser penser son nom, *Free University Compiler Kit*, ce compilateur n'est pas un programme libre. Il s'agit de Vuck, dont l'acronyme fait référence au statut de cette université et non au logiciel libre, cf. chap. 7 — NdT.

30. Dans son livre intitulé *Operating System Design and Implementation* (Pearson Education, 1987), Tanenbaum décrit Minix comme un système d'exploitation, mais l'objet du livre ne concerne que la partie du système qui correspond au noyau d'Unix. Il y a en effet dans ce livre deux usages courants de l'expression « système d'exploitation », et l'un d'eux désigne le noyau en terminologie Unix. Ce n'est cependant pas la seule difficulté terminologique. Cette partie de Minix consiste en un micronoyau sur lequel étaient basés des serveurs, ce qui rendait l'ensemble comparable au couple GNU Hurd plus Mach. Si le micronoyau et ses serveurs sont certes comparables au noyau d'Unix, lorsque cet ouvrage fait référence au « noyau », il ne s'agit que du micronoyau. Voir Andrew Tanenbaum, *Operating System Design and Implementation.*

aussi l'émulation du terminal, une fonctionnalité qui aurait permis à Torvalds de mimer le comportement d'un terminal d'affichage, rendant ainsi possible la connexion vers le MicroVAX depuis chez lui.

Au début de l'année 1991, l'étudiant entreprit d'écrire un programme d'émulation de terminal. Il utilisa Minix pour le développer ; l'émulateur ne s'exécutait pas sur Minix mais seul, indépendamment. Torvalds ajouta alors des fonctionnalités pour accéder au système de fichiers de Minix. Cette année-là, il parlait de son travail en cours comme d'un « GNU Emacs des programmes d'émulation de terminal »[31].

Comme il manquait à Minix de nombreuses et importantes fonctions, Torvalds commença à transformer son émulateur de terminal en un noyau comparable à celui de Minix, mais monolithique. Non sans ambition, il sollicita un groupe de discussion sur Minix pour trouver des copies de la norme Posix, c'est-à-dire les spécifications qu'un noyau doit respecter pour être compatible avec Unix[32].

Quelques semaines plus tard, après avoir fait tourner conjointement son noyau et quelques programmes GNU adaptés par ses soins, Torvalds envoya un message rappelant celui qu'avait posté Stallman en 1983 pour lancer le projet GNU :

```
Bonjour à tous ceux ici utilisant Minix,

Je suis en train de créer une distribution logicielle (libre) – ce n'est
qu'un passe-temps, ça ne sera pas aussi grand et professionnel que GNU – pour
les 386/486 pour PC-AT[33]. Ça mijote depuis le mois d'avril, et c'est presque
prêt. J'aimerais avoir des retours sur ce que les gens aiment ou n'aiment
```

31. *Ibid.*

32. Les normes Posix ont par la suite été étendues à plusieurs fonctions en ligne de commande, mais ce n'était pas le cas en 1991.

33. *Personal Computer-Advanced Technology* (IBM) qui allaient devenir les « compatibles PC ».

pas dans Minix en général, puisque mon OS y ressemble d'une certaine façon (entre autres, le même arrangement physique du système de fichiers [pour des raisons pratiques]). Pour l'instant, sur mon système, j'ai porté Bash (1.08) et GCC (1.40)…[34]

Le message reçut quelques vagues réponses et un mois plus tard, Torvalds avait publié la version 0.01 de sa distribution logicielle — c'est-à-dire la toute première version conçue pour un examen extérieur — sur un site FTP. L'étudiant devait par ailleurs trouver un nom pour ce nouveau noyau. Sur son propre disque dur, il avait en effet sauvegardé le programme sous le nom « Linux », respectant ainsi la convention logicielle qui consiste à donner à chaque variante d'Unix un nom finissant avec la lettre X. Comme il trouvait le nom trop « égocentrique », Torvalds le changea en « Freax », pour finalement voir le responsable du site FTP le renommer d'après le premier nom.

Torvalds dit avoir cherché à écrire un système d'exploitation complet et libre, comme le laisse penser sa comparaison avec GNU. Or, il n'écrivit en fait qu'un noyau pur et simple, parce qu'il savait que les autres composants étaient déjà disponibles, grâce au travail de GNU et des autres projets libres. De plus, le projet GNU consistant à tous les intégrer dans un seul système, tous les éléments étaient de fait conçus pour être compatibles les uns avec les autres. Ainsi, pendant qu'il poursuivait le développement du noyau, il y adapta (et plus tard ses collaborateurs) les nombreux programmes GNU.

Initialement, Linux n'était pas un logiciel libre. En effet, la licence que Torvalds avait choisie n'autorisait pas les distributions commerciales. Il avait peur, en effet, de voir une entreprise rentrer dans le projet puis lui prendre son noyau. Pourtant, au vu de la popularité de GNU/Linux, il réalisa que la vente de copies serait utile pour la communauté. Son inquiétude s'amenuisa et il reconsidéra la question de la licence[35].

34. *Ibid.*, p.85 [voir aussi p.22]
35. *Ibid.*, pp.94—95

Ni la compilation de Linux avec GCC, ni l'exécution de GCC avec Linux, n'impliquaient pour lui l'obligation légale de publier Linux sous licence GPL. Mais en utilisant GCC, Torvalds se trouvait d'une certaine façon obligé de laisser les autres emprunter à leur tour. Comme Torvalds le dirait plus tard : « Je m'étais élevé en prenant appui sur les épaules de géants. » [36]. Comme on pouvait s'y attendre, il commença à se demander ce qu'il adviendrait si d'autres venaient à leur tour lui demander une aide similaire.

Une décennie après cette décision, Torvalds fait écho à Robert Chassell de la FSF, résumant ainsi 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 abuser. Je voulais qu'ils puissent voir [Linux], et qu'ils puissent faire des modifications autant qu'ils le voulaient. Mais je voulais aussi m'assurer que j'obtiendrais en retour de pouvoir voir ce qu'ils en faisaient. Je voulais avoir accès aux sources à tout moment, afin que, s'ils faisaient des améliorations, je puisse les reproduire moi-même. » [37]

Quand vint le temps de publier la version 0.12 de Linux, la première fonctionnant pleinement avec GCC, Torvalds décida de se ranger au côté du mouvement du logiciel libre. Il remisa l'ancienne licence de Linux et la remplaça par la GPL. Trois ans plus tard, les développeurs proposaient une version 1.0 du noyau, qui fonctionnait correctement avec le quasi complet système GNU, composé de programmes issus du projet GNU et d'ailleurs. En pratique, ils avaient complété le projet GNU en y ajoutant le noyau Linux. Le résultat, en gros, était GNU *plus* Linux. Pourtant, Torvalds et ses amis se référaient à ce système en le nommant abusivement « Linux » tout court.

36. *Ibid.*, pp.95—97
37. *Ibid.*, pp.94—95

En 1994, le système ainsi constitué avait obtenu une large reconnaissance de la part du monde des hackers. Au point que certains observateurs du monde commercial se demandaient si Torvalds n'avait pas laissé s'échapper la poule aux œufs d'or en optant pour la GPL durant les premiers mois du projet. Dans le premier numéro de *Linux Journal*, l'éditeur Robert Young s'entretenait avec Torvalds. Quand il demanda au programmeur finlandais si celui-ci regrettait d'avoir abandonné la propriété exclusive du code source, Torvalds répondit que non. « Même rétrospectivement », Torvalds dit qu'il considérait la GPL « comme l'une des meilleures décisions de conception » qu'il adopta durant les premières phases du projet Linux. [38]

Que la décision ait été prise sans aucun appel ni déférence envers Stallman ou la FSF illustrait la transposabilité grandissante de la GPL. S'il fallut bien deux ans à Stallman pour le reconnaître, le boom du développement de Linux rappellait celui d'Emacs. Cette fois, cependant, l'innovation à l'origine de l'explosion n'était pas un *hack* logiciel comme Control-R, mais la possibilité inédite de faire fonctionner un système *à la* Unix sur une architecture PC. Les motivations étaient différentes, sans doute, mais le résultat final convenait très bien à l'éthique du logiciel libre : une distribution logicielle complète et fonctionnelle, entièrement composée de logiciels libres.

Comme l'indique son message initial au groupe de discussion *comp.os.minix*, il fallut plusieurs mois avant que Torvalds considère Linux comme étant plus qu'un simple substitut provisoire au noyau Hurd, sur lequel œuvraient les développeurs GNU. Torvalds ne se voyait que comme dernier exemple en date de ces gamins s'amusant à démonter et réassembler tout un tas de choses. Malgré tout, résumant le succès fulgurant d'un projet qui aurait pu

38. Young, 1994

passer le reste de ses jours abandonné sur un disque dur, il se félicite d'avoir, dans sa jeunesse, eu la sagesse d'abandonner toute volonté de contrôle sur son code, et accepté le marché qu'offrait la GPL.

« Je n'ai peut-être pas vu la lumière », écrit Torvalds en évoquant le discours de Stallman à l'université d'Helsinki en 1991 et sa décision subséquente d'opter pour la GPL. « Mais je pense que quelque chose, dans son discours, s'est ancré en moi. » [39]

39. Torvalds and Diamond, 2001a, *op. cit.*, p.59

Chapitre

10

GNU/Linux

En 1993, le mouvement du logiciel libre en était à la croisée des chemins. Pour les optimistes, tous les signes semblaient indiquer le succès de la culture « hacker ». *Wired*[1], un nouveau magazine branché dans lequel on pouvait lire des articles sur le chiffrement des données, *Usenet* ou encore la liberté des logiciels, faisait un malheur dans les rayons de la presse informatique.

L'Internet, terme argotique qui avait été tout d'abord l'apanage des hackers et des scientifiques, avait fini par trouver sa place dans le dictionnaire courant. Même le président Clinton l'utilisait. L'ordinateur personnel, au départ simple jouet réservé aux amateurs, avait acquis ses lettres de noblesse, donnant à toute

1. *Wired* signifie câblé ou branché, au sens propre comme au figuré — NdT.

une nouvelle génération d'utilisateurs accès aux logiciels conçus par les hackers. Et si le projet GNU n'avait pas encore atteint son objectif de fournir un système d'exploitation libre complètement opérationnel, il était déjà possible d'essayer sa variante GNU/Linux.

Quelle que soit la façon d'aborder les choses, les nouvelles étaient bonnes, du moins en apparence. Après dix années de lutte, les hackers et leurs valeurs finissaient par faire école au sein du grand public. Des valeurs que les gens finissaient, semblait-il, par comprendre peu à peu.

Mais était-ce vraiment le cas ? À chaque signe positif, les pessimistes voyaient, quant à eux, une contrepartie inquiétante.

Certes, être un hacker était soudain à la mode, mais était-ce une bonne chose que d'être à la mode pour une communauté construite dans la marginalité ? Certes, la Maison Blanche ne tarissait pas d'éloges au sujet de l'Internet, jusqu'à avoir créé son propre nom de domaine, *whitehouse.gov*, mais il y avait aussi les entreprises, les comités de censures et les représentants de l'exécutif qui n'attendaient qu'une chose : soumettre ce territoire indompté qu'était alors la culture Internet. Certes, les PC avaient gagné en puissance, mais en inondant le marché de ses puces, Intel avait favorisé la mainmise des marchands de logiciels non libres.

Pour chaque utilisateur gagné à la cause du logiciel libre grâce à GNU/Linux, des centaines, voire des milliers, faisaient leur premiers pas sous Microsoft Windows. En effet, les interfaces graphiques de GNU/Linux étant alors des plus rudimentaires, l'emploi de ce système était peu aisé. En 1993, seuls les spécialistes pouvaient l'utiliser. Les premières tentatives du projet GNU visant à développer une interface graphique n'avaient encore jamais abouti.

Quant au contexte politique du moment, il n'était guère favorable. L'application de droits d'auteurs aux interfaces utilisateur constituait encore une réelle menace — l'idée n'ayant pas encore été déboutée en justice. Parallèlement, les brevets portant sur les algorithmes et les fonctions représentaient un danger croissant, menaçant de s'étendre à d'autres pays.

Enfin, il y avait l'étrange nature de GNU/Linux lui-même. Non entravée par des désaccords juridiques (comme c'était le cas pour BSD), son évolution extrêmement rapide avait été si inattendue, son succès si imprévu, que les programmeurs les plus impliqués dans son développement ne savaient finalement pas quoi en faire.

Ressemblant davantage à un recueil de morceaux choisis qu'à un projet unifié, c'était une sorte d'anthologie hacker qui contenait tout, de GCC, GDB et Glibc (la toute nouvelle bibliothèque C du projet GNU) à X (une interface graphique inspirée d'Unix développée par le Laboratoire de sciences informatiques du MIT), en passant par des outils créés par le projet BSD comme Bind [2] (*Berkeley Internet Naming Daemon,* système DNS permettant de remplacer les nombres des adresses IP par des noms de domaines faciles à mémoriser) et une pile TCP/IP.

À tout cela s'ajoutait le noyau Linux, lui-même conçu pour remplacer Minix. Torvalds et son équipe toujours plus nombreuse, plutôt que de développer un tout nouveau système d'exploitation, avaient greffé leur travail à ce gisement GNU de départ. C'est cela qu'exprimerait Torvalds plus tard, évoquant son propre succès : « Je suis en fait une personne très paresseuse, qui aime s'attribuer le travail d'autres personnes. » [3]

2. Bind est le sigle de *Berkeley Internet Naming Daemon*, démon (programme) de nommage pour l'Internet de Berkeley, devenu aujourd'hui *Berkeley Internet Naming Domain*. Bind est ce qu'on appelle un serveur de noms de domaines (DNS) — NdT.

3. Torvalds a cité cette phrase à de nombreuses reprises. La référence la plus notoire figure dans l'essai de Eric S. Raymond, *La Cathédrale et le Bazar*

Une telle paresse, tout admirable qu'elle fût sous l'angle de la productivité, était gênante d'un point de vue politique. Avant tout, elle faisait ressortir l'absence de toute visée idéologique de la part de Torvalds. Contrairement aux développeurs du projet GNU, l'étudiant finlandais n'avait pas bâti son noyau par volonté d'offrir la liberté à ses confrères hackers. Il ne l'avait fait que pour avoir quelque chose avec quoi il pourrait lui-même s'amuser. Quelle était donc la nature de ce système mixte et à quelle philosophie serait-il rattaché ? Était-ce l'incarnation de l'esprit du logiciel libre explicité par Stallman au sein du *Manifeste GNU* ? Ou n'était-ce qu'un amalgame d'outils ingénieux que n'importe quel utilisateur, tout aussi motivé, aurait pu lui-même assembler sur son système maison ?

***

Fin 1993, le nombre des utilisateurs de GNU/Linux penchant pour la seconde définition et commençant à bricoler leurs propres variations sur le thème s'était considérablement accru. Diverses « distributions » furent ainsi développées et diffusées, parfois gratuitement, parfois non, et avec des résultats on ne peut plus inégaux.

« C'était bien avant Red Hat et les autres distributions commerciales, se souvient Ian Murdock, alors étudiant en informatique à l'université de Purdue. Vous n'aviez qu'à feuilleter un magazine Unix pour trouver toutes sortes de petits encarts publicitaires proclamant 'Linux'. La plupart de ces compagnies couvraient en fait

(mai 1997). Voir E. S. Raymond, *The Cathedral and the Bazaar*, O'Reilly, 1999, 2001 (http ://www.catb.org/~esr/writings/cathedral-bazaar/). Traduction en français disponible sur http ://www.linux-france.org/article/these/cathedrale-bazar/cathedrale-bazar.html.

des opérations douteuses, et ne se gênaient pas pour glisser un peu de leur code maison [non libre] dans le mélange. »

Murdock, qui était un programmeur Unix, se souvient avoir été « emballé » par GNU/Linux après l'avoir téléchargé et installé sur son PC pour la première fois. « C'était si excitant, dit-il, que cela m'a donné envie de m'impliquer ». Mais l'explosion des distributions de mauvaise qualité refroidit son enthousiasme naissant. Décidant alors que la meilleure façon de s'impliquer était de créer une version débarrassée de tout additif, il entreprit de recenser tous les meilleurs logiciels libres disponibles, dans l'intention de les intégrer à sa propre distribution. « Je voulais faire quelque chose qui soit à la hauteur de la réputation de Linux », déclare-t-il.

Afin « d'éveiller un intérêt », Murdock envoya un message avec ses intentions sur l'Internet, y compris auprès du groupe de discussion Usenet *comp.os.linux*. L'une des toutes premières réponses qu'il reçut émanait de *rms@ai.mit.edu*. Murdock reconnut l'adresse aussitôt. Il s'agissait de Richard M. Stallman, le fondateur du projet GNU, l'homme qu'il connaissait déjà comme « le hacker des hackers ». À la vue de cette adresse parmi ses courriels en attente, Murdock n'en revenait pas. Pourquoi Stallman, alors qu'il dirigeait son propre projet de système d'exploitation, pouvait-il bien s'intéresser à ses récriminations concernant les distributions « Linux » ?

Murdock ouvrit le message. « Il disait que la Free Software Foundation commençait à regarder Linux de près et qu'il se pourrait bien qu'elle soit aussi intéressée par créer un système Linux [*sic*]. En gros, Stallman avait l'air de penser que nos objectifs étaient compatibles avec leurs principes. »

Ce message représentait un tournant dans la stratégie de Stallman. Jusqu'en 1993, il s'était contenté de ne pas se mêler des affaires de la communauté Linux. Après avoir entendu parler pour

la première fois de ce nouveau noyau, il avait demandé à un ami de vérifier s'il possédait les qualités requises. Ainsi se remémore-t-il les faits : « Il m'a rapporté que le programme était modelé sur System V, c'est-à-dire une version inférieure d'Unix. Il m'a dit aussi qu'il n'était pas portable. »

Le rapport de cet ami était correct. Conçu pour fonctionner sur des machines à base d'Intel 386, Linux était fortement lié à cette plate-forme bon marché. Ce que cet ami manqua de rapporter, en revanche, c'était l'énorme avantage du fait que Linux soit alors le seul noyau libre sur le marché. Autrement dit, tandis que Stallman allait passer l'année et demie suivante à lire les rapports d'activité du développeur de Hurd, dont la progression était plutôt lente, Torvalds ralliait les programmeurs qui porteraient ensuite Linux et GNU sur de nouvelles plates-formes.

Précision À propos de Hurd

Hurd est un double acronyme récursif signifiant « Hird of Unix-Replacement Daemons », et Hird : « Hurd of Interfaces Representing Depth ». De plus, comme l'explique Richard Stallman, la sonorité similaire entre Hurd et le mot « herd » (troupeau, en anglais), reflète bien l'idée d'un troupeau de programmes serveurs GNU, rassemblés autour du micro-noyau GNU Mach, et constituant ainsi la moitié supérieure du noyau du système GNU. Mach, quant à lui, fut développé à l'université de Carnegie Mellon entre 1985 et 1994. Après l'arrêt de son développement, le projet GNU en produisit une version modifiée : GNU Mach.

Le très ambitieux projet Hurd connut une histoire mouvementée, si bien que le souhait de voir naître une distribution complète du système GNU ne s'est toujours pas réalisé. De plus, GNU/Linux ayant fait plus que combler la principale faiblesse du projet GNU (le noyau manquant), le développement de Hurd s'en trouva d'autant plus retardé. Le projet Hurd se poursuit néanmoins aujourd

hui et une version de Debian propose Hurd comme noyau à la place de Linux : Debian GNU/Hurd. Sur l'histoire de Hurd, on peut se référer à : http://www.gnu.org/software/Hurd/history.html. Pour en apprendre plus sur Mach : http://www.cs.cmu.edu/afs/cs.cmu.edu/project/mach/public/www/mach.html.

***

En cette année 1993, l'incapacité du projet GNU à fournir un noyau fonctionnel entraînait de nombreux problèmes, aussi bien au sein même du projet que dans le mouvement du logiciel libre en général. Dans un article du magazine *Wired* paru au mois de mars, Simson Garfinkel décrivait le projet GNU comme s'étant « enlisé », en dépit du succès de ses nombreux outils [4].

Les membres du projet et de l'institution sans but lucratif qu'était la Free Software Foundation se souviennent que l'atmosphère était encore pire que ne le laissait entendre l'article de Simson Garfinkel. « Il était très clair, pour moi au moins à cette époque, qu'il y avait une fenêtre ouverte pour l'introduction d'un nouveau système d'exploitation, témoigne Chassell. Et dès l'instant où celle-ci serait refermée, les gens s'en désintéresseraient. Et c'est exactement cela qui a fini par se produire. » [5]

4. Garfinkel, 1993

5. La préoccupation de Chassell au sujet d'une « fenêtre » de trente-six mois qui allait se refermer pour le lancement d'un nouveau système d'exploitation ne se limitait pas au projet GNU. Au début des années 1990, le développement des versions libres de la Berkeley Software Distribution était entravé par les poursuites intentées par Unix System Laboratories, restreignant la distribution des logiciels dérivés de BSD. Alors que de nombreux utilisateurs considèrent les dérivées de BSD, comme FreeBSD et OpenBSD,

Les difficultés du projet GNU durant la période 1990-1993 ont fait couler beaucoup d'encre. Tandis que certains blâment Stallman, Eric Raymond, un de ses vieux amis qui avait soutenu le projet GNU sans grand enthousiasme, affirme que le problème était en grande partie institutionnel. « La FSF est devenue arrogante, explique-t-il. 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 se croyaient hors d'atteinte de tout ce qui se passait en dehors d'eux-mêmes. »

Murdock porte un jugement moins sévère : « Je pense que le problème est qu'ils étaient un peu trop ambitieux et qu'ils ont dépensé de l'argent pour rien. Au tournant des années 1990, les micro-noyaux étaient à la mode. Malheureusement, c'est à ce moment que le projet GNU a commencé à concevoir son propre noyau. Il s'est donc créé un très gros bagage, et il aurait fallu beaucoup d'efforts pour revenir en arrière. »

Et Stallman de réagir : « L'arrogance perçue par Raymond est un pur fruit de son imagination. Par contre, il voit juste quant à l'une des causes du retard de Hurd : son développeur a repensé et récrit à plusieurs reprises des portions importantes du code, en fonction de ce qu'il avait appris, au lieu d'essayer de rendre le noyau opérationnel le plus rapidement possible. C'était une bonne habitude de conception, mais elle n'était pas adaptée à notre objectif d'obtenir quelque chose de fonctionnel le plus tôt possible. »

Stallman évoque encore d'autres raisons à ce retard. D'une part, les procès Lotus et Apple réquisitionnaient en grande partie son attention ; d'autre part, des douleurs à la main, en l'empêchant

supérieures à GNU/Linux tant sur les performances que sur la sécurité, le nombre de leurs utilisateurs ne représente qu'une fraction de celui des utilisateurs de GNU/Linux. Pour une analyse du succès de ce dernier en comparaison des autres systèmes libres, se référer à l'ouvrage du hacker néo-zélandais Liam Greenwood [Greenwood, 1999].

de se servir d'un clavier pendant trois ans, l'avaient tenu éloigné de la programmation. Il cite enfin la mauvaise communication entre les différentes parties du projet GNU. « Faire fonctionner l'environnement de débogage demandait un gros travail, se souvient-il. Et l'équipe qui maintenait GDB à l'époque n'était pas très coopérative. » Les développeurs de GDB avaient en effet donné la priorité au support des plates-formes existantes sur lesquelles tournait GDB, au lieu de se consacrer au but général : l'obtention d'un système GNU complet.

Mais surtout, Stallman reconnaît que lui et les développeurs de Hurd avaient foncièrement sous-estimé la difficulté d'étendre le micro-noyau Mach à un noyau Unix complet. « Je m'étais imaginé que la partie [de Mach] parlant avec la machine avait déjà été déboguée, raconte-t-il au cours d'un discours prononcé en l'an 2000, se souvenant des déboires de l'équipe de Hurd. Grâce à cette longueur d'avance, nous aurions dû pouvoir finir le travail plus vite. Mais au contraire, il est apparu que déboguer ces programmes asynchrones et multithreadés [6] était vraiment difficile. Il y avait des bogues de synchronisation qui corrompaient les fichiers, et c'était vraiment l'enfer. Et au final, il a fallu de très nombreuses années pour obtenir une version de test » [7] (voir encart ci-après).

Note À propos des bogues de synchronisation

Dans les courriels qui ont suivi ce discours, j'ai demandé à Stallman ce qu'il voulait dire exactement par « bogues de synchronisation » (*timing bugs*). Stallman expliqua que l'expression « erreurs

6. Le multithread est la capacité d'un programme à disposer de plusieurs traitements simultanément actifs au sein d'un même processus. (Source : Le Jargon Français, http://jargonf.org) — NdT.

7. Extrait du discours du Maui High Performance Computing Center (cf. chap. 8).

temporelles » était plus exacte pour résumer le problème et fournit une explication technique de la manière dont ces erreurs temporelles peuvent gravement handicaper les performances d'un système d'exploitation : « Les erreurs temporelles se produisent dans un système asynchrone où des tâches parallèles devraient en principe pouvoir se dérouler dans n'importe quel ordre. Or, un ordre particulier amène des problèmes. Imaginez qu'un programme A fasse une tâche X, et qu'un programme B une tâche Y, où X et Y sont toutes deux de courtes tâches de routine qui examinent et mettent à jour une même structure de données. Presque toujours, l'ordinateur exécutera X avant Y ou bien Y avant X, et il n'y aura donc aucun problème. En de rares occasions, par pur hasard, l'ordonnanceur laissera le programme A s'exécuter jusqu'à mi-parcours de X, puis lancera B qui effectuera Y. Ainsi, la tâche Y sera accomplie tandis que X n'est faite qu'à moitié. Comme ces processus mettent à jour la même structure de données, il y aura des interférences. Par exemple, peut-être X aura-t-il déjà examiné la structure de données et ne verra-t-il pas qu'il y a eu une modification, qui causera un échec non reproductible. Cette erreur ne peut en effet être reproduite, car elle dépend de facteurs aléatoires (c'est-à-dire à quel moment l'ordonnanceur décide d'exécuter quel programme et sur quelle durée). Le meilleur moyen d'éviter ce genre d'erreur est d'utiliser un verrou pour s'assurer que X et Y ne se déroulent jamais au même moment. Les programmeurs de systèmes asynchrones connaissent bien l'importance générale des verrous, mais parfois, la nécessité d'un verrou à un certain endroit ou sur une certaine structure de données peut leur échapper. Le programme connaît alors une erreur de synchronisation. »

Avec le temps et le succès grandissant de GNU associé à Linux, il devint évident que le projet GNU devait prendre le train en marche, sans attendre l'achèvement de Hurd. En outre, la communauté entourant GNU/Linux présentait certaines faiblesses. Certes, Linux s'était vu doté de la licence GPL, mais ainsi que

l'avait remarqué Murdock, la volonté de traiter Linux comme un système d'exploitation entièrement libre était loin de faire l'unanimité.

Fin 1993, la population totale des utilisateurs de Linux était passée d'une douzaine de fans à un chiffre situé entre vingt mille et cent mille personnes[8]. Ce qui n'était au début qu'un hobby constituait désormais un marché mûr pour l'exploitation, et certains ne voyaient aucune objection à en tirer profit avec du logiciel non libre. Tel Winston Churchill observant les troupes soviétiques déferler sur Berlin, Stallman éprouvait un sentiment mitigé bien compréhensible à l'heure de célébrer la « victoire » de GNU/Linux[9].

***

Or, bien qu'en retard à la fête, Stallman avait toujours de l'influence. Aussitôt que la FSF eut annoncé qu'elle fournirait son argent et son soutien moral au projet logiciel de Murdock, les autres offres de soutien commencèrent à affluer. Murdock baptisa le nouveau projet Debian — fusion du prénom de sa femme, Deborah, et du sien — et en quelques semaines, la première version de la distribution fut livrée. Selon ses propres mots, « [l'appui de

8. Les chiffres concernant la population des utilisateurs de GNU/Linux sont on ne peut plus lacunaires, d'où l'étendue de la marge que j'avance ici. Le chiffre 100 000 provient du site de Red Hat « Milestones » .

9. J'avais déjà écrit cette analogie avec Winston Churchill quand Stallman m'envoya de lui-même son propre commentaire sur Churchill : « La seconde guerre mondiale et la détermination nécessaire pour vaincre ont été des souvenirs prédominants durant mon enfance. Les déclarations telles que celle de Churchill — 'Nous les combattrons sur les zones de débarquement, nous les combattrons sur les plages… jamais nous ne nous rendrons !' — ont toujours retenti en moi. »

Richard] catapulta Debian du jour au lendemain d'un petit projet intéressant à un événement qui allait retenir l'attention de la communauté. »

En janvier 1994, Murdock publia le *Manifeste Debian*. Dans l'esprit du *Manifeste GNU* de Stallman, publié dix ans plus tôt, il expliquait l'importance de travailler en étroite collaboration avec la FSF.

« La Free Software Foundation joue un rôle extrêmement important pour le futur de la distribution Debian. Le simple fait que ce soit elle qui la diffuse est un message envoyé au monde : Linux [*sic*] n'est pas un produit commercial et jamais il ne devrait l'être ; cela ne signifie pas pour autant que Linux ne sera jamais en mesure d'être compétitif. Pour ceux d'entre vous qui en douteraient, je vous mets au défi d'expliquer le succès de GNU Emacs et de GCC, qui, sans être des logiciels commerciaux, ont pourtant eu un sacré impact sur le marché économique.

Le temps est venu de se concentrer sur le futur de Linux [*sic*] plutôt que sur l'objectif destructeur consistant à s'enrichir aux dépens de toute la communauté Linux et de son avenir. Le développement et la distribution de Debian ne sont peut-être pas la réponse aux problèmes que j'ai exposés dans ce Manifeste, mais j'espère au moins qu'ils attireront suffisamment l'attention sur ces problèmes afin qu'ils puissent être résolus. » [10]

Peu après la publication du Manifeste, la FSF émit sa première requête d'importance : Stallman voulait que Murdock appelle sa distribution « GNU/Linux ». Il avait proposé pour commencer le terme « Lignux », combinant les noms Linux et GNU, mais les premières réactions très négatives l'avaient convaincu d'adopter « GNU/Linux », plus long mais moins critiqué.

10. Murdock, 1994

La démarche de Stallman consistant à ajouter le préfixe « GNU » fut interprétée par certains comme une quête tardive de reconnaissance, qu'il l'eût méritée ou non. Murdock, lui, considérait les choses différemment. Il y vit plutôt une tentative d'agir face à la tension grandissante entre les développeurs du projet GNU et ceux qui adaptaient les programmes GNU au noyau Linux. « On était au bord de la rupture, évoque-t-il. Richard était inquiet. »

⁂

Dès 1990, un « responsable du développement » était désigné pour chaque programme GNU. Certains programmes tournaient sur de nombreux systèmes différents, et les utilisateurs proposaient souvent des changements pour les porter sur de nouveaux systèmes. Cependant, ne connaissant souvent que leur propre système, ceux-ci ne cherchaient pas spécialement à garder le code propre pour qu'il reste portable facilement sur d'autres systèmes. Prendre en charge un nouveau système tout en gardant le code compréhensible — pour être maintenu de façon fiable pour tous les systèmes — demandait donc la récriture d'une bonne partie des modifications.

Le responsable d'un programme était ainsi chargé d'évaluer les modifications des auteurs/utilisateurs et de leur expliquer comment reprendre certaines parties du portage. Généralement, ces derniers étaient très empressés à s'exécuter, pour voir leurs modifications intégrées à la version standard. Enfin, le responsable incorporait le code ainsi retravaillé, qui serait désormais pris en compte dans la maintenance du programme. Pour certains programmes, ce processus s'était répété des dizaines de fois, pour autant de systèmes différents.

Quant aux programmeurs qui adaptaient les divers programmes GNU au noyau Linux, ils adoptaient tous peu ou prou la même attitude : ils ne se préoccupaient que de leur plate-forme. Mais lorsque les responsables des programmes GNU les sollicitaient pour nettoyer leurs modifications en vue de la maintenance successive, plusieurs d'entre eux faisaient la sourde oreille. Peu leur importait de faire les choses correctement ou de faciliter la maintenance des paquets GNU qu'ils avaient adaptés. Seules leurs propres versions les intéressaient, la tendance étant alors à les maintenir sous forme de branches séparées, ou *forks*.

Dans l'univers des hackers, la question des *forks* est du plus grand intérêt. Bien qu'il soit éthiquement permis à un programmeur de faire ce qu'il veut avec le code source d'un programme, il est de bon ton de proposer au développeur original de travailler conjointement pour maintenir une version unique. Les hackers trouvent ainsi habituellement plus utile et approprié de reverser leurs améliorations dans la version principale du programme. Et si une licence de logiciel libre autorise tout un chacun à créer une nouvelle branche d'un programme (le « forker ») — ce qui s'avère parfois nécessaire, le faire sans raison particulière est considéré un acte quelque peu désobligeant.

En tant que leader du projet GNU, Stallman avait déjà fait l'expérience des méfaits d'un fork logiciel en 1991. Il raconte : « Une entreprise, Lucid, a recruté quelques personnes pour écrire des améliorations à GNU Emacs, améliorations censées contribuer à ce programme. Mais les développeurs ne m'ont pas tenu informé de leur projet, développant au lieu de cela plusieurs nouvelles fonctionnalités, de leur propre chef. Comme on pouvait s'y attendre, j'ai approuvé certaines de leurs décisions et récusé les autres. Ils m'ont alors demandé d'intégrer la totalité de leur code, mais apprenant mon intention de n'en utiliser que la moitié, ils ont alors refusé de m'aider à l'adapter. J'ai dû me débrouiller seul. » Le fork

avait donné naissance à une version parallèle, Lucid Emacs, ainsi qu'à un amer ressentiment [11].

Plusieurs des principaux paquets GNU avaient ainsi donné lieu très rapidement à des forks. Au début, Stallman considérait ces forks comme le produit de l'impatience des développeurs. À l'opposé de la dynamique rapide et informelle de l'équipe en charge de Linux, les mainteneurs du code source de GNU tendaient à être plus lents et circonspects avec des changements susceptibles d'affecter la viabilité à long terme d'un programme. De plus, ils n'hésitaient pas à critiquer sévèrement le code des autres. Avec le temps, à la lecture des courriels envoyés par les développeurs de Linux, Stallman commença à réaliser qu'au fond, peu d'entre eux avaient vraiment conscience du projet GNU et de ses objectifs.

« Nous avons découvert que les gens qui se targuaient d'être des « utilisateurs de Linux » se moquaient bien du projet GNU, exprime Stallman. 'Pourquoi devrais-je m'embêter à faire tout ça ? disaient-ils. Je n'ai que faire du projet GNU, puisque ça [le programme] marche pour moi. Ça marche pour nous, utilisateurs de Linux, et c'est tout ce qui compte à nos yeux.' C'était vraiment étonnant, que ces personnes utilisant pour l'essentiel une variante du système GNU s'en soucient si peu. Ils se souciaient de GNU encore moins que quiconque ! »

Abusés par leur propre habitude d'appeler la combinaison des deux programmes « Linux », ils ne se rendaient pas compte que leur système tenait plus de GNU que de Linux.

11. Jamie Zawinski, un ancien programmeur de Lucid, qui passa ensuite à la tête de l'équipe de développement de Mozilla, possède un site Internet documentant le fork Lucid/GNU Emacs, intitulé « *The Lemacs/FSFmacs Schism* » : http://www.jwz.org/doc/lemacs.html. La réponse de Stallman se trouve sur son site personnel, dans l'article intitulé « *The origin of XEmacs* » : http://stallman.org/articles/xemacs.origin.

Afin de préserver l'unité, Stallman demanda aux responsables de développement de faire eux-mêmes le travail qui aurait dû incomber aux auteurs des modifications. Dans la plupart des cas, la tâche était envisageable, mais pas dans celui de Glibc. Abréviation de *GNU C Library*, Glibc est le paquet qui permet à tous les programmes d'adresser des « appels système » au noyau, en l'occurrence Linux. En effet, tout programme utilisateur, sur un système de type Unix, ne communique avec le noyau qu'à travers la bibliothèque C.

Les modifications visant à faire fonctionner Glibc comme canal de communication entre Linux et tous les autres programmes du système étaient importantes et ciblées, écrites sans tenir compte de leurs conséquences possibles une fois portées sur d'autres plates-formes. Pour le responsable de Glibc, le travail de nettoyage avait de quoi faire peur. Dès lors, la FSF préféra le payer presque une année durant pour réimplémenter ces changements à partir de zéro, et livrer une version 6 de Glibc « clé en main » pour GNU/Linux.

***

Murdock rapporte que c'est la raison qui aurait précipité Stallman à vouloir à tout prix ajouter le préfixe GNU lorsque Debian publia sa distribution logicielle. « Depuis, le fork a réintégré la branche initiale. Mais, à l'époque, grande était l'inquiétude dans la communauté GNU de voir la communauté Linux faire bande à part, au risque d'aggraver la désunion. »

Si certains considéraient qu'assimiler la combinaison de GNU et de Linux à une « variante » de GNU était faire preuve d'avidité politique, Murdock, déjà sympathisant de la cause des logiciels libres, jugea honnête la requête de Stallman demandant à ce que

la version de Debian soit estampillée « GNU/Linux ». « Elle traduisait davantage la recherche d'une unité que celle d'une reconnaissance », déclare-t-il.

Des demandes de nature plus technique s'ensuivirent. Or, bien que Murdock eût été arrangeant sur les questions d'ordre politique, il se montra plus ferme au sujet de la structure et du modèle de développement du logiciel. Ce qui avait commencé comme une démonstration de solidarité devint rapidement un désaccord permanent.

« Ah pour ça, j'ai eu ma part d'oppositions avec lui, assure Murdock en riant. En toute honnêteté, Richard est une personne avec qui il peut être assez difficile de travailler. » Le principal différend portait sur le débogage. Stallman voulait inclure des informations de débogage dans tous les programmes exécutables, de manière à permettre aux utilisateurs/développeurs d'identifier immédiatement les bogues qu'ils auraient pu rencontrer. Au contraire, Murdock estimait que cela rendrait le système de fichiers trop volumineux et affecterait la distribution. Or aucun des deux n'était disposé à changer d'avis.

***

En 1996, ayant obtenu son diplôme à 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, qui était bien connu pour son travail sur Electric Fence [12]. Perens, tout comme Murdock, était un programmeur Unix tombé amoureux de GNU/Linux sitôt que ses propriétés Unix furent devenues manifestes. Et tout comme Murdock, Perens sympathisa avec les objectifs politiques de Stallman et de la FSF, quoique de plus loin.

12. Sur Electric Fence, cf. note chap. 9.

« Je me rappelle que bien après que Stallman ait publié le Manifeste GNU, GNU Emacs et GCC, j'avais lu un article disant qu'il travaillait comme consultant pour Intel, dit Perens, évoquant son premier accrochage avec 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 pensé que sa réponse était parfaitement logique. »

En revanche, Perens, éminent développeur Debian, assistait avec consternation aux batailles opposant Stallman et Murdock au sujet de la conception de Debian. Dès son accession à la tête de l'équipe de développement, il prit la décision de mettre de la distance entre Debian et la FSF. « Je ne voulais pas que Richard ait son mot à dire sur la moindre de nos décisions », explique-t-il.

Selon lui, Stallman fut pris de court par la décision, mais eut la sagesse de s'en accomoder : « Il a laissé les choses se calmer, puis a envoyé un message comme quoi nous avions vraiment besoin d'être en relation. Il a demandé que nous appelions le système GNU/Linux, et en est resté là. Cela m'a semblé convenable et j'ai accepté. J'ai pris seul cette décision. Tout le monde a soupiré de soulagement. »

***

Petit à petit, Debian allait se forger une réputation de « version hacker » de GNU/Linux, aux côtés de Slackware, autre distribution populaire née à la même période en 1993-94. Cependant, Slackware contenait des programmes non libres et Debian, après sa séparation d'avec le projet GNU, l'imita[13]. Bien que signalés

13. En juin 1996, Debian Buzz contenait le programme non libre Netscape 3.1, dans sa section Contrib.

comme « non libres » et ne faisant « pas officiellement partie de Debian », ces paquets ne pouvaient être proposés aux utilisateurs sans que cela ne leur donne une caution implicite. Lorsque le projet GNU prit connaissance de ces pratiques, il dut convenir que ni Debian ni Slackware ne pouvaient être recommandées au public en tant que distributions GNU/Linux.

Hors du royaume des systèmes pour hackers, pourtant, GNU/-Linux commençait à faire parler de lui sur le marché des Unix commerciaux. En Caroline du Nord, une entreprise Unix nommée Red Hat était en pleine réorganisation de ses activités pour les recentrer sur GNU/Linux. Son président directeur général était Robert Young, l'ancien éditeur du *Linux Journal* qui, en 1994, avait demandé à Linus Torvalds s'il regrettait d'avoir mis le noyau sous licence GPL.

La réponse de Torvalds avait alors profondément influencé la vision qu'avait Young de GNU/Linux. Au lieu de chercher à s'imposer sur le marché GNU/Linux par les tactiques habituelles, il se demanda ce qui se passerait si une entreprise adoptait la même démarche que Debian, à savoir assembler un système d'exploitation uniquement à partir de logiciels libres. Cygnus Solutions, l'entreprise montée par Michael Tiemann et John Gilmore en 1990, démontrait déjà qu'il était possible de vendre des logiciels libres en s'appuyant sur la qualité et la personnalisation. Et si Red Hat adoptait cette approche avec GNU/Linux ?

« Dans la tradition scientifique occidentale, on s'élève en s'appuyant sur les épaules des géants, affirme Young, faisant écho à la fois Torvalds et à Sir Isaac Newton avant lui. Dans le monde des affaires, cela signifie éviter de réinventer la roue. La beauté du modèle [de la GPL], c'est que le code est placé dans le domaine public [14]. Mettons que vous soyez un fournisseur de logiciels indépendant et que vous essayiez de concevoir une application, pour

14. Young utilise le terme « domaine public » de manière erronée ici. Le domaine public signifie « non couvert par copyright ». Or le code sous licence

laquelle vous ayez besoin d'un composeur pour modem, eh bien… pourquoi réinventer les composeurs ? Vous n'avez qu'à récupérer PPP [15] sur Red Hat [GNU/]Linux et en faire le cœur de votre outil de composition pour modems. Si vous avez besoin d'outils graphiques, pas besoin d'écrire votre propre bibliothèque graphique… Il suffit de télécharger GTK [16] ! Vous vous retrouvez soudainement avec la possibilité de réutiliser le meilleur de ce qui a déjà été créé avant vous. Et tout aussi soudainement, votre attention, en tant que vendeur de logiciels, se trouve moins portée sur la gestion des logiciels que sur l'écriture d'applications répondant spécifiquement aux besoins de vos clients. »

En dépit de ces déclarations, Young n'était pas un militant du logiciel libre, incluant volontiers des logiciels non libres dans le système GNU/Linux Red Hat.

***

Young n'était pas le seul professionnel du logiciel à être attiré par l'efficacité commerciale des logiciels libres. Fin 1996, la plupart des compagnies Unix commençaient à se réveiller et à flairer l'attrait du code source. Et si on en était encore à une bonne année ou deux du passage au plein déploiement commercial du secteur GNU/Linux, quelque chose d'énorme était en train de se produire

GNU GPL ne peut être dans le domaine public car il doit être sous copyright pour que la GNU GPL ait force de loi.

15. PPP est le composeur pour modem (*modem-dialer*) inclus dans la distribution Red Hat.

16. GTK (*The Gimp ToolKit*) ou GTK+ est un ensemble de bibliothèques logicielles permettant de réaliser des interfaces graphiques. Cet outil a été développé à l'origine pour le logiciel de graphisme Gimp, mais est utilisé dans de nombreux projets, notamment les environnements de bureau (Gnome, XFCE, etc.).

— tous les familiers du milieu hacker le sentaient. Le processeur Intel 386, l'Internet et le World Wide Web avaient déferlé sur le marché comme autant de raz-de-marée. Quant au logiciel libre, il semblait déjà former la plus grosse lame de fond en préparation.

Aux yeux de Ian Murdock, cette déferlante était tout autant un hommage qu'une punition bien mérités pour l'homme qui s'était consacré si longtemps à forger l'identité du mouvement du logiciel libre. Comme de nombreux fans de GNU/Linux, Murdock avait lu les tout premiers messages et échanges postés. Il avait lu l'avertissement de Torvalds mentionnant Linux comme un simple « passe-temps » et son aveu au créateur de Minix, Andrew Tanenbaum : « Si le noyau GNU avait été prêt le printemps dernier, je ne me serais même pas donné la peine de lancer mon projet » [17]. Comme beaucoup, Murdock savait que certaines occasions avaient été manquées. Il connaissait aussi l'excitation ressentie en voyant de nouvelles opportunités émerger de la structure même de l'Internet.

« Être impliqué dans Linux à ses débuts était amusant, se rappelle-t-il. En même temps, cela occupait et permettait de passer le temps. Si on relit les vieux échanges sur *[comp.os.minix]*, c'est ce sentiment qui se dégage : Linux était quelque chose avec quoi jouer en attendant que le noyau Hurd soit prêt. Les gens étaient impatients. C'est drôle, mais j'ai l'impression qu'il n'y aurait jamais eu de Linux si Hurd était sorti plus rapidement. »

Fin 1996, toutefois, de telles considérations n'avaient plus lieu d'être, le noyau de Torvalds ayant déjà atteint une masse critique

17. Cette citation est tirée de l'échange houleux et très médiatisé sur l'Internet, qui opposa Torvalds à Tannenbaum après la parution de la première version de Linux. Pour défendre son choix d'un noyau monolithique non portable, Torvalds affirme avoir commencé à travailler sur Linux dans le but de connaître davantage son nouveau PC 386. Voir DiBona, Ockman & Stone, 1999, p. 224.

d'utilisateurs. La fenêtre des trente-six mois s'était refermée, signifiant que même si le projet GNU sortait son noyau Hurd, les chances auraient été bien minces pour que, en dehors du noyau dur de la communauté hacker, quelqu'un le remarque. Linux, en comblant la dernière lacune du système GNU, avait accompli la mission du projet GNU, à savoir la production d'un système d'exploitation libre de type Unix.

Quoi qu'il en soit, la plupart des utilisateurs ne comprirent pas ce qu'il s'était vraiment passé, croyant que Linux était le système entier et que Torvalds était son créateur. La plupart installèrent des distributions incluant des programmes non libres et, avec Torvalds comme guide éthique, n'y virent aucune raison de les rejeter.

Malgré tout, une liberté précaire restait à la portée de ceux qui savaient l'apprécier.

Chapitre

11

Richard Stallman et l'open source

Notes de Richard Stallman sur ce chapitre

Dans ce chapitre (uniquement), j'ai effacé plusieurs citations concernant l'open source, qui ne concernaient pas directement mon œuvre ou ma vie.

En novembre 1995, Peter Salus, membre de la FSF et auteur d'un ouvrage intitulé *A Quarter Century of Unix* (1994) [1], lança un appel à contribution aux membres de la liste de discussion *system-discuss* du projet GNU. Salus voulait ainsi avertir ses collègues de

1. En français, le titre pourrait être traduit « Unix, un quart de siècle ! ». Peter Salus, *A Quarter Century of Unix*, Addison-Wesley, 1994.

la tenue de la conférence sur le « logiciel librement redistribuable » à Cambridge (Massachusetts) qu'il devait présider. Prévue pour février 1996 et sponsorisée par la FSF, la manifestation s'annonçait comme la première dédiée entièrement au logiciel libre. Dans un esprit d'unité avec tous les développeurs de logiciels libres, elle était ouverte aux interventions sur « tout aspect de GNU, Linux, NetBSD, 386BSD, FreeBSD, Perl, Tcl/tk, et autres outils dont le code est accessible et redistribuable ».

Tel fut l'appel de Salus : « Depuis les quinze dernières années, les logiciels gratuits et bon marché sont devenus omniprésents. Cette conférence réunira des développeurs de logiciels librement redistribuables, ainsi que leurs éditeurs (tous supports de distribution confondus). Elle proposera des tutoriels, des débats, ainsi que des conférences plénières de Linus Torvalds et Richard Stallman. » [2]

Parmi les destinataires du message, on comptait Eric S. Raymond, membre du comité organisateur. Bien que, contrairement aux autres membres, il ne soit pas le dirigeant d'un projet ou d'une entreprise, il s'était acquis une bonne réputation dans la communauté des hackers grâce à plusieurs projets de logiciels et à l'édition du *New Hacker's Dictionary* — une version revue et largement augmentée du *Hacker's Dictionary* publié dix ans plus tôt par Guy Steele. [3]

Pour Raymond, la conférence de 1996 était une occasion. S'il n'était pas un défenseur inconditionnel des idées du mouvement pour le logiciel libre, il avait contribué à plusieurs programmes

2. Voir Peter Salus, « *FYI-Conference on Freely Redistributable Software, 2/2, Cambridge* », 1995 (archives de Terry Winograd) — http://bat8.inria.fr/%7Elang/hotlist/free/licence/fsf96/call-for-papers.html.

3. G. Steele, E. S. Raymond, *The New Hacker's Dictionary*, MIT Press, 1991, 1993, 1996. G. Stelle, *The Hacker's Dictionary : A Guide to the World of Computer Wizards*, Harpercollins, 1984.

GNU, en particulier GNU Emacs. Il avait mis fin à ses contributions en 1992, après avoir exigé l'autorisation d'apporter des modifications à la version officielle de GNU Emacs et ceci sans en discuter préalablement avec Stallman, alors responsable du logiciel. Stallman refusa ces exigences, et Raymond l'accusa de *micro-management*, c'est-à-dire de vouloir tout contrôler dans les moindres détails. « Richard avait fait un scandale sur le fait que j'aie procédé à des modifications non autorisées alors que je nettoyais les bibliothèques Lisp d'Emacs, se souvient Raymond. Ça m'a tellement mis en colère que j'ai décidé de ne plus travailler avec lui. »

Malgré cette brouille, 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'orateurs principaux, il appuya l'idée avec enthousiasme. L'unité symbolique de l'association entre Stallman, représentant le contingent des vieux sages ITS/Unix, et Torvalds, figure de proue des jeunes et énergiques partisans de Linux, ne pouvait se révéler que bénéfique, surtout pour les jeunes hackers ambitieux (c'est-à-dire âgés de moins de quarante ans), à l'image de Raymond. « J'avais pour ainsi dire un pied dans chaque camp », commente-t-il.

Lorsqu'arriva le moment de la conférence, la tension entre les deux groupes était devenue palpable. Un point commun les réunissait pourtant : la conférence leur offrait l'occasion de voir pour la première fois le jeune prodige finlandais en chair et en os. Étonnamment, Torvalds se révéla un orateur charmant et affable. Avec seulement un léger accent suédois [4], il surprit son public par sa

4. Bien que Linus Torvalds soit finlandais, sa langue maternelle est le suédois. « *The Rampantly Unofficial Linus FAQ* » l'explique brièvement : une minorité significative (environ 6 %) de la population finlandaise parle suédois. S'appelant eux-mêmes *finlandssvensk* ou *finlandssvenskar*, ils 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.

vivacité d'esprit et sa modestie. Encore plus surprenant, rapporte Raymond, Torvalds se montrait tout aussi décidé à tirer à boulets rouges sur d'autres éminents hackers, y compris le plus auguste d'entre eux, Richard Stallman. À la fin de la conférence, le style « hacker indolent » du jeune homme allait remporter les suffrages des conférenciers, tous âges confondus.

« C'était un moment clef, se rappelle Raymond. Avant 1996, Richard était le seul prétendant crédible pour être le leader idéologique de la culture toute entière. Personne ne le désapprouvait en public. Ce fut Torvalds qui brisa ce tabou. »

La rupture fut définitivement consommée à la fin de la conférence. Lors d'une discussion relative à la domination croissante du marché par Microsoft Windows ou un sujet du même accabit, Torvalds admit être un amateur du programme de présentation PowerPoint. Du point de vue de la vieille garde et de ses puristes, c'était comme se vanter d'avoir des esclaves au beau milieu d'une conférence abolitionniste. Du point de vue de Torvalds et de ses supporters, dont le nombre allait grandissant, ce n'était que du bon sens. Pourquoi fuir les bons logiciels, même non libres, par pure idéologie ? Être un hacker ne signifiait pas faire vœu d'abnégation, mais produire des résultats. Or ces résultats, selon eux, ne pouvaient être appréciés que dans la pratique.

Il faut bien dire que cette idéologie n'était pas la leur. N'accordant pas de valeur à la liberté, ils voyaient les sacrifices qu'elle exigeait non comme un moyen d'obtenir une chose importante, mais comme un déni de soi. « C'était une déclaration sacrément choquante, se souvient Raymond. Mais encore une fois, Torvalds pouvait se le permettre, parce qu'en 1995-96, son influence n'avait cessé de croître. »

Stallman, de son côté, ne se remémore aucune tension lors de cette conférence en 1996. Probablement n'avait-il pas assisté à la

fameuse sortie de Torvalds. En revanche, il se souvient avoir été par la suite piqué au vif par l'effronterie tant acclamée du Finlandais. « Dans la documentation de Linux, on trouvait un passage qui incitait les gens à imprimer les standards de codage GNU pour ensuite les déchirer, cite par exemple Stallman. Si on y regarde de plus près, la partie avec laquelle il n'était pas d'accord était en fait la moins importante : la recommandation sur la manière d'indenter le code C. »

Et d'ajouter : « Qu'il ne soit pas d'accord avec certaines de nos conventions, ce n'est pas un problème… mais il a choisi une manière étonnamment méchante pour le faire savoir. Il aurait pu dire simplement : 'Voilà comment je pense qu'il faut mettre en forme le code'. Parfait. Je ne vois pas pourquoi il devrait y avoir matière à hostilité ici. »

⁂

Pour Raymond, l'accueil chaleureux réservé aux propos de Torvalds par les autres hackers ne fit que confirmer ses soupçons. La séparation entre les développeurs de Linux et ceux de GNU était essentiellement générationelle. Un grand nombre de programmeurs Linux, comme Torvalds, avaient grandi dans un monde dominé par le logiciel privateur. Ils avaient commencé à contribuer au logiciel libre sans percevoir le caractère illégitime des logiciels non libres.

Ainsi, pour la plupart d'entre eux, seule la commodité d'utilisation comptait. À moins qu'un logiciel ne soit techniquement inférieur, ils ne voyaient pas pourquoi le refuser pour de simples raisons de licence. Un jour, les hackers finiraient par mettre au point une alternative libre à Powerpoint. Mais en attendant, pourquoi critiquer Microsoft ou PowerPoint, et pourquoi ne pas utiliser ce logiciel ?

C'était un bon exemple du clivage de plus en plus marqué dans la communauté du logiciel libre, entre ceux qui accordaient de l'importance à la liberté en soi, et ceux qui ne prenaient en compte que la fiabilité et la puissance des logiciels. Stallman désignait ces deux camps comme deux partis politiques au sein d'une même communauté. Il nomma le premier le « parti pour la liberté ». Et comme le second camp ne daignait pas se donner un nom, il l'appela ironiquement « le parti des opportunistes » ou « le parti pour le succès ». Nombre des partisans de ce dernier considéraient en effet que l'objectif premier était d'attirer toujours « plus d'utilisateurs ».

⁂

Lors de la décennie suivant le lancement du projet GNU, Stallman s'était construit une réputation redoutable en tant que programmeur. À cela s'ajoutait une réputation d'intransigeance en termes de conception de logiciel et de gestion humaine. Tout cela était partiellement mérité, mais donnait aussi une bonne excuse pour accabler Stallman dès lors qu'il n'agissait pas comme on l'aurait voulu. Sans compter que cette réputation était exagérée par des suppositions infondées.

Peu avant la conférence de 1996, la FSF dut faire face à des démissions d'équipes généralisées. Brian Youmans, un employé engagé par Salus suite à ces départs, se souvient de la scène : « Il arriva un moment où Peter [Salus] était le seul employé restant à travailler dans les locaux. »

En réalité, l'équipe précédente était mécontente de la directrice générale. Comme l'explique Bryt Bradley à ses amis en décembre 1995 : « [Anonyme], directrice générale de la FSF, a décidé de revenir de son congé médical de complaisance la semaine dernière.

Nous, employés du bureau (Gena Bean, Mike Drain et moi-même), avons décidé que nous ne pouvions plus travailler avec elle comme supérieure hiérarchique, étant donné les nombreuses erreurs qu'elle a commises dans ses tâches professionnelles avant son départ en congé. De plus, à de nombreuses reprises, certains ont subi des menaces de licenciement abusif, sans compter les nombreux exemples de ce qui a été ressenti comme des violences verbales envers *tous* les membres du bureau. Nous avons demandé (plusieurs fois) à ce qu'elle ne soit plus notre supérieure, tout en déclarant que nous étions prêts à continuer à collaborer avec elle sur un pied d'égalité. Nos demandes ayant été ignorées, nous présentons notre démission. »

La directrice générale en question imposa alors un ultimatum à Stallman : soit il lui donnait une autonomie totale au bureau, soit il acceptait sa démission. En tant que président de la FSF, il refusa de lui donner un contrôle exclusif sur les activités de la fondation. Elle démissionna et Stallman recruta Peter Salus pour la remplacer.

Lorsque Raymond, pourtant étranger à ces histoires, apprit que ces personnes quittaient la FSF, il présuma que Stallman en était responsable. Cela semblait confirmer sa théorie que c'était la personnalité de Stallman qui était la cause du moindre problème survenant dans le projet GNU.Raymond avait encore une autre théorie : les récents retards comme celui de Hurd et le schisme Lucid-Emacs reflétaient des problèmes de gestion de projet, plus que de développement de code.

***

Peu après la conférence sur « le logiciel librement redistribuable », Raymond se mit à travailler sur son propre projet de

logiciel, un utilitaire de récupération de courriers électroniques 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 s'était attendu à se retrouver devant une pagaille monstrueuse, trouva que le logiciel était tout compte fait étonnamment robuste.

Analysant le succès de l'approche initiée par Torvalds, il en tira une rapide conclusion : en utilisant l'Internet comme sa « boîte de Petri » [5] et la surveillance sans complaisance de la communauté hacker comme moyen de sélection naturelle, l'étudiant finlandais avait créé un modèle évolutionniste débarrassé de toute planification centrale.

Raymond s'aperçut que cette méthode permettait en outre de contourner la loi de Brooks. Énoncée pour la première fois par Fred P. Brooks, responsable du projet OS/360 chez IBM et auteur du livre *The Mythical Man-Month* en 1975, cette loi postulait que l'ajout de développeurs à un projet ne faisait qu'entraîner des retards supplémentaires. Croyant comme la plupart de ses confrères programmeurs que le logiciel, à l'instar de la soupe, a tout avantage à être préparé par un nombre limité de cuisiniers, Raymond sentit que quelque chose de révolutionnaire était à l'œuvre. En invitant toujours plus de cuisiniers à la cuisine, Torvalds avait effectivement réussi à rendre le logiciel « meilleur » [6].

5. Il s'agit d'une boîte cylindrique en verre ou en plastique permettant la mise en culture de micro-organismes. Ce dispositif fut inventé par le bactériologiste allemand Julius Richard Petri (1852-1921). — NdT.

6. La loi de Brooks est la forme courte de la citation suivante, tirée du livre de l'auteur : « Puisque la fabrication de logiciels est intrinsèquement le fruit de l'effort conjugué de différents systèmes — un exercice d'interactions complexes — l'effort de communication est grand, et il contrebalance rapidement la diminution du temps de travail individuel rendue possible par le partage des

Raymond coucha ses observations sur le papier et les mit en forme dans un discours, qu'il lut rapidement devant un groupe d'amis et de voisins, dans le comté de Chester, en Pennsylvanie. Intitulé *La cathédrale et le bazar*, le texte établissait un contraste entre le management de type « bazar » initié par Torvalds, et celui de style « cathédrale », utilisé habituellement. D'après Raymond, les réactions furent enthousiastes, et elles le furent encore plus au printemps suivant, lors du Linux Kongress de 1997 qui réunissait des utilisateurs GNU/Linux allemands.

« Au Kongress, ils se levèrent pour me faire une ovation à la fin du discours, se rappelle Raymond. J'y ai vu un symbole, pour deux raisons. D'une part, cela traduisait leur enthousiasme sur le fond du discours, d'autre part, cet enthousiasme était manifeste en dépit du fait que le discours ait été prononcé dans une langue étrangère. »

Raymond finit par convertir ce discours en article, intitulé lui-aussi *La cathédrale et le bazar*. Le texte tirait son nom de l'analogie centrale de Raymond. Auparavant, les programmes étaient des « cathédrales », monuments impressionnants dont la construction était planifiée de façon centralisée, et conçus pour résister aux assauts du temps. À l'opposé, Linux ressemblait davantage à « un grand bazar bruyant », un projet logiciel développé grâce aux dynamiques floues et décentralisées de l'Internet.

L'article de Raymond associait le style « cathédrale » — que lui, Stallman et bien d'autres avaient utilisé — tout spécialement au projet GNU et à Stallman, faisant ainsi glisser l'opposition entre deux modèles à une opposition entre deux personnes.

Là où Stallman incarnait le modèle classique du bâtisseur de cathédrale, c'est-à-dire un programmeur « maître » qui pouvait

tâches. Au lieu de réduire les délais, ajouter plus de personnel les allonge. » Voir [Brooks, 1995].

disparaître pendant dix-huit mois et revenir avec un programme comme le compilateur C GNU, Torvalds était plutôt un aimable organisateur de soirées. En laissant les autres diriger la discussion sur la conception de Linux, n'intervenant que lorsque le débat requérait un arbitre, il avait créé un mode de développement reflétant parfaitement sa personnalité décontractée. Pour lui, la tâche d'encadrement la plus importante ne consistait pas à imposer un contrôle, mais à veiller à maintenir un flux continu d'idées.

Raymond résume ainsi : « Je pense que le coup de maître de Linus n'a pas été le noyau Linux en lui-même, mais plutôt l'invention du modèle de développement associé. » [7]

Or, si la description des deux styles de développement était pénétrante, associer le mode « cathédrale » à Stallman en particulier (plutôt qu'à n'importe qui d'autre l'ayant utilisé, ce qui inclut Raymond lui-même) était pure calomnie. En effet, bien avant que Raymond ne s'y intéresse, les développeurs de plusieurs paquetages GNU, y compris GNU Hurd, avaient découvert et adopté les méthodes de Torvalds, sans toutefois les analyser en profondeur, ni s'en faire publiquement le champion à l'instar de Raymond dans son article.

Dire qu'à cause de ce texte diffamant, des milliers de hackers ont sans doute nourri des sentiments négatifs envers le projet GNU…

⁂

En rapportant les secrets du succès de la gestion de projet à la Torvalds, Raymond attira l'attention des membres de la communauté du logiciel libre pour qui la liberté n'était pas une priorité.

7. Raymond, 2000

Ceux-là cherchaient plutôt à susciter l'intérêt des entreprises pour l'usage et le développement de logiciels libres. Ainsi décidèrent-ils de traduire les enjeux de ces logiciels en termes relevant de la logique commerciale : puissants, fiables, bon marché, avancés.

Raymond devint le principal porte-parole de ces idées, qui parvinrent à l'oreille des dirigeants de Netscape, dont le navigateur — non libre — perdait des parts de marché face à celui — non libre — de Microsoft, Internet Explorer. Intrigués par un discours de Raymond, des cadres 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 la contribution des hackers dans les développements futurs.

Quand le PDG de Netscape, Jim Barksdale, cita le traité de Raymond *La cathédrale et le bazar* comme principale source d'influence de la décision de l'entreprise, il éleva immédiatement Raymond au rang de célébrité parmi les hackers. Il organisa une réunion entre Larry Augustin, fondateur de VA Research, qui commercialisait des postes de travail équipés de GNU/Linux, Tim O'Reilly de la maison d'édition O'Reilly & Associates, et Christine Peterson, présidente de l'Institut Foresight, une *think tank* de la Silicon Valley chargée de la réflexion sur les nanotechnologies.

« L'ordre du jour de la réunion était focalisé sur un point : comment utiliser la décision de Netscape pour donner envie à d'autres entreprises de l'imiter ? »

Raymond ne se souvient pas de toute la discussion, mais il se rappelle de la première plainte formulée. En effet, malgré les efforts de Stallman et des autres hackers pour rappeler aux gens que *free* dans *free software* signifiait liberté et non gratuité, le message ne passait toujours pas. La plupart des hommes d'affaires, après avoir entendu le terme pour la première fois, y voyaient un synonyme

de « coût zéro », écartant rapidement toute autre interprétation. En attendant que les hackers trouvent un moyen de dissiper ce malentendu, le mouvement pour le logiciel libre était face à une difficulté, même après l'initiative de Netscape.

***

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

Rétrospectivement, elle dit avoir pensé à cette expression alors qu'elle évoquait la décision de Netscape, au cours d'une discussion avec un ami travaillant dans les relations publiques de l'industrie du logiciel. À défaut de se rappeler l'origine de ce terme, peut-être emprunté à un domaine tout autre, elle se souvient très bien que son ami n'avait pas aimé cette proposition. [8]

Durant la réunion, raconte Peterson, la réaction avait été totalement différente : « J'hésitais à suggérer ce terme. Je n'avais pas vraiment de poids au sein du groupe, alors j'ai commencé à l'utiliser l'air de rien, sans souligner que c'était une nouvelle expression ». À sa grande surprise, celle-ci reçut bon accueil et à la fin de l'entrevue, la plupart des gens présents, y compris Raymond, semblaient l'apprécier.

Ce dernier affirme n'avoir pas utilisé publiquement l'expression *open source* pour « logiciel libre » avant la fête inaugurale de Mozilla, à l'occasion de laquelle Tim O'Reilly avait organisé une rencontre autour du logiciel libre. En lui donnant le nom de « Sommet du Freeware », l'organisateur voulait attirer l'attention des

8. Voir Malcolm Maclachlan, « Profit Motive Splits Open Source Movement », *TechWeb News* (26 Août 1998). http://content.techweb.com/wire/story/TWB19980824S0012.

médias et du public sur les autres projets méritants qui avaient incité Netscape à publier Mozilla. « Tous ces gars avaient tant de choses en commun, que j'étais surpris qu'ils ne se connûssent pas déjà les uns les autres, raconte O'Reilly. Je voulais aussi montrer au monde quel impact avait déjà eu la culture du logiciel libre. En effet, les gens passaient à côté d'une grande partie de cette tradition. »

Pourtant, en mettant au point la liste des invités, O'Reilly prit une décision qui allait avoir des conséquences politiques à long terme. Il décida de restreindre la liste aux développeurs de la côte ouest comme Larry Wall, créateur de Perl, Eric Allman, créateur de Sendmail, et Paul Vixie, créateur de Bind. Il y avait certes des exceptions : Raymond, résident en Pennsylvanie, était déjà sur place du fait du lancement de Mozilla et fut vite invité, de même que Guido van Rossum, de Virginie, créateur de Python. « Frank Willison, mon éditeur en chef et expert de Python dans l'entreprise, l'invita sans même me demander, se souvient O'Reilly. J'étais ravi de l'avoir là, mais au début, ce ne devait être qu'un simple rassemblement local. »

Pour certains, le refus d'inclure le nom de Stallman sur la liste relevait du plus grand mépris. « J'ai décidé de ne pas y aller à cause de ça », se souvient Perens. Raymond, qui lui était présent, dit avoir plaidé la cause de Stallman, sans résultat. La rumeur prit de l'ampleur, O'Reilly, hôte de l'événement, s'étant publiquement disputé avec son confrère au sujet du copyright des manuels d'utilisation des logiciels. Selon Stallman, les manuels des logiciels libres devaient pouvoir être librement copiés et modifiés, ce à quoi O'Reilly rétorquait qu'un marché à valeur ajoutée pour les livres permettait d'accroître l'utilité du logiciel libre, le rendant accessible à un public plus vaste.

De plus, tous deux s'étaient opposé à propos du titre de l'évènement, Stallman insistant pour *Free Software* au lieu de *Free-*

ware puisque ce dernier terme est utilisé le plus souvent pour désigner des programmes disponibles gratuitement, mais dont le code source n'est pas publié.

A posteriori, O'Reilly ne considère pas comme une preuve de mépris sa décision de ne pas compter Stallman parmi les invités : « Je n'avais encore jamais rencontré Richard en personne mais, dans nos échanges de courriels, il s'était montré inflexible et peu enclin au dialogue. Comme 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. Je savais qu'ils étaient très attachés aux valeurs de la GPL, tout en se montrant davantage prêts à un échange franc sur les forces et faiblesses des divers projets et traditions du logiciel libre. Sachant tout le brouhaha qui s'ensuivrait, je regrette effectivement de ne pas avoir invité Richard ; et je ne souhaite pas que mon manque de l'époque soit interprété comme un manque de respect envers le projet GNU ni envers Richard en personne. »

Au-delà de cette question, O'Reilly et Raymond s'accordent sur le fait que l'expression *open source* remporta l'adhésion d'un nombre tout juste suffisant de participants au congrès pour être qualifiée de succès. Les présents partagèrent leurs idées et expériences, et discutèrent des moyens d'améliorer l'image du logiciel libre. L'une des préoccupations majeures consistait à trouver comment en souligner les succès, tout spécialement dans le domaine de l'infrastructure Internet, plutôt que de jouer la carte de l'opposition entre GNU/Linux et Microsoft Windows. Mais comme ce fut le cas au cours de la réunion chez VA Research, la discussion dériva bientôt sur les problèmes associés à l'expression « *free software* ».

O'Reilly se souvient d'un commentaire de la part de Torvalds, présent au meeting : « Linus venait tout juste d'emménager dans la Silicon Valley et il expliqua comment il n'avait appris que peu

de temps auparavant que le mot *free*' avait deux sens en anglais : libre et gratuit. »

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

Bien que l'expression fût attrayante, l'idée d'un changement de la terminologie officielle était loin de faire l'unanimité. À la fin de la journée de conférence, les trois expressions furent soumises au vote — *free software*, *open source* et *sourceware*. Selon O'Reilly, neuf des quinze présents votèrent pour *open source*. Mais malgré les quelques réserves émises par certains, tous s'accordèrent pour l'employer dans leurs futures déclarations à la presse. « Nous voulions repartir avec un message de solidarité », explique O'Reilly.

L'expression ne mit pas longtemps à intégrer le lexique national. Peu après le sommet, O'Reilly rassembla certains de ses membres à une conférence de presse organisée pour les journalistes du *New York Times*, du *Wall Street Journal* et d'autres journaux d'envergure. Dans les quelques mois qui suivirent, le portrait de Torvalds fit la couverture du magazine *Forbes*, tandis que ceux de Stallman, de Larry Wall et de Brian Behlendorf, chef de l'équipe d'Apache, figuraient en pages intérieures. L'open source s'ouvrait au monde des affaires.

Pour les personnes présentes au sommet, tel Tiemann, le message de solidarité était l'élément le plus important. Bien que son entreprise ait connu un succès non négligeable dans la vente d'outils et de services liés aux logiciels libres, il n'ignorait pas la difficulté que les programmeurs et les entrepreneurs rencontraient.

« Il ne fait aucun doute que l'utilisation du terme 'free' était problématique dans nombre de situations, dit-il. L'open source

s'est positionné comme une solution utile et ouverte aux entreprises, tandis que le logiciel libre se positionnait comme étant moralement juste. Pour le meilleur ou pour le pire, nous avons décidé qu'il était plus avantageux de nous aligner sur le mouvement de l'open source. »

⁂

Après cette rencontre, Raymond appela Stallman pour lui parler de l'expression *open source* et lui demander s'il l'utiliserait. Ce dernier sembla songer un moment à l'adopter, mais finit par la rejeter, d'après son interlocuteur. « Je le sais, car j'avais des conversations directes avec lui à ce propos », précise Raymond.

La réponse immédiate de Stallman fut : « Je vais devoir y réfléchir ». Le jour suivant, il en était arrivé à la conclusion que les valeurs défendues par Raymond et O'Reilly finiraient certainement par dominer le discours de l'open source dans l'avenir, et que pour continuer à faire exister les idées du mouvement pour le logiciel libre dans le débat public, il valait mieux en rester à l'expression traditionnelle.

C'est plus tard au cours de l'année 1998 qu'il annonça sa position : *open source*, bien que commode pour qui voulait mettre en avant les avantages techniques du logiciel libre, encourageait aussi les intervenants à négliger la question des libertés logicielles. Ce problème le forçait à rester fidèle à l'expression « logiciel libre ». Certes, *open source* évitait la confusion avec « logiciel gratuit », mais il supprimait par la même occasion le sens de « logiciel respectueux de la liberté » et n'avait donc plus d'utilité pour exprimer ce dernier sens. Dans les faits, Raymond et O'Reilly avaient donné un nom à celui des deux « partis politiques » de la communauté qui était dénué d'idéaux — celui que Stallman désapprouvait.

En outre, pour Stallman, l'idée d'*open source* conduisait les gens à trop rechercher l'appui des entreprises. Or si ce soutien n'était pas mauvais en soi, le rechercher coûte que coûte pouvait conduire à des compromis nuisibles. « Il est bien connu que lors d'une négociation, si vous cherchez désespérément à passer un marché avec quelqu'un, ce marché sera mauvais, explique-t-il. Vous devez être prêt à dire non. »

Récapitulant sa position à la conférence-exposition LinuxWorld en 1999, événement étiqueté par Torvalds lui-même comme événement de *coming out* de la communauté « Linux », Stallman implora ses confrères de ne pas céder à la facilité apparente de ce genre de compromis. « Parce que nous avons montré ce que nous savons faire, nous ne devons pas être prêts à tout pour travailler avec des entreprises ou compromettre nos objectifs, déclare Stallman lors d'une table ronde. Laissez-les faire leurs propositions, et nous accepterons. Nous ne devons pas changer ce que nous faisons uniquement pour obtenir leur aide. C'est en faisant un pas vers un objectif, puis un autre, et ainsi de suite, qu'on finit par l'atteindre. Ou bien on peut se contenter d'une demi-mesure qui empêchera toute avancée ultérieure, et ne jamais arriver à ses fins. »

***

Cependant, avant même le salon LinuxWorld, Stallman avait fait preuve d'une tendance croissante à s'aliéner les partisans de l'open source. Quelques mois après le « Sommet du Freeware », O'Reilly avait organisé sa deuxième conférence annuelle sur Perl. Cette fois-ci, Stallman était dans le public.

Durant une table ronde glorifiant la décision d'IBM d'utiliser le serveur web libre Apache dans ses offres commerciales, Stallman, s'emparant d'un micro destiné au public, s'attaqua violemment à

John Ousterhout, l'un des participants, et créateur du langage de script Tcl[9]. Il le qualifia de « parasite » de la communauté du logiciel libre pour avoir commercialisé une version non libre de Tcl via sa propre start-up, Scriptics. Ousterhout avait en effet annoncé que sa société ne contribuerait que le strict minimum nécessaire à l'amélioration de la version libre de Tcl. Juste de quoi, finalement, gagner l'approbation générale pour légitimer ensuite le développement d'une grande quantité de logiciels non libres. Stallman refusait cette position et dénonça les plans de l'entreprise : « Je ne pense pas que Scriptics soit indispendable à la suite de l'existence de Tcl », déclara-t-il, sous les huées des autres membres du public. [10]

« C'était une scène des plus déplaisantes, se souvient Rich Morin, de chez Prime Time Freeware. John a réalisé des choses plutôt respectables : Tcl, Tk, Sprite. C'est un vrai contributeur. » Malgré sa sympathie pour Stallman et ses idées, Morin comprenait ceux que l'agressivité de ses paroles avaient dérangé.

Stallman quant à lui ne regrette rien : « Ce n'est pas critiquer les logiciels non libres qui est critiquable, mais bien les logiciels non libres eux-mêmes. Ousterhout avait en effet réalisé de belles contributions par le passé, mais le problème était que Scriptics devenait quasiment 100 % un éditeur commercial propriétaire. Durant cette conférence, défendre la liberté revenait à se mettre à dos presque tout le monde. Comme je prenais la parole depuis le public, j'étais limité à quelques phrases. La seule manière de soulever le problème sans qu'il ne soit aussitôt oublié, c'était d'employer des termes forts. »

« Si on me reproche de 'faire une scène' lorsque je critique sérieusement la conduite de quelqu'un, alors que Torvalds n'est

9. *Tool Command Language* (abrégé en Tcl).
10. Maclachlan, *op. cit.*

qualifié que 'd'impertinent' lorsqu'il balance des choses encore plus désagréables sur des questions sans intérêt, ajoute-t-il, il me semble qu'il y a alors deux poids, deux mesures. »

Cette critique controversée fit fuir momentanément un autre sympathisant potentiel, Bruce Perens. En 1998, Eric Raymond proposa le lancement de l'*Open Source Initiative*, dite OSI, une organisation qui régulerait l'utilisation de l'expression *open source* et en fournirait une définition aux entreprises souhaitant développer leurs propres programmes. Raymond engagea Perens pour en rédiger les termes [11].

Perens allait par la suite démissionner de l'OSI, regrettant que l'organisation se soit mise en place en opposition à Stallman et à la FSF. Il reste que, considérant rétrospectivement la nécessité d'une définition du logiciel libre hors du giron de la FSF, Perens comprend pourquoi d'autres hackers aient pu ressentir le besoin de prendre leurs distances. « J'aime et j'admire vraiment Richard, dit Perens. Mais je pense qu'il réussirait mieux s'il était plus mesuré. Et cela passerait notamment par une retraite de quelques mois hors du monde du logiciel libre. »

***

Toute l'énergie de Stallman ne pouvait contrer la dynamique des partisans de l'open source en matière de relations publiques. En août 1998, lorsque le fabriquant de processeurs Intel acquit des parts de Red Hat, la société qui commercialisait GNU/Linux, un article du *New York Times* décrivit l'entreprise comme un produit du mouvement « connu alternativement comme logiciel libre et open source » [12]. Six mois plus tard, un article de John Markoff

11. Perens, 1999
12. Harmon, 1998

sur *Apple Computer* annonçait dans son titre l'adoption du serveur « open source » Apache par l'entreprise. [13]

Ce dynamisme allait coïncider avec l'importance croissante des entreprises ayant délibérément choisi l'expression *open source*. En août 1999, Red Hat, qui se prévalait volontiers de cette terminologie, vendait ses actions au Nasdaq. En décembre, VA Linux — anciennement VA Research - voyait sa cotation en bourse battre des sommets historiques. Ouvrant à 30 dollars l'action, la valeur dépassa vite la barre des 300 dollars avant de se stabiliser au niveau des 239 dollars. Les actionnaires ayant eu la chance de s'être engagés au prix de départ et qui étaient restés jusqu'au bout, avaient vu la valeur de leurs titres augmenter de 698 %, un record pour le Nasdaq. Eric Raymond, membre du conseil d'administration, possédait ainsi des parts à valeur de 36 millions de dollars. Cette forte croissance en bourse fut toutefois temporaire, et chuta avec l'explosion de la bulle Internet.

Le message des promoteurs de l'open source était simple : tout ce dont vous avez besoin pour vendre le concept de logiciel libre, c'est de le rendre intéressant pour les entreprises. De leur point de vue, Stallman et le mouvement pour le logiciel libre étaient en lutte contre le marché, tandis qu'eux cherchaient à l'exploiter. Plutôt que de jouer les cancres bannis du lycée, ils préféraient jouer la carte de la popularité, accroissant leur pouvoir par la même occasion.

Si ces méthodes servaient la cause de l'open source, les idéaux du logiciel libre étaient perdants. Tout ce qui avait été fait pour « répandre le bon tuyau » avait occulté la composante du message de départ la plus importante : la liberté comme enjeu éthique. Les effets de cette omission se font sentir aujourd'hui encore : en 2009, presque toutes les distributions GNU/Linux contiennent des

13. Markoff, 1999

logiciels non libres, la version de Linux maintenue par Torvalds contient des firmwares non libres et la compagnie anciennement nommée VA Linux base ses activités sur du logiciel non libre. Plus de la moitié des serveurs web dans le monde tournent avec une version d'Apache, qui est à la base un logiciel libre. Mais combien de ces sites ont-ils recours à une version modifiée non libre distribuée par IBM ?

« Dans ses mauvais jours, Richard croit que Linus Torvalds et moi avons conspiré pour détourner sa révolution, dit Raymond. Son rejet de l'expression *open source* et sa création délibérée d'un schisme idéologique provient selon moi d'un étrange mélange d'idéalisme et de volonté de défendre son territoire. Certains mettent tout cela sur le compte de son ego. Je ne suis pas de cet avis. La raison en est plutôt qu'il s'associe si personnellement à l'idée du logiciel libre qu'il prend toute menace envers cette idée comme une menace personnelle. »

Stallman oppose : « Raymond dénature mon point de vue. Je ne pense pas que Torvalds ait 'conspiré' avec qui que ce soit, pour la bonne raison que ce n'est pas quelqu'un de sournois. Quant à Raymond, son comportement mauvais est flagrant dans ses propos mêmes. Au lieu de répondre à mes arguments sur le fond (et ce, même sous la forme distordue qu'il leur donnait), il en propose une interprétation psychologique. Il attribue à d'autres — sans les nommer — l'interprétation la plus négative de mes actes, pour ensuite prendre ma 'défense' en opposant une autre interprétation à peine moins péjorative. Il m'a souvent 'défendu' de cette manière. »

***

Ironie du sort, le succès de l'open source et de ses supporters n'amoindrit pas pour autant le rôle de leader tenu par Stallman,

quoique cela créa une certaine confusion sur *ce dont* il était censé être le leader. Le mouvement pour le logiciel libre n'ayant pas eu la reconnaissance accordée par les entreprises et les médias à l'open source, la plupart des utilisateurs de GNU/Linux n'en ont tout simplement jamais entendu parler, et encore moins de la vision qu'il propose. Ils connaissent les idées et valeurs de l'open source sans même pouvoir imaginer que Stallman en propose d'autres.

Ainsi, aujourd'hui encore, Stallman reçoit régulièrement des messages de félicitations pour sa défense de l'open source, ce qui lui permet, en retour, d'expliquer qu'il n'en a jamais été partisan, tout en saisissant l'occasion d'informer l'expéditeur sur ce qu'est le logiciel libre.

Certains auteurs reconnaissent l'expression « logiciel libre » dans l'emploi de l'acronyme FLOSS, *Free/Libre Open Source Software*, qui signifie à peu près « logiciel libre et open source ». Toutefois, ils ne reconnaissent souvent qu'un seul mouvement, le « FLOSS », un peu comme s'ils prétendaient qu'il existe aux États-Unis un mouvement « *liberal* - conservateur » [14] Quant à leur perception de ce mouvement supposé unique, elle relève la plupart du temps des seuls arguments de l'open source.

En dépit de ces obstacles, le mouvement pour le logiciel libre parvient parfois à faire entendre sa voix, et continue de croître en termes de chiffres absolus. En ne déviant pas de sa position et en présentant ses idées en opposition à celles de l'open source, il gagne du terrain. « Un des traits de caractères principaux de Stallman est son entêtement, dit Ian Murdock. S'il faut attendre dix ans pour que les gens se rallient à son point de vue, il attendra. »

14. Note de Richard Stallman — Aux États-Unis, le mouvement *liberal* promeut la sécurité sociale, la réglementation de l'économie au bénéfice de l'intérêt général, la jouissance des droits de l'homme pour tous et des relations internationales paisibles.

Murdock, quant à lui, trouve à la fois rafraîchissante et précieuse cette nature inébranlable. Stallman n'est peut être plus le seul et unique leader du mouvement pour le logiciel libre, mais il n'en reste pas moins le guide de la communauté. « On peut être sûr qu'il sera cohérent dans ses idées, dit Murdock. Ce n'est pas le cas de tout le monde, loin de là. Qu'on soit ou non d'accord avec lui, cela est digne du plus grand respect. »

Chapitre

12

Une brève incursion dans l'enfer hacker

À travers le pare-brise de notre voiture de location, Richard Stallman fixe le feu rouge, attendant sans ciller qu'il passe au vert, alors que nous nous frayons un chemin 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.

Deux heures se sont écoulées depuis le discours de Stallman au Maui High Performance Center. Kihei, une ville d'apparence si accueillante avant cette conférence, semble à présent profondément repoussante. Comme la plupart des cités balnéaires, elle

est le fruit d'un exercice unidimensionnel d'étalement périurbain. En descendant la rue principale, bordée d'une infinie succession d'échoppes de hamburgers, d'agences immobilières et de magasins de bikinis, difficile de ne pas se sentir comme un morceau d'acier transitant dans le tube digestif d'un immense ver solitaire publicitaire. Un sentiment exacerbé par le manque de rues transversales. Avec comme unique solution d'aller tout droit, le trafic avance par à-coups. Quelque deux-cents mètres devant, un feu passe au vert. À peine commençons-nous à avancer qu'il est de nouveau orange.

Pour Stallman, résident permanent de la côte Est, la seule idée qu'il va passer à Hawaï la plus grande partie de cette journée ensoleillée coincé dans les embouteillages, suffit à déclencher une embolie.

Comme je conduisais, je prenais du retard sur mes mails, chose insupportable pour moi qui ai déjà du mal à tenir le rythme.

D'autant plus qu'avec quelques rapides bifurcations à droite moins d'un kilomètre auparavant, cette situation aurait pu être évitée facilement. Malheureusement, nous sommes dépendants du conducteur qui nous précède, un programmeur du laboratoire connaissant le chemin et qui a décidé de nous mener à Pa'ia via la route pittoresque, plutôt que par l'autoroute de Pilani toute proche.

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

À nouveau, le feu passe au vert, et à nouveau, nous avançons péniblement de quelques longueurs de voiture. Le procédé 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, levant les bras, frustré. Non mais tu le crois, ça ? »

Je m'abstiens de répondre. Je trouve que le fait même d'être assis sur 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 qu'il avait son permis de conduire. Maintenant, écoutant le violoncelle de Yo-Yo Ma jouant les sombres notes de basse de *Appalachian Journey* à la radio, tout en admirant le coucher de soleil sur notre gauche, je m'efforce de me faire tout petit.

Dès qu'une autre occasion de bifurquer se présente, Stallman met son clignotant, 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. Mon interlocuteur est à présent livide.

« C'est comme s'il nous ignorait délibérément », râle-t-il, gesticulant et pantomimant comme un sémaphore sur le pont d'un porte-avions, dans une vaine tentative d'attirer l'attention 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 jette un œil à travers 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. Du genre à rendre ce type de situation un peu plus acceptable si vous êtes un Hawaïen natif, je suppose. J'essaye de montrer le phénomène à Stallman. Il n'en fait que peu de cas, désormais obsédé par l'inattention du conducteur qui nous précède.

Quand ce dernier passe un nouveau feu vert, ignorant complètement le panneau « Autoroute Pilani, prochaine à droite », je

grince des dents. Je me souviens de l'avertissement du programmeur BSD Keith Bostic. « Stallman ne supporte pas bien les idiots, m'a-t-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 désagrément, qui rend Stallman furieux.

« C'est comme s'il avait choisi cette route sans avoir du tout réfléchi à la manière la plus efficace de se rendre là-bas », lance-t-il.

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 résultant du fait d'avoir à vérifier l'imprimante Xerox deux à trois fois par jour qui a déclenché l'enquête initiale de Stallman au sujet du code source du pilote. C'était l'inefficacité provoquée par l'obligation d'avoir à récrire des outils logiciels kidnappés par les vendeurs de logiciels commerciaux qui l'a amené à se battre contre Symbolics et à lancer le Projet GNU. Si, comme le pensait Jean-Paul Sartre, l'enfer c'est les autres, l'enfer hacker, c'est la réitération des erreurs bêtes des autres, et on peut dire sans exagérer que la vie entière de Stallman a été une tentative visant à protéger l'humanité de ce travers dantesque.

La métaphore de l'enfer se fait plus évidente alors que nous faisons route lentement dans ce paysage. Avec sa multitude de boutiques, de parkings, et de lampadaires mal réglés, Kihei ressemble moins à une ville qu'à un gros logiciel mal conçu. Au lieu de diriger le trafic et de distribuer les véhicules le long des rues adjacentes et des voies rapides, les urbanistes 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 exaspèrent 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 sont la cause de fichus délais tellement 'non nécessaires' [l'emphase vient de Levy] qu'ils résistent difficilement à l'envie de réarranger les panneaux, d'ouvrir les boîtiers des feux rouges… de repenser tout le système. » [1]

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

Avant que je puisse faire part de cette réflexion à Stallman, le conducteur met finalement son clignotant à droite. Les épaules crispées de mon accompagnateur se relaxent un peu, et durant un instant, la tension dans la voiture se dissipe. Elle revient cependant au galop quand le véhicule devant nous se met à ralentir. Des panneaux « Travaux » bordent la rue, et bien que l'autoroute de Pilani ne soit qu'à quelques centaines de mètres, la 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 qui 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 en 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 ! »

1. Levy, 1984, p.40

Le conducteur finit sa manœuvre et nous croise en sens inverse, retournant vers l'artère principale. Ce faisant, il hausse les épaules comme pour s'excuser. Ajoutée à un sourire toutes dents dehors, sa gestuelle 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 semblait nous parvenir un message succinct : « Hé, c'est Maui… Qu'est-ce qu'on peut y faire ? »

Stallman n'en peut plus. « 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 ! »

La transcription de ce monologue n'est pas exacte semble-t-il, car je n'utilise pas le mot « putain » utilisé ainsi comme adverbe (« It's your fucking fault »*). Comme il ne s'agissait pas d'une interview, Williams n'enregistrait donc pas. Je suis sûr que les choses se sont passées comme ça dans l'ensemble, mais ce qu'il cite de moi reflète plus ce qu'il en a interprété que mes paroles réelles.*

Stallman insiste sur les mots « à ma façon » en agrippant le volant et en se ramenant vers lui à deux reprises. L'image qu'il donne alors est celle d'un enfant piquant une crise dans son siège auto. Une image soutenue par le ton de sa voix. À mi-chemin entre colère et angoisse, il semble au bord des larmes.

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

Cinq minutes plus tard, nous atteignons le croisement suivant. En quelques secondes, bénéficiant cette fois d'un accès facile à l'autoroute, nous fonçons vers Pa'ia à une vitesse réconfortante. Le soleil qui irradiait tout à l'heure, couvrant d'une lueur dorée l'épaule gauche de Stallman, brûle maintenant d'un beau rouge orangé dans notre rétroviseur. Il colore les arbres que nous croisons des deux côtés de l'autoroute.

Durant les vingt minutes suivantes, le ronronnement des pneus et du moteur, ainsi que le trio de violons et de violoncelles jouant les sombres notes de basse de Appalachian Journey sur le lecteur CD de la voiture, sont les seuls sons à résonner dans l'habitacle.

Chapitre

13

Le combat vers la liberté

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

Quatre ans après *La cathédrale et le bazar*, le père du logiciel libre 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. À ce propos, il se souvient d'un tee-shirt populaire apparu pour la première fois aux conventions Linux aux alentours de 1999. Parodie de l'affiche originale de *Star Wars*, le vêtement montre Torvalds brandissant un sabre laser à la façon de Luke Skywalker, alors que la tête de Stallman se trouve au sommet de R2D2. Ce tee-shirt lui tape toujours sur les nerfs, non seulement parce qu'il le représente comme un sous-fifre de Torvalds, mais aussi parce qu'il élève ce

dernier au rang de leader dans la communauté du logiciel libre, un rôle que l'intéressé lui-même refuse d'endosser. « C'est ironique, commente Stallman désabusé. 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, alors ? »

Mais en contrepartie, c'est le refus du Finlandais de « porter le sabre » qui a permis à Stallman de conserver sa réputation d'arbitre éthique de la communauté hacker. Malgré ses griefs, il 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 triomphe de GNU/Linux — un triomphe teinté d'ironie si l'on considère le nombre d'utilisateurs qui le nomment simplement « Linux », Stallman s'est néanmoins réapproprié cette initiative avec succès. Son agenda d'orateur entre janvier 2000 et décembre 2001 comprenait des interventions sur les six continents et des visites dans des pays où la notion de liberté logicielle comporte de lourds sous-entendus : la Chine et l'Inde, par exemple.

En dehors de la tribune de leader, Stallman a aussi profité de l'influence de la GNU GPL, dont il reste le gardien. Durant l'été 2000, alors qu'éclatait la bulle créée par l'offre publique d'achat de VA Linux en 1999, Stallman et la FSF remportèrent deux victoires majeures. En juillet, Trolltech, une entreprise norvégienne développant Qt, une bibliothèque logicielle de composants graphiques pour GNU/Linux, annonça qu'elle allait enregistrer ses logiciels sous licence GPL. Quelques semaines plus tard, Sun Microsystems, qui jusque là avait tenté maladroitement de suivre le mouvement open source sans céder son code à la communauté, se laissa finalement convaincre et annonça la publication de la nouvelle suite OpenOffice[1] sous double licence : la *Lesser GNU*

1. Sun a été contraint, suite à un problème de marque déposée, d'utiliser le nom plus maladroit *OpenOffice.org*.

Public Licence (LGPL, Licence publique générale limitée GNU) et la *Sun Industry Standards Source Licence* (SISSL, Licence source des standards industriels de Sun).

Dans le cas de Trolltech, cette victoire fut le résultat d'un long effort de la part du projet GNU. En effet, le statut non libre de Qt posait un grave problème à la communauté puisque KDE, un environnement de bureau graphique de plus en plus populaire, dépendait beaucoup de cette bibliothèque logicielle. Qt n'était pas libre mais Trolltech avait invité des projets libres tels que KDE à l'utiliser gratuitement. Bien que KDE soit un logiciel libre, il ne pouvait s'exécuter sans Qt, qui n'était pas libre. KDE ne pouvait donc être utilisé par ceux qui voulaient utiliser exclusivement du logiciel libre.

Stallman reconnut que de nombreux utilisateurs allaient vouloir un bureau graphique sous GNU/Linux, et que la plupart d'entre eux ne chérissaient pas assez la liberté pour résister aux sirènes de KDE. Dès lors, GNU/Linux risquait de susciter l'installation massive KDE, et donc de la bibliothèque non libre Qt, compromettant ainsi l'objectif du projet GNU.

Afin de remédier à cette situation, Stallman recruta des gens pour lancer deux contre-projets. Le premier était Gnome, un environnement graphique libre pour le bureau GNU, l'autre, Harmony, une bibliothèque graphique libre qui devait remplacer Qt. Si Gnome réussissait, KDE deviendrait superflu. Si Harmony réussissait, KDE n'aurait plus besoin de Qt. Dans les deux cas, les utilisateurs pourraient utiliser un environnement graphique sous GNU/Linux sans passer par un Qt non libre.

En 1999, les deux projets étaient bien avancés, et Trolltech commençait à sentir la pression monter. Trolltech commença tout d'abord par écrire sa propre licence, la QPL (*Q Public Licence*). Bien qu'elle fût reconnue comme licence libre, Stallman souligna

l'inconvénient de son incompatibilité avec la GPL : combiner du code sous licence GPL avec Qt dans un même programme revenait systématiquement à violer l'une ou l'autre licence. La direction de Trolltech finit par reconnaître que la GPL servait aussi bien ses intérêts. Elle publia alors le code source de Qt sous double licence, QPL et GPL. Une victoire qui couronnait trois ans d'efforts pour le projet GNU.

Une fois Qt sous licence libre, il n'y avait plus de raison de poursuivre le développement de Harmony, alors pas encore assez abouti pour une utilisation réelle. A contrario, Gnome acquit une dynamique propre, de sorte que son développement se poursuivit, jusqu'à incarner l'environnement graphique GNU de référence.

Sun était pour sa part disposé à respecter les conditions de la FSF. À la conférence Open Source-O'Reilly de 1999, le cofondateur et directeur scientifique de Sun Microsystems, Bill Joy, se fit l'avocat de la licence *Community Source* créée par son entreprise. Il s'agissait essentiellement d'un compromis permettant aux utilisateurs de copier et modifier les logiciels dont Sun était propriétaire, sans toutefois pouvoir en vendre des copies tant qu'un accord de royalties avec l'entreprise mère n'était pas négocié (de ce point de vue, la licence n'était alors ni libre ni open source).

Un an après le discours de Joy, le vice-président de Sun Microsystems, Marco Boerries, se trouvait sur la même scène, à détailler le nouveau compromis de licence concernant OpenOffice, une suite d'applications bureautiques conçue spécialement pour la distribution logicielle GNU/Linux.

« Je peux l'épeler en trois lettres, dit Boerries. G.P.L. »

À l'époque, il déclara que la décision de Sun avait moins à voir avec Stallman qu'avec le dynamisme des programmes protégés par la GPL. « Ce qu'il s'est passé, fondamentalement, c'est la prise de

conscience que différents produits attirent différentes communautés, et que la licence utilisée dépend du type de communauté que l'on souhaite attirer. Pour OpenOffice, il était évident que nous avions un lien plus étroit avec la communauté GPL. » [2] Mais ce ne fut qu'une demi-victoire pour le projet GNU, car OpenOffice recommande l'utilisation d'extensions non libres.

***

Ces commentaires soulignent la puissance trop souvent sous-estimée de la GPL et, indirectement, le génie politique de l'homme qui a joué le rôle principal dans sa création. « Il n'est pas un avocat sur terre qui aurait rédigé la GPL telle quelle, dit Eben Moglen, professeur de droit à la Columbia University, et conseiller général de la FSF. Mais elle fonctionne. Et elle fonctionne grâce aux principes conçus par Richard. »

Ancien programmeur professionnel, Moglen dit avoir commencé son travail bénévole avec Stallman à partir de 1990, lorsque ce dernier lui demanda son aide juridique pour une affaire privée. Moglen, qui travaillait à l'époque avec l'expert en cryptographie Phillip Zimmerman durant ses batailles juridiques contre le gouvernement fédéral [3], fut honoré par cette demande. « Je lui ai dit que j'utilisais Emacs chaque jour de ma vie, et qu'il me faudrait fournir énormément de conseil juridique pour rembourser cette dette. »

2. Marco Boerries, entretien avec l'auteur (juillet 2000).

3. Pour plus d'informations sur les travaux légaux de Zimmerman, lisez Levy, 2002, p. 287-288. Dans la version originale du livre que vous avez en main, je rapportais que Moglen aidait Zimmerman dans sa lutte 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.

Depuis lors, Moglen, peut-être plus que quiconque, a été à même d'observer la transposition des principes philosophiques hackers de Stallman dans la sphère juridique. Selon lui, Stallman appréhende les codes juridique et logiciel d'une manière largement similaire.

« En tant qu'avocat, je dois avouer que l'idée qui voudrait qu'on supprime tous les bogues d'un texte de loi n'a pas vraiment de sens, dit Moglen. Il y a une certaine dose d'incertitude dans tout processus juridique, et les avocats cherchent à l'exploiter au bénéfice 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 rédiger une licence qui tienne compte de toutes les circonstances dans tous les systèmes juridiques à travers le monde entier. Mais si jamais vous essayez de le faire, il faudrait le faire à la façon de Richard. Et l'élégance, la simplicité dans la conception de départ qui en résultent atteignent au mieux l'objectif fixé. À partir de là, avec un peu de travail de juriste, vous pouvez aller loin. »

Chargé de promouvoir les objectifs de Stallman, Moglen comprend la frustration de ses alliés potentiels. « Richard est un homme qui ne veut faire aucune concession sur les questions qu'il considère comme fondamentales, dit-il, et il accepte difficilement qu'on joue sur les mots ou même qu'on cherche le flou artistique — ce qu'on est bien souvent amené à faire dans la société. »

En plus d'aider la FSF, Moglen dispensa un soutien juridique à ceux qui étaient attaqués pour violation de copyright, tels Dmitri Sklyarov ou les distributeurs du programme de décryptage de DVD appelé DeCSS.

Employé par une société russe, Sklyarov avait écrit et publié un programme permettant de briser le verrouillage anti-copie des livres électroniques d'Adobe. En Russie, aucune loi ne l'interdisait.

Il fut cependant arrêté lors d'une visite aux États-Unis alors qu'il donnait une conférence scientifique au sujet de son travail. Stallman participa avec enthousiasme aux manifestations à l'encontre d'Adobe, qui avait joué un rôle dans cette arrestation.

De son côté, la FSF dénonça le *Digital Millenium Copyright Act* (DMCA) [4] comme un outil de « censure des logiciels », mais ne put intervenir en faveur de Sklyarov, car son logiciel n'était pas libre. C'est donc à travers la *Electronic Frontier Foundation* (EFF) [5] que Moglen put assurer indirectement la défense du programmeur russe.

La FSF évita de participer à la diffusion de DeCSS, celle-ci étant illégale, mais Stallman condamna tout de même la volonté du gouvernement américain de l'interdire. Moglen travailla alors directement comme conseiller juridique auprès des hackers qui diffusaient ce programme.

Si la FSF ne voulait pas s'engager sur de telles questions hors du champ du développement de GNU et du renforcement de la GPL, Moglen apprit néanmoins 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 la prochaine action. Voilà ce qu'il faut faire'. Et sa réponse a toujours été : 'Nous n'avons pas à faire quoi que ce soit. Il suffit d'attendre. Ce qui doit être fait le sera.' »

4. Il s'agit de la loi sur laquelle reposait le chef d'accusation de Sklyarov. La DMCA fut votée aux États-Unis en 1998 afin de couvrir les droits d'auteur à l'ère du numérique. Parmi ses dispositions, on trouve l'interdiction de contourner toute technologie de menotte digitale utilisée pour restreindre l'utilisateur des œuvres publiées. La Communauté Européenne a d'ailleurs emboîté le pas pour étendre ces restrictions à tous les états membres.

5. L'EFF est une organisation à but non lucratif, fondée en 1990 aux États-Unis dans le but de défendre la liberté d'expression sur Internet — NdT.

« Et vous savez quoi ? ajoute Moglen. En général, il avait raison. »

Ces mots contredisent ce que Stallman dit de lui-même : « Je ne suis pas doué pour les petits jeux, confesse-t-il, répondant aux nombreux détracteurs anonymes qui le voient comme un habile stratège. Je ne suis pas doué pour prévoir et anticiper ce que les autres vont faire. Ma démarche a toujours été de me concentrer sur les idées fondamentales du mouvement et de dire 'Rendons-les aussi fortes que nous le pouvons'. »

***

La popularité grandissante et l'indéfectible force d'attraction de la GPL constituent le meilleur hommage au succès de la fondation créée par Stallman et ses collègues du projet GNU. Bien qu'il n'ait jamais été le seul au monde à publier des logiciels libres, il peut cependant se prévaloir d'avoir tissé le canevas éthique du mouvement. Que les programmeurs actuels se sentent ou non à l'aise au sein de cette structure n'est pas la question. Le simple fait qu'ils aient le choix est le plus grand leg du personnage.

Cela dit, parler de leg semble un peu prématuré pour l'instant. Stallman, âgé de quarante-huit ans au moment de l'écriture de la première édition [6], a encore quelques années devant lui pour ajouter ou soustraire à ce qu'il transmettra. Reste que l'élan donné au mouvement pour le logiciel libre encourage à examiner sa vie indépendamment de ses batailles quotidiennes contre l'industrie du logiciel, et davantage du point de vue général et historique.

À son crédit, Stallman refuse toute spéculation sur ce point. « Je n'ai jamais réussi à imaginer en détail ce que serait le futur, indique-t-il, proposant prématurément sa propre épitaphe.

6. En 2002.

J'ai juste dit : 'Je vais me battre. Qui sait dans quelle mesure je réussirai ?' »

Il va sans dire que par le choix de ses causes, il a fait fuir ceux qui, en d'autres occasions, auraient pu être ses plus grands défenseurs, eût-il accepté de se battre pour leurs idéaux plutôt que les siens. Cela prouve aussi sa nature franche et vertueuse, qui amène invariablement ses anciens ennemis politiques à finalement glisser quelque louange à son encontre. Le biographe est néanmoins contraint de se demander comment, entre l'idéologue et le hacker de génie, Stallman sera considéré lorsque sa propre personnalité ne sera plus là pour jouer les éclaireurs…

Dans les versions préliminaires de ce livre, j'ai intitulé ce point 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 tenter 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 contemporain, il est aisé de voir des points communs entre Stallman et ces Américains du passé qui, bien que marginaux au cours de leur carrière, ont acquis une dimension historique en vieillissant. On trouve facilement une similitude avec Henry David Thoreau, philosophe transcendentaliste et auteur de *La désobéissance civile*, ou John Muir, fondateur du Sierra Club, père du mouvement environnementaliste moderne [7]. Mais on peut aussi penser à William Jennings Bryan, dit « le Grand Roturier »,

7. Henry David Thoreau (1817-1862) est l'auteur de *Walden ou la vie dans les bois*, classique de la littérature américaine, tiré de l'expérience de deux années passées dans une cabane au fond des bois. Dans cette critique en règle de l'humanité moderne, Thoreau ne livre pas un plaidoyer contre le progrès technique, mais contre les défauts d'une société privilégiant le bien-être matériel au spirituel. John Muir (1838-1914) est à ranger du côté des contemplatifs. Ses récits de voyages dans les contrées reculées des États-Unis et son activisme en faveur, notamment, de la sauvegarde de la vallée du Yosemite ont directe-

leader du mouvement populiste, ennemi des monopoles, et qui, bien qu'il fût un homme puissant, semble être tombé dans l'oubli [8].

S'il n'a pas été le premier à considérer le logiciel comme un bien public, Stallman aura assurément une place dans les livres d'histoire grâce à la GPL. On peut ensuite prendre du recul et se détacher du contexte actuel : la GPL sera-t-elle toujours utilisée par les programmeurs en 2102, ou sera-t-elle reléguée aux oubliettes depuis longtemps ? L'expression *free software* sera-t-elle demain obsolète, comme l'est aujourd'hui l'expression *free silver* [9], ou sera-t-elle considérée comme étonnamment visionnaire à la lumière des événements politiques qui auront eu lieu ?

ment inspiré la politique de Théodore Roosevelt pour la création de plusieurs parcs naturels sur le modèle de Yellowstone (1872). Le Sierra Club qu'il fonda en 1892 est (encore aujourd'hui) une association lobbyiste à forte influence – NdT.

8. William Jennings Bryan (1860-1925) fut un avocat presbytérien et démocrate qui eut une forte influence dans la vie politique américaine du début du xx^{e} siècle. Leader du parti démocrate, il se réclamait d'un mouvement populiste résumant à lui seul l'ensemble des valeurs incarnées par « l'Américain moyen », réactionnaire (pour la Prohibition), croyant (contre le darwinisme) et volontaire (la crise financière de 1893 n'est pas loin). En permettant notamment à Woodrow Wilson (dont il fut le secrétaire d'État) d'accéder à la présidence, W. J. Bryan a porté pour la première fois sur la scène politique l'archétype américain en prise avec l'avènement du capitalisme industriel, celui qui mena pourtant l'Amérique jusqu'à la crise de 1929 — NdT.

9. La libre frappe de l'argent fut une revendication du mouvement populiste américain, essentiellement rural, apparue vers la fin du XIX^{e} siècle. Cette revendication, visant à insuffler davantage de liquidités sur le marché, s'oppose au choix de l'étalon or comme unique monnaie de référence : la spéculation sur la rareté de l'or provoquant une hausse de l'endettement des agriculteurs, ces derniers réclament donc un étalon argent qui circulerait plus rapidement pour accroître l'accès au crédit. William Jenning Bryan (voir plus haut) fut l'un des porte-parole populistes qui soutenaient cet argument. La croisade *Free Silver* traversa les crises successives jusqu'à la Conférence de Bretton Woods qui changea le système après la Seconde Guerre mondiale : chaque monnaie se vit attribuer une parité fixe par rapport à l'or, ce qui rendit le mouvement obsolète — NdT.

Prédire l'avenir est une entreprise hasardeuse. Stallman s'y refuse d'ailleurs, dans la mesure où se demander ce que les gens penseront dans cent ans laisse penser que nous n'avons pas d'influence sur le futur. Aussi préfère-t-il se poser la question : « Que devons-nous faire pour un avenir meilleur ? » Pourtant, lorsqu'on leur pose la question, nombreux sont ceux qui se risquent à des prévisions.

« Dans cent ans, Richard et quelques autres mériteront davantage qu'une note dans les livres d'histoire, dit Moglen. Ils seront considérés comme les personnages principaux du récit. »

Les « quelques autres » cités par Moglen incluent John Gilmore, grand contributeur à la cause du logiciel libre et père de l'Electronic Frontier Foundation, et Theodor Holm Nelson, dit Ted Nelson, auteur en 1982 du livre *Literary Machines*. Selon Moglen, les trois personnages — Stallman, Nelson et Gilmore — se singularisent historiquement, bien que de manière différente. Il accorde à Nelson, communément reconnu pour avoir proposé le terme *hypertexte*, le mérite d'avoir identifié le problème épineux de la propriété de l'information à l'âge numérique. Gilmore et Stallman, quant à eux, ont eu le grand mérite d'identifier les effets politiques néfastes du contrôle de l'information et d'avoir mis sur pied des organisations pour lutter contre ses effets : l'Electronic Frontier Fondation dans le cas de Gilmore et la FSF dans le cas de Stallman. Des deux, cependant, Moglen voit Stallman comme celui qui mène les actions les plus personnelles et les moins politiques par nature.

« Le cas de Richard est unique, car il a vu très tôt les implications éthiques des logiciels non libres, dit Moglen. C'est lié à sa personnalité. De nombreuses personnes, lorsqu'elles écriront à son sujet, tenteront de décrire cette personnalité comme un épiphénomène, voire comme un handicap dans l'œuvre de sa vie. »

Gilmore, qui considère sa position entre l'erratique Nelson et l'irascible Stallman comme un honneur « mitigé », approuve néanmoins l'argument de Moglen. Il écrit : « Mon intuition est que les

écrits de Stallman résistent autant que ceux de Thomas Jefferson ; c'est un auteur très clair, y compris dans ses principes… L'influence de Richard égalera celle de Jefferson selon que, dans un siècle, le concept de 'droits civiques' primera ou non sur celui de 'logiciel' ou de 'restrictions techniques'. »

Autre élément de l'héritage de Stallman à ne pas sous-estimer, écrit Gilmore, le modèle collaboratif de développement logiciel initié par le projet GNU. Bien qu'il ait révélé parfois des imperfections, ce modèle n'en est pas moins devenu un standard au sein de l'industrie du développement logiciel. En fin de compte, selon lui, le modèle de développement collaboratif pourrait devenir plus important que le projet GNU, que la licence GPL ou qu'aucun des logiciels développés par Stallman.

« Avant l'Internet, dit encore Gilmore, 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 au 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 de courriel, 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. »

Pour Stallman, ce jugement, bien que positif, néglige l'essentiel : « Il situe les méthodes de développement avant la question de la liberté, ce qui reflète davantage le mouvement open source que celui du logiciel libre. Si les futurs utilisateurs ne voient le projet GNU que par ce biais, j'ai bien peur que cela conduise à un monde où les développeurs tiendront les utilisateurs enchaînés,

les laissant parfois apporter leurs contributions, mais sans jamais relâcher leurs entraves. »

Lawrence Lessig, professeur de droit à Stanford et auteur en 2001 du livre *The Future of Ideas*, pense de même. À l'instar de nombreux universitaires juristes, Lessig considère la GPL comme l'un des plus importants remparts au service des « biens communs numériques », comme on les appelle aujourd'hui, c'est-à-dire la vaste accumulation de logiciels, de standards réseau et de télécommunication, qui ont déclenché sous contrôle communautaire 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 Joseph Carl R. Licklider, qui amenèrent une vision plus large de la technologie informatique, Lessig voit dans l'impact de Stallman quelque chose de personnel, d'introspectif et, enfin, d'unique.

« [Stallman] a fait évoluer le débat du constat à l'action. Il a fait comprendre l'importance de l'enjeu et a construit tout un appareil pour diffuser ces idéaux… Cela dit, je ne saurais le situer par rapport à Cerf ou Licklider. Ce n'est pas le même type d'innovation. Il ne s'agit pas seulement d'un certain type de code, ou d'avoir permis l'avènement de l'Internet. Il s'agit plus encore d'avoir amené les gens à chérir une certaine conception de l'Internet. Je ne pense pas qu'il existe quelqu'un d'autre dans sa catégorie, que ce soit dans le passé ou dans le futur. »

Tout le monde ne considère pas cet héritage comme acquis, bien sûr. Eric Raymond, qui proposa l'open source et pour qui le rôle de leader de Stallman a diminué de façon significative depuis 1996, voit des signes mitigés pour 2102 : « Je pense que les artéfacts de Stallman (GPL, Emacs, GCC) seront perçus comme des travaux révolutionnaires, comme des fondements de la technologie de l'information. Je pense que l'histoire sera moins tendre avec certaines des théories à partir desquelles RMS a opéré, et

sans aucune complaisance à l'égard de sa tendance personnelle à la territorialité et au culte de la personnalité. »

Les prévisions de Stallman lui-même ne sont pas particulièrement optimistes. « Ce que l'histoire dira du projet GNU, d'ici vingt ans, dépendra de qui gagnera la bataille de la liberté de disposer du savoir public. Si nous perdons, nous serons à peine cités. Si nous gagnons, il n'est pas sûr que les gens reconnaîtront le rôle du système d'exploitation GNU — s'ils croient qu'il s'agit de « 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 en retiendront dans cent ans dépendra de qui domine politiquement. »

Cherchant lui-même une figure à qui se comparer dans l'histoire du XIX^{e} siècle, Stallman évoque le militant abolitionniste John Brown, considéré comme un héros d'un côté de la ligne Mason Dixon, et comme un fou de l'autre. 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, l'homme était un héros ; cent ans après, et pour la plus grande partie du vingtième siècle, les manuels d'histoire enseignèrent qu'il était fou. À l'ère de la ségrégation institutionnalisée, à une époque où les racismes s'affichaient sans honte, les États-Unis laissaient en partie les états du Sud réécrire l'Histoire, et les manuels scolaires recelaient bien des contrevérités à propos de la Guerre civile et des évènements qui s'y rapportaient.

Cette comparaison illustre à la fois la nature du travail de Stallman (qu'il considère lui-même comme accessoire) et le caractère bivalent de sa célébrité. On imagine difficilement sa réputation abaissée au même niveau d'infamie que celle de Brown après la Reconstruction. Malgré d'occasionnelles analogies martiales, Stallman n'a jamais rien fait qui puisse paraître inciter à la violence.

Reste qu'il est facile de concevoir un futur où ses idées finiraient au rebut.

Dans le texte original, Sam Williams a ajouté à la fin de ce paragraphe : « 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 des logiciels privateurs ». Cela ne correspond pas aux faits. Dès la première annonce du projet GNU, j'ai demandé au public de soutenir la cause. Le mouvement pour le logiciel libre a vocation à devenir un mouvement de masse. Tout dépend de savoir à partir de combien de supporters vous définissez le mot « masse ». En 2009, la Free Software Fondation compte quelques 3 000 membres à jour de leurs coûteuses cotisations, et plus de 20 000 abonnés à la lettre électronique mensuelle.

Mais c'est peut-être cette volonté inébranlable qui pourrait se révéler son plus grand leg. 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 comme un simple épiphénomène au regard des « artéfacts » de sa vie. Sans cette personnalité, dit Moglen, il y aurait eu bien peu de réalisations à considérer.

En tant qu'ancien conseiller juridique auprès de la Cour suprême des États-Unis, il ajoute : « Voyez-vous, le plus grand homme pour qui j'avais jamais travaillé était Thurgood Marshall[10]. Je savais ce qui faisait 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 d'établir une comparaison, car les deux hommes ne pourraient être plus différents :

10. Juriste américain, premier Noir à avoir siégé, de 1967 à 1991, à la Cour suprême des États-Unis.

Thurgood Marshall était un homme intégré dans la société, représentant certes un peuple exclu par la société même dont il faisait partie, mais quand même un homme intégré dans la société. Son talent était un talent social. Cependant il était d'une intégrité absolue. La personne avec qui je le compare désormais le plus sur ce point, c'est Stallman, aussi différents soient-ils à tout autre égard : intègre, compact, fait de la substance qui compose les étoiles, jusqu'au-boutiste. »

Dans un effort pour donner corps à cette image, Moglen se remémore un moment qu'ils partagèrent au printemps 2000. Le succès du rachat de VA Linux résonnait toujours dans le milieu des affaires, et une demi-douzaine de numéros dédiés 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 un réfugié dans l'œil du cyclone. Durant l'heure suivante, dit-il, la conversation n'avait tranquillement porté que sur un seul point : 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'imaginai 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 nœuds dans ses cheveux, se comportant à sa manière habituelle. Quiconque aurait écouté notre conversation nous aurait pensé fous, mais je savais… je savais que la révolution était là, à cette table. Elle se jouait à cette table, et l'homme qui la menait était devant moi. »

Pour Moglen, ce moment-là plus que tout autre mettait en lumière la simplicité fondamentale du style de Stallman. « C'était

drôle, se souvient-il. Je lui ai dit : 'Richard, tu sais, toi et moi sommes les deux seuls à ne gagner aucun argent avec cette révolution'. Puis j'ai payé pour le repas, parce que je savais qu'il n'en avait pas les moyens. »

Je n'ai jamais refusé qu'on m'invite au restaurant car ma fierté n'est pas fondée sur le fait de payer ou non la tournée. Je devais d'ailleurs avoir assez d'argent sur moi pour payer ce repas avec Eben Moglen. Mes revenus, qui proviennent pour près de la moitié des discours que je donne, sont sans doute moindres que ceux d'un professeur de droit. Mais je ne suis pas pauvre.

Et en parlant de liberté...

de Richard Stallman

Aux Français, j'explique le logiciel libre en trois mots qui devraient leur être familiers : liberté, égalité, fraternité. Liberté, parce que chaque utilisateur est libre de l'usage qu'il veut faire du programme. Égalité, parce que le logiciel libre ne confère à personne de pouvoir sur personne. Fraternité, parce que les utilisateurs peuvent s'entraider, en partageant des copies et en développant en collaboration leurs versions.

Il va sans dire qu'aujourd'hui, vue l'omniprésence de l'informatique dans la vie, la fraternité en informatique ne se limite plus aux seuls logiciels. Partager des copies des œuvres publiées est une pratique commune, fort utile. Cette pratique ne doit souffrir aucune entrave.

Or, dans le monde, les États qui sont dominés par l'empire des entreprises mènent une guerre contre la pratique du partage, au point d'en faire paraître la simple notion comme aberrante, antinaturelle, voire barbare. Ils l'appellent « piraterie » comme si partager équivalait moralement à attaquer et piller un navire.

Cette guerre est orchestrée par l'industrie du divertissement. Avec la loi HADOPI, la France en est la victime emblématique. Avec cette loi, je crois que l'actuel gouvernement français porte atteinte du même coup aux trois valeurs de liberté, d'égalité, et de fraternité. Liberté, parce que cette loi institutionnalise la chasse à ceux qui osent partager. Égalité, parce que cette loi n'octroie qu'à quelques organisations privées le pouvoir exclusif de la dénonciation. Fraternité, parce que son but est d'écraser l'entraide qui lie les citoyens.

L'aspect qui révèle le plus clairement la nature tyrannique de la loi HADOPI est qu'elle cherche à imposer à chaque Français le

rôle de soldat d'une guerre contre les autres : celui qui ne « sécurise » pas son réseau, c'est-à-dire, qui n'aide pas les maîtres à maintenir leur joug sur les autres, risque d'être puni pour être resté neutre. Cette pratique de « responsabilisation collective » est le recours classique des gouvernements injustes dont le but est d'exploiter leurs sujets.

Défier la responsabilisation collective pour protéger les concitoyens contre l'empire est le premier pas naturel de la résistance.

J'ai l'ardent espoir que les citoyens français sortent vainqueurs de la bataille qui les oppose aux entreprises impérialistes qui ont exigé l'HADOPI, et qu'ils résistent aux politiciens serviles qui, sous leurs ordres, l'ont imposée.

Il faut en finir avec cette loi et son premier essai, DADVSI, mais aussi avec toute loi qui interdit aux gens de partager entre eux les copies d'une oeuvre publiée.

Richard Stallman

Épilogue de Sam Williams : une écrasante solitude

Notes de Richard Stallman sur ce chapitre

Ce chapitre est très personnel de la part de Sam Williams. Je n'ai fait qu'y clarifier certains points techniques ou juridiques, et supprimer des passages hostiles n'apportant pas d'information. J'ai également inséré quelques réponses, présentées ainsi. Sam Williams a aussi apporté quelques modifications à ce chapitre ; elles ne sont pas indiquées explicitement.

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 bien pâle en comparaison de celui qui a lieu dans les coulisses.

Dans *L'Autobiographie de Malcom X*, Alex Haley propose lui-même au lecteur un épilogue de quelques pages où, se débarrassant un instant du rôle de narrateur objectif, il écrit à la première personne. On y comprend 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 ses barrières personnelles et politiques afin de coucher la vie de Malcom X sur le papier.

J'hésite à comparer ce livre avec *L'Autobiographie de Malcom X*, bien sûr, mais je dois faire part de ma gratitude à Haley pour cet épilogue d'une grande sincérité. Durant ces douze derniers mois, il m'a servi de manuel d'instructions pour gérer un sujet biographique qui a bâti toute sa carrière sur son caractère désagréable.

> *J'ai bâti ma carrière en disant non à des choses que les autres acceptent sans trop se poser de questions, et si je peux paraître ou être désagréable à l'occasion, c'est sans en avoir l'intention.*

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 s'est poursuivie à travers les diverses localités mentionnées dans ce livre : Silicon Valley, Maui, Boston et Cambridge. Au final toutefois, 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. À cette époque, j'écrivais des récits pour le malchanceux site Internet *BeOpen.com*. Une de mes premières missions fut un entretien téléphonique avec Richard M. Stallman. Ce fut un tel succès que *Slashdot*[1], quotidien de référence des *nerds* appartenant à VA Software, Inc. (anciennement VA Linux Systems et encore avant, VA Research) le référença en première page de sa liste quotidienne d'articles. Quelques heures après, les serveurs de *BeOpen.com* surchauffaient de l'afflux massif de lecteurs.

L'histoire aurait dû se terminer là. Trois mois après ce premier entretien, alors que j'assistais à la conférence O'Reilly sur l'open source à Monterey en Californie, je reçus 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 avec grand intérêt votre entretien avec Richard Stallman sur
BeOpen. Je m'intéresse à RMS et à son travail depuis pas mal de temps
maintenant, et j'ai été ravie de lire votre contribution. Je pense vraiment
que vous avez fait du bon travail en capturant un peu l'esprit de ce que
Stallman essaye de faire avec GNU-Linux et la Fondation pour le logiciel
libre. Toutefois, j'apprécierais énormément d'en lire davantage, et je ne
pense pas être la seule. Croyez-vous qu'il y a des informations et/ou des
sources supplémentaires pour compléter votre entretien et l'adapter comme
une biographie ? Peut-être inclure des éléments plus anecdotiques sur sa
personnalité et son histoire qui pourraient vraiment intéresser et éclairer
les lecteurs hors du milieu des programmeurs hardcore ?
```

Tracy terminait le message en me demandant de lui téléphoner pour parler plus longuement de cette idée. Quand je l'appelai, elle me dit que son entreprise lançait une nouvelle série de livres électroniques, et qu'elle cherchait des récits qui puissent attirer

1. Voir : http://www.slashdot.org.

des lecteurs adeptes de nouveauté. Le format du livre électronique était de trente mille mots, soit environ cent pages, et Tracy avait vendu l'idée auprès de ses supérieurs de dresser le portrait d'un acteur majeur de la communauté hacker. Ses chefs apprécièrent la proposition, et dans ses recherches de personnes intéressantes, elle avait trouvé l'entretien de Stallman sur *BeOpen*. D'où son courrier électronique.

C'est ainsi qu'elle me demanda si je pourrais transformer l'entretien en un portrait complet. J'acceptai immédiatement.

Avant d'aller plus loin, ma correspondante me suggéra de mettre au point une proposition de récit qu'elle pourrait présenter à sa hiérarchie. Deux jours plus tard, je lui envoyai une proposition soignée, à laquelle elle répondait une semaine plus tard pour m'annoncer que ses chefs lui 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 l'homme 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 ».

Cela dit, je savais que Stallman était à la recherche d'un moyen de diffuser son message auprès d'un public plus large. Peut-être serait-il plus réceptif au projet ainsi présenté ? Sinon, je pouvais toujours m'en remettre à la grande quantité de documents, d'entretiens et de conversations enregistrées qu'il avait laissé traîner sur l'Internet, et en faire une biographie non autorisée.

Durant mes recherches, je trouvai un essai intitulé *Liberté ou copyright ?* écrit par Stallman et publié en juin 2000, dans la *MIT Technology Review*. L'essai fustigeait les livres électroniques pour

l'assortiment de péchés qu'ils impliquaient en matière de logiciels. Non seulement les lecteurs devaient utiliser des logiciels privateurs pour pouvoir les lire, se lamentait Stallman, mais les méthodes employées pour empêcher les copies non autorisées étaient exagérément brutales. Au lieu de télécharger un fichier transférable au format HTML ou PDF, les lecteurs téléchargeaient un fichier chiffré.

Ainsi, acheter un livre électronique revenait à acheter une clef non transférable permettant de traduire le contenu chiffré. Et toute tentative d'ouvrir un livre électronique sans la clef autorisée constituait une violation criminelle du *Digital Millennium Copyright Act*, la loi de 1998 censée étayer le respect du copyright sur l'Internet. Des pénalités étaient prévues, même dans le cas de personnes convertissant le contenu d'un livre dans un format de fichiers ouvert pour le lire sur un autre ordinateur, chez eux. Contrairement à ce qu'ils pouvaient faire avec un livre classique, les lecteurs d'un livre électronique n'avaient plus le droit de le prêter, de le copier ou de le revendre. Ils n'avaient la permission de le lire que sur une machine autorisée, nous avertissait Stallman.

« Nous avons toujours les mêmes libertés lorsque nous utilisons des livres papier. Mais si les livres électroniques venaient à les remplacer, cette exception serait caduque. Avec 'l'encre électronique', qui rend possible le téléchargement d'un nouveau texte sur une feuille de papier apparemment imprimée, même les journaux pourraient devenir éphémères. Imaginez : plus de marchands de livres anciens, 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, si l'on en croit les publicités pour le Reader de Microsoft, plus d'anonymat possible lors de l'achat d'un livre). Voilà le monde que les éditeurs envisagent pour nous. » [2]

2. Voir l'article « Liberté — ou copyright ? » (mai 2000). http://www.technologyreview.com/articles/stallman0500.asp.

Il va sans dire que cet essai souleva certaines interrogations. Ni Tracy ni moi n'avions parlé du logiciel que son entreprise allait utiliser, ni même du type de licence qui encadrerait l'usage du livre électronique. 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. Elle me promit de me contacter à nouveau.

Impatient de commencer, je décidai d'appeler Stallman malgré tout et de lui présenter l'idée du livre. Quand je le fis, il exprima tant un intérêt qu'une inquiétude immédiats. « 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 », il énonça deux conditions : d'une part, il ne voulait pas apporter son soutien à un mécanisme de licence e-book auquel il s'opposait fondamentalement, et d'autre part, il ne voulait pas paraître le cautionner non plus. « 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 la notice légale que les lecteurs étaient autorisés à faire et distribuer des copies mot pour mot du contenu du livre électronique. Il désigna *The Plant* de Stephen King comme modèle possible. En effet, en juin 2000, l'écrivain avait annoncé sur son site Internet officiel qu'il allait auto-éditer *The Plant* sous forme d'épisodes. Selon l'annonce, le coût total du livre serait de 13 dollars, étalés sur une série de chapitres à 1 dollar. Tant qu'au moins 75 % des lecteurs payaient pour chaque chapitre, King promit de continuer à publier les nouveaux épisodes. En août, le plan semblait fonctionner, car il 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 permis d'en faire des copies exactes. »

Je me souviens d'avoir également soulevé la question du chiffrement, comme le confirment les deux paragraphes suivants. Je n'aurais pas accepté de publier le livre si sa lecture avait « nécessité » l'usage d'un programme non libre.

Je fis suivre cette information à Tracy. J'étais convaincu qu'elle et moi pourrions trouver un arrangement équitable. J'appelai ensuite Stallman et nous convînmes d'un premier entretien pour le livre. Il accepta de me rencontrer sans redemander où en étaient les questions de licence. Peu après cette première entrevue, je me hâtai de prévoir une deuxième interview (à Kihei cette fois), m'arrangeant pour rencontrer mon interlocuteur avant son départ pour quatorze jours de congés à Tahiti.

Il ne s'agissait pas que de vacances puisque j'y ai aussi donné une conférence.

C'est au cours des vacances de Stallman que Tracy m'annonça la mauvaise nouvelle : le département des affaires juridiques de son entreprise ne voulait pas adapter sa notice légale sur les livres électroniques. Les lecteurs qui voulaient rendre leur livre transférable devraient d'abord casser le verrou de chiffrement pour pouvoir convertir le livre dans un format libre et public comme le HTML. Un acte illégal qui les exposait à des sanctions juridiques.

Avec deux entretiens frais dans ma besace, je ne voyais pas comment écrire mon livre sans les y inclure. J'arrangeai rapidement un voyage à New York pour rencontrer mon agent et Tracy afin de voir si un compromis était possible.

À 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 considéraient le format du livre électronique avec déjà suffisamment de suspicion, et n'étaient pas dans un état d'esprit idéal pour expérimenter des innovations de copyright qui permettaient plus facilement au lecteur de ne pas payer.

Cependant, en tant qu'agent spécialisé dans les ouvrages techniques, Henning était intrigué par la nature inédite 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 d'une 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, disait-il, 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 de la communauté des hackers. Le bâton serait l'ensemble des risques encourus s'il ne la respectait pas.

Neuf mois avant que l'affaire Dmitri Sklyarov ne devienne célèbre sur l'Internet, nous savions que viendrait tôt ou tard le moment où un programmeur entreprenant révèlerait comment craquer les livres électroniques. Nous savions aussi que voir une grande maison d'édition publier un livre électronique verrouillé par chiffrement et dont le sujet était Richard M. Stallman équivalait à lui apposer une couverture proclamant « Volez ce livre électronique ».

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 OpenOffice.org, la suite bureautique libre ? L'éditeur aurait pu ensuite publier des versions du livre électronique restreintes par DRM au format maison, tirant avantage des revenus qui venaient avec le logiciel dédié, tout en publiant une version copiable dans un format HTML moins esthétique.

> *Williams avait ici écrit à tort « commerciales », mais c'était inexact, puisque commercial signifie « à but de commerce ». Toutes ces versions seraient commerciales dès lors qu'elles seraient publiées par un éditeur.*

Stallman dit qu'il ne s'opposait pas à l'idée d'une double licence, mais qu'il n'aimait pas l'idée de diffuser la version librement copiable dans une qualité inférieure. Par ailleurs, dit-il, à la réflexion, ce cas était différent car il avait un moyen de contrôler le résultat : il pouvait refuser de coopérer.

> *La question était de savoir si je serais en tort d'accepter une version restreinte. Je peux cautionner la version libre OpenOffice de Sun, car c'est du logiciel libre et beaucoup mieux que rien, tout en rejetant la version non libre. Il n'y a de ma part aucune contradiction, car Sun n'avait pas besoin de mon accord pour cette version non libre — et ne me l'a pas demandé ; je n'étais pas responsable de l'existence de cette version. Dans le cas présent, si je donnais mon accord pour la version non librement copiable, c'était de ma responsabilité.*

Je fis quelques autres suggestions sans grand effet. La seule chose que je pus à peu près obtenir de lui fut une concession sur le fait que la licence du livre électronique restreigne à une redistribution *non commerciale* toute forme d'échange du fichier.

Il parle de concession, mais c'était, plus exactement, un compromis de plus…

Avant de raccrocher, Stallman suggéra que je dise à l'éditeur que je lui avais promis que le résultat serait librement partageable. Je lui répondis que je ne pouvais m'engager à faire une telle déclaration, mais que je pouvais toujours leur dire que je considérais le livre comme non achevable sans sa coopération. Apparemment satisfait, Stallman raccrocha avec sa phrase traditionnelle : « Hacke bien. »

Williams dit qu'il ne pouvait s'engager à une telle déclaration, mais cela n'aurait pourtant pas été mentir, puisque Williams avait accepté mes conditions au départ.

Henning et moi rencontrâmes Tracy le lendemain. Elle annonça que son entreprise était prête à publier des extraits copiables mais qu'elle les limiterait à cinq cents mots. Henning lui signifia que cela ne serait pas suffisant pour m'affranchir de mon obligation éthique envers Stallman. Tracy évoqua les obligations contractuelles de son entreprise envers divers distributeurs en ligne tels qu'Amazon.com. Même si la maison d'édition décidait de n'ouvrir le contenu du livre électronique qu'à titre exceptionnel, il y avait un risque que les partenaires dénoncent une rupture de contrat.

Sachant que ni la direction de l'éditeur, ni Stallman ne céderaient sur rien, la décision me revenait. Je pouvais soit trahir mon premier engagement envers Stallman et utiliser quand même mes entretiens, soit plaider l'éthique journalistique et m'affranchir de l'engagement oral d'écrire le livre.

À 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 comme 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-il.

La conversation prit mauvaise tournure dès le début. Je relayai à Stallman le commentaire de Tracy concernant les obligations contractuelles de l'éditeur.

« Et alors ? dit Stallman abruptement. En quoi est-ce que leurs obligations contractuelles devraient m'intéresser ? »

« Parce que demander à une grosse maison d'édition de prendre le risque d'une bataille juridique avec ses distributeurs, juste pour un livre électronique de 30 000 mots, semble exagéré », suggérai-je.

Williams partait alors du principe implicite que je ne pourrais refuser par principe.

« Tu ne comprends donc pas ? 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 lorsque j'étais musicien. Tel le gigolo éternellement frustré que décrit la

chanson, je pouvais sentir les choses s'écrouler aussi vite qu'elles s'étaient construites. L'ironie était palpable. Après des semaines passées à écouter attentivement les lamentations des autres, je me retrouvais dans une position où je devais obtenir le plus rare des mets : un compromis de Richard Stallman.

Alors que je continuais à parler, plaidant la position de l'éditeur et révélant ma sympathie grandissante pour lui, Stallman, tel un animal, sentit le sang et attaqua. « Alors c'est tout ? Tu vas simplement m'arnaquer ? Tu vas simplement te soumettre à leur volonté ? »

Voilà qui montre combien Williams interprétait mes propos. Il me compare à un prédateur, mais je ne faisais qu'opposer un « non » au marché qu'il tentait de me faire accepter. J'avais déjà fait plusieurs concessions, décrites plus haut. Je refusais juste de compromettre complètement mes principes. Ce genre de choses m'arrive souvent : ceux qui ne sont pas satisfaits disent que je « refuse tout compromis » mais c'est une exagération (voir http://www.gnu.org/philosophy/compromise.html). En réalité, je craignais à ce moment-là qu'il ne revienne sur les conditions qu'il avait d'abord acceptées et qu'il publie le livre avec des DRM, malgré mon refus. Je ne sentais pas le « sang » mais une possible trahison.

Je soulevai à nouveau la question du double copyright.

« Tu veux dire licence », rétorqua sèchement Stallman.

« Oui, licence. Copyright. Peu importe », dis-je, me sentant subitement dans la peau d'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.

Je crois que cette citation a été brouillée, non seulement parce que l'utilisation de « putain » (fucking) comme adverbe n'a jamais été dans mes habitudes linguistiques, mais aussi parce que ces mots ne correspondent pas aux circonstances. On dirait une remontrance à un subordonné. Pour moi, il avait une obligation éthique envers moi, mais n'était pas mon subordonné et je ne me serais pas adressé à lui en tant que tel. Comme Williams prenait des notes sans enregistrer, on ne peut lui tenir rigueur de sa retranscription approximative.

J'avais dû soutenir la position de l'éditeur jusqu'à l'extrême, car dans mes notes, j'ai réussi à consigner la châtaigne finale de Stallman : « Ça m'est égal. Ce qu'ils font est mal. Je ne peux cautionner le mal. Au revoir. »

Il semble que j'étais arrivé à la conclusion qu'il ne pourrait entendre un « non » et que la seule façon de terminer la conversation sans accepter sa proposition était de raccrocher.

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, ajouta-t-il en riant. Je t'ai vu trembler sur la fin. »

Je tremblais effectivement. Ce tremblement ne cesserait qu'après une bonne moitié de Guiness. Ça faisait bizarre de m'entendre qualifié d'émissaire du « mal ».

Mes mots visaient l'éditeur, pas Williams en tant que personne. S'il l'a pris pour lui, peut-être est-ce le signe qu'il commençait à se sentir responsable, moralement, du contrat qu'il m'avait pressé d'accepter.

D'autant plus bizarre que trois mois auparavant, j'étais dans un appartement d'Oakland, cherchant l'idée de mon prochain récit. À 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 ne connaissais que depuis la veille. Tout cela était trop surréaliste, comme si je voyais ma vie mise sous la forme d'un montage de cinéma.

À ce moment, mon compteur d'absurdité interne pris le relais. Le tremblement initial laissa la place à des rires convulsifs. Pour mon agent, je devais ressembler à l'un de ces auteurs fragiles subissant une rupture émotionnelle inopportune. Quant à moi, j'avais juste l'impression de commencer 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 plus 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 en trinquant avec mon agent. Autant en tirer notre parti. »

Si cette histoire avait été une pièce de théâtre, c'est à ce moment-là que j'aurais placé un intermède romantique. Découragée par la tension de notre rendez-vous, 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 les rattrapâmes.

Une fois 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. À nouveau, la conversation fut agréable, au point que le livre électronique sur Stallman était devenu un lointain souvenir.

Quand je rentrai à Oakland, j'appelai divers journalistes, amis ou fréquentations. Je leur racontai mes misères. La plupart me tancèrent pour en avoir trop cédé à Stallman dans la négociation préliminaire.

Ceux qui ont lu le livre savent à présent que jamais je n'aurais cédé sur ces conditions.

Un ancien professeur en école de journalisme me suggéra d'ignorer le commentaire de Stallman sur l'hypocrisie et d'écrire mon histoire malgré tout. Ceux connaissant Stallman et son sens des médias m'exprimèrent leur sympathie, mais tous offraient 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 nous partagions 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. Nous nous mariâmes le 11 mai. Merci aux contrats d'édition ratés.

Durant l'été, je commençai à arranger les notes de mes 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 documents 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 sa part. Les

deux me blâmaient pour avoir utilisé « Linux » au lieu de « GNU/-Linux » dans deux articles pour le magazine en ligne *Upside Today*. En dehors de ça, ce fut le silence. En juin, une semaine environ après son discours à l'université de New York, je pris l'initiative d'écrire un article de magazine de cinq mille mots à son propos. 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 reçus un appel de Henning. Il me dit que O'Reilly & Associates, une maison d'édition basée à Sebastopol, Californie, souhaitait publier l'histoire de Stallman sous la forme d'une biographie.

> *J'ai le vague souvenir d'avoir suggéré de contacter O'Reilly, mais je ne peux en être sûr après tant d'années.*

La nouvelle me ravit. De toutes les maisons d'édition de par le monde, O'Reilly, qui avait publié *La cathédrale et le bazar* d'Eric Raymond, semblait la plus attentive aux problèmes qui avaient tué le livre électronique et dont je parlais précédemment. En tant que journaliste, je m'étais beaucoup servi du livre *Open Sources* de chez O'Reilly comme référence historique. Je savais aussi que divers chapitres du livre, dont celui écrit par Stallman, avaient été publiés avec des notices (de licences) autorisant la redistribution. De telles informations seraient utiles si le problème de la publication électronique faisait à nouveau surface.

Bien entendu, ce fut le cas. J'appris par Henning que O'Reilly voulait publier la biographie à la fois sous une forme papier traditionnelle, mais aussi dans sa nouvelle offre de service payante *Safari Tech Books Online*. La licence utilisateur de Safari impliquait

des restrictions spécifiques [3], m'avertit Henning, mais O'Reilly souhaitait permettre un copyright qui donnait aux utilisateurs le droit de copier et partager le texte, quel que soit son support. En tant qu'auteur, j'avais en gros le choix entre deux licences : la licence OPL (*Open Publication Licence*) ou la GNU FDL (*GNU Free Documentation Licence*).

Je consultai alors le contenu et l'histoire de chacune. La licence OPL donne aux lecteurs le droit de reproduire et de distribuer une réalisation, en partie ou entièrement, de manière « physique ou électronique », tant que la copie reste sous cette même licence [4]. Elle permet aussi les modifications de la réalisation, sous certaines conditions. Enfin, elle 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 accord préalable de l'auteur.

Quant à la GNU FDL, elle permet la reproduction et la distribution d'un document sous n'importe quelle forme, tant que les versions résultantes portent la même licence [5]. 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 leur donne pas non plus le droit de rejeter les modifications qui pourraient aboutir à un livre concurrençant la première version. Elle oblige cependant tout tiers (non détenteur du copyright) souhaitant publier plus de cent copies d'un travail protégé, à inscrire en première et quatrième de couverture certaines informations.

3. Voir les conditions d'utilisation du service *Safari Tech Books Online* : http://my.safaribooksonline.com/termsofservice.

4. Voir *The Open Publication licence*, version 1.0 (8 juin 1999) — http ://open content.org/openpub.

5. Voir *The GNU Free Documentation licence*, version 1.1 (mars 2000) — http://www.gnu.org/copyleft/fdl.html.

Durant mon processus de recherche sur les licences, je m'assurai aussi de lire la page du site Internet du projet GNU consacrée à cette question : « Diverses licences et commentaires à leur propos » [6]. Sur cette page, je trouvai une critique de Stallman concernant la licence Open Publication. Elle était relative à la création de versions modifiées et à la capacité conférée à l'auteur de choisir l'une des options de l'OPL pour restreindre la possibilité de modifier le travail. Si un auteur ne souhaitait 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 ultérieures du document.

L'importance accordée aux modifications dans ces deux licences reflétait leur but originel : donner aux propriétaires de manuels de logiciels une chance de les améliorer et de publier les 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 préoccupation était d'offrir aux lecteurs le droit de copier ou d'échanger le contenu — la même liberté dont ils auraient joui en achetant une version imprimée du livre. Considérant ces deux licences pertinentes par rapport à mon objectif, je signai le contrat avec O'Reilly dès qu'il me fut envoyé.

La notion de modifications non restreintes continuait malgré tout de m'intriguer. Dans les premières négociations avec Tracy, j'avais mis en avant 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

6. Voir http://www.gnu.org/philosophy/license-list.html.

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[7], le légendaire concept logiciel 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 (libre) du projet, et j'avais été impressionné. Au cours de la démonstration, Udanax affichait côte à côte le document parent et la version modifiée, dans une mise en forme similaire sur deux colonnes en format texte. D'un simple clic on pouvait introduire des lignes reliant chaque phrase du document de départ 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 au vu d'un tel potentiel, pourquoi ne pas donner aux utilisateurs la possibilité de s'amuser ? (J'aurais d'ailleurs soutenu avec enthousiasme quiconque aurait porté ce livre vers Udanax, la version libre de Xanadu — http ://www.udanax.com.)

Quand Laurie Petrycki, mon éditrice 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 dans ce domaine en raison du 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, ces collections auraient-elles survécu ?

Après avoir signé le contrat, j'informai Stallman qu'un projet de livre était à nouveau sur les rails. J'évoquai le choix que

7. http://www.xanadu.com

O'Reilly me donnait entre la licence *Open Publication* et la licence *GNU Free Documentation Licence*. 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 argumenta en faveur de la GFDL. Il me fit remarquer que O'Reilly l'avait utilisée à plusieurs reprises auparavant. Malgré les événements de l'année passée, je proposai alors un arrangement. Je choisirais la GFDL si cela me permettait de faire plus d'entretiens et s'il aidait O'Reilly à faire la promotion du livre. Stallman accepta de participer, mais précisa que sa participation aux événements promotionnels dépendrait du contenu du livre. Trouvant le marché honnête, je convins d'un entretien le 17 décembre 2001 à Cambridge.

Je plaçai la rencontre de manière à coïncider avec un voyage d'affaires 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 a permis notre rencontre ».

J'écrivis un courrier électronique au hacker, qui répondit promptement, acceptant l'offre. Après avoir roulé vers Boston le lendemain, je passai prendre Tracy à son hôtel avant de sauter dans les transports en commun en direction du MIT. En arrivant à Tech Square, nous surprîmes notre hôte au milieu d'une conversation, alors que nous frappions à sa porte.

« Excusez-moi », dit-il, tenant la porte ouverte de suffisamment loin pour que Tracy et moi ne puissions que difficilement entendre son interlocuteur. C'était une jeune femme, dans les vingt-cinq ans aurais-je dit, nommée Sarah.

« Je me suis permis d'inviter une autre personne à dîner », dit-il, nous mettant devant le fait accompli et me lançant ce même sourire félin qu'il m'avait adressé dans le 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 sa propre mère. « 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 du restaurant, j'appris les circonstances de la première rencontre entre Richard et Sarah. Curieusement, le cas de figure m'était très familier. Travaillant sur son propre livre de fiction, Sarah avait entendu parler de Stallman et de quel personnage intéressant il était. Elle avait rapidement décidé de s'en inspirer pour l'un des personnages de son livre, et, dans ses recherches, elle avait arrangé un entretien avec lui. À partir de là, les choses s'étaient enchaînées. Tous deux étaient ensemble depuis le début 2001, rapportait-elle.

« J'ai vraiment admiré la façon dont Richard a construit un mouvement politique tout entier afin de traiter un problème profondément personnel », ajouta Sarah, expliquant son attirance vers Stallman.

« Quel problème ? », demanda aussitôt mon épouse.

« Une solitude écrasante », répondit-elle.

Durant le dîner, je laissai les femmes faire la conversation et passai le plus clair de mon temps à essayer de repérer des indices indiquant si les douze derniers mois avaient adouci Stallman de façon significative. Je ne vis rien qui puisse le suggérer. Certes plus

8. Hélas, je n'ai appris la décision de la fondation Takeda de récompenser Stallman, ainsi que Linus Torvalds et Ken Sakamura, avec le prix nouvellement créé de la « Réalisation technico-entrepreneuriale pour le bien-être social et économique » qu'après le départ de Stallman pour le Japon. Pour plus d'information sur cette récompense assortie d'un prix d'un million de dollars, voir le site de la Fondation Takeda : http://www.takeda-foundation.jp

charmeur que dans mes souvenirs, Stallman conservait ce même niveau d'acidité. Lorsque ma femme entama un emphatique « Dieu nous en garde », elle reçut instantanément une réprimande typique de Stallman : « Je suis désolé d'avoir à te l'apprendre, mais Dieu n'existe pas », dit-il.

> *J'ai dû être trop pince-sans-rire. Il aurait pu m'accuser, à juste titre, de faire le malin, mais pas de l'avoir blâmée.*

Plus tard, quand le dîner fut terminé et Sarah partie, il semblait avoir un peu baissé sa 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. « Je pensais que j'allais être seul pour toujours, dit-il. Je suis heureux de m'être trompé. »

Avant de nous quitter, il me tendit sa « carte de plaisir », une carte de visite avec son adresse, son numéro de téléphone, et ses passe-temps favoris (« s'échanger de bons livres, la bonne cuisine, les danses et les musiques exotiques ») afin que je puisse le contacter pour un entretien final.

Le lendemain, autour d'un nouveau repas vapeur chinois, il semblait encore plus amoureux que le soir précédent. Se souvenant de ses débats dans le dortoir de la Currier House sur les avantages et les inconvénients des sérums d'immortalité, il exprima le souhait que les scientifiques puissent un jour trouver la clef de la vie éternelle. « Maintenant que je commence enfin à être heureux, je souhaite vivre plus longtemps », dit-il.

Alors que j'évoquai le commentaire de Sarah sur son « écrasante solitude », Stallman dit ne pas voir de relation entre la solitude physique et spirituelle, et celle des hackers. « L'envie de

partager le code a un rapport avec l'amitié, mais à un niveau bien moindre », dit-il. Plus tard, cependant, quand le sujet refit surface, il admit que la solitude, ou la crainte de la solitude éternelle, avait joué un rôle majeur comme moteur de sa motivation durant les premiers jours du Projet GNU.

> *Je parlais de solitude au niveau communautaire, de hacker à hacker.*

« Ma fascination pour les ordinateurs n'est la conséquence de rien d'autre. Je n'aurais pas été moins fasciné par eux, même si j'avais été populaire et que les femmes s'étaient attroupées autour de moi. Cependant, il est vrai que le fait de ne pas avoir de chez-moi, d'en trouver un et de le perdre, d'en trouver une autre pour la voir détruite, m'a profondément affecté. Celle que j'ai perdue, c'est le dortoir, celle qui a été détruite, c'est le AI Lab. Le sentiment de précarité induit par le fait de ne me sentir chez moi nulle part, de ne pas avoir de communauté, a été très fort. Cela m'a donné envie de me battre pour retrouver cela. »

Après l'entretien, je ne pus m'empêcher d'éprouver une certaine proximité émotionnelle. En entendant Sarah décrire ce qui l'avait attirée vers Stallman, et en entendant l'homme décrire lui-même les sentiments qui l'avaient entraîné à défendre la cause du logiciel libre, j'étais ramené à mes propres motivations présidant à l'écriture de ce livre. Depuis juillet 2000, j'ai appris à apprécier tant les aspects séduisants 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. À de nombreux égards, les deux se définissaient mutuellement à tel point qu'ils en devenaient indissociables.

> *Williams objective ses réactions, tant positives que négatives, comme des parties de moi, mais elles sont aussi fonction de ses propres attitudes face à l'apparence, à la conformité, et à la réussite professionnelle.*

Je ne suis pas sûr que tous les lecteurs ressentiront ce même niveau d'affinité avec Stallman… mais je suis certain que la majorité conviendront que peu d'individus offrent un portait aussi singulier que lui. Mon souhait le plus sincère est que, grâce à ce portrait initial complété, et avec l'aide de la GFDL, d'autres ressentiront l'envie d'y ajouter leur propre perspective.

Annexe

A

À propos du terme hacker

Afin de comprendre le sens exact du mot *hacker*, il est bon de se pencher sur l'évolution de son étymologie au fil des années.

The New Hacker Dictionnary[1], un dictionnaire en ligne du jargon des programmeurs, donne officiellement neuf connotations différentes au mot *hack* et autant pour le mot *hacker*. Un essai joint à cette publication cite toutefois Phil Agre, hacker du MIT qui invite les lecteurs à ne pas se laisser duper par l'apparente flexibilité du mot. « *Hacker* n'a qu'une seule signification, indique-t-il. Une signification profonde et subtile qui échappe à toute tentative de

1. http://www.ccil.org/jargon/jargon_toc.html

l'exprimer par des mots. » Richard Stallman, quant à lui, définit le *hacker* par son « ingéniosité espiègle » [2].

Quelle que soit l'étendue de la définition, la plupart des hackers modernes font remonter son origine au MIT, où le terme s'imposa dans le jargon étudiant au début des années 1950. En 1990, le musée du MIT publia un journal traitant du phénomène hacker. D'après ce document, les étudiants de l'institut utilisaient à cette époque le mot *hack* dans le sens de « blague » ou « bricole » [3]. Suspendre un vieux tacot à une fenêtre de dortoir était un *hack*, mais toute farce plus dure ou malveillante, comme bombarder d'œufs les fenêtres d'un bâtiment rival ou dégrader une statue du campus, dépassait ce cadre. La définition du *hack* laissait transparaître implicitement un esprit d'amusement inoffensif et créatif.

Cet état d'esprit allait inspirer le gérondif *hacking*, d'où provient le verbe « hacker », en français. Un étudiant de 1950 qui aurait passé une bonne partie de son après-midi au téléphone ou à démonter une radio aurait pu qualifier son activité de *hacking*. Là encore, en français, on pourrait employer les verbes « bricoler », « bidouiller »» pour décrire la même activité [4].

Les années 1950 avançant, le mot *hack* acquit un sens plus précis, plus rebelle. Le MIT était à cet époque extrêmement compétitif, et le *hacking* s'imposa à la fois comme une réaction et une extension à cette culture de compétition. Farces et blagues étaient soudain devenues 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 à son comportement créatifs, que le rigoureux programme de l'Institut étouffait.

2. *Playful cleverness*, en anglais — NdT.

3. Dans la version originale, on donne comme équivalent en argot étudiant américain moderne le terme *goof* — NdT.

4. Dans le texte original : *goofing* ou *goofing off* — NdT.

Avec ses myriades de couloirs et de conduits de ventilation souterrains, l'institution offrait de nombreux endroits à explorer pour l'étudiant qui ne se laissait pas intimider par les portes fermées ou les panneaux « entrée interdite ». Les étudiants commencèrent à nommer « spéléo-hacking » [5] leurs explorations hors-piste. Au-dessus du niveau du sol, le système téléphonique du campus offrait des possibilités similaires. Entre expérimentation informelle et diligence de rigueur, les étudiants apprirent à jouer toutes sortes de tours. S'inspirant de la plus traditionnelle course dans les souterrains, les étudiants nommèrent vite cette nouvelle activité « hack téléphonique » [6].

C'est ce mélange de jeu créatif et d'exploration sans restriction qui allait servir de fondement aux mutations ultérieures du terme *hacker*. Les premiers à se décrire eux-mêmes comme des hackers de l'informatique sur le campus du MIT au début des années 1960 avaient fait partie d'un groupe d'étudiants appelé le *Tech Model Railroad Club* [7] à la fin des années 1950. Parmi eux, on trouvait la fine équipe du *Signals and Power (S&P) Committee*, à l'origine du circuit électrique alimentant le Railroad Club. Le circuit lui-même était un agencement sophistiqué de relais et de commutateurs similaires à ceux qui contrôlaient le central téléphonique du campus. Pour le diriger, un des membres du groupe n'avait qu'à composer les commandes sur un téléphone pour voir les trains obéir à ses ordres.

Les jeunes ingénieurs électriciens responsables de la construction de ce système voyaient dans leur activité un esprit proche de celui du « hacking téléphonique ». En adoptant le terme *hacking*, ils en affinèrent encore davantage les contours. Pour les hackers de S&P, utiliser un relais de moins pour commander une partie

5. *Tunnel-hacking*, en anglais — NdT.
6. *Phone-hacking*, en anglais — NdT.
7. Il s'agit d'un club de modélisme ferroviaire — NdT.

du chemin de fer signifiait en avoir un de plus sous la main pour s'amuser plus tard. Le sens de *hacking* évolua subtilement d'un synonyme de « passe-temps » à celui de « passe-temps améliorant la performance ou l'efficacité générale du système du Railroad Club ». Bientôt, les membres du comité S&P firent fièrement usage du mot *hacking* pour désigner l'activité entière d'amélioration et de remodelage du circuit électrique du chemin de fer, et ils nommèrent *hackers* ceux qui pratiquaient cette activité.

Étant donnée leur affinité pour l'électronique sophistiquée — sans parler de la défiance traditionnelle de l'étudiant du MIT face aux portes fermées et aux « accès interdits », il fallut peu de temps aux hackers pour avoir vent de la présence d'une nouvelle machine sur le campus. Baptisée TX-0, il s'agissait de l'un des tout premiers ordinateurs commercialisés. À la fin des années 1950, toute la bande de S&P avait migré en masse vers la salle de contrôle de la machine, apportant son esprit de jeu créatif.

Le royaume grand ouvert de la programmation allait entraîner une nouvelle mutation étymologique. « Hacker » ne signifiait alors plus assembler des bouts de circuits disparates les uns avec les autres, mais concocter des programmes informatiques sans prêter attention aux procédures d'écriture « officielles ». Cela impliquait aussi la capacité d'améliorer des logiciels existants tendant à accaparer trop de ressources système. Et fidèle à l'origine du terme, cela voulait dire aussi écrire des programmes sans autre utilité que celle de s'amuser.

Un exemple classique de cette definition étendue du *hacking* est Spacewar, le premier jeu vidéo sur ordinateur. Développé par des hackers du MIT au début des années 1960, Spacewar possédait toutes les caractéristiques du *hacking* traditionnel : c'était un jeu sans objectif précis, sinon de fournir un passe-temps nocturne à la douzaine de hackers qui adoraient y jouer. D'un point de vue strictement logiciel, cependant, c'était un témoin magistral

de l'innovation amenée par la 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 largement avec les autres programmeurs. À la fin des années 1960, Spacewar était devenu la distraction des développeurs du monde entier, pour peu qu'ils aient disposé d'un affichage graphique, ce qui était encore assez rare.

Cette notion d'innovation collective et de propriété commune des logiciels éloigna le hacker informatique des années 1960 des hackers téléphoniques ou spéléos des années 1950. À l'époque, en effet, il s'agissait davantage d'un acte individuel ou réalisé par un petit groupe, car les hackers spéléos et téléphoniques avaient beau utiliser l'équipement du campus comme matière première, la nature illégale de ces activités décourageait la divulgation des nouvelles découvertes. Cependant, les hackers informatiques travaillaient dans un environnement scientifique basé sur la collaboration et la récompense de l'innovation. Et si les informaticiens « officiels » ne furent pas toujours leurs meilleurs amis, avec l'évolution rapide de leur domaine d'étude, les deux espèces évoluèrent vers une relation de coopération, que certains qualifient même de symbiose.

En revanche, les hackers montraient peu de respect pour les règles bureaucratiques. Pour eux, les systèmes de sécurité qui bloquaient l'accès aux machines n'étaient qu'un bogue comme un autre… un bogue qu'il fallait retravailler et corriger autant que possible. Ainsi, briser un système de sécurité (sans mauvaise intention) fut l'une de leurs activités notoires dans les années 1970, savoir-faire très utile à la fois pour leurs facéties (les victimes pouvaient alors dire : « Quelqu'un est en train de me 'hacker'! ») et pour l'accès aux machines en soi. Cette activité n'était pourtant pas le cœur de l'esprit hacker. Face à un système de sécurité, les hackers étaient fiers de montrer leur esprit apte à le dépasser ; mais

lorsqu'ils en avaient la possibilité, comme au AI lab, ils préféraient travailler sans obstacle et se livrer à d'autres activités de *hacking*. En l'absence de système de sécurité, nul besoin de le briser.

C'est en hommage au talent prodigieux de ces hackers originels que leurs successeurs, Richard M. Stallman compris, cherchèrent à perpétuer cette identité. Dans la deuxième moitié des années 1970, le terme « hacker » avait acquis une connotation élitiste. Au sens général, un hacker informatique désignait quiconque écrivait du code pour le seul intérêt d'écrire du code. Mais plus particulièrement, cela impliquait une reconnaissance en termes de virtuosité en programmation. Tout comme le mot « artiste », le terme « hacker » comportait une connotation tribale. Qualifier un autre programmeur de hacker était un signe de respect. Se décrire comme hacker faisait montre d'une très grande confiance en soi. Toujours est-il que la flexibilité originelle du mot diminua à mesure que les ordinateurs se démocratisaient.

Tout en voyant sa définition se restreindre, le *hacking* informatique se vit doté de nouvelles connotations sémantiques. Les hackers du AI Lab avaient en effet de nombreux points en commun : la cuisine chinoise, le rejet du tabac, de l'alcool et de toute autre drogue addictive. Ces caractéristiques finirent par s'ajouter à l'identité perçue du hacker, tandis que la communauté exerçait son influence sur les nouveaux venus, sans exiger pour autant de s'y conformer. Toutefois, cet arrière-sens culturel disparut avec la communauté hacker du MIT. À cet égard, la plupart des hackers d'aujourd'hui sont le reflet de la société qui les entoure.

Par ailleurs, lorsque dans les institutions d'élite, comme au MIT, à Stanford et à Carnegie Mellon, les hackers discutaient des *hacks* qu'ils admiraient le plus, ils prenaient en compte les règles qui les sous-tendaient, commençant à parler ouvertement d'une « éthique hacker » : les règles non écrites du comportement quotidien du hacker. Ainsi, en 1984, le livre *Hackers*, de Steven Levy,

après moult recherches et consultations, codifiait l'éthique des hackers en cinq principes fondamentaux[8].

Dans les années 1980, l'utilisation de l'ordinateur connut une grande expansion, ce qui alla de pair avec les attaques des systèmes de sécurité. Bien que ces dernières aient été en majorité le fait d'initiés n'ayant aucun lien avec la communauté hacker, il arrivait que la police et les administrateurs — pour qui toute désobéissance était nécessairement mal intentionnée — remontent la trace de l'intrusion jusqu'à un hacker, dont la règle éthique consistait uniquement à « ne blesser personne ». Dès lors, les journalistes se mirent à publier des articles dans lesquels le *hacking* était considéré comme une activité consistant uniquement à briser les systèmes de sécurité, définition approuvée par la plupart des administrateurs. Et l'ouvrage de Steven Levy eut beau faire pour rendre compte de l'esprit originel d'exploration qui animait la culture hacker, pour la plupart des journalistes et des lecteurs, le « hacker informatique » devint synonyme de « cambrioleur numérique ».

Par la suite, à la fin des années 1980, de nombreux adolescents américains eurent accès aux ordinateurs. Certains étaient des exclus sociaux qui s'inspiraient de l'image du hacker véhiculée par les journalistes, exprimant alors leur malaise par l'attaque des systèmes de sécurité, là où d'autres adolescents mal dans leur peau auraient cassé quelques vitres. Ils se donnèrent le nom de hackers, sans toutefois intégrer le principe du MIT consistant à refuser tout acte malveillant. En conséquence, tandis qu'ils employaient leurs ordinateurs à des fins nuisibles (créer et disséminer des virus, s'introduire dans des systèmes, faire délibérément planter des

8. Selon S. Levy, l'éthique hacker obéit à ces cinq principes : 1) Toute information est par nature libre. 2) Être anti-autoritariste. 3) Les hackers se jugent par leurs prouesses, non par d'autres hiérarchies sociales. 4) Art et beauté peuvent être créés avec un ordinateur. 5) Les ordinateurs peuvent changer et améliorer la vie — NdT.

machines…), le terme « hacker » acquit une signification punk et nihiliste, qui attira encore davantage de personnes de ce type.

Les hackers se débattirent contre cette méprise durant presque vingt ans. Stallman, qui n'était pas du genre à se laisser faire, inventa la notion de « craquage » (*cracking*) pour désigner le fait de briser un système de sécurité, et ainsi éviter plus facilement que les gens emploient le terme « hacker » dans ce contexte. Cependant, la distinction entre « hacker » et « cracker » est souvent mal comprise : les deux termes ne sont pas censés être exclusifs ; il s'agit en fait de deux attributs d'une même activité, tout comme « jeune » et « grand » sont deux attributs pouvant convenir à une seule personne.

La plupart des *hacks* ne concernent pas les systèmes de sécurité, ce ne sont donc pas des *cracks*. Le craquage, par contre, est bien souvent une activité lucrative, ou de simple malveillance, sans jamais être ludique, ce n'est donc en rien du *hacking*. Il peut certes arriver, parfois mais rarement, qu'un acte soit les deux à la fois. Or, même s'il est irrévérencieux, le hacker respecte les règles. Quant au craquage, il a beau être par définition un acte de désobéissance, il n'est pas nécessairement nuisible. Ainsi, dans le domaine de la sécurité informatique, on fait habituellement la distinction entre « chapeaux noirs » et « chapeaux blancs », c'est-à-dire entre les « crackers » qui cherchent à nuire et ceux qui cherchent seulement à éprouver un système de sécurité dans le but de le corriger.

Le refus de la malveillance reste un dénominateur commun entre l'idée du hacker des années 1950 et celle du début du XXI^{e} 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 à explorer des souterrains, est demeurée intacte. À la fin de l'an 2000, le musée du MIT à rendu hommage à la longue tradition des hackers de l'Institut en leur dédiant une exposition, le *Hall of Hacks*. Cette dernière montre de nombreuses

photographies, remontant pour certaines jusqu'aux années 1920. L'une d'elles fait apparaître une fausse voiture de police. En 1993, les étudiants saluèrent l'esprit originel des hackers en plaçant le même véhicule, gyrophare en marche, sur le toit du dôme principal de l'institut. La plaque d'immatriculation de la voiture indiquait IHTFP, un sigle populaire au MIT doté de plusieurs sens. La version la plus intéressante, remontant sans doute à l'atmosphère très tendue du MIT dans les années 1950, est sans doute « *I Hate This Fucking Place* » (« Je hais ce putain d'endroit »). Il n'empêche qu'en 1990, le musée utilisa ce sigle comme base d'une publication sur l'histoire des hackers. Intitulée *The Journal of the Institute for Hacks, Tomfoolery and Pranks* (Le journal de l'Institut des hacks, niaiseries et autres farces), elle offre un aperçu pertinent de ce qu'est le hacking.

« Dans la culture du *hacking*, une création simple et élégante est tout 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 de la farce estudiantine habituelle par sa planification élaborée, son ingéniosité, sa finesse, et en ce qu'il exige à la fois de l'esprit et de l'inventivité. La règle implicite est que le hacking doit être de bonne intention, non destructif et sûr. D'ailleurs, il arrive même que les hackers aident à défaire leur propre ouvrage. »

Vouloir délimiter la culture du hacking informatique au sein des mêmes frontières éthiques est plein de bonne volonté, mais impossible. Bien que la plupart des *hacks* informatiques aspirent au même esprit, à la même élégance et à la même simplicité, le médium logiciel est bien moins réversible. Démonter un véhicule de police est plus facile que défaire une idée, tout particulièrement lorsque celle-ci s'est imposée.

Autrefois terme obscur du jargon étudiant, le mot *hacker* est devenu aujourd'hui une boule de billard linguistique, sujette aux

retournements politiques et aux nuances éthiques. Peut-être est-ce la raison pour laquelle tant de hackers et de journalistes aiment l'utiliser. Il est certes difficile de savoir comment le mot sera employé à l'avenir, mais nous pouvons déjà décider de la manière dont nous-mêmes allons l'employer : lorsqu'il s'agit de briser un système de sécurité, l'emploi du mot « cracker » au lieu de « hacker » est une preuve de respect envers Stallman et tous les hackers mentionnés dans ce livre.

Cela permet en tout cas de préserver une chose dont tous les utilisateurs d'ordinateur ont bénéficié un jour ou l'autre : l'esprit hacker.

Annexe

B

La GNU Free Documentation License

Version 1.3, 3 November 2008

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.

The "publisher" means any person or entity that distributes copies of the Document to the public.

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 under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

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. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM : How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page :

```
Copyright (C) YEAR YOUR NAME.

Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3, or
any later version published by the Free Software Foundation; with no
Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy
```

```
of the license is included in the section entitled "GNU Free Documentation
License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with … Texts." line with this :

```
with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover
Texts being LIST, and with the Back-Cover Texts being LIST
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

```
Official GNU FDL webpage : http://www.gnu.org/copyleft/fdl.html
```

Bibliographie

[Barnes and Ard, 2000]
Barnes, C. and Ard, S. (2000). *Court Grants Stay of Napster Injunction*, dans *News.com* (28 juillet 2000)
http://news.cnet.com/2100-1023-243817.html

[Ben-David, 2000]
Ben-David, P. (2000). *Whatever Happened to the 'Killer App'?*
http://www.ecommercetimes.com/story/5893.html

[Bennahum and Stallman, 1996]
Bennahum, D. and Stallman, R.M. (1996). *Stallman interview*, dans *MEME 2.04*
http://memex.org/meme2-04.html

[Betz and Edwards, 1986]

Betz, D. and Edwards, J. (1986). *Richard Stallman discusses his public-domain Unix-compatible software system with BYTE editors*, dans *Byte* (Juillet)

http://www.gnu.org/gnu/byte-interview.html

[Brooks, 1995]

Brooks, F.P. (1995). *The Mythical Man-Month*. Reading, Mass. : Addison Wesley Publishing.

[Brown, 1999]

Brown, A. (1999). *Programmer on moral high ground. Free software is a moral issue for Richard Stallman believes in freedom and free software*, dans *London Guardian* (6 novembre 1999)

http://www.guardian.co.uk/Archive/Article/0,4273,3926604,00.html

[Davenport, 1975]

Davenport, J. (1975). *Skinning the Tiger : Carmine DeSapio and the End of the Tammany Era*, dans *New York Affairs* **3** (1), pp.72–93

[DiBona, Ockman and Stone, 1999]

DiBona, C., Ockman, S. and Stone, M. (1999). *Open Sources. Voices from the Open Source Revolution*. Sebastopol, Cal. : O'Reilly Media Inc..

[Garfinkel, 1993]

Garfinkel, S. (1993). *Is Stallman Stalled ?*, dans *Wired* (1.01)

http://www.wired.com/wired/archive/1.01/stallman.html

[Garfinkel and Abelson, 1999]

Garfinkel, S. and Abelson, H. (1999). *Architects of the Information Society : Thirty-Five Years of the Laboratory for Computer Science at MIT*. Boston : MIT Press.

[Gates, 1976]

Gates, B. (1976). *An Open Letter to Hobbyists*.

http://www.blinkenlights.com/classiccmp/gateswhine.html

[Ghosh, 2000]
Ghosh, S. (2000). *Revealing the Microsoft Windows Source Code*. (Janvier 2000)
http://www.gigalaw.com/

[Greenwood, 1999]
Greenwood, L. (1999). *Why is Linux Successful ?*, dans *UniForum NZ 99*
http://www.freebsddiary.org/linux.php

[Gross, 2000]
Gross, M. (2000) Richard Stallman : High School Misfit, Symbol of Free Software, MacArthur-Certified Genius, In : *My Generation. Fifty Years of Sex, Drugs, Rock, Revolution, Glamour, Greed, Valor, Faith, and Silicon Chips*. New York : Cliff Street Books,
http://www.mgross.com/MoreThgsChng/interviews/stallman1.html

[Haley, 1993]
Haley, A. (1993). *L'autobiographie de Malcom X*. Paris : Grasset.

[Harmon, 1998]
Harmon, A. (1998). *For Sale : Free Operating System*.
http://www.nytimes.com/library/tech/98/09/biztech/articles/28linux.html

[Kahney, 1999]
Kahney, L. (1999). *Linux's Forgotten Man*, dans *Wired News* (5 mars 1999)
http://www.wired.com/science/discoveries/news/1999/03/18291

[Leibovitch, 2000]
Leibovitch, E. (2000). *Who's Afraid of Big Bad Wolves*, dans *ZDNet Tech Update* (14 décembre 2000)
http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2664992,00.html

[Leonard, 1998]
Leonard, A. (1998). *The Saint of Free Software*. (août 1998)
http://archive.salon.com/21st/feature/1998/08/cov_31feature.html

[Lerner, 1990]
Lerner, R. (1990). *Stallman wins $240,000 MacArthur award*, dans *MIT, The Tech* (Juillet)
http://the-tech.mit.edu/V110/N30/rms.30n.html

[Lessig, 2001]
Lessig, L. (2001). *The Future of Ideas*. New York : Random House Inc.
http://www.the-future-of-ideas.com/download/

[Lessig, 2005]
Lessig, L. (2005). *L'avenir des idées : Le sort des biens communs à l'heure des réseaux numériques*. Lyon : Presses Universitaires de Lyon.
http://presses.univ-lyon2.fr/?q=node/20

[Levy, 1984]
Levy, S. (1984). *Hackers. Heroes of the computer revolution*. New York : Penguin USA.

[Levy, 2002]
Levy, S. (2002). *Crypto : How the Code Rebels Beat the Government Saving Privacy in the Digital Age*. New York : Penguin Book.

[Mak, 1998]
Mak, M.L. (1998). *A Mae Ling Story*.
http://www.crackmonkey.org/pipermail/crackmonkey/1998-December/001777.html

[Markoff, 1999]
Markoff, J. (1999). *Apple Adopts 'Open Source' for its Server Computers*, dans *New York Times* (17 Mars 1999)
http://www.nytimes.com/library/tech/99/03/biztech/articles/17apple.html

[McHugh, 1998]
McHugh, J. (1998). *For the Love of Hacking*, dans *Forbes* (10 août 1998)
http://www.forbes.com/forbes/1998/0810/6203094a.html

[McKusick, 1999]
McKusick, M.K. (1999) Twenty Years of Berkeley Unix : From AT&T-Owned to Freely Redistributable, In : *Open Sources. Voices from the Open Source Revolution*. Sebastopol, Cal. : O'Reilly Media Inc., pp.31–46

[Mundie, 2001]
Mundie, C. (2001). *The Commercial Software Model*.
http://www.microsoft.com/presspass/exec/craig/05-03sharedsource.asp

[Murdock, 1994]
Murdock, I. (1994). *A Brief History of Debian*. debian.org.
http://www.debian.org/doc/manuals/project-history/index.en.html#contents

[Nelson, 1982]
Nelson, T.H. (1982). *Literary Machines*. Sausalito, Cal. : Mindful Press.

[Newitz, 2000]
Newitz, A. (2000). *If Code is Free Why Not Me ?*. (26 mai 2000)
http://archive.salon.com/tech/feature/2000/05/26/free_love/index.html

[Newquist, 1994]
Newquist, H.P. (1994). *The Brain Makers : Genius, Ego, and Greed in the Quest for Machines that Think*. IndianapolisSam Publishing.

[Perens, 1999]
Perens, B. (1999) The Open Source Definition, In : Dibona, C., Ockman, S. and Stone, M. (Eds.). *Open Sources : Voices from the*

*Open Source Revolution.*Sebastopol, Cal. : O'Reilly Media Inc., pp.171–88,
http://oreilly.com/catalog/opensources/book/perens.html

[**RIAA, 2001**]
RIAA (2001). *A Clear Victory for Recording Industry in Napster Case.*
http://riaa.com

[**Raymond, 1999a**]
Raymond, E.S. (1999a). *Shut Up and Show Them the Code.*
http://www.catb.org/~esr/writings/shut-up-and-show-them.html

[**Raymond, 1999b**]
Raymond, E.S. (1999b). *Surprised by Wealth*, dans *Linux Today* (10 Décembre 1999)
http://www.linuxtoday.com

[**Raymond, 1999c**]
Raymond, E.S. (1999c). *Guest Interview : Eric S. Raymond*, dans *Linux.com* (18 mai)
http://www.linux.com/interviews/19990518/8/

[**Raymond, 2000**]
Raymond, E.S. (2000). *The Cathedral and the Bazaar.*
http://www.catb.org/~esr/writings/cathedral-bazaar/

[**Salus, 1994**]
Salus, P.H. (1994). *A Quarter Century of UNIX.* Reading, Mass. : Addison-Wesley Professional.

[**Silberman, 2001**]
Silberman, S. (2001). *The Geek Syndrome*, dans *Wired* (Décembre 2001)

[**Stallman, 1981**]
Stallman, R.M. (1981). *EMACS : The Extensible, Customizable, Display Editor*, dans *ACM Conference on Text Processing*
http://www.gnu.org/software/emacs/emacs-paper.html

[Stallman, 1983]
Stallman, R.M. (1983). *Initial GNU Announcement.*
http://www.gnu.org/gnu/initial-announcement.html

[Stallman, 1985]
Stallman, R.M. (1985). *The GNU Manifesto.*
http://www.gnu.org/gnu/manifesto.html

[Stallman, 1986]
Stallman, R.M. (1986). *Lecture at KTH.* (30 octobre 1986)
http://www.gnu.org/philosophy/stallman-kth.html

[Stallman, 1987]
Stallman, R.M. (1987). *Emacs the Full Screen Editor.*
http://www.lysator.liu.se/history/garb/txt/87-1-emacs.txt

[Stallman, 1999]
Stallman, R.M. (1999) The GNU Operating System and the Free Software Movement, dans : Dibona, C., Ockman, S. and Stone, M. (Eds.). *Open Sources. Voices from the Open Source Revolution.* Sebastopol, Cal. : O'Reilly Media Inc., pp.53–70,
http://oreilly.com/catalog/opensources/book/stallman.html

[Stallman, 2000]
Stallman, R.M. (2000). *Freedom or copyright ? By legalizing the copying of e-books, we can turn copyright back into the industrial regulation it once was.*, dans *MIT Technology Review* (Mai 2000)
http://www.technologyreview.com/communications/12110/

[Stallman, 2001]
Stallman, R.M. (2001). *Free Software : Freedom and Cooperation.*
http://www.gnu.org/events/rms-nyu-2001-transcript.txt

[Stallman, 2008]
Stallman, R.M. (2008). *Vous avez dit 'Propriété intellectuelle' ? Un séduisant mirage.*
http://www.gnu.org/philosophy/not-ipr.fr.html

[Steed, 2000]
Steed, J. (2000). *C03 Section Business*, dans *Toronto Star* (9 Octobre 2000)

[Tiemann, 1999]
Tiemann, M. (1999). *Future of Cygnus Solutions : An Entrepreneur's Account*, dans *Open Sources. Voices from the Open Source Revolution*, pp.71–90
http://oreilly.com/catalog/opensources/book/tiemans.html

[Torvalds and Diamond, 2001a]
Torvalds, L. and Diamond, D. (2001a). *Il était une fois Linux. L'extraordinaire histoire d'une révolution accidentelle*. Paris : Eyrolles.

[Torvalds and Diamond, 2001b]
Torvalds, L. and Diamond, D. (2001b). *Just for Fun : The Story of an Accidental Revolutionary*. New York : Harper Collins Publisher Inc.

[Weizenbaum, 1976] Weizenbaum, J. (1976). *Computer Power and Human Reason : From Judgment to Calculation*. San Francisco : W. H. Freeman and Co..

[Young, 1994]
Young, R. (1994). *Interview with Linus, the Author of Linux*, dans *Linux Journal* (01 mars)
http://www.linuxjournal.com/article/2736

Index des noms propres