Livre como em Liberdade (2.0)

Richard Stallman e a Revolução do Software Livre

Autor original: Sam Williams

Edição revista e comentada: Richard M. Stallman (RMS)

Tradução, Interpretação e Prefácio: Maurício da Silva Maia

Licença: GNU Free Documentation License (GFDL) v1.3+

Cópia Transparente (GFDL): Disponibilizada em mauriciomaia.com.br/livre

📥 Baixar EPUB 📥 Baixar PDF 📥 Baixar Markdown

🏠 mauriciomaia.com.br

Apresentação e Prefácio da Edição em Língua Portuguesa

Prefácio do Tradutor

O movimento do software livre transformou profundamente a forma como a humanidade desenvolve, partilha e interage com a tecnologia. No centro dessa revolução cultural e tecnológica está uma das figuras mais singulares e polarizadoras do nosso tempo: Richard Stallman.

Quando Sam Williams publicou a primeira edição de *Free as in Freedom* em 2002, o ecossistema tecnológico era muito diferente. Anos mais tarde, numa reviravolta literária rara, o próprio Richard Stallman pegou na obra, corrigiu imprecisões históricas e adicionou as suas próprias notas e reflexões na primeira pessoa, resultando na versão 2.0 publicada pela GNU Press.

Embora o manifesto e a filosofia do software livre tenham uma forte base de apoiantes nos países de língua portuguesa, esta importante versão revista e atualizada da biografia permaneceu, durante mais de uma década, acessível apenas em inglês.

A decisão de traduzir, interpretar e publicar esta obra em formato físico e digital através da Amazon nasce de dois objetivos claros e práticos:

Acessibilidade: Permitir que estudantes, programadores, ativistas e curiosos da lusofonia compreendam a gênese do projeto GNU e as nuances filosóficas do *copyleft* sem a barreira do idioma.
Preservação e Difusão: Utilizar os canais modernos de distribuição impressa e digital para que o livro chegue facilmente a quem valoriza o prazer da leitura em papel ou em dispositivos dedicados, mantendo viva a memória de uma revolução que ainda molda o código que roda nos nossos computadores e servidores.
Como a própria essência deste livro é a liberdade, esta tradução respeita integralmente a licença GNU Free Documentation License (GFDL v1.3+). O trabalho aqui apresentado não me pertence de forma exclusiva: o texto em português é livre, pode ser compartilhado, estudado, modificado e reaproveitado pela comunidade, sob a única condição de manter as mesmas liberdades para os próximos leitores. Nas páginas finais e nas notas editoriais desta edição, você encontrará o link para acessar gratuitamente todos os arquivos-fonte editáveis e a cópia transparente.

Traduzir os termos técnicos e a retórica muito própria de Stallman — que sempre insistiu na distinção crucial entre *free software* (software livre, alicerçado na liberdade ética) e *open source* (código aberto, focado primordialmente no modelo corporativo de desenvolvimento) — foi um desafio gratificante.

Contudo, disponibilizar esta obra em nosso idioma hoje não é apenas um registro histórico do passado. É, acima de tudo, um posicionamento urgente diante do futuro que estamos construindo — reflexão que aprofundo a seguir, na introdução que preparei para esta edição.

Maurício da Silva Maia
Tradutor da Edição em Língua Portuguesa
mauriciomaia.com.br

Introdução: Reacendendo a Chama da Liberdade na Era da Inteligência Artificial

Por Maurício da Silva Maia

Decidi publicar esta versão em língua portuguesa com um propósito pessoal, ético e inadiável: **reacender a chama da liberdade de software, que, sob a sedução da conveniência imediata e o ritmo vertiginoso da indústria, corre hoje o risco real de se apagar.**

Vivemos um momento de inflexão sem precedentes na história da computação. Com o avanço acelerado da Inteligência Artificial, a consequência que se desenha com clareza alarmante é que a criatividade humana genuína e o pensamento crítico vão sendo postos de lado em nome de uma corrida desenfreada por pura produtividade. O programador, outrora um artesão do intelecto que compreendia as entranhas do que criava, é incentivado a se transformar em mero operador de respostas pré-fabricadas.

Mas a perda mais grave não reside apenas na técnica — reside na essência humana da colaboração.

A verdadeira liberdade na tecnologia nunca brotou do isolamento nem da eficiência fria. Ela sempre nasceu da vontade solidária de ver o próprio trabalho crescendo por meio da colaboração comunitária e de saber que a sua linha de código ajudaria outras pessoas a criarem coisas inteiramente novas. É o prazer hacker de compartilhar, de ensinar, de construir em conjunto o bem comum.

Com a onipresença dos assistentes de inteligência artificial, essa dinâmica está sendo silenciosamente desfeita. A colaboração passa a ocorrer entre o desenvolvedor isolado e a IA, e não mais entre o desenvolvedor e a sua comunidade. Cria-se, assim, uma barreira invisível para o nascimento de novos projetos colaborativos: quando cada indivíduo interage apenas com uma máquina proprietária em seu terminal fechado, o tecido comunitário que deu vida ao ecossistema aberto esgarça-se e perde sua força vital.

Ao mesmo tempo, as promessas de abertura no campo da inteligência artificial revelam-se uma miragem. O movimento recente do mercado global — marcado por aquisições estratégicas e pela centralização das grandes plataformas e repositórios comunitários sob a tutela dos gigantes do hardware e da computação em nuvem — já sinaliza sem disfarces o fechamento da colaboração no terreno dos grandes modelos de linguagem (LLMs). Aquilo que começou sob o pretexto de ser "aberto" rapidamente se curva aos interesses de monopólios que detêm as fábricas de silício e os datacenters globais.

É vital, portanto, resgatar o que as palavras realmente significam: **Software Aberto e Livre significa, essencialmente, liberdade irrestrita de uso e de plataforma.**

Se você não tem o direito de executar seu software onde quiser — se depende de uma infraestrutura em nuvem que pode alterar termos ou ser descontinuada unilateralmente, de uma GPU proprietária de um único fabricante hegemônico ou de licenças comerciais com cláusulas restritivas —, você não usufrui de liberdade real. Você é apenas um inquilino digital sujeito às vontades de corporações que controlam as chaves da sua ferramenta de trabalho e de pensamento.

É exatamente por essa razão que as ideias de **Richard Matthew Stallman** precisam ser resgatadas com vigor redobrado.

Quando Stallman se rebelou no Laboratório de IA do MIT contra o fechamento do código de uma impressora e recusou-se a assinar acordos de não-divulgação que o proibiam de ajudar seus colegas, ele formulou o princípio basilar que ecoa através das décadas:

> *« Diante da tecnologia, existem apenas duas possibilidades: ou os usuários controlam o programa, ou o programa controla os usuários. Se o programa é proprietário, ele é um instrumento de poder injusto. »*

A luta do Software Livre nunca foi meramente sobre código; sempre foi sobre soberania humana, solidariedade e justiça social. Resgatar as Quatro Liberdades Fundamentais hoje — a liberdade de executar o programa como quiser, de estudar seu funcionamento, de redistribuir cópias aos seus pares e de distribuir suas modificações para o benefício coletivo — é a única vacina duradoura contra a subserviência tecnológica e a pasteurização da criatividade diante da IA fechada.

Que a leitura desta biografia extraordinária não seja uma viagem nostálgica a uma era que já passou, mas sim uma centelha viva. Que ela desperte em você a coragem de questionar a comodidade passiva, de defender a autonomia de suas plataformas e de voltar a programar com liberdade, paixão e fraternidade comunitária.

A chama da liberdade de software precisa de novas mãos para continuar acesa. Que este livro seja o seu convite para mantê-la viva.

Maurício da Silva Maia
Organizador e Tradutor da Edição em Língua Portuguesa
mauriciomaia.com.br

Nota Editorial: A Transcriação para a Língua Portuguesa

Esta obra é a versão integral em português de ***Free as in Freedom (2.0): Richard Stallman and the Free Software Revolution***, biografia originalmente escrita pelo jornalista **Sam Williams** (lançada pela O'Reilly em 2002) e substancialmente revista, retificada e comentada pelo próprio **Richard M. Stallman** na Edição 2.0 (publicada pela Free Software Foundation em 2010).

O compromisso assumido nesta edição não foi o de uma tradução mecânica e literal, mas sim o de uma **transcriação interpretativa**: preservar com absoluto rigor o ritmo investigativo, a ironia sutil e o vigor jornalístico de Williams, ao mesmo tempo em que se garante a máxima clareza conceitual, a exatidão técnica e a naturalidade da língua portuguesa.

A Dinâmica das Duas Vozes

A Edição 2.0 apresenta um formato ímpar na biografia moderna: 1. **A narrativa de Sam Williams:** O biógrafo independente que investiga a trajetória de Stallman com fascínio, mas sem complacência, explorando seus conflitos, suas contradições pessoais e o impacto sísmico de suas escolhas. 2. **As intervenções diretas de Richard Stallman (RMS):** Inseridas ao longo dos capítulos entre colchetes demarcados como `[RMS: ...]` e em notas complementares. Stallman não interveio para censurar as opiniões do biógrafo, mas sim para corrigir imprecisões históricas de engenharia de software e refinar os argumentos éticos contra a apropriação indevida dos ideais do Projeto GNU pelo mercado.

Nesta edição, essa conversa cruzada foi meticulosamente preservada com tratamentos tipográficos diferenciados para permitir que o leitor sinta toda a tensão viva do diálogo entre biógrafo e biografado.

Critérios e Rigor Terminológico

Algumas convenções conceituais fundamentais foram adotadas nesta versão para resguardar a coerência do pensamento de Stallman:

Software Livre (*Free Software*): Em inglês, *free* confunde gratuidade (*free beer*) e liberdade (*free speech*). Na língua portuguesa, temos o termo inequívoco: **software livre**, que remete diretamente à liberdade humana e aos direitos inalienáveis da comunidade.
Software Proprietário ou Privativo (*Proprietary Software*): Expressão que designa qualquer programa que negue aos usuários as quatro liberdades essenciais, mantendo o controle sob as mãos exclusivas do detentor dos direitos.
Hacker e Hacking: Mantidos em sua pureza original do MIT — a exploração engenhosa, lúdica e virtuosa do conhecimento (*playful cleverness*) —, repudiando a vulgarização que confunde hackers com criminosos digitais.
Cracker e Cracking: Termos cunhados pelo próprio Stallman para isolar as ações de quebra maliciosa de sistemas e invasão de segurança.
Código Aberto (*Open Source*): Tratado em sua distinção filosófica real. Enquanto o "código aberto" defende a transparência de código como mera metodologia de eficiência corporativa, o "software livre" posiciona a questão no terreno da ética, dos direitos civis e da fraternidade humana.
GNU/Linux: Denominação estrita do sistema operacional livre (o sistema GNU dotado do núcleo Linux), fazendo justiça histórica aos autores e militantes que desenvolveram as ferramentas essenciais antes mesmo do nascimento do kernel de Torvalds.
Copyleft: Mantido com sua consagração internacional e com a tradução conceitual precisa: *"Todos os direitos revertidos"*.

Licença e Distribuição (GFDL 1.3+)

Fiel à filosofia que defende, esta edição é compartilhada com o mundo sob os termos da **GNU Free Documentation License (GFDL)**, Versão 1.3 ou posterior, sem Seções Invariantes, sem Textos de Capa Frontal

e sem Textos de Quarta-Capa.

Em estrito cumprimento à Seção 3 da GFDL, uma **Cópia Transparente (*Transparent Copy*)** integral desta obra (em formatos abertos legíveis por máquina: leitor HTML interativo, EPUB livre de DRM, PDF e arquivos-fonte em Markdown) está disponibilizada de forma pública, permanente e gratuita em:

👉 **https://www.mauriciomaia.com.br/livre**

Você tem a liberdade plena e irrevogável de copiar, estudar, modificar e redistribuir esta obra para quem desejar, sob a única condição de preservar as mesmas liberdades para as próximas gerações.

Boa leitura e que o código (e o conhecimento) continuem livres.

Prefácio de Richard M. Stallman

Meu objetivo com esta edição foi reunir o melhor de dois mundos: o conhecimento detalhado que tenho dos fatos e a visão externa de Sam Williams, construída a partir de suas entrevistas. Cabe ao leitor julgar até que ponto fui bem-sucedido nessa empreitada.

Li o texto publicado da edição original em inglês pela primeira vez apenas em 2009, quando fui convidado a colaborar com a tradução francesa de *Free as in Freedom*. Logo percebi que a obra exigia muito mais do que meros retoques pontuais.

Diversos fatos precisavam ser corrigidos, mas eram necessárias alterações ainda mais profundas. Williams, por não ser programador, acabou misturando e obscurecendo distinções técnicas e jurídicas essenciais — como a diferença fundamental entre alterar o código de um software já existente e, por outro lado, reimplementar as ideias desse software em um programa novo do zero. Por conta disso, a primeira edição afirmava que tanto o Gosmacs quanto o GNU Emacs haviam sido desenvolvidos a partir de modificações do Emacs original para o PDP-10 — o que não é verdade para nenhum dos dois. Da mesma maneira, descrevia equivocadamente o Linux como uma "versão do Minix". Anos mais tarde, a SCO usou exatamente essa alegação fraudulenta em seu famigerado processo contra a IBM, e tanto Linus Torvalds quanto Andrew Tanenbaum a desmentiram cabalmente.

A primeira edição também dramatizava eventos em excesso, projetando neles emoções artificiais. Afirmava, por exemplo, que eu havia "praticamente ignorado" o Linux em 1992 e que, em seguida, teria feito uma "reviravolta dramática" ao decidir apoiar o Debian GNU/Linux em 1993. Ora, tanto o meu interesse em 1993 quanto o meu desinteresse relativo em 1992 foram escolhas estritamente pragmáticas voltadas para o mesmo objetivo de sempre: concluir o sistema GNU. O lançamento do núcleo GNU Hurd em 1990 também foi uma decisão pragmática com essa exata finalidade.

Por todas essas razões, muitas passagens da primeira edição mostravam-se incorretas ou incoerentes. Era indispensável corrigi-las, mas não seria simples fazer isso com integridade sem reescrever o livro inteiro — algo indesejável por vários motivos. Chegou-se a sugerir o uso de notas explícitas para cada correção, mas, na maioria dos capítulos, a quantidade de ajustes tornaria esse recurso insuportável para o leitor. Alguns erros eram tão estruturais e frequentes que não caberiam em notas de rodapé; em outros trechos, notas intercaladas poluiriam o texto, tornando a leitura truncada e cansativa, além de serem ignoradas por leitores exaustos de descer a vista para o pé da página. Decidi, portanto, aplicar as correções diretamente no corpo do texto.

Entretanto, não tentei verificar exaustivamente todos os fatos e depoimentos alheios ao meu conhecimento direto; a maior parte deles foi mantida sob a responsabilidade de Williams.

A versão original de Williams continha diversas declarações bastante críticas a mim. Fiz questão de preservar todas elas, inserindo minhas respostas e contra-argumentos quando cabível. Não excluí nenhuma declaração, exceto no Capítulo 11, onde cortei algumas falas sobre o movimento de "código aberto" que não tinham relação com a minha vida ou com o meu trabalho. Do mesmo modo, mantive (e por vezes comentei) quase todas as interpretações do próprio Williams com críticas à minha postura, desde que não partissem de má compreensão dos fatos ou da tecnologia. Porém, corrigi livremente asserções incorretas sobre meu trabalho, meus pensamentos

e meus sentimentos. Preservei as impressões pessoais de Williams sempre que foram apresentadas como tais; por isso, a palavra "eu" na narrativa desta edição refere-se invariavelmente a Williams, exceto em notas explicitamente identificadas como "[RMS:]".

Nesta edição, o sistema operacional completo que reúne o GNU e o kernel Linux é sempre denominado **"GNU/Linux"**; o termo **"Linux"**, isolado, refere-se estritamente ao núcleo desenvolvido por Linus Torvalds — ressalvadas algumas citações diretas em que o uso impreciso é assinalado com *"[sic]"*. Para uma explicação mais aprofundada sobre por que é incorreto e injusto chamar o sistema todo de "Linux", consulte: http://www.gnu.org/gnu/gnu-linux-faq.html.

Agradeço imensamente a John Sullivan por suas inúmeras críticas e sugestões fundamentais para a revisão deste trabalho.

Prefácio de Sam Williams

Este verão marca o décimo aniversário daquela primeira troca de e-mails que desencadeou a escrita de *Free as in Freedom: Richard Stallman's Crusade for Free Software* e, por extensão, da obra que apresento agora: *Richard Stallman and the Free Software Revolution*.

Escusado dizer que o mundo mudou de forma vertiginosa ao longo dessa década.

Originalmente concebido no auge do triunfalismo norte-americano do início dos anos 2000, o eixo central do livro — a jornada profética, quase ao estilo de Jeremias, de um homem tentando alertar seus colegas desenvolvedores para a miopia ética (quando não econômica) de um modelo comercial empenhado em transformar a cultura intelectual e livre que gerou a ciência da computação em um amontoado ríspido de condomínios fechados proprietários — hoje soa quase nostálgico. Lembra os dias em que o sistema tecno-capitalista parecia operar às mil maravilhas, imune a tudo, exceto às críticas de meia dúzia de céticos isolados.

Hoje, quando desconfiar do sistema virou praticamente uma virtude cívica comum, vale a pena observar quais fios condutores da narrativa resistiram intactos ao teste desses dez anos.

Embora eu já não acompanhe a indústria de software tão de perto quanto antes, há algo que salta aos olhos hoje com uma força ainda maior: a incrível facilidade com que consumidores comuns abrem mão de nacos gigantescos de sua privacidade e de sua liberdade individual em troca de desfilar com a "plataforma" tecnológica da moda ou surfar a última onda das redes sociais.

Alguns anos atrás, eu teria apelidado esse fenômeno de "Efeito iPod" — uma reverência irônica ao sucesso sem precedentes do cofundador da Apple, Steve Jobs, em convencer tanto a indústria fonográfica quanto os ouvintes de música digital a engolirem anos de ressentimentos mútuos para se curvarem diante de um único aparelho sedutor — o iPod da Apple — e de seu regime de licenciamento asfixiante, o iTunes. Se eu estivesse vendendo essa pauta a um jornal ou revista hoje, provavelmente teria de chamá-la de "Efeito iPad" ou talvez "Efeito Kindle", tanto para acompanhar a dança das marcas quanto para registrar as transformações tectônicas que atingiram o jornalismo diário e o mercado editorial de livros eletrônicos.

Para que não pareça que estou fazendo propaganda gratuita dos nomes citados, RMS sugere que eu conceda espaço equivalente a dois sites que esmiúçam as armadilhas dessas tecnologias, sobretudo no que tange à liberdade de software. Concordei com a sugestão em nome do bom e velho direito de resposta. Os sites recomendados por ele são http://DefectiveByDesign.org e http://BadVista.org.

Independentemente do rótulo da vez, a ideia de que uma marca corporativa é a única garantia de qualidade de software em um mundo em constante mutação continua difícil de desarmar — mesmo numa época em que o valor dessas grandes corporações vive sob desconfiança crônica.

Dez anos atrás, era comum participar de conferências de tecnologia e flagrar algum veterano — Richard Stallman entre eles — discursando ou ensinando diretamente a quem quisesse ouvir uma visão arrebatadora de um caminho alternativo. Com o tempo, entendi que a missão desses pioneiros era esfregar na cara de nós, jornalistas novatos,

que as ferramentas que tanto nos orgulhávamos de dominar — Microsoft Word, PowerPoint, Internet Explorer, só para citar o menu básico de uma única empresa dominante — não passavam de uma sombra pálida e empobrecida do monumento imponente que os arquitetos originais do computador pessoal haviam sonhado erguer.

Hoje em dia, vivemos quase o inverso. O monumento virou um ecossistema labiríntico e desordenado, uma selva fervilhando de ideias, mas com pouquíssimos nichos estáveis para um crescimento sustentável. Embora ainda existam hackers dispostos a protestar contra os eternos defeitos estruturais do Windows Vista, a mão de ferro da Apple sobre a App Store do iPhone ou as convenientes e elásticas redefinições do Google para o lema *"não seja mau"* (*don't be evil*), cada novo ano despeja no mercado uma safra fresca de consumidores "nativos digitais" perfeitamente confortáveis em terceirizar suas vidas e confiar nos rumos ditados pelas grandes corporações nesse território hobbesiano. Talvez isso aconteça porque boa parte dos problemas que antes faziam o uso de um computador de mesa ser uma experiência de ranger os dentes foi asfaltada e aplainada justamente com a ajuda do software livre.

Enfim. Conforme o software de consumo foi ficando mais estável e confiável, a corrida para se antecipar aos caprichos do público jogou os desenvolvedores nos braços do capital financeiro com uma força ainda maior. Não estou dizendo que a ética hacker tenha desaparecido ou que tenha definhado a ponto de virar poeira. O que quero dizer é que duvido muito que o programador que escreveu o algoritmo do Facebook para associar palavras-chave a páginas patrocinadas — mesmo com uma margem de erro semântico de 80% — passe muito tempo ao volante do seu novo Porsche se lamentando pelo que aquele código poderia ter sido se os engravatados da diretoria não tivessem metido o bedelho.

É verdade que milhões de pessoas hoje rodam softwares predominantemente livres em seus computadores — e muitos usam exclusivamente programas livres. No entanto, para o usuário comum, palavras como "software" e "computador" foram ficando cada vez mais abstratas e distantes. Qualquer celular básico de 2010 dá um banho de funcionalidade em um notebook do ano 2000. E mesmo assim, na hora de comprar um aparelho novo, quantos usuários dedicam um único pensamento ao sistema operacional que torna tudo aquilo viável? A esmagadora maioria decide a compra quase que exclusivamente pela quantidade de aplicativos disponíveis, pela estabilidade do sinal da operadora e, acima de tudo, pelo valor da fatura mensal.

Convencer um consumidor nessa situação a enxergar suas escolhas tecnológicas pelo prisma da liberdade individual, e não pelo mero conforto imediato, tornou-se — se não impossível — uma tarefa incomparavelmente mais complexa.

Diante de um preâmbulo tão cético, por que raios alguém ainda deveria ler este livro?

Vejo duas razões primordiais.

A primeira é de ordem pessoal. Como relatei no epílogo da primeira edição, Richard e eu nos desentendemos e rompemos relações pouco antes da publicação original. A culpa, em grande medida, foi minha. Eu havia colaborado estreitamente com Richard para garantir que meu esboço biográfico não ferisse os princípios do software livre — um esforço pioneiro do qual muito me orgulho, pois transformou *Free as in Freedom* em uma das primeiras obras publicadas sob a Licença de Documentação Livre GNU (GFDL). Contudo, cortei a cooperação de maneira abrupta na hora de editar o material final e incorporar a alentada lista de correções e pedidos de esclarecimento enviados por ele.

Embora na época eu tenha me refugiado atrás do escudo da minha independência editorial e da objetividade jornalística, passei a lamentar amargamente o fato de não ter pedido mais prazo à editora, a O'Reilly and Associates. Como a O'Reilly já havia aceitado minha exigência mais radical — publicar o livro sob a GFDL — e já vinha tolerando uma torrente de alterações de última hora da minha parte, fiquei com receio de esticar demais a corda.

Nos anos seguintes ao lançamento, eu costumava justificar a minha decisão lembrando que a GFDL, exatamente como a GPL no universo do software, concede a qualquer leitor o direito soberano de modificar o livro e redistribuí-lo em livre concorrência. Como dizia Ernest Hemingway, *"o primeiro rascunho de qualquer coisa é uma droga"*. Se Stallman ou a comunidade hacker considerassem meu livro apenas um primeiro rascunho imperfeito, bem, ao menos eu havia poupado a eles o esforço de começar a obra do zero.

Agora que Richard finalmente realizou o que equivale a uma reescrita substancial, só me resta ser fiel à minha coerência de outrora e aplaudir o resultado. Eu saúdo esta edição com entusiasmo. Minha única esperança é que, como o fôlego de Stallman não dá o menor sinal de esmorecer, novos registros e relatos continuem sendo incorporados à história.

Antes de passar à segunda razão, vale mencionar que um dos frutos mais felizes deste projeto foi a reabertura dos canais de comunicação por e-mail entre Richard e eu. Esse contato me reconectou com o estilo afiado e cortante da escrita de Stallman.

Há uma anedota saborosa e bastante reveladora para quem já travou debates textuais com Richard: ao discutirmos o trecho em que registro o momento exato em que ele perde as estribeiras preso no engarrafamento da hora do rush em Kihei, no Maui — passagem que serviu de matéria-prima para o antigo Capítulo 7 (*"Uma Breve Jornada pelo Inferno Hacker"*), mencionei a ele uma queixa recorrente dos críticos da obra: a de que aquele episódio parecia deslocado, uma espécie de perfil de revista intrusivo quebrando o fôlego de uma biografia em livro. Disse a Richard que ele tinha total liberdade para limpar o episódio se quisesse, mas pontuei que mantive a cena por duas razões. Primeiro, porque ela oferecia um vislumbre raro do temperamento vulcânico de Stallman — algo sobre o qual eu havia sido fartamente alertado, mas que ainda não tinha presenciado ao vivo. Segundo, porque aquela cena caótica guardava um valor metafórico flagrante. Daí o título do capítulo.

Para minha surpresa, Stallman concordou com ambos os argumentos. Sua bronca se concentrava em duas palavras pontuais da transcrição. Em dado momento, eu o citava acusando o motorista que liderava o nosso comboio de estar nos conduzindo "deliberadamente" para uma rua sem saída — uma acusação que, se literal, sugeria um grau de malícia incompatível com a realidade da situação. Sem a gravação em áudio para tirar a dúvida — na hora eu contava apenas com meu bloco de anotações —, admiti que era muito provável que eu tivesse me equivocado na escolha das palavras de Stallman, tornando a frase mais agressiva do que ele pretendia.

Além disso, em outro ponto, Stallman questionou a frase em que ele aparecia soltando o palavrão *"fucking"*. Mais uma vez, eu não tinha a fita do momento gravada, mas respondi afirmando que me lembrava com absoluta nitidez de um festival formidável de impropérios — um daqueles rompantes típicos de quem cresceu em Nova York —, e que eu sustentava aquela lembrança.

A resposta de Richard por e-mail, chegada no dia seguinte, rebateu a questão com uma sutileza desconcertante que me obrigou a reler a mensagem anterior com muito mais cuidado. Descobri que Stallman não estava contestando o uso do palavrão *"fuck"*, mas especificamente o sufixo *"-ing"* na citação:

"Uma das razões de eu duvidar dessas palavras", escreveu Stallman, *"é que elas pressupõem o uso de 'fucking' como advérbio. Eu nunca falei desse jeito na minha vida. Portanto, tenho certeza de que as palavras foram alteradas."*

Touché.

A segunda razão pela qual acredito que vale a pena ler este livro nos devolve ao tema que abriu este prefácio: a distância abissal entre o futuro com que nos deparamos em 2010 e aquele que vislumbrávamos com esforço lá atrás, em 2000. Serei muito franco: como aconteceu com tantos norte-americanos (e cidadãos do mundo todo), minha visão de mundo foi profundamente abalada pelos atentados de 11 de setembro de 2001. Tanto que, pouco tempo depois do lançamento da primeira edição, minha atenção se distanciou bruscamente do movimento do software livre e da incansável batalha de Stallman para mantê-lo nos trilhos. Embora eu tenha continuado acompanhando as grandes tendências e os debates cruciais, o drama cotidiano em torno de padrões de interoperabilidade, direitos autorais e patentes de software virou para mim algo quase invisível — o equivalente digital àquelas enfadonhas atas de reunião da companhia de saneamento publicadas nos classificados do jornal local.

> **[RMS:** Os ataques de setembro de 2001, que não voltam a ser mencionados ao longo do livro, merecem um breve comentário aqui. Longe de "mudar tudo", como muitos bradam aos quatro ventos, os ataques, na realidade, mudaram muito pouco nos Estados Unidos: os patifes que odeiam as nossas liberdades continuam no poder. A única diferença substancial é que agora eles têm a desculpa do "terrorismo" para aprovar leis que nos despojam dessas liberdades. Para mais reflexões a esse respeito, consulte as notas políticas em http://stallman.org.**]**

Esse meu distanciamento foi lamentável, sobretudo porque, passados dez anos, começo a enxergar este século XXI com muito mais clareza. Se eu tivesse de apresentar esse conceito a um editor em tom de manchete, eu o chamaria de *"O Século dos Processos"*.

Com isso, quero dizer que vivemos um momento singular da história no qual, despindo-nos de todo o cinismo, o poder de transformar a realidade de fato desceu ao nível do cidadão comum. A pegadinha, é claro, é que esse poder que pertence a você pertence simultaneamente a todo o resto da humanidade. Se em épocas passadas era possível emplacar uma reforma simplesmente conquistando a simpatia de meia dúzia de burocratas bem posicionados nos bastidores do poder, hoje o reformador precisa alinhar um campo vetorial caótico de ideias, disputas e interesses conflitantes. Em suma, ser um reformador eficaz nos dias de hoje exige muito mais do que um fôlego titânico e a disposição de pregar no deserto por uma década ou mais: exige saber articular propostas sólidas, escaláveis e duradouras — exige saber derrotar o sistema jogando pelas regras do próprio jogo.

Em todos esses quesitos, não hesito em afirmar que Richard M. Stallman, se não for o arquétipo definitivo, é no mínimo o modelo pioneiro e primal dessa estirpe de reformador bem-sucedido.

Enquanto alguns lamentam um futuro em que qualquer problema parece demandar décadas de audiências públicas e comitês burocráticos arrastados para alcançar alguma melhora, eu, particularmente, considero a alternativa oposta — um mundo tão suscetível à ação vertiginosa de indivíduos ou pequenos grupos que qualquer justiceiro de ocasião decida testar essa velocidade na marra — assustadora demais para sequer cogitar.

Em resumo: se você é o tipo de pessoa que, como eu, deseja ver o século XXI trilhar um caminho menos sangrento do que o século XX, a "reunião do comitê de saneamento" — sob suas mais diversas e exasperantes roupagens — é exatamente o campo onde essa guerra está sendo travada. Como bem insinua a epígrafe inspirada em Virgílio na abertura do primeiro capítulo, guardo a esperança de que esta obra possa, de algum modo, funcionar como uma espécie de poema épico para a Era da Internet. Construída ao redor de uma figura central heroica e cheia de arestas, é natural e saudável que a assinatura original de seu autor vá se dissipando com o passar dos anos.

Com esse espírito, encerro este prefácio como sempre fiz: estendendo o convite a qualquer leitor que queira propor correções, sugestões e melhorias ao texto. O Apêndice B detalha todos os seus direitos como leitor para submeter alterações ou até mesmo criar sua própria versão derivada deste livro. Se preferir encaminhar suas notas diretamente a Richard ou a mim, os canais de contato estão disponíveis no site da Free Software Foundation.

Até lá, boa leitura e vida longa à liberdade!

Sam Williams
Staten Island, Nova York, EUA

Capítulo 1: Por Falta de uma Impressora

> *Temo os gregos, mesmo quando trazem presentes.*
> *— Virgílio, Eneida*

A impressora nova estava atolada. De novo.

Richard M. Stallman, programador da equipe do Laboratório de Inteligência Artificial do Massachusetts Institute of Technology (o famoso AI Lab do MIT), descobriu o enguiço da pior maneira possível. Uma hora após despachar um arquivo de cinquenta páginas para a impressora a laser do departamento, Stallman, então com 27 anos, interrompeu uma sessão produtiva de trabalho para buscar seus documentos. Ao chegar lá, encontrou míseras quatro páginas na bandeja. Para piorar a frustração, as folhas pertenciam a outro usuário — o que significava que o trabalho de Stallman e o resto da impressão alheia continuavam presos em algum desvão do encanamento eletrônico da rede local do laboratório.

Ficar esperando por máquinas é um daqueles ossos do ofício inevitáveis para qualquer programador, de modo que Stallman costumava encarar esses contratempos com certa paciência. Ainda assim, há uma distância abissal entre esperar que uma máquina trabalhe e ter de ficar de prontidão servindo a uma máquina emperrada. Não era a primeira vez que ele se via obrigado a plantar diante do aparelho, vigiando as folhas saírem uma a uma. Para alguém que passava a maior parte dos dias e das noites aprimorando o rendimento de computadores e dos softwares que os controlavam, a reação instintiva era abrir a carcaça da máquina, fuçar suas entranhas e ir direto à raiz do problema.

Infelizmente, os talentos de Stallman como programador não alcançavam os domínios da engenharia mecânica. Enquanto os documentos recém-impressos iam caindo na bandeja, Stallman aproveitou para refletir sobre maneiras mais inteligentes de contornar os atolamentos crônicos de papel.

Havia quanto tempo a equipe do AI Lab tinha recebido aquela impressora de braços abertos? Stallman se pôs a pensar. O equipamento fora uma doação da Xerox Corporation: um protótipo de ponta, adaptado a partir de uma copiadora rápida da marca. A diferença crucial era que, em vez de tirar cópias físicas, ela recebia instruções e dados transmitidos pela rede de computadores, transformando essas informações em documentos com acabamento profissional. Criada pelos brilhantes engenheiros do lendário Centro de Pesquisas da Xerox em Palo Alto (o Xerox PARC), ela representava, pura e simplesmente, um vislumbre pioneiro da revolução da editoração eletrônica que varreria a indústria da informática até o final daquela década.

Movidos pelo impulso nato de brincar com o que houvesse de mais moderno em matéria de tecnologia, os programadores do AI Lab logo integraram o novo aparelho à sofisticada infraestrutura computacional do laboratório. Os resultados causaram encantamento imediato. Ao contrário da antiga impressora, a máquina da Xerox era veloz como um raio. As páginas voavam à velocidade de uma por segundo, reduzindo uma tarefa de vinte minutos para meros dois minutos. Além disso, a precisão era cirúrgica: círculos saíam redondos de verdade, e não ovais; linhas retas mantinham-se impecavelmente retas, em vez de parecerem ondas senoidais trêmulas.

Era, para todos os efeitos, um presente bom demais para recusar.

Porém, bastou o equipamento entrar na rotina do laboratório para que seus defeitos começassem a brotar. O pior deles era a facilidade irritante com que o papel enroscava. Os programadores com veia de engenharia logo mataram a charada: concebida originalmente como copiadora, a máquina pressupunha a presença física constante de um operador humano ao seu lado. Partindo da premissa de que esse operador estaria sempre ali para puxar uma folha presa caso houvesse engasgo, os engenheiros da Xerox concentraram seus esforços na solução de outros gargalos técnicos. Em termos de engenharia, a diligência do usuário estava embutida no projeto original do sistema.

Ao adaptarem a máquina para funcionar como impressora de rede, contudo, os engenheiros da Xerox alteraram a relação entre o homem e a máquina de forma sutil, porém profunda. Em vez de subordinar o aparelho a um único operador postado diante dele, colocaram-no a serviço de uma população inteira de usuários dispersos em rede. Em vez de ficar em cima do equipamento, o sujeito em uma ponta do prédio enviava seu comando de impressão por meio de um longo revezamento em cadeia de máquinas, confiando que o documento chegaria são e salvo ao destino final. Só quando descia até a sala para apanhar as folhas é que ele se dava conta de quão pouco havia sido realmente impresso.

Stallman esteve longe de ser o único habitante do AI Lab a reparar no problema, mas foi ele quem bolou o remédio. Anos antes, para a impressora anterior do laboratório, Stallman havia resolvido um transtorno semelhante alterando o software que a controlava — instalado em um pequeno PDP-11 — e o próprio Incompatible Timesharing System (ITS) que rodava no computador principal, o PDP-10. Ele não tinha como impedir que o papel enroscasse na mecânica da máquina, mas podia perfeitamente inserir linhas de código para fazer o PDP-11 monitorar a impressora em intervalos regulares e reportar qualquer travamento de volta ao PDP-10. Stallman também inseriu instruções no PDP-10 para avisar automaticamente cada usuário que estivesse com trabalhos pendentes na fila: "A impressora emperrou; por favor, dê um pulo lá para consertar". Como o alerta chegava direto às pessoas mais interessadas em ver a fila andar, a probabilidade de alguém correr para destravar o aparelho num instante era altíssima.

Como solução, o *hack* de Stallman foi oblíquo, mas de uma elegância primorosa. Não consertava o lado mecânico da máquina, mas realizava o que havia de melhor ao fechar o ciclo de informação entre o usuário e o equipamento. Graças a um punhado de linhas de código, os pesquisadores do AI Lab economizavam os dez ou quinze minutos que antes perdiam toda semana indo e voltando à toa para conferir a bandeja. Em termos de programação, o remendo de Stallman aproveitava a inteligência amplificada da rede como um todo.

— Se você recebia aquela mensagem, não podia presumir que outra pessoa fosse resolver — relembra Stallman, reconstituindo a lógica do sistema. — Você tinha que ir até a impressora. Um ou dois minutos depois da pane, as duas ou três pessoas notificadas chegavam juntas diante da máquina. E, daquele pequeno grupo, pelo menos uma delas quase sempre sabia exatamente como desatar o nó.

Soluções engenhosas desse tipo eram a marca registrada do AI Lab e de sua fauna nativa de programadores. Na verdade, os melhores cérebros do laboratório desdenhavam o rótulo convencional de "programador", preferindo o termo bem mais livre e informal de **hacker**[1]. O termo abraçava uma infinidade de atividades — desde brincadeiras espirituosas e trotes criativos até o aperfeiçoamento minucioso de sistemas e programas existentes. Implícita no título, no entanto, residia a boa e velha noção de engenhosidade prática norte-americana. Para um hacker, escrever um programa que funcionasse era apenas o ponto de partida. O verdadeiro desafio — e a chance

de exibir sua destreza e impressionar os colegas — consistia em tornar o programa incrivelmente rápido, enxuto, poderoso, elegante ou de alguma forma brilhante.

Empresas como a Xerox costumavam adotar a política de doar seus lançamentos (e softwares) aos redutos onde os hackers costumavam se aglomerar. Afinal, se os hackers usassem seus produtos, aumentavam as chances de irem trabalhar para essas companhias no futuro. Nas décadas de 1960 e no início dos anos 1970, esses especialistas frequentemente desenvolviam programas e melhorias que os próprios fabricantes depois redistribuíam com orgulho aos seus demais clientes.

Quando Stallman constatou a tendência da nova impressora a laser a engasgar, seu primeiro reflexo foi aplicar o velho remendo, o consagrado "hack", à nova máquina. Ao procurar os arquivos do software da Xerox, porém, deu de cara com uma descoberta perturbadora: a impressora não trazia nenhum software — pelo menos nada que Stallman ou qualquer outro programador conseguisse ler. Até então, a grande maioria dos fabricantes mantinha a cortesia elementar de fornecer os arquivos com o **código-fonte** — arquivos de texto legíveis por seres humanos que documentam passo a passo as instruções dadas à máquina. Naquele caso específico, a Xerox havia entregue o programa compilado, exclusivamente em formato binário. Se os programadores abrissem os arquivos, a única coisa que encontrariam seria um oceano interminável de uns e zeros: puro jargão indecifrável de máquina.

Existem programas chamados "descompiladores" ou "desassembladores" que convertem esses uns e zeros de volta em instruções de máquina de baixo nível, mas decifrar o que cada uma dessas instruções faz na prática é uma tarefa hercúlea e exaustiva, conhecida como "engenharia reversa". Desvendar aquele programa por engenharia reversa exigiria mais tempo do que cinco anos inteiros de papel atolado na impressora. Stallman não estava desesperado a esse ponto; engoliu a frustração e deixou o problema de lado.

A postura mesquinha da Xerox colidia de forma gritante com a convivência comunitária dos hackers. Por exemplo: para programar o PDP-11 da antiga impressora e outro PDP-11 responsável pelos terminais de vídeo, o AI Lab precisava de um "montador cruzado" (*cross-assembler*) para construir programas para o PDP-11 a partir do computador central PDP-10. Os hackers do laboratório poderiam muito bem ter escrito um do zero, mas Stallman, então estudante em Harvard, encontrou um programa com essa exata função no laboratório de informática de lá. O software havia sido feito para rodar na mesma família de computadores, o PDP-10, embora sob um sistema operacional diferente. Stallman nunca soube quem fora o autor da obra, já que o código-fonte não trazia assinatura. Mas copiou o arquivo e o levou para o AI Lab. Em seguida, alterou o código-fonte para que funcionasse redondo no Incompatible Timesharing System (ITS) do laboratório. Sem dramas e sem burocracia, o AI Lab ganhou a ferramenta de que precisava para sua infraestrutura. Stallman ainda foi além: acrescentou recursos que não existiam na versão original, tornando o utilitário ainda mais robusto.

— Acabamos usando esse programa durante vários anos — lembra Stallman.

Para a mentalidade de um programador dos anos 1970, aquela troca era o equivalente exato a bater na porta do vizinho para pegar uma furadeira emprestada ou pedir uma xícara de açúcar. Com uma diferença fulcral: ao pegar uma cópia do software para o AI Lab, Stallman não havia privado o dono original de continuar usando o seu programa. Pelo contrário: outros hackers saíram ganhando no processo, já que Stallman havia introduzido novas funções que qualquer um tinha a liberdade de pegar emprestado de volta. Ele se lembra, por exemplo, de um programador da empresa de engenharia Bolt, Beranek & Newman (BBN) que pegou o programa emprestado. O colega fez o software rodar no Twenex (TOPS-20), adicionou novos recursos e Stallman, tempos depois, reintegrou

tudo ao repositório central do AI Lab. Os dois decidiram manter uma versão única e colaborativa, com o código preparado para rodar tanto no ITS quanto no Twenex, à escolha do usuário.

— Um programa ia se desenvolvendo da mesma maneira que uma cidade se desenvolve — resume Stallman, recordando a atmosfera do laboratório. — Partes velhas eram substituídas e reconstruídas. Novas alas eram puxadas. Você podia bater o olho numa rotina específica e dizer: "Hum, pelo estilo das instruções, vejo que este trecho foi feito no início dos anos 60, e esta outra parte foi escrita em meados da década de 70".

Por meio desse mecanismo espontâneo de acumulação intelectual, os hackers do AI Lab e de outros centros ergueram obras monumentais. Nem todo programador que participava daquele ambiente se autodenominava hacker, mas a grande maioria compartilhava da mesma ética de Richard Stallman: se um programa ou remendo serviu para resolver o seu problema, era bom o bastante para resolver o problema do seu próximo. Por que não compartilhá-lo, nem que fosse pelo simples desejo de gerar bom carma?

Esse ecossistema de cooperação fraterna, entretanto, vinha sendo minado gradualmente pelo segredo comercial e pela ganância corporativa, gerando bizarros híbridos de sigilo e partilha. Por exemplo: pesquisadores da Universidade da Califórnia em Berkeley haviam construído um sistema operacional poderoso chamado BSD, baseado no sistema Unix que haviam obtido da AT&T. Berkeley distribuía o BSD cobrando apenas o custo de gravação da fita magnética, mas só fornecia essas fitas para instituições de ensino que apresentassem uma licença prévia de código-fonte de 50 mil dólares, comprada diretamente da AT&T. Os hackers de Berkeley continuavam compartilhando o máximo que a AT&T tolerava, sem enxergar ainda a contradição latente entre as duas posturas.

Do mesmo modo, Stallman estava aborrecido pelo fato de a Xerox não ter fornecido o código-fonte da impressora, mas ainda não propriamente furioso. Jamais lhe passou pela cabeça pedir uma cópia à empresa:

— Eles já tinham nos doado uma impressora a laser novinha — pondera Stallman. — Eu não tinha como bater o pé dizendo que nos deviam mais coisas. Além disso, dei como certo que a ausência do código-fonte refletia uma decisão deliberada e que pedir a eles que mudassem de ideia seria pura perda de tempo.

Tempos depois, contudo, correu uma notícia animadora: diziam que um pesquisador do departamento de ciência da computação da Carnegie Mellon University (CMU) tinha conseguido uma cópia do código-fonte do software daquela impressora.

A simples menção à Carnegie Mellon, porém, não trazia bons presságios. Em 1979, Brian Reid, um estudante de doutorado daquela universidade, havia escandalizado a comunidade ao se recusar a compartilhar seu programa de formatação de textos, batizado de Scribe. O Scribe fora o primeiro formatador a incorporar comandos orientados à semântica desejada (como "enfatize esta palavra" ou "este parágrafo é uma citação"), em vez de se perder em detalhes gráficos de baixo nível ("ponha esta palavra em itálico" ou "estreite as margens deste bloco"). Em vez de liberar o código para os colegas, Reid vendeu o Scribe para uma empresa de software de Pittsburgh chamada Unilogic. Concluindo sua pós-graduação, Reid alegou que apenas procurava uma maneira de descarregar o software nas mãos de uma equipe que se comprometesse a impedir que ele caísse no domínio público — por que cargas d'água alguém consideraria o domínio público um desfecho indesejável é algo que jamais ficou claro. Para adoçar o negócio, Reid ainda aceitou embutir uma série de rotinas condicionadas ao tempo — as famosas "bombas-relógio", no jargão dos programadores — que desativavam sumariamente cópias livres do programa após o prazo de noventa dias. Para evitar o bloqueio, os usuários tinham de pagar à empresa, que fornecia uma chave para desarmar a armadilha interna.

Para Stallman, aquilo tinha sido uma traição pura e simples à ética hacker. Em vez de honrar o princípio da partilha mútua, Reid havia plantado um dispositivo para forçar programadores a pagarem pelo direito elementar de acessar a informação. Ainda assim, na época, Richard não matutou muito sobre a gravidade da coisa, até porque quase não usava o Scribe.

A Unilogic chegou a ceder uma cópia gratuita do programa para o AI Lab usar, sem mencionar uma única palavra sobre a bomba-relógio. O formatador funcionou bem por um tempo, até que, em um belo dia, um usuário reclamou que o Scribe havia parado de responder do nada. O hacker de sistemas Howard Cannon passou horas a fio escovando os binários do programa até localizar o gatilho da armadilha e aplicar um remendo que a extirpou. Cannon ficou indignado e não poupou palavras para desabafar com os colegas a sua revolta contra a Unilogic por ter feito o laboratório perder tempo precioso por causa de um bug proposital.

Alguns meses depois, Stallman teve um compromisso de trabalho que o levou ao campus da Carnegie Mellon. Durante a estada, fez questão de ir atrás do sujeito que, segundo diziam, guardava o código-fonte da impressora da Xerox. Por sorte, o homem estava em sua sala.

Numa abordagem direta de engenheiro para engenheiro, a conversa foi cordial, mas sem rodeios. Após se apresentar brevemente como visitante vindo do MIT, Stallman pediu uma cópia do código-fonte da impressora a laser para poder modificá-lo e corrigir os gargalos da máquina. Para sua absoluta consternação, o pesquisador se recusou a entregar o arquivo.

— Ele me disse na cara que havia prometido não me dar uma cópia — relembra Stallman.

A memória humana é um terreno caprichoso. Vinte anos após o ocorrido, o arquivo mental de Stallman apresenta algumas lacunas. Ele já não se lembra do motivo exato daquela viagem nem da época do ano em que ela aconteceu; tampouco guarda na lembrança quem estava do outro lado daquela mesa. Segundo Brian Reid, o interlocutor mais provável de Stallman teria sido Robert Sproull, ex-pesquisador do Xerox PARC e, anos mais tarde, diretor dos Sun Laboratories, o braço de pesquisas da Sun Microsystems. Durante a década de 1970, Sproull havia sido o principal desenvolvedor do software da impressora a laser no Xerox PARC. Por volta de 1980, assumiu uma cadeira de pesquisador na Carnegie Mellon, onde continuou tocando projetos ligados ao desenvolvimento da impressora, entre outras pesquisas.

Quando questionado diretamente sobre o episódio, contudo, Sproull não se lembra de nada. "Não tenho como fazer um relato factual", respondeu Sproull por e-mail. "Não guardo lembrança alguma desse incidente."

"O código que Stallman estava pedindo era o que havia de mais avançado, um software de ponta que Sproull havia desenvolvido cerca de um ano antes de ir para a Carnegie Mellon", relembra Reid. Se foi isso mesmo, é possível que tenha havido um mal-entendido técnico: Stallman queria apenas o código da versão que o MIT já vinha usando há bastante tempo, e não uma versão experimental mais recente. Mas a questão de qual versão estava em jogo sequer chegou a ser ventilada naquele breve e tenso diálogo.

Ao discursar para plateias pelo mundo afora, Stallman sempre cita esse episódio como um divisor de águas, frisando que a recusa do pesquisador em compartilhar o código-fonte decorria de um **acordo de não divulgação** (*non-disclosure agreement*, ou **NDA**): um contrato formal assinado entre ele e a Xerox Corporation que garantia acesso privilegiado ao código do software em troca do compromisso de guardá-lo sob estrito sigilo. Hoje um instrumento banal no mundo corporativo, o termo de confidencialidade era uma novidade absoluta na

época, reflexo tanto do valor comercial que a impressora a laser representava para a Xerox quanto do valor estratégico das informações necessárias para fazê-la funcionar. "A Xerox estava tentando transformar a impressora a laser em um produto de prateleira", recorda Reid. "Teria sido uma loucura comercial entregar o código-fonte de mão beijada."

Para Stallman, contudo, aquele NDA significava algo completamente diferente. Significava a recusa deliberada por parte de um cientista da computação em fazer parte de uma comunidade fraterna que, até então, encorajava os programadores a enxergarem o software como um bem comum. Stallman se sentiu como um camponês que, ao ver secar o canal de irrigação que abastecia sua aldeia havia séculos, sobe o leito do riacho apenas para encontrar uma imponente represa hidrelétrica novinha em folha com o logotipo da Xerox estampado no concreto.

A ficha de que a Xerox havia coagido um colega de profissão a se submeter àquele regime artificial de sigilo forçado demorou um pouco para cair. No primeiro instante, Stallman só conseguiu processar a recusa no plano estritamente pessoal:

— Fiquei tão furioso que não consegui nem articular o que estava sentindo. Apenas dei meia-volta e saí da sala sem dizer mais uma única palavra — recorda Stallman. — Talvez até tenha batido a porta; não lembro. A única coisa de que me lembro com clareza é da vontade incontrolável de sair dali. Fui até aquela sala esperando encontrar cooperação mútua; não havia passado pela minha cabeça como reagir caso ele recusasse. Quando ele disse "não", fiquei mudo de choque, além de profundamente desapontado e revoltado.

Vinte anos depois, as marcas daquela indignação ainda não se apagaram. Stallman apresenta o incidente como o momento em que foi confrontado pela primeira vez com um dilema ético incontornável — ainda que não tenha sido o único solavanco pelo caminho. Nos meses seguintes, uma sucessão de acontecimentos se abateria sobre Stallman e sobre a comunidade hacker do AI Lab com tamanha virulência que aqueles trinta segundos de constrangimento em uma sala remota da Carnegie Mellon pareceriam café pequeno por comparação. Não obstante, quando se trata de mapear os pontos de inflexão que transformaram Stallman de um hacker solitário e desconfiado de autoridades centrais no ativista combativo que levou os ideais de Liberdade, Igualdade e Fraternidade para o universo do software, ele sempre destaca aquele encontro na CMU com reverência especial.

— Foi o meu primeiro contato com um acordo de não divulgação, e ele me ensinou de imediato que termos de confidencialidade fazem vítimas — diz Stallman com firmeza. — E, naquele caso, a vítima fui eu. O meu laboratório e eu fomos as vítimas.

Mais tarde, Stallman detalharia o raciocínio:

— Se ele tivesse se recusado a cooperar comigo por uma implicância pessoal, isso não teria levantado um problema de princípios. Eu poderia simplesmente achá-lo um sujeito antipático e vida que segue. Mas o fato de a recusa ter sido impessoal — o fato de ele ter prometido com antecedência ser egoísta e recusar ajuda não apenas a mim, mas a qualquer ser humano — elevava aquilo à condição de uma grave questão ética.

Embora episódios anteriores já houvessem despertado seu incômodo, Stallman afirma que só depois do choque na Carnegie Mellon percebeu que a maré comercial começava a invadir e poluir uma cultura que ele sempre considerara sagrada:

— Eu já cultivava a convicção íntima de que o software devia ser compartilhado, mas ainda não sabia direito como articular esse pensamento. Minhas ideias não estavam organizadas com a clareza necessária para expressá-las de modo sintético para o mundo. Foi a partir daquela experiência que comecei a discernir qual era a raiz do problema e a dimensão real do que estava em jogo.

Na condição de programador de elite em uma das instituições científicas mais prestigiadas do planeta, Stallman até então vinha ignorando com relativa facilidade os pactos e concessões que seus colegas faziam com empresas privadas, desde que isso não interferisse em seu próprio trabalho. Até a chegada daquela impressora da Xerox, ele se contentava em olhar de cima e com desdém as máquinas travadas e os programas indigestos que os outros usuários de computador tinham de engolir a contragosto no mercado.

Porém, agora que a impressora a laser havia fincado suas garras dentro da rede interna do AI Lab, o cenário mudara de figura. A máquina imprimia lindamente — tirando os engasgos de papel —, mas a capacidade de adaptar o software de acordo com as preferências individuais ou com as necessidades da comunidade havia sido confiscada. Sob a ótica da indústria de informática, o software fechado da impressora representava apenas uma mudança natural de estratégia comercial: programas haviam se tornado ativos valiosos demais, e as corporações já não viam motivos para tornar o código-fonte público — especialmente quando a divulgação permitia que concorrentes duplicassem suas inovações a custo zero. Sob a ótica de Stallman, contudo, a máquina era nada menos que um **Cavalo de Troia**. Após uma década de fracassos das empresas em dobrar a resistência hacker, o software que os usuários não podiam alterar nem redistribuir — que no futuro seria classificado pelos ativistas como software **"proprietário"** — havia conseguido uma cabeça de ponte dentro do santuário do AI Lab através do mais pérfido dos disfarces: viera travestido de presente.

Saber que a Xerox havia oferecido a outros pesquisadores privilégios adicionais em troca da promessa de sigilo causava espécie, mas Stallman é honesto o bastante para admitir que, se tivesse sido exposto a um acordo de toma-lá-dá-cá quando era mais jovem, talvez ele próprio pudesse ter aceitado a oferta da empresa. O choque na Carnegie Mellon, porém, operou como um antídoto radical contra qualquer lassidão moral. Não apenas forneceu o combustível de indignação indispensável para enxergar propostas semelhantes com desconfiança visceral, mas também o forçou a inverter a perspectiva da mesa: e se, em um belo dia, um colega hacker batesse à sua porta pedindo uma cópia de código e coubesse a Stallman o papel de negar a ajuda?

— Quando alguém me convidou para trair todos os meus companheiros assinando um papel daqueles, lembrei na hora da fúria que senti quando alguém fez exatamente isso comigo e com o meu laboratório inteiro — enfatiza Stallman. — Então respondi: "Muito obrigado por me oferecer este belo pacote de software, mas não posso aceitá-lo sob as condições de sigilo que vocês exigem; prefiro ficar sem ele".

Foi uma lição que Stallman carregaria consigo ao longo dos turbulentos anos 1980 — uma década vertiginosa durante a qual muitos de seus colegas do MIT abandonariam o laboratório para assinar seus próprios termos de confidencialidade no mercado corporativo. Aqueles programadores consolavam suas consciências dizendo a si mesmos que o sigilo era um mal necessário para trabalhar nos projetos mais avançados da época. Para Stallman, porém, o acordo de confidencialidade contaminava a própria legitimidade moral de qualquer projeto: que valor tem uma pesquisa tecnicamente deslumbrante se o seu propósito deliberado é ser ocultada e negada à comunidade humana?

Como Stallman logo descobriria na pele, rejeitar essas propostas exigia muito mais do que simples sacrifício material. Exigia isolar-se deliberadamente de companheiros hackers que, embora partilhassem de uma aversão

parecida pelo sigilo, costumavam modular esse desgosto com uma flexibilidade moral bem mais elástica. Recusar o código-fonte a quem precisa dele, concluiu Stallman, não era apenas trair a missão científica e colaborativa que impulsionara o avanço da computação desde o fim da Segunda Guerra Mundial: era uma violação frontal da Regra de Ouro da civilização — o mandamento moral basilar que nos ensina a não fazer aos outros aquilo que não gostaríamos que fizessem conosco.

Daí a magnitude histórica daquela impressora a laser e do choque que ela provocou. Sem aquele entrevero, diz Stallman, sua vida bem poderia ter corrido por trilhas ordinárias, equilibrando o conforto financeiro de um emprego na iniciativa privada com a frustração crônica de passar a existência escrevendo linhas de código invisíveis e acorrentadas. Não teria havido a clareza fulgurante; não teria havido o senso de urgência para combater uma tragédia que ninguém mais parecia disposto a enfrentar. E, acima de tudo, não teria havido aquela raiva sagrada — uma indignação moral que, como logo veremos, impulsionaria a trajetória de Richard Stallman com muito mais potência do que qualquer ideologia política abstrata ou dogma intelectual.

— Daquele dia em diante, decidi que isso era algo de que eu jamais participaria — decreta Stallman, referindo-se à barganha faustiana de trocar a liberdade fundamental pela conveniência passageira, bem como à cultura rasteira que normalizava tais concessões em primeiro lugar. — Decidi que jamais faria de outras pessoas vítimas como eu havia sido feito de vítima.

Para uma análise aprofundada sobre as origens e a evolução do termo "hacker", consulte o Apêndice A. ↩

Capítulo 2: 2001: Uma Odisseia Hacker

O departamento de ciência da computação da New York University (NYU) fica abrigado no Warren Weaver Hall, um edifício com ares de fortaleza situado a dois quarteirões a leste do Washington Square Park. Dutos de ar-condicionado de potência industrial criam uma espécie de fosso invisível de ar quente ao redor da fachada, espantando desocupados e vendedores ambulantes na mesma medida. Os visitantes que conseguem atravessar essa barreira térmica dão de cara com outro obstáculo respeitável: um balcão de controle de segurança logo após a única porta de entrada do prédio.

Passado o ponto de checagem, a atmosfera fica um pouco mais descontraída. Ainda assim, inúmeras placas espalhadas pelo saguão do térreo alertam sobre os perigos de portas destrancadas e saídas de emergência escoradas abertas. Vistos em conjunto, os avisos funcionam como um lembrete implacável: mesmo nos confins relativamente tranquilos da Nova York que antecedeu o 11 de setembro de 2001, cautela e desconfiança nunca eram demais.

Os avisos oferecem um contraponto temático curioso ao contingente crescente de pessoas que vai se aglomerando no átrio do prédio. Alguns parecem estudantes típicos da NYU; a grande maioria, no entanto, lembra uma plateia cabeluda de festival de rock, perambulando diante de uma casa de espetáculos na expectativa do início do show principal. Por uma manhã atípica, as massas tomaram conta do Warren Weaver Hall, deixando a recepcionista de plantão sem outra ocupação além de assistir a um programa de auditório na TV e apontar com os ombros em direção ao auditório mais próximo sempre que alguém se aproxima para perguntar sobre "o discurso".

Lá dentro, o visitante finalmente descobre quem é o responsável por desarmar a rotina solene de segurança do edifício: trata-se de Richard M. Stallman, fundador do Projeto GNU, presidente e criador da Free Software Foundation, bolsista da prestigiosa Fundação MacArthur em 1990, agraciado com o Grace Murray Hopper Award pela Association of Computing Machinery (também em 1990), correcipiente do Takeda Award de 2001 por Melhoria Socioeconômica e lendário veterano do AI Lab do MIT. Conforme amplamente anunciado em dezenas de fóruns e sites hackers pela internet — a começar pelo portal oficial do próprio Projeto GNU (http://www.gnu.org) —, Stallman está em Manhattan, sua antiga terra natal, para proferir uma conferência aguardadíssima, planejada como resposta direta à agressiva ofensiva desencadeada pela Microsoft contra a Licença Pública Geral GNU (a GNU GPL).

O tema da palestra é a história e o futuro do movimento do software livre. O local escolhido está longe de ser um mero detalhe: menos de um mês antes, o vice-presidente sênior da Microsoft, Craig Mundie, havia subido ao púlpito da vizinha Stern School of Business, também na NYU, para proferir um ataque virulento contra a GNU GPL — o instrumento jurídico concebido por Stallman dezesseis anos antes. Criada para conter a onda avassaladora de sigilo que ameaçava engolir a indústria de informática — fenômeno que Stallman testemunhou em primeira mão em 1980, durante seu calvário com a impressora da Xerox —, a GPL consolidou-se como a ferramenta mestra da comunidade do software livre. Em linhas gerais, a GPL estabelece uma propriedade comum e compartilhada — o que juristas modernos chamam hoje de "bens comuns digitais" (*digital commons*) — respaldada pelo próprio peso legal do direito autoral (*copyright*). A GPL torna essa condição irrevogável: uma vez que o autor doa seu código à comunidade sob esses termos, ninguém mais tem o direito de torná-lo proprietário. Versões derivadas devem obrigatoriamente carregar a mesma licença se utilizarem partes substanciais do código

original. Por conta dessa cláusula inegociável, os detratores da licença passaram a rotulá-la de "viral", sugerindo maliciosamente que ela infectaria qualquer programa que encostasse nela[1].

Em uma economia da informação cada vez mais dependente de softwares e amarrada a padrões tecnológicos, a GPL tornou-se o famoso "porrete" do ecossistema. Até empresas que outrora a ridicularizavam chamando-a de "socialismo de software" acabaram curvando-se aos seus benefícios. O Linux, núcleo de sistema desenvolvido em 1991 pelo estudante finlandês Linus Torvalds, é licenciado sob a GPL, assim como a maior parte das engrenagens do sistema GNU: o GNU Emacs, o depurador GNU Debugger (GDB), o compilador GNU C (GCC), entre outros. Reunidas, essas ferramentas dão corpo ao sistema operacional livre GNU/Linux, desenvolvido, cuidado e mantido pela comunidade hacker planetária. Em vez de enxergarem essa comunidade como ameaça, gigantes da tecnologia como IBM, Hewlett-Packard e Sun Microsystems passaram a se apoiar nela, comercializando serviços e programas corporativos construídos para rodar sobre os ombros dessa imensa infraestrutura aberta[2].

Essas corporações também aprenderam a usar o software livre como uma arma estratégica na guerra secular do mundo da informática contra a Microsoft — a potência de Redmond, no estado de Washington, que monopolizou o mercado de softwares para computadores pessoais desde o fim dos anos 1980. Na qualidade de dona do onipresente sistema operacional Windows, a Microsoft era quem mais tinha a perder diante de uma migração em massa da indústria para a licença GPL. Cada programa que compõe o colosso do Windows é blindado por camadas de direitos autorais e contratos leoninos (os Termos de Licença para Usuário Final, ou EULAs), garantindo o monopólio comercial sobre o executável e trancando a sete chaves o código-fonte que o usuário comum jamais verá. Incorporar qualquer trecho de código protegido pela "viral" GPL a esses pacotes é terminantemente proibido pelas regras da empresa: para cumprir os requisitos legais da licença, a Microsoft seria obrigada por lei a libertar o programa inteiro como software livre. Se isso acontecesse, empresas rivais poderiam copiar, aprimorar e vender suas próprias versões melhoradas, desmantelando de uma vez por todas o cativeiro que aprisionava os usuários ao ecossistema de Redmond.

Daí a preocupação crescente da cúpula da Microsoft diante da aceleração no ritmo de adoção da GPL. Daí o tom belicoso do discurso de Craig Mundie disparando contra a GPL e contra o modelo de desenvolvimento em "código aberto" — a Microsoft, aliás, recusa-se até mesmo a pronunciar o termo "software livre", preferindo concentrar sua artilharia na vertente puramente comercial do "código aberto" (*open source*), esvaziada de conotação política, para desviar a atenção do movimento social fundado por Stallman. E daí, enfim, a decisão de Stallman de subir ao mesmo palco nova-iorquino para desferir um desmentido público e categórico.

Vinte anos representam uma eternidade na linha do tempo da informática. Basta lembrar: em 1980, enquanto Richard Stallman praguejava contra a impressora da Xerox no laboratório do MIT, a mesma Microsoft que hoje manda e desmanda no mercado mundial era apenas uma startup minúscula de capital fechado. A IBM, então tida como a força mais temível da computação corporativa, sequer havia lançado seu primeiro computador pessoal — o marco fundador que daria partida ao mercado do PC acessível. Tecnologias que hoje consideramos banais — a internet comercial, a televisão via satélite, os consoles de videogame de 32 bits — simplesmente não existiam. O mesmo vale para corporações que hoje ocupam o topo do establishment empresarial, como AOL, Sun Microsystems, Amazon, Compaq e Dell.

Para aqueles que colocam o progresso técnico acima da liberdade, o fato de o mercado ter marchado tão longe em tão pouco tempo costuma ser invocado tanto a favor quanto contra a GNU GPL. Os defensores da licença apontam para a obsolescência acelerada dos equipamentos de informática. Diante do risco constante de comprar uma

máquina que se tornará sucata em meses, os consumidores tendem a correr para os braços das empresas dominantes com maior chance de sobrevivência a longo prazo. Como resultado, o mercado de software virou uma arena implacável onde o vencedor leva tudo. O modelo de software proprietário, argumentam esses analistas, estimula a formação de monopólios sufocantes e a estagnação: os gigantes corporativos sugam todo o oxigênio do setor, inviabilizando a concorrência e abortando startups inovadoras.

Os detratores, por sua vez, sustentam exatamente o oposto. Comercializar software, afirmam eles, é tão ou mais arriscado do que comprá-lo. Sem a blindagem jurídica das licenças proprietárias e a expectativa de lucros estratosféricos garantida pela exclusividade de um "aplicativo matador" (*killer app*)[3], os investidores perderiam qualquer estímulo financeiro para inovar. O mercado entraria em paralisia. Como o próprio Mundie fez questão de frisar em seu pronunciamento de 3 de maio naquele mesmo campus, o caráter "viral" da GPL "constitui uma ameaça frontal" a qualquer companhia que faça da exclusividade de seus algoritmos o seu diferencial de mercado:

> *"A GPL compromete de forma basilar o setor de software comercial independente, pois torna inviável, na prática, distribuir programas em um modelo em que os clientes paguem pelo produto intelectual em si, e não apenas pelos custos materiais de distribuição."*

O sucesso simultâneo do GNU/Linux e do Windows ao longo da última década demonstra que ambos os lados têm seus momentos de razão na esfera do mercado. Contudo, para ativistas do software livre como Richard Stallman, toda essa discussão financeira passa ao largo do que realmente importa. A questão primordial, insistem eles, não é descobrir qual modelo gera mais riqueza ou vence a corrida das vendas, mas sim qual modelo é **eticamente justo**.

Ainda assim, a disputa de narrativa é feroz no setor de tecnologia. Até impérios gigantescos como a Microsoft dependem da boa vontade de desenvolvedores terceirizados, cujos programas e jogos tornam uma plataforma como o Windows atraente para o consumidor final. Mencionando a marcha acelerada da tecnologia nos últimos vinte anos — e ostentando o faturamento formidável da sua própria empresa no período —, Mundie advertiu os ouvintes para não se deixarem seduzir pelo canto da sereia do software livre:

> *"Duas décadas de experiência prática demonstraram que um arcabouço econômico alicerçado na proteção da propriedade intelectual, somado a um modelo de negócios capaz de recuperar os pesados investimentos em pesquisa e desenvolvimento, é capaz de gerar benefícios econômicos extraordinários e difundi-los amplamente pela sociedade."*

Foi exatamente essa a provocação que serviu de combustível para a palestra de Stallman. Menos de trinta dias após o pronunciamento do executivo da Microsoft, Stallman encontra-se encostado no quadro-negro da frente da sala, impaciente para assumir o microfone.

Se as duas últimas décadas transformaram radicalmente o mapa da informática, transformaram ainda mais o próprio Richard Stallman. Ficou para trás o jovem hacker magro e de rosto liso que passava dias inteiros em comunhão mística com seu querido PDP-10. Em seu lugar surge um homem de meia-idade, corpulento, cabelos longos e barba profética, que hoje gasta a maior parte dos dias respondendo a milhares de e-mails, batendo boca

com outros desenvolvedores e percorrendo o globo em conferências como a de hoje. Vestindo uma camiseta verde-água desbotada e calças marrons de poliéster, Stallman parece um eremita saído diretamente de um provador do Exército da Salvação.

A plateia está repleta de entusiastas que compartilham das mesmas preferências estéticas de seu líder. Vários trazem notebooks pesados e modems celulares à tira-colo, prontos para registrar e transmitir cada frase de Stallman em tempo real para os fóruns da internet. A proporção entre os sexos gira em torno de quinze homens para cada mulher; uma das raras mulheres na sala carrega orgulhosamente no colo um pinguim de pelúcia — o mascote oficial do Linux —, enquanto outra aperta um ursinho de brinquedo.

Visivelmente inquieto, Stallman deixa o quadro-negro e se senta em uma cadeira na primeira fila, batendo comandos rápidos em um laptop aberto. Pelos dez minutos seguintes, permanece completamente alheio ao burburinho de estudantes, acadêmicos e fãs que se espremem ao redor do palco.

Antes do espetáculo começar, contudo, é preciso cumprir o protocolo acadêmico. A presença de Stallman atrai nada menos que duas apresentações formais. A primeira fica a cargo de Mike Uretsky, codiretor do Center for Advanced Technology da Stern School.

— A missão de uma universidade é estimular o contraditório e abrigar debates enriquecedores — declara Uretsky. — Este seminário se encaixa com perfeição nesse espírito. Considero a discussão sobre *código aberto* particularmente fascinante...

Antes que Uretsky consiga engatar a oração seguinte, Stallman já está de pé no meio do auditório, agitando os braços feito um náufrago no acostamento tentando parar um caminhão.

— O que eu faço é **software livre**! — brada Stallman, provocando as primeiras gargalhadas da plateia. — "Código aberto" é outro movimento completamente diferente!

O riso logo se converte em palmas calorosas. A sala está lotada de simpatizantes de Stallman — veteranos que conhecem sua obsessão cirúrgica com a precisão dos termos, sem falar do célebre racha público ocorrido em 1998 com os defensores pragmáticos do termo *open source*. Para muitos ali, presenciar aquele choque ao vivo era como ouvir o comediante de rádio soltar seu famoso bordão de abertura.

Desconcertado, Uretsky encerra sua fala às pressas e passa a palavra a Edmond Schonberg, professor do departamento de ciência da computação da NYU. Sendo ele próprio programador e colaborador do Projeto GNU, Schonberg sabe exatamente onde pisar para desviar das minas terrestres da linguagem. Ele resume a trajetória de Stallman com a habilidade de quem conhece a mente de um hacker moderno:

— Richard é o exemplo acabado daquele sujeito que, ao agir localmente, passou a pensar globalmente sobre os dilemas morais da retenção do código-fonte — resume Schonberg. — Ele articulou uma filosofia coesa e contundente que obrigou todos nós a repensar a maneira como produzimos software, o verdadeiro significado da propriedade intelectual e o papel que a comunidade de desenvolvedores desempenha na sociedade[4].

Schonberg saúda Stallman sob aplausos entusiasmados. Richard desliga o laptop com calma, levanta-se da cadeira e sobe ao palco.

No início, sua apresentação lembra mais um show de comédia stand-up do que um manifesto político.

— Gostaria de agradecer de coração à Microsoft por ter me proporcionado a oportunidade de subir a esta tribuna — ironiza Stallman, arrancando risos da sala. — Nas últimas semanas, estive me sentindo como um escritor que teve a imensa sorte de ver seu livro proibido em praça pública.

Para nivelar a conversa e acolher os leigos, Stallman puxa uma de suas metáforas mais clássicas: compara um programa de computador a uma **receita de culinária**. Ambos oferecem instruções práticas, passo a passo, para cumprir uma tarefa determinada, e ambos podem ser livremente alterados se o cozinheiro quiser adaptá-los ao seu paladar ou às suas restrições:

— Você não é obrigado a seguir uma receita ao pé da letra — explica Stallman. — Se não gostar de um ingrediente, você tira. Se adora cogumelos, joga cogumelos na panela. Se o seu médico mandou cortar o sal, você reduz o sal. Você adapta como bem entender.

E o mais importante: programas e receitas culinárias são extremamente fáceis de compartilhar. Ao ensinar uma receita a um convidado no jantar, o cozinheiro não perde nada além de alguns minutos e o custo da folha de papel em que anotou os ingredientes. Com softwares o gasto é menor ainda: bastam alguns cliques do mouse e um fiapo de energia elétrica. Em ambos os casos, contudo, quem compartilha o que sabe colhe dois frutos imediatos: estreita laços de amizade e ganha o direito moral de receber receitas deliciosas em troca no futuro.

— Agora imaginem como seria o mundo se as receitas de cozinha fossem empacotadas em caixas-pretas lacradas — provoca Stallman, mudando o tom da voz. — Você não saberia quais temperos foram usados, não teria como adaptá-los ao gosto dos seus filhos e, se ousasse copiar a receita para dar a um amigo, seria tachado de pirata e ameaçado de passar anos na cadeia! Um mundo desses causaria revolta e indignação generalizada entre qualquer um que goste de cozinhar. Pois bem: esse é exatamente o mundo do software proprietário. Um mundo doentio em que a decência humana elementar de ajudar o próximo foi declarada proibida ou inviabilizada pela força.

Construída a ponte da receita, Stallman engata o relato de sua epopeia com a impressora da Xerox. Assim como a culinária, o causo da impressora a laser funciona como uma parábola perfeita: dramatiza com crueza a velocidade assustadora com que o chão desapareceu sob os pés dos pioneiros da informática. Ao transportar a plateia de volta a um tempo anterior às compras com um clique na Amazon, aos cliques no Windows e aos bancos de dados da Oracle, ele convida o ouvinte a pensar sobre o software despido das marcas corporativas da moda.

Stallman desfia a história com o ritmo treinado de um promotor de justiça em sustentação oral. Ao atingir o clímax — o instante em que o pesquisador da Carnegie Mellon bate o pé e nega a cópia do código-fonte da impressora —, Richard faz uma pausa teatral.

— Aquele sujeito nos traiu — dispara Stallman. — Mas ele não fez isso apenas conosco. A verdade é que ele muito provavelmente fez isso com você também!

Ao pronunciar a palavra "você", Stallman estica o dedo indicador de forma acusatória na direção de um espectador desavisado na plateia. O sujeito pisca, surpreso, mas o olhar de Stallman já se deslocou. De forma lenta e deliberada, escolhe um segundo alvo em meio aos risos nervosos do auditório:

— E acho muito provável que ele tenha feito isso com você também! — repete, apontando para alguém três fileiras atrás.

Quando seleciona a terceira vítima, as risadas já tomaram conta da sala. O número parece encenado — e é mesmo. Ainda assim, na hora de fechar a história da Xerox, Stallman o faz com a precisão de um grande mestre:

— Ele muito provavelmente traiu a imensa maioria das pessoas aqui presentes — exceto, talvez, alguns poucos jovens que ainda não haviam nascido em 1980 — arremata, arrancando novas risadas. — E traiu porque assinou um papel se comprometendo a recusar cooperação com quase toda a população do planeta Terra!

Stallman deixa o peso da acusação planar no ar por um segundo.

— Ele havia assinado um acordo de não divulgação.

A metamorfose de Richard Matthew Stallman ao longo de duas décadas — de acadêmico recluso e frustrado a líder político internacional — revela muitas coisas. Revela a sua teimosia lendária e uma força de vontade titânica. Revela a coerência inabalável dos princípios e dos valores que estruturam o movimento do software livre que ele ergueu. Revela a excelência magistral das ferramentas de código que ele programou com as próprias mãos, programas que cimentaram seu nome no panteão dos maiores hackers de todos os tempos. E revela, com nitidez cristalina, a expansão formidável da GPL, uma invenção jurídica que muitos consideram o seu maior legado para a humanidade.

Acima de tudo, porém, essa trajetória escancara as profundas transformações na natureza do próprio poder político em uma sociedade umbilicalmente atrelada à tecnologia digital e aos códigos invisíveis que governam as máquinas.

Talvez resida aí a razão pela qual, mesmo em uma época em que tantas estrelas brilhantes do Vale do Silício despencam no esquecimento, a relevância de Stallman continue a crescer. Desde o lançamento do Projeto GNU em 1984[5], Stallman já foi sucessivamente ignorado, satirizado, caluniado e atacado com ferocidade — tanto por inimigos externos quanto por facções dentro do próprio movimento. Apesar de tudo, o Projeto GNU seguiu cumprindo suas metas históricas — ainda que acumulando alguns atrasos lendários pelo caminho — e manteve-se vivo e influente em um mercado infinitamente mais complexo e hostil do que aquele em que deu seus primeiros passos.

Para decifrar o mistério por trás dessa longevidade, é preciso encarar Richard Stallman tanto através de seus próprios depoimentos quanto pelas impressões daqueles que conviveram, lutaram e romperam com ele ao longo dos anos. O perfil humano de Stallman não guarda artimanhas: se existe alguém que encarna à perfeição o ditado *"o que você vê é o que você tem"*, esse alguém é ele.

— Acredito que, se você deseja compreender Richard Stallman como ser humano, precisa enxergar todas as partes de sua personalidade como um todo coerente — ensina Eben Moglen, professor da Columbia Law School e diretor jurídico histórico da Free Software Foundation. — Todas aquelas excentricidades que muita gente enxerga como barreiras para se aproximar dele são, na verdade, a própria essência de Stallman: o seu profundo sentimento de frustração diante do erro, seu apego intransigente a princípios éticos fundamentais e sua recusa categórica em fazer concessões, sobretudo naquilo que considera inegociável. Foram exatamente esses traços que levaram Richard a fazer o que fez, na hora exata em que fez.

Explicar de que maneira uma rusga técnica com uma impressora a laser desembocou em um embate de proporções bíblicas contra a corporação mais rica e poderosa do planeta não é tarefa trivial. Exige dissecar com serenidade as forças ocultas que transformaram a apropriação do software em um dos negócios mais rentáveis e disputados do capitalismo moderno. Exige também analisar com lucidez a trajetória de um homem que, como tantos outros líderes históricos, compreende o papel crucial da memória humana na disputa de poder. Exige destrinchar os mitos e os chavões carregados de viés ideológico que se acumularam ao redor de sua figura com o passar do tempo. E exige, por fim, reconhecer tanto a genialidade pioneira de Stallman como programador quanto seus triunfos e tropeços ao tentar transpor essa mesma genialidade para os campos minados da política e da sociedade.

Ao fazer o balanço de sua própria jornada, Stallman não hesita em reconhecer essa fusão indissolúvel entre temperamento e convicção ética notada por Eben Moglen:

— A teimosia é o meu forte — reconhece ele. — A maioria das pessoas que tenta realizar algo muito difícil acaba desanimando e jogando a toalha no meio do caminho. Eu nunca desisti.

Ele também dá crédito ao acaso e à sorte cega. Se não tivesse havido aquele choque por causa da impressora da Xerox; se não tivesse enfrentado as rupturas pessoais e políticas que selaram o fim de sua carreira como funcionário do MIT; se não tivessem convergido tantas variáveis raras naquele início dos anos 1980, Stallman admite que sua vida bem poderia ter tomado um rumo muito mais pacato e convencional. Dito isso, agradece profundamente às circunstâncias que o colocaram na trincheira para fazer a diferença:

— Eu tinha exatamente as habilidades necessárias — conclui Stallman, explicando para a plateia por que decidiu fundar o Projeto GNU. — Não havia mais ninguém ali além de mim. Então senti que havia sido convocado. Eu tinha que assumir essa missão. Se não eu, quem?

Na prática, o alcance da GPL está longe de ter esse poder avassalador: o simples fato de colocar seu código no mesmo disco rígido onde roda um programa coberto pela GPL não submete seu programa à licença. "Comparar a licença a um vírus é de uma grosseria desmedida", argumenta Stallman. "Uma comparação muito mais fiel seria com uma muda de clorófito (*spider plant*): ela só brota em outro vaso se você deliberadamente for lá e plantar uma muda cortada". Para mais informações sobre a GNU GPL, consulte: http://www.gnu.org/copyleft/gpl.html. ↩

Embora esses aplicativos comerciais rodem em cima do GNU/Linux, isso não significa que eles próprios sejam softwares livres. Pelo contrário: a maior parte desses programas é composta por softwares estritamente proprietários, que respeitam a liberdade do usuário tanto quanto o Windows respeita. Eles podem até colaborar para a popularização do sistema GNU/Linux, mas não contribuem para o objetivo maior de liberdade que justifica a existência do sistema. ↩

Softwares revolucionários não precisam ser necessariamente proprietários. Ainda assim, a metáfora é clara: a indústria do software funciona como uma loteria — quanto maior o prêmio em jogo, maior o número de apostadores. ↩

Se esse pronunciamento fosse feito hoje, Stallman certamente interromperia Schonberg para protestar contra a própria expressão "propriedade intelectual", que ele denuncia categoricamente como um termo confuso e enviesado, criado para

misturar artificialmente leis tão distintas quanto direitos autorais, patentes e marcas registradas. Para uma explicação detalhada, consulte: http://www.gnu.org/philosophy/not-ipr.html. ↩

O acrônimo **GNU** significa recursivamente *"GNU's Not Unix"* ("GNU Não É Unix"). Na mesma conferência da NYU, Stallman recordou o batismo do sistema: "Nós, hackers, sempre procurávamos nomes engraçados ou espirituosos para os nossos programas, porque dar o nome era metade da diversão de programar. Tínhamos também a velha tradição dos acrônimos recursivos, uma forma bem-humorada de indicar que o programa que você estava escrevendo era parecido com algum sistema já existente... Busquei então um acrônimo recursivo para a ideia de *Algo Não É Unix*. Testei todas as vinte e seis letras do alfabeto e descobri que nenhuma formava uma palavra de verdade em inglês. Resolvi então apelar para uma contração: assim eu poderia ter uma sigla de três letras para *Something's Not Unix*. Fui testando combinações até bater o olho na palavra **GNU** [o antílope africano gnu]. Era perfeito!" Embora seja apaixonado por trocadilhos, Stallman recomenda pronunciar claramente o "g" inicial do acrônimo (como *"g-nu"*, e não *"nu"*, como no inglês), tanto para não confundir o sistema com o animal quanto para não confundi-lo com a palavra inglesa *new* ("novo"). "Já estamos trabalhando nele há mais de dezessete anos, de modo que ele não é exatamente 'novo'!", brincou. ↩

Capítulo 3: Retrato do Hacker Quando Jovem

Alice Lippman, mãe de Richard Stallman, ainda guarda viva na memória a tarde exata em que se deu conta de que o filho tinha um talento fora do comum.

— Acho que ele tinha uns oito anos — recorda Alice.

O ano era 1961. Divorciada e criando o filho sozinha, Alice passava uma tarde de fim de semana preguiçosa no modesto apartamento de um quarto da família no Upper West Side, em Manhattan. Folheando um exemplar da conceituada revista *Scientific American*, foi direto à sua seção predileta: a famosa coluna de Martin Gardner intitulada "Jogos Matemáticos". Professora substituta de artes, Alice adorava os enigmas propostos por Gardner pelo puro estímulo mental. Com o filho já devidamente enfiado em um livro no sofá da sala, ela resolveu encarar o quebra-cabeça principal daquela edição.

— Eu nunca fui das mentes mais brilhantes na resolução de enigmas matemáticos — admite. — Mas, como artista plástica, achava que aqueles quebra-cabeças ajudavam a arejar o cérebro e a romper bloqueios criativos.

Alice conta que sua tentativa de solucionar o enigma esbarrou em um beco sem saída logo de cara. Preste a jogar a revista de lado com um suspiro de irritação, sentiu um puxãozinho suave na manga de sua blusa.

— Era o Richard — lembra ela. — Ele veio me perguntar se eu queria ajuda.

Olhando espantada da revista para o menino e do menino para a revista, Alice diz ter recebido a oferta com compreensível ceticismo:

— Perguntei ao Richard se ele tinha lido a revista. Ele respondeu que sim, que tinha lido e que, além disso, já havia decifrado o enigma todinho. Quando me dei conta, ele começou a me explicar passo a passo o raciocínio para chegar à resposta.

Ao ouvir a lógica impecável da demonstração do filho, a desconfiança de Alice converteu-se em espanto absoluto:

— Eu sempre soube que ele era um garoto esperto, mas aquela foi a primeira vez que presenciei algo revelando o quão adiantado ele realmente era em relação à idade.

Trinta anos depois, Alice pontua a lembrança com uma gargalhada cúmplice:

— Para ser bem sincera, acho que eu nunca entendi como se resolvia aquele maldito quebra-cabeça. A única coisa de que me lembro com nitidez é do queixo caindo ao ver que ele sabia a resposta.

Acomodada à mesa de jantar de seu segundo apartamento em Manhattan — o mesmo imóvel amplo de três quartos para onde se mudou com o filho após seu casamento em 1967 com Maurice Lippman, já falecido —, Alice

irradia aquele clássico misto de orgulho transbordante e perplexidade afetuosa típico de uma mãe judia novaiorquina ao rememorar os primeiros passos do primogênito. Sobre o aparador da sala destaca-se um retrato ampliado de Stallman, ostentando sua farta barba profética e o capelo de gala dos trajes doutorais. O quadro se agiganta diante das fotos bem menores dos outros sobrinhos da família; mas, antes que qualquer visitante elabore suposições solenes, Alice se antecipa com uma alfinetada irônica:

— O Richard fez questão absoluta de que eu ficasse com esse retrato depois que ele recebeu o doutorado honorário pela Universidade de Glasgow — diz ela, rindo. — Ele me soltou: "Adivinha só, mãe? É a primeiríssima cerimônia de formatura que eu assisto na vida!"[1].

Comentários como esse sintetizam a dose indispensável de bom humor necessária para criar um jovem prodígio. Não há engano: para cada relato de teimosia obsessiva ou esquisitice que Alice ouve sobre o filho ilustre, ela tem pelo menos uma dúzia de causos de infância prontos na ponta da língua.

— Ele costumava ser tão conservador! — comenta Alice, jogando as mãos para o alto em sinal de exasperação teatral. — Nós travávamos as piores brigas do mundo bem aqui nesta mesma mesa! Eu fiz parte da primeira leva de professoras da rede pública de Nova York que cruzou os braços em greve para fundar nosso sindicato, e o Richard ficou furioso comigo. Ele achava que sindicatos eram corruptos por definição. Também era contra a Previdência Social: dizia que qualquer trabalhador ganharia muito mais dinheiro investindo suas economias por conta própria no mercado financeiro. Quem haveria de imaginar que, em menos de dez anos, ele se tornaria esse idealista radical? Lembro-me até hoje da meia-irmã dele me puxando pelo braço e sussurrando, apavorada: "Mãe, o que é que o Richard vai ser quando crescer? Um fascista?!"[2].

Mãe solo ao longo de quase uma década — ela e o pai de Richard, Daniel Stallman, casaram-se em 1948, divorciaram-se em 1958 e dividiram a guarda do filho nos anos seguintes —, Alice é testemunha ocular da aversão congênita do menino a qualquer imposição de autoridade. Do mesmo modo, atesta sua fome insaciável por conhecimento. E era exatamente quando essas duas forças colidiam, ressalta ela, que as maiores tempestades domésticas desabavam sobre a casa:

— Parecia que ele simplesmente não precisava de comida — relembra Alice, descrevendo um padrão que se fixou por volta dos oito anos e só arrefeceu quando Richard concluiu o ensino médio, em 1970. — Eu chamava para o jantar e ele simplesmente não ouvia. Tinha que chamá-lo nove, dez vezes só para conseguir tirá-lo do transe. Ele ficava absolutamente imerso no que estava fazendo.

Stallman guarda lembrança muito semelhante daquele período, embora enquadre a questão sob uma ótica estritamente política:

— Eu adorava ler — recorda ele. — Se eu estava compenetrado na leitura e minha mãe mandava eu ir para a cozinha comer ou ir para a cama dormir, eu não obedecia. Não via sentido algum em ter de interromper a minha leitura. Não via razão plausível para que ela pudesse mandar em mim e ponto final. Em essência, tudo aquilo que eu lia a respeito de democracia e liberdades individuais, eu aplicava diretamente a mim mesmo. Não enxergava justificativa alguma para excluir as crianças desses princípios universais.

Essa fé intransigente na soberania individual contra a autoridade arbitrária transbordou, inevitavelmente, para as salas de aula. Adiantado dois anos inteiros em relação aos colegas aos onze anos de idade, Stallman amargou todas

as agruras típicas de uma criança superdotada confinada à rotina engessada da escola pública. Pouco tempo depois do episódio da revista científica, Alice precisou comparecer à primeira de uma série interminável de reuniões pedagógicas tensas.

— Ele se recusava terminantemente a fazer redações — conta Alice, resgatando um dos primeiros impasses escolares. — Acho que a última redação que ele escreveu na vida antes do último ano do colegial foi um pequeno texto sobre a história do sistema numérico ocidental para uma professora da quarta série.

Ser forçado a escrever sobre um tema específico quando não havia nenhum interesse genuíno pelo assunto era uma tortura para Stallman — um suplício penoso o bastante para fazê-lo mover mundos para boicotar o dever de casa.

Dotado de um raciocínio lógico brilhante, Stallman canalizava toda a sua energia para a matemática e para as ciências, deixando as demais matérias em segundo plano. O que alguns professores diagnosticavam como pura teimosia, no entanto, Alice compreendia como impaciência: as ciências exatas ofereciam um terreno fértil de descobertas, em contraste gritante com disciplinas pelas quais ele não nutria atração natural. Por volta dos dez ou onze anos, quando os garotos da vizinhança começaram a organizar peladas diárias de futebol americano na rua, Alice lembra de vê-lo chegar em casa bufando de raiva:

— Ele queria jogar desesperadamente, mas simplesmente não tinha coordenação motora para aquilo. Isso o deixava furioso da vida.

A frustração motora acabou empurrando o menino ainda mais fundo para o refúgio das fórmulas e das equações. Mesmo no território da ciência, contudo, a precipitação do garoto costumava provocar atritos. Debruçado sobre manuais de cálculo diferencial antes dos oito anos, Richard não via razão alguma para nivelar seu vocabulário por baixo ao conversar com adultos. Em dado momento do ensino fundamental, Alice tentou contratar um estudante da prestigiosa Columbia University para atuar como tutor e "irmão mais velho" do menino. O universitário encerrou o primeiro encontro com a criança, foi embora e nunca mais pisou na casa. "Acho que as coisas sobre as quais o Richard falava passaram muito por cima da cabeça do rapaz", especula Alice.

Outra lembrança materna marcante remonta ao início dos anos 1960. Aos sete anos, pouco após o divórcio e a mudança da família do Queens para Manhattan, Richard tomou gosto por lançar pequenos foguetes de brinquedo no parque da Riverside Drive. O que começara como uma brincadeira ingênua logo ganhou ares de pesquisa rigorosa quando o menino passou a registrar minuciosamente as métricas de trajetória de cada disparo. Assim como acontecera com a matemática, o hobby não chamava tanta atenção até que, em uma tarde em que a NASA preparava uma transmissão histórica de lançamento espacial na TV, Alice foi ao quarto do filho ver se ele queria assistir à transmissão.

— Ele estava lívido de raiva — recorda Alice. — A única coisa que conseguia balbuciar entre dentes era: "Mas eu ainda não fui publicado!". Aparentemente, ele achava que tinha descoberto algo crucial que precisava com urgência mostrar aos cientistas da NASA.

Stallman diz não se recordar do episódio, mas pondera que, muito provavelmente, o motivo de sua angústia infantil não era ter sido rejeitado pela NASA, mas sim a dor de constatar que ainda não tinha nada de valor palpável para apresentar à agência espacial.

Episódios como esses revelam as primeiras manifestações daquela intensidade vulcânica que viria a ser a assinatura vitalícia de Stallman. Enquanto as outras crianças iam para a mesa, Stallman ficava no quarto lendo. Enquanto os outros garotos brincavam de heróis do esporte, Stallman brincava de astronauta.

— Eu era esquisito — resume ele em uma entrevista em 1999, ao falar de seus primeiros anos. — A partir de certa idade, os únicos amigos que eu conseguia manter eram os meus professores.

Stallman não sentia vergonha de suas esquisitices intelectuais, distinguindo-as com clareza da inaptidão social prática, que esta sim encarava como um defeito penoso. Seja como for, a combinação de ambas pavimentou seu doloroso isolamento entre os colegas de classe.

Mesmo ciente de que aquilo atrairia atritos na escola, Alice optou por respaldar as obsessões do filho. Aos doze anos, Richard já passava as férias em acampamentos científicos de verão e cursava o ano letivo em colégios particulares. Quando um professor recomendou que o garoto prestasse os exames para o Columbia Science Honors Program — um programa de ponta criado no pós-Sputnik para atender alunos nova-iorquinos superdotados de ensino fundamental e médio —, Stallman ampliou sua rotina e passou a cruzar a ilha de Manhattan todos os sábados rumo ao campus da Columbia University.

Dan Chess, colega de turma no programa de ciências de Columbia, recorda que Stallman chamava a atenção por sua excentricidade até mesmo em um ambiente apinhado de jovens fascinados por números:

— Todos nós éramos nerds e geeks incorrigíveis, mas o Richard era desajustado em um grau completamente fora da curva — relembra Chess, hoje professor emérito de matemática do Hunter College. — E ele era assombrosamente inteligente. Já convivi com muita gente genial nesta vida, mas acho que ele foi o ser humano mais brilhante que já conheci.

Seth Breidbart, outro colega egresso do mesmo programa que manteve contato com Stallman graças ao amor compartilhado pela ficção científica, reforça a impressão. Ele recorda a figura de Stallman aos quinze anos, com corte de cabelo militar rente ao couro cabeludo, como alguém "aterrorizante" — sobretudo para outro adolescente da mesma idade:

— É difícil de descrever com palavras — explica Breidbart. — Não é que ele fosse agressivo ou inacessível. O negócio é que ele era de uma intensidade desconcertante. Tinha uma bagagem monstruosa de conhecimento, mas era de uma teimosia e de uma rigidez mental quase impenetráveis.

Retratos dessa natureza inevitavelmente alimentam especulações retrospectivas: seriam adjetivos carregados como "intenso" ou "irredutível" meros eufemismos para o que hoje a psiquiatria enquadra no espectro dos transtornos comportamentais? Em dezembro de 2001, uma reportagem de grande repercussão da revista *Wired*, intitulada *"The Geek Syndrome"* ("A Síndrome Geek"), traçou o perfil de várias crianças e jovens cientificamente superdotados diagnosticados com autismo de alto funcionamento ou Síndrome de Asperger. Sob diversos aspectos, os relatos parentais citados na matéria da *Wired* guardam semelhanças desconcertantes com as memórias de Alice Lippman. O próprio Stallman já refletiu abertamente sobre a hipótese. Em uma entrevista de perfil concedida em 2000 ao jornal canadense *Toronto Star*, ele admitiu que já se perguntara se não estaria "no limiar do autismo" — fala que o jornal, de forma sensacionalista, acabou publicando como se fosse um diagnóstico médico fechado[3].

Toda especulação retrospectiva esbarra, é claro, na elasticidade com que a medicina contemporânea define tais rótulos. Como bem aponta Steve Silberman, autor do clássico ensaio da *Wired*, foi apenas muito recentemente que a psiquiatria norte-americana passou a aceitar o diagnóstico de Asperger como um guarda-chuva amplo para um conjunto diverso de características: desde incoordenação motora e dificuldades de traquejo social até superdotação analítica e uma fixação quase obsessiva por números, linguagens computacionais e sistemas ordenados[4].

— É perfeitamente plausível que eu tenha tido algo nessa linha — pondera Stallman. — Por outro lado, um dos traços marcantes dessa síndrome é a dificuldade crônica em assimilar e acompanhar ritmos musicais. E eu danço! Na verdade, eu sou apaixonado por dançar acompanhando os ritmos mais intrincados e complexos do mundo. As coisas não são nítidas o bastante para bater o martelo.

Outra hipótese aventada por especialistas é que Stallman se encaixe no que a literatura médica chama de "síndrome-sombra": manifestações que tangenciam os traços do espectro sem romper as fronteiras da funcionalidade cotidiana[5].

Dan Chess, por sua vez, torce o nariz para qualquer tentativa de diagnóstico tardio:

— Nunca me passou pela cabeça enxergá-lo sob esse prisma médico — afirma o matemático. — Ele apenas era completamente destituído de verniz social; mas, convenhamos, naquela turma todos nós éramos.

Alice Lippman, no entanto, olha para a tese com olhos bem mais receptivos. Ela resgata episódios dos primeiros meses de vida do menino que fornecem farto material para reflexão. Um sintoma recorrente do espectro autista é a hipersensibilidade extrema a ruídos e cores fortes, e Alice tem duas lembranças nítidas desse período:

— Quando o Richard ainda era bebê de colo, nós o levávamos para passear na praia. Ele começava a berrar desesperadamente a dois ou três quarteirões da arrebentação. Só na terceira vez que fomos é que conseguimos matar a charada: o estrondo das ondas quebrando na praia machucava fisicamente os ouvidos dele.

Ela recorda também uma reação alérgica idêntica em relação a estímulos visuais:

— Minha mãe tinha cabelos ruivos muito vivos. Toda vez que ela se inclinava para pegá-lo no colo, o Richard abria um berreiro histérico.

Nos últimos anos, Alice dedicou-se a ler literatura sobre o autismo e passou a acreditar que aqueles episódios estavam longe de ser meras coincidências:

— Tenho a convicção íntima de que o Richard carregava muitos dos traços de uma criança no espectro autista. Lamento profundamente que a ciência soubesse tão pouco a respeito disso naquela época.

Com o tempo, contudo, o menino aprendeu a se defender. Aos sete anos, Alice conta que a maior paixão do filho era ficar colado ao vidro dianteiro do primeiro vagão do metrô novaiorquino, mapeando e decorando o labirinto infinito de trilhos que serpenteava sob o subsolo da metrópole. Era um passatempo que exigia absorver estoicamente o barulho ensurdecedor que acompanhava a marcha das composições nos túneis:

— O estrondo só o incomodava no primeiro segundo — observa Alice. — Era como se ele tomasse um choque inicial com o barulho, mas seu sistema nervoso aprendesse depressa a compensar o impacto.

Na maior parte da infância, Alice insiste em que o menino exibia a curiosidade, a vivacidade e a saúde de qualquer criança comum. Foi apenas após uma sucessão de abalos sísmicos dentro da família que ele se recolheu para trás de uma muralha de silêncio e distanciamento emocional.

O primeiro golpe traumático foi o divórcio entre Alice e Daniel Stallman, pai de Richard. Embora ambos tenham se esforçado para suavizar o impacto sobre o garoto, o baque foi devastador:

— Ele meio que fez de conta que não estava ouvindo quando contamos que iríamos nos separar — relembra Alice. — Mas a realidade caiu feito uma marreta sobre a cabeça dele no instante em que nos mudamos para o novo apartamento. A primeira frase que ele soltou ao entrar na casa vazia foi: "Cadê os móveis do papai?".

Pela década seguinte, Stallman passaria os dias de semana no apartamento da mãe em Manhattan e os fins de semana com o pai no Queens. Esse trânsito contínuo permitiu-lhe testemunhar dois estilos parentais radicalmente opostos que, até hoje, alimentam sua recusa convicta em ter filhos. Ao falar do pai — um veterano condecorado da Segunda Guerra Mundial que faleceu no início de 2001 —, Stallman oscila entre o respeito moral e a ferida aberta. De um lado, admirava o homem cujo senso de dever o levara a aprender francês por conta própria apenas para ser mais útil aos Aliados na luta contra o nazismo na Europa. Do outro, ressentia-se do patriarca que dominava a arte cruel de desferir humilhações verbais calibradas para esmagar a autoestima[6].

— Meu pai tinha um temperamento horrível — desabafa Stallman. — Ele nunca gritava, mas sempre encontrava um jeito cirúrgico de rebaixar você com uma frieza calculada para destruir sua moral.

Quanto à convivência na casa da mãe, Stallman é igualmente categórico:

— Aquilo era uma praça de guerra — resume ele. — No auge do meu desespero, eu costumava chorar e repetir: "Eu quero ir para casa!", referindo-me àquele lugar acolhedor e inexistente que eu jamais teria.

Nos primeiros anos após a separação dos pais, Stallman encontrou o oásis de serenidade que lhe faltava na casa dos avós paternos. Contudo, um faleceu quando Richard tinha oito anos, e o outro, quando ele completou dez. Para o garoto, a perda foi irreparável:

— Eu adorava visitá-los; sentia que ali estava em um ambiente de doçura, afeto e acolhimento — recorda Stallman. — Foi o único refúgio seguro que conheci na infância, até o dia em que saí de casa para ir para a faculdade.

Alice aponta a morte dos avós paternos como o segundo grande trauma na formação do filho:

— Aquilo acabou com ele. O Richard era apegadíssimo aos dois. Antes do falecimento deles, ele era uma criança extrovertida, quase um líder de turma entre os meninos da rua. Depois que eles morreram, fechou-se numa concha e distanciou-se do mundo.

Pela perspectiva de Stallman, o recolhimento emocional não passava de um escudo elementar para sobreviver à agonia da puberdade. Classificando sua adolescência como um "horror absoluto", ele conta que se sentia frequentemente como um surdo perdido no meio de uma roda de pessoas comentando fascinadas sobre uma música:

— Eu tinha a nítida sensação de que era incapaz de entender o que as outras pessoas estavam dizendo — relembra Stallman sobre a dor do isolamento. — Eu entendia o significado literal das palavras, mas havia uma corrente subterrânea por baixo daquelas conversas que me escapava completamente. Eu simplesmente não conseguia alcançar por que diabos as pessoas se interessavam pelas bobagens que os outros falavam.

Apesar de todo o sofrimento, a adolescência forjou em Stallman um senso de individualidade granítico. Numa época em que todos os seus colegas deixavam os cabelos crescerem até os ombros, Stallman insistia em mantê-los cortados rentes. Numa era em que a juventude mundial só respirava rock and roll, ele refugiava-se no universo erudito da música clássica. Leitor voraz de ficção científica, devorador das sátiras da revista *Mad* e espectador assíduo da televisão na madrugada, Stallman foi sedimentando uma personalidade excêntrica e imprevisível que despertava incompreensão generalizada entre adultos e contemporâneos.

— E os trocadilhos! — suspira Alice, ainda exausta ao recordar o humor do filho naquela época. — Não havia uma única frase dita à mesa que ele não rebatesse na hora em forma de trocadilho infame!

Fora de casa, Richard reservava seus chistes intelectuais para os poucos adultos que sabiam apreciar seu cérebro brilhante. Um dos primeiros foi um monitor de colônia de férias que, ao vê-lo fascinado por números aos oito ou nove anos, presenteou-o com o manual de instruções do computador **IBM 7094**. Para um pré-adolescente sedento por lógica pura, aquele livro foi nada menos que uma revelação[7]. Sem demora, Stallman passou a escrever programas inteiros no papel, grafados na linguagem de máquina do 7094. Não havia computador algum à vista para rodar aquelas linhas, tampouco ele tinha qualquer utilidade prática para processar dados; mas sua alma ardia pelo desejo de programar — qualquer coisa, qualquer rotina que fosse. Chegava a pedir ao monitor que inventasse problemas fictícios só para ter o prazer de codificar a solução.

Com o primeiro microcomputador pessoal a uma década de distância no horizonte, Stallman teve de esperar a adolescência avançar para colocar as mãos em uma máquina de verdade. A oportunidade de ouro finalmente bateu à porta no seu último ano de colégio: o Centro Científico da IBM em Nova York, um laboratório de pesquisas já desativado no centro de Manhattan, abriu suas portas para que o jovem prodígio escrevesse seus primeiros programas no mundo real. Sua ideia era ousada: construir um pré-processador para a linguagem PL/I, concebido para incorporar à sintaxe a convenção de soma para álgebra tensorial.

— Primeiro escrevi tudo em PL/I; mas, quando vi que o programa compilado ficara grande demais para a memória da máquina, joguei tudo fora e reescrevi do zero em linguagem *Assembly* — recorda ele.

No verão logo após a formatura no ensino médio, o Centro Científico contratou-o formalmente como estagiário. Incumbido de redigir um programa de análise numérica em Fortran, liquidou a tarefa em pouquíssimas semanas — desenvolvendo uma aversão tão visceral pela sintaxe do Fortran que jurou solenemente nunca mais escrever uma linha sequer naquela linguagem pelo resto da vida. Passou então o restante do verão dedicando-se a construir um editor de texto na exótica linguagem APL.

Paralelamente, Stallman assumira uma vaga de assistente de pesquisa nos laboratórios de biologia da prestigiada Rockefeller University. Embora sua vocação já pendesse irresistivelmente para a matemática ou para a física teórica, o rigor do seu cérebro impressionou tanto o diretor do laboratório que, anos mais tarde, Alice Lippman recebeu um telefonema inesperado:

— Era o professor catedrático da Rockefeller — conta ela. — Ele queria saber por onde andava o Richard. Ficou embasbacado quando contei que ele havia migrado para a computação. O professor jurava de pés juntos que o Richard tinha uma carreira brilhante pela frente como biólogo.

A destreza analítica de Stallman também causava furor entre os docentes de Columbia — mesmo quando o próprio Richard se tornava o terror dos professores em sala de aula:

— Uma ou duas vezes por hora ele flagrava algum erro na lousa — relata Breidbart. — E não tinha a menor cerimônia em interromper o professor na hora para esfregar o equívoco em público. Aquilo rendia a ele um respeito colossal, mas popularidade zero.

Ouvir o relato de Breidbart arranca um sorriso irônico dos lábios de Stallman:

— Eu posso ter sido meio antipático algumas vezes — reconhece ele. — Mas eu encontrava almas gêmeas nos professores, porque eles também amavam aprender. Os garotos da minha idade, na maioria, não estavam nem aí para o conhecimento. Ao menos não com aquela fome.

Conviver com os prodígios de Nova York aos sábados, porém, forçou Stallman a encarar o custo prático de sua inadaptação social. Com o vestibular se aproximando, Richard — a exemplo de quase todos os colegas de Columbia — afunilou suas opções entre dois monumentos da academia norte-americana: **Harvard** e o **MIT**. Ao tomar conhecimento da ambição do filho, Alice sentiu um frio na espinha. Aos quinze anos, no penúltimo ano do colegial, Richard colecionava boletins com notas máximas em História Americana, Química, Francês e Álgebra, mas manchados por um vergonhoso zero em Inglês, reflexo de seu boicote intransigente às tarefas de redação. Enquanto uma excentricidade dessas arrancava risadas complacentes nos corredores do MIT, em Harvard equivalia a um cartão vermelho sumário.

Diante do impasse, Alice marcou uma consulta com um terapeuta. O especialista foi taxativo: com aquele histórico de desobediência e recusa a redigir textos, o garoto correria sério risco de implodir na universidade. Ele sem dúvida possuía capacidade intelectual para Harvard, mas teria maturidade para sentar a bunda na cadeira e entregar os ensaios exigidos pelas disciplinas humanísticas? O terapeuta propôs um teste de fogo: se Richard conseguisse sobreviver a um ano letivo completo na rede pública regular de Nova York, cumprindo todas as exigências das aulas de inglês, ele provavelmente estaria apto a encarar a rotina de Harvard. Stallman engoliu o orgulho, matriculou-se no curso público de verão e compensou as matérias atrasadas que havia negligenciado.

No outono seguinte, estava devidamente integrado à massa de alunos do colégio Louis D. Brandeis High School, na West 84th Street. Não foi fácil suportar aulas que pareciam jardim de infância em comparação com o rigor dos sábados em Columbia, mas Alice recorda com orgulho o sacrifício do filho:

— Ele foi obrigado a engolir sapos e a se curvar à disciplina, mas cumpriu o combinado — gaba-se Alice. — Fui chamada à diretoria uma única vez, o que foi um verdadeiro milagre. Era o professor de cálculo reclamando que o Richard interrompia as aulas o tempo todo. Perguntei de que jeito ele atrapalhava. O professor respondeu que o Richard vivia acusando-o diante de todos de apresentar demonstrações matemáticas falsas. Eu perguntei: "Bem, mas ele está com a razão?". O professor respondeu: "Está, mas eu não posso dizer isso para a turma; os outros alunos jamais entenderiam!".

Ao término do primeiro semestre na Brandeis High School, as peças haviam se encaixado: uma nota 96 em Inglês apagou a mancha do boletim anterior. Para consolidar o feito, Richard acumulou notas máximas em História Americana, Cálculo Avançado e Microbiologia, coroando o histórico escolar com um 100 absoluto em Física. Embora continuasse sendo um pária social nos corredores, Stallman concluiu seus dez meses na Brandeis como o quarto melhor aluno de uma turma de 789 formandos.

Longe das salas de aula, ele dobrava a aposta nos estudos: cumpria as tarefas de assistente no laboratório da Rockefeller University durante a semana e desviava das passeatas contra a Guerra do Vietnã para chegar pontualmente às aulas de sábado em Columbia. Foi lá que, enquanto os colegas do programa debatiam suas expectativas universitárias, Stallman finalmente se dignou a participar da conversa:

— A maioria estava rumando para Harvard ou para o MIT, como de praxe — relata Breidbart. — Conforme o assunto dava a volta na sala, percebemos que o Richard ainda não tinha dito uma única palavra. Não lembro quem teve a coragem de perguntar o que ele pretendia fazer da vida.

Trinta anos depois, Breidbart ainda descreve a cena com detalhes vívidos. Assim que Stallman soltou a notícia de que também havia sido aceito e que desembarcaria em Harvard no outono, um silêncio pesado e desconcertado engoliu a sala. Quase em câmera lenta, os cantos da boca de Stallman curvaram-se para cima em um sorriso de triunfo e satisfação:

— Foi o jeito silencioso dele nos dizer: "É isso mesmo, seus idiotas. Vocês ainda não conseguiram se livrar de mim!".

Uma das fontes biográficas centrais para a reconstituição deste capítulo foi a longa entrevista *"Richard Stallman: High School Misfit, Symbol of Free Software, MacArthur-Certified Genius"*, conduzida por Michael Gross para seu livro de 1999, *Talking About My Generation*. Embora Stallman não tenha entrado na seleção final do livro impresso, Gross disponibilizou a íntegra na web: http://www.mgross.com/MoreThgsChng/interviews/stallman1.html. ↩

[RMS: Não guardo nenhuma lembrança de ter dito isso a ela. A única coisa que posso afirmar com convicção é que discordo radicalmente de todas essas opiniões hoje em dia. Na minha adolescência, eu não tinha a menor empatia pelas agruras e dificuldades que a maioria dos trabalhadores enfrenta na vida cotidiana; os meus problemas eram de outra natureza. Eu simplesmente não alcançava como os muito ricos reduzem o restante da sociedade à miséria se as pessoas não se organizarem em todas as esferas para contê-los. Tampouco compreendia como é difícil para a imensa maioria dos cidadãos resistir à pressão social do consumo e às tolices cotidianas, como torrar todo o salário em vez de poupar — até porque eu próprio mal sentia essas pressões sociais. Além disso, nos anos 1960, quando os sindicatos norte-americanos desfrutavam de enorme poder, muitos eram de fato arrogantes ou corruptos. Hoje, contudo, estão enfraquecidos, e o resultado dessa debilidade sindical é evidente: qualquer crescimento econômico beneficia quase que exclusivamente os super-ricos.**]** ↩

Ver Judy Steed, *Toronto Star* (9 de outubro de 2000): "Sua visão de software livre e cooperação social contrasta fortemente com a natureza isolada de sua vida pessoal. Excêntrico ao estilo do pianista clássico canadense Glenn Gould, Stallman é igualmente brilhante, articulado e solitário. Considera-se afetado, em certo grau, pelo autismo: uma condição que, segundo ele, torna árdua a interação cotidiana com as pessoas." ↩

Ver Steve Silberman, *"The Geek Syndrome"*, *Wired* (dezembro de 2001). ↩

Ver John Ratey e Catherine Johnson, *Shadow Syndromes*. ↩

Lamentavelmente, não tive a oportunidade de entrevistar Daniel Stallman para esta obra. Durante as pesquisas iniciais, Richard me informou que o pai sofria da doença de Alzheimer em estágio avançado. Ao retomar as apurações em 2001, recebi a triste notícia de que Daniel havia falecido meses antes. ↩

Stallman, sendo ateu convicto, certamente implicaria com a expressão "graça divina" ou "revelação". Fato é que foi um achado providencial. Como declarou a Gross em 1999: "Assim que ouvi falar de computadores pela primeira vez, minha única obsessão na vida passou a ser encontrar um e brincar com ele". ↩

Capítulo 4: O Impeachment de Deus

Embora o relacionamento entre mãe e filho vivesse marcado por atritos e fagulhas, Richard Stallman herdaria de Alice Lippman um traço de personalidade definidor: a paixão visceral pela política progressista.

Tratava-se, no entanto, de uma herança genética que levaria algumas décadas para aflorar na prática. Nos primeiros anos de vida, Stallman cresceu no que hoje reconhece ter sido um "vácuo político total"[1]. A exemplo de milhões de lares norte-americanos sob o governo Eisenhower, a família Stallman passou a década de 1950 tentando recuperar a estabilidade e a normalidade perdidas nos anos de tempestade da Segunda Guerra Mundial.

— O pai do Richard e eu éramos eleitores do Partido Democrata, mas ficávamos nisso — resume Alice ao recordar os anos no Queens. — Não nos metíamos em política municipal nem nos grandes debates nacionais.

O cenário começou a ruir no final daquela mesma década, quando Alice divorciou-se de Daniel Stallman. O retorno a Manhattan foi muito além de uma simples troca de bairro: representou a reinvenção de uma identidade independente e um choque frontal com a aspereza do mundo real.

— Acho que o meu primeiro contato com o ativismo político aconteceu no dia em que fui à biblioteca pública do Queens e descobri estarrecida que havia um único exemplar sobre divórcio no prédio inteiro — recorda Alice. — A Igreja Católica exercia um controle moral asfixiante sobre as instituições, ao menos em Elmhurst, onde morávamos. Aquele episódio acendeu o meu primeiro alerta sobre as forças invisíveis que controlam e engessam a nossa vida cotidiana.

Ao retornar ao bairro onde passara a infância, o Upper West Side de Manhattan, Alice tomou um susto com a metamorfose ocorrida nos quinze anos desde sua partida para o Hunter College. A explosão na demanda por moradia no pós-guerra havia transformado a região em um campo de batalha implacável. De um lado da trincheira perfilavam-se os figurões da prefeitura e especuladores imobiliários, empenhados em pôr quarteirões inteiros abaixo para erguer condomínios sofisticados para a classe média alta de colarinho branco que inundava a ilha. Do outro lado resistia uma população humilde de inquilinos de origem irlandesa e porto-riquenha, que haviam encontrado ali o seu último refúgio acessível.

No início, Alice hesitou sobre qual bandeira empunhar. Como recém-chegada, reconhecia a urgência de novos prédios; como mãe solo sobrevivendo com renda enxuta, contudo, partilhava da aflição dos moradores pobres diante de projetos de "revitalização" pensados sob medida para os ricos. Revoltada, decidiu combater a máquina partidária que tentava transformar seu bairro em uma cópia estéril e aristocrática do Upper East Side.

A primeira ida de Alice ao diretório distrital do Partido Democrata deu-se em 1958. Em busca de uma creche pública para deixar o filho enquanto lecionava, deparou-se com o horror de uma instalação municipal voltada à população de baixa renda:

— A única coisa que guardo daquele lugar é o fedor insuportável de leite azedo, os corredores escuros e mofados e a penúria de materiais. Eu havia dado aulas em jardins de infância particulares; o contraste era estarrecedor. Batemos o olho naquela sala e fomos embora na mesma hora. Aquilo mexeu com os meus brios.

A recepção no diretório do partido, no entanto, foi um banho de água fria. Descrevendo o local como a clássica sala enfumaçada de coronéis da política, Alice percebeu que a corrupção entranhada na máquina era a causa deliberada do desprezo da cidade pelos mais pobres. Em vez de se curvar, uniu-se a uma dissidência reformista empenhada em varrer os resquícios da legendária e corrupta máquina do Tammany Hall. Sob a bandeira do Woodrow Wilson/FDR Reform Democratic Club, Alice e seus companheiros passaram a marcar presença constante em audiências públicas e sessões da câmara de vereadores, exigindo voz ativa nas decisões.

— Nosso alvo primordial era derrubar o Tammany Hall, Carmine DeSapio e seus capangas — explica Alice[2]. — Tornei-me delegada do comitê junto à câmara e mergulhei de cabeça na elaboração de um plano diretor de renovação urbana que fosse além de simplesmente erguer prédios de luxo para milionários.

Aquele batismo de fogo desabrochou em ativismo direto ao longo dos anos 1960. Em 1965, Alice já despontava como uma das cabos eleitorais mais combativas de parlamentares progressistas como William Fitts Ryan — um dos primeiros deputados federais a denunciar a intervenção norte-americana no Vietnã.

Não tardou para que a própria Alice levantasse a voz contra o atoleiro militar no Sudeste Asiático:

— Fui contra a Guerra do Vietnã desde o primeiro minuto em que John F. Kennedy enviou os primeiros contingentes de tropas — afirma ela. — Lia avidamente os despachos dos jornalistas enviados ao conflito e tinha a certeza absoluta de que aquilo se converteria em um pesadelo sem saída.

A repulsa à guerra permeava o ar do lar dos Stallman-Lippman. Em 1967, Alice casou-se novamente. Seu novo marido, Maurice Lippman, major da Guarda Aérea Nacional, pediu baixa imediata de seu posto militar como protesto aberto contra o conflito. O enteado de Alice, Andrew Lippman, estudava no MIT e contava com uma dispensa estudantil temporária; mesmo assim, a ameaça de convocação forçada caso a dispensa caísse — o que de fato aconteceu mais tarde — tornava o fantasma da escalada militar assustadoramente palpável. E havia ainda Richard que, embora mais novo, via o conflito arrastar-se rumo à década de 1970 com o risco iminente de ser engolido pelo alistamento militar obrigatório.

— O Vietnã era o assunto permanente da casa — relata Alice. — Falávamos daquilo o tempo todo: o que faríamos se a guerra continuasse, que atitude Richard ou o meio-irmão tomariam caso fossem convocados. Éramos contra a guerra e contra o recrutamento obrigatório. Considerávamos aquilo profundamente imoral.

Para Richard Stallman, a Guerra do Vietnã despertava uma tempestade contraditória de sentimentos: confusão, horror e uma sensação asfixiante de impotência cívica. Para um garoto que mal suportava as exigências disciplinares moderadas de um colégio particular, o simples vislumbre de um campo de treinamento militar causava calafrios na espinha. Tinha convicção de que não sobreviveria a um quartel com a sanidade intacta.

— Eu vivia paralisado de pavor, mas não fazia a menor ideia do que fazer e não tinha estômago para ir para as ruas protestar — confessa Stallman, cuja data de nascimento (16 de março) lhe rendera um número perigosamente baixo no temido sorteio do alistamento militar. O sorteio não o atingiu de imediato por conta da dispensa universitária — uma das últimas concedidas antes de o governo norte-americano abolir o benefício —, mas a ameaça pairava sobre sua cabeça. — Eu não me imaginava fugindo para o Canadá ou para a Suécia. A perspectiva de juntar minhas coisas sozinho e me mudar para outro país me apavorava. Como eu faria isso? Eu sequer sabia me virar sozinho na vida cotidiana. Eu não era o tipo de pessoa que tinha autoconfiança para encarar uma ruptura daquelas.

Stallman diz ter ficado impressionado com os familiares que tiveram a coragem de agir. Ao lembrar-se de um adesivo impresso e distribuído por seu pai que comparava o massacre de My Lai às atrocidades nazistas na Segunda Guerra Mundial, recorda a emoção provocada por aquele gesto de indignação:

— Admirei imensamente meu pai por ter feito aquilo — reconhece ele. — Mas eu não conseguia me ver fazendo nada parecido. Tinha um medo mortal de ser triturado pelo rolo compressor do recrutamento militar.

Por outro lado, o tom estridente da contracultura juvenil afastava Stallman das passeatas. A exemplo de outros colegas do programa de ciências de Columbia, ele encarava as manifestações estudantis no campus como pouco mais do que um espetáculo histérico[3]. No limite, as forças irracionais que moviam a agitação antiguerra pareciam indistinguíveis da histeria geral da cultura jovem. Em vez de suspirarem pelos Beatles, as garotas da sua idade agora idolatravam incendiários contraculturais como Abbie Hoffman e Jerry Rubin. Para um adolescente que já sofria para decifrar as regras sociais dos colegas, slogans como *"faça amor, não faça guerra"* soavam como deboche pessoal: Stallman não tinha o menor interesse em fazer a guerra nas selvas do Vietnã, mas ninguém parecia disposto a convidá-lo a fazer amor também.

— Eu não engolia aquela contracultura — dispara Stallman. — Detestava as músicas deles. Detestava as drogas; morria de medo de drogas. Sobretudo, abominava o anti-intelectualismo e o preconceito tolo contra a tecnologia. Afinal de contas, eu amava os computadores! E odiava aquele antiamericanismo cego que brotava em cada esquina. Havia pessoas cujo raciocínio era tão simplista e rasteiro que, pelo simples fato de condenarem os crimes dos Estados Unidos no Vietnã, sentiam-se na obrigação moral de apoiar cegamente o regime do Vietnã do Norte. Imagino que fossem incapazes de formular um pensamento um pouco mais sofisticado do que isso.

Observações como essa escancaram uma chave essencial para entender a maturidade política de Stallman: nele, a coragem para agir politicamente sempre dependeu umbilicalmente da sua autoconfiança intelectual. Até 1970, Stallman só se sentia seguro no terreno da matemática e das ciências duras. Contudo, essa mesma segurança matemática lhe deu a base necessária para radiografar os exageros da contracultura pelas lentes da lógica pura — e, ao fazê-lo, encontrou furos imperdoáveis no raciocínio alheio. Embora se opusesse com veemência à carnificina no Vietnã, não via sentido algum em renunciar em abstrato ao uso da força armada quando necessária para defender a liberdade ou estancar tiranias.

Somente na década de 1980, já senhor de sua própria voz e segurança pessoal, Stallman decidiu redimir sua inércia juvenil participando ativamente de marchas multitudinárias em Washington pelos direitos reprodutivos e pelo aborto legal: "Passei a sentir vergonha do meu 'eu' do passado por ter falhado no dever cívico de protestar contra a Guerra do Vietnã", explicaria ele mais tarde.

Em 1970, Stallman deixou para trás as tensas discussões políticas à mesa de jantar e rumou para **Harvard**. Olhando em retrospecto, descreve a transição do apartamento sufocante da mãe para os dormitórios universitários de Cambridge como uma autêntica "fuga". Em Harvard, podia enfim trancar-se no quarto e desfrutar de paz absoluta sempre que quisesse. Os contemporâneos que acompanharam seus primeiros meses no campus, no entanto, viram poucos indícios de uma libertação interior.

— Ele parecia profundamente infeliz no início — lembra Dan Chess, seu colega no programa de ciências que também ingressara em Harvard. — Dava para notar como qualquer interação humana básica era um tormento para ele; e em Harvard não havia como fugir disso. Harvard era um ambiente socialmente asfixiante e gregário.

Para não sucumbir ao desespero, Stallman refugiou-se em seu porto seguro: a matemática de vanguarda. Como quase todos os egressos do programa de ciências de Columbia, gabaritou sem esforço o exame seletivo para a lendária disciplina **Math 55** — o mítico "campo de concentração" para calouros de ciências exatas da universidade. Dentro daquela turma de elite, os garotos de Nova York fecharam fileiras: "Éramos a máfia da matemática", diverte-se Chess. "Harvard perto do nosso cursinho de Nova York não era nada!".

Contudo, para ostentar tamanho orgulho, era preciso sobreviver ao Math 55. Prometendo condensar quatro anos de currículo avançado em míseros dois semestres, o curso selecionava apenas mentes extraordinárias.

— Foi uma experiência assombrosa — define David Harbater, antigo membro da máfia matemática e hoje professor titular da University of Pennsylvania. — É seguro dizer que jamais existiu no planeta uma turma para calouros que fosse tão brutalmente intensa e adiantada. A frase que costumo soltar para que as pessoas compreendam a dimensão da coisa é que, no segundo semestre, já estávamos debatendo a geometria diferencial de variedades de Banach. É nessa hora que os olhos dos matemáticos saltam das órbitas, porque a imensa maioria dos mortais só ouve falar de variedades de Banach no segundo ano de doutorado!

Começando com setenta e cinco estudantes no primeiro dia, o Math 55 derreteu até sobrar apenas uma vinte alunos no segundo semestre. Daquele grupo de sobreviventes, relata Harbater, "apenas dez realmente sabiam o que estavam fazendo ali". Desses dez prodígios, oito se tornariam professores catedráticos de matemática e um lecionaria física teórica.

— O décimo restante — faz questão de enfatizar Harbater — era Richard Stallman.

Seth Breidbart recorda que Stallman já se destacava dos demais gênios da sala por sua precisão obsessiva:

— Ele era irredutível de um jeito muito singular. Há uma técnica metodológica consagrada na matemática que praticamente todo mundo executa de forma incorreta — um vício de notação em que você precisa definir uma função para algo e o que você faz é definir a função e só depois provar que ela está bem definida. Na primeira vez em que o Richard subiu ao quadro para apresentar a resolução de um problema, ele primeiro definiu uma relação matemática e só depois provou que aquela relação era, rigorosamente falando, uma função. Era rigorosamente a mesma demonstração, mas ele usava a terminologia matematicamente impecável que nenhum outro aluno empregava. Ele simplesmente operava assim.

Foi nas salas do Math 55 que Richard Stallman cimentou sua fama de mente genial. Breidbart concorda, mas Chess, cujo instinto competitivo recusava-se a ceder, confessa que só se curvou à superioridade avassaladora de Stallman no ano seguinte:

— Foi durante uma aula avançada de Análise Real — relata Chess. — Lembro-me com nitidez de uma demonstração intrincada sobre medidas de valor complexo em que o Richard sacou uma solução alicerçada em uma metáfora vinda diretamente do cálculo variacional. Foi a primeiríssima vez na minha vida em que vi alguém resolver um problema cabeludo com uma originalidade tão deslumbrante.

Para Chess, o choque foi doloroso: foi como uma ave que bate de frente contra uma vidraça cristalina, compreendendo tarde demais que certas alturas do intelecto humano simplesmente não estavam ao seu alcance:

— O fascínio da matemática é esse: você não precisa ser um matemático de primeira grandeza para reconhecer instantaneamente um talento de primeira grandeza ao seu lado. Eu sabia que estava entre os melhores, mas ali me dei conta de que eu não era um gênio de primeira linha. Se Richard tivesse optado por seguir carreira na matemática pura, ele teria sido um dos maiores matemáticos do século [4].

No entanto, as vitórias intelectuais de Stallman continuavam contrastando com sua inaptidão nas relações humanas. Enquanto os colegas de turma se reuniam para resolver os exercícios hercúleos do Math 55 em grupo, ele insistia em trabalhar isolado. O mesmo padrão repetiu-se na moradia estudantil: no formulário de habitação de Harvard, expressou sua preferência em termos literais: "Gostaria de ter um colega de quarto invisível, inaudível e intangível". Em um raro lampejo de sensibilidade burocrática, a administração acolheu o pedido e designou-lhe um quarto individual para o primeiro ano.

Breidbart, o único membro da "máfia da matemática" a partilhar do mesmo prédio que Stallman naquele outono, relata que o colega foi aprendendo, a duras penas, a interagir com os demais estudantes. Impressionados com a argúcia implacável do rapaz, os vizinhos de dormitório passaram a acolher suas ponderações nas tradicionais sessões de debate filosófico nas madrugadas da faculdade:

— Tínhamos aquelas velhas conversas sem fim tentando consertar os dilemas do mundo — lembra Breidbart. — Imaginávamos, por exemplo: e se alguém descobrisse o soro da imortalidade? O que a humanidade deveria fazer? Quais seriam os impactos políticos? Se você distribuísse o soro a todos, a Terra entraria em colapso populacional e todos morreriam de fome. Se você limitasse o acesso aos que já estivessem vivos, proibindo a próxima geração de recebê-lo, criaria uma casta inferior de mortais. O Richard sempre foi incomparavelmente melhor do que qualquer um de nós em prever os desdobramentos invisíveis e as consequências não intencionais de qualquer decisão.

Stallman guarda lembrança vívida daquelas sessões:

— Eu sempre fui calorosamente a favor da imortalidade humana — explica ele. — De que outra maneira poderíamos testemunhar com os próprios olhos como estará o mundo daqui a duzentos anos?

Curioso, começou a abordar conhecidos nos corredores para sondar se aceitariam a imortalidade caso lhes fosse oferecida:

— Fiquei profundamente escandalizado ao constatar que a maioria encarava a imortalidade como algo perverso. Muitos me diziam que a morte era algo bom porque não fazia sentido viver uma velhice decrépita, e que o envelhecimento era bom porque preparava as pessoas para a morte — sem jamais se darem conta do círculo vicioso ridículo desse argumento!

Apesar de ser reverenciado como um matemático magistral e debatedor de primeira linha, Stallman fugia de competições formais que pudessem chancelar seu brilho acadêmico. Perto do final do primeiro ano, Breidbart estranhou quando Richard boicotou ostensivamente o prestigiadíssimo exame William Lowell Putnam — a maratona anual de matemática aberta a estudantes universitários de todos os Estados Unidos e do Canadá. Além de permitir medir forças com os melhores talentos do continente, o Putnam servia de vitrine de recrutamento para as cátedras de pós-graduação: a lenda nos campi dizia que o vencedor do exame garantia uma bolsa integral de doutorado em qualquer universidade do mundo que escolhesse.

Assim como o Math 55, o exame Putnam era uma peneira brutal: seis horas de prova divididas em duas etapas, concebidas expressamente para separar o joio do trigo. Breidbart classifica o exame como a prova mais implacável que realizou na vida: "Para se ter uma ideia do nível de crueldade, a pontuação máxima era 120; no meu primeiro ano, fiz pouco mais de 30 pontos — e essa nota me garantiu a 101ª posição no ranking de todo o país!".

Incrédulo diante da ausência da mente mais brilhante da classe, Breidbart e um colega encurralaram Stallman no refeitório exigindo explicações:

— Ele me confessou que tinha medo de não ir bem — revela Breidbart.

Ali mesmo, os dois rabiscaram algumas questões da prova de cabeça e passaram a folha para Stallman:

— Ele resolveu todas com uma facilidade desconcertante — prossegue Breidbart. — Aquilo me levou a concluir que, na cabeça dele, "não ir bem" significava tirar o segundo lugar ou errar uma única vírgula.

Stallman reconstrói o episódio com menos triunfalismo:

— Lembro-me de que eles me trouxeram as perguntas e é bem possível que eu tenha resolvido uma ou outra, mas tenho certeza absoluta de que não resolvi todas.

Contudo, Stallman corrobora o cerne da recordação de Breidbart: o medo paralisante do fracasso era o motor do seu boicote. A despeito de não hesitar em apontar as falhas intelectuais de mestres e colegas em plena aula, ele detestava competições diretas mano a mano. Para que se arriscar se podia simplesmente ignorá-las?

— É a mesmíssima razão pela qual sempre detestei o xadrez — desabafa Stallman. — Toda vez que eu me sentava para jogar, era engolido por um pavor tão avassalador de cometer um único deslize e perder a partida que começava a fazer jogadas estúpidas logo na abertura. O medo virava uma profecia autorrealizável.

A solução pragmática foi simples: extirpou o xadrez de sua vida.

Se esse temor visceral acabou podando uma promissora carreira na matemática teórica é matéria de especulação. Fato é que, ao término do primeiro ano em Harvard, outras forças poderosas começavam a puxá-lo para longe do departamento de matemática: a programação de computadores, fascínio adormecido desde a adolescência, desabrochava em paixão arrebatadora. Enquanto os outros calouros de exatas buscavam descanso em aulas de literatura ou história da arte, Stallman encontrava seu verdadeiro refúgio nas salas do centro de informática.

O primeiro contato nos laboratórios da IBM em Nova York plantara nele a semente da curiosidade insaciável:

— No final do meu primeiro ano em Harvard, criei coragem para começar a visitar os centros de processamento de dados e ver o que eles tinham por lá. Eu pedia para ver se tinham cópias sobrando de manuais técnicos que eu pudesse levar para casa e ler.

Devorando as brochuras no quarto, Stallman estudava as especificações de cada arquitetura para compreender as minúcias das diferentes filosofias de computação da época.

Foi então que, nos estertores de seu ano de calouro, ouviu falar de um laboratório singular instalado nas cercanias do MIT. O centro ficava encravado no nono andar de um edifício no Tech Square — um complexo de escritórios comerciais e acadêmicos que o instituto erguera do outro lado da rua de seu campus principal. Segundo os boatos, aquele santuário era dedicado à ciência futurista da Inteligência Artificial e contava com computadores de última geração e softwares incomparáveis para a época.

Fascinado pelos relatos, Stallman resolveu atravessar o rio para conferir.

A distância era curta: menos de três quilômetros a pé ou meros dez minutos de metrô. Ao desembarcar do outro lado, contudo, Stallman descobriu que Harvard e o MIT operavam como dois hemisférios opostos do mesmo planeta. Com seu labirinto infinito de prédios industriais interconectados por corredores sombrios e tubulações expostas, o campus do MIT oferecia a antítese visual da harmonia colonial e dos gramados aristocráticos de Harvard. Para o temperamento de Stallman, aquele caos arquitetônico do MIT combinava muito mais com sua alma. O mesmo valia para a fauna de alunos: uma legião de jovens esquisitos, desajustados do ensino médio que preferiam gastar a inteligência aprontando trotes científicos geniais do que polindo currículos para cortejar o establishment corporativo.

A mesma dualidade imperava no interior do **AI Lab**. Ao contrário dos centros de computação de Harvard, ali não havia monitores de gravata fazendo triagem na porta, não havia pranchetas burocráticas disputando turnos de acesso nem aquele ambiente gélido de "olhe mas não toque". Stallman deparou-se com uma constelação de terminais abertos e braços robóticos espalhados pelas salas — resquícios de algum experimento de visão computacional. Ao cruzar com um funcionário do laboratório, perguntou candidamente se teriam manuais extras para emprestar a um estudante curioso:

— Eles tinham alguns, mas a maior parte das coisas sequer estava documentada — recorda Stallman com um sorriso irônico. — Afinal de contas, eles eram hackers! — arremata, aludindo ao hábito clássico da cultura hacker de pular de cabeça no projeto seguinte sem perder tempo redigindo manuais para o anterior.

Stallman saiu daquela primeira visita com algo infinitamente mais valioso do que um calhamaço de manuais de papel: saiu com uma **proposta de trabalho**. Sua primeira missão seria escrever um simulador do minicomputador PDP-11 para rodar dentro da máquina central PDP-10. Na semana seguinte, apresentou-se pontualmente no nono andar do Tech Square, sentou-se diante do primeiro terminal desocupado que encontrou e começou a digitar as primeiras linhas de código.

Olhando para trás, Stallman não enxerga nada de surpreendente na naturalidade com que o AI Lab acolheu um novato desconhecido no primeiro minuto:

— As coisas funcionavam desse jeito naquela época — explica ele. — E é assim que ainda deveriam funcionar hoje. Se eu conheço alguém e percebo que o sujeito é bom de serviço, contrato na hora! Para que esperar? Essa gente engessada que faz questão de enfiar burocracia em cada fresta da vida perde completamente o foco das coisas. Se uma pessoa é brilhante, ela não tem que passar por uma maratona humilhante de entrevistas com o departamento de recursos humanos; ela tem que sentar diante de um teclado e escrever código.

Para experimentar na pele o veneno do engessamento burocrático, bastava a Stallman retornar aos computadores de Harvard. Lá, o tempo de tela era distribuído de acordo com a hierarquia da cátedra acadêmica. Na qualidade de simples estudante de graduação, Stallman frequentemente tinha de amargar horas na fila. Esperar por uma vaga

nos terminais públicos sabendo que meia dúzia de máquinas idênticas e operantes acumulavam poeira em salas trancadas a chave por professores ausentes parecia-lhe o cúmulo do desperdício estúpido. Embora ainda fizesse visitas esporádicas aos laboratórios de Harvard, seu coração já pertencia à política igualitária e desprovida de salamaleques do AI Lab:

— Foi uma lufada de ar fresco — desabafa ele. — No AI Lab do MIT, as pessoas estavam genuinamente preocupadas com o trabalho prático, e não com diplomas ou títulos nobiliárquicos.

Stallman não demorou a entender que a política de livre acesso do laboratório dependia da vigília feroz de um punhado de rebeldes. Vários eram veteranos do mítico **Projeto MAC** — o ousado programa de pesquisas financiado pelo Departamento de Defesa dos EUA que dera à luz os primeiros sistemas operacionais com compartilhamento de tempo (*time-sharing*). Alguns já eram lendas vivas da ciência da computação: **Richard Greenblatt**, o mago residente da linguagem Lisp e criador do *MacHack* — o pioneiro programa de xadrez digital que calara a boca do filósofo Hubert Dreyfus, crítico feroz da inteligência artificial; **Gerald Sussman**, autor do programa pioneiro de manipulação robótica *HACKER*; e **Bill Gosper**, prodígio da matemática que passara nada menos que dezoito meses ininterruptos em um transe obsessivo de programação, explorando as implicações filosóficas e matemáticas do célebre jogo *LIFE*, inventado pelo matemático britânico John Conway[5].

Os membros desse clã fechado autodenominavam-se orgulhosamente **hackers**. Com o passar dos meses, acolheram o jovem calouro de Harvard em sua irmandade e passaram a chamá-lo de hacker também. Ao fazê-lo, transmitiram a Stallman as tábuas da lei da **Ética Hacker**: movidos pelo desejo obsessivo de explorar até os limites o que um computador era capaz de realizar, aqueles sujeitos podiam permanecer plantados diante do terminal por 36 horas seguidas se estivessem diante de um enigma fascinante. Acima de tudo, exigiam acesso irrestrito às máquinas quando ninguém estivesse usando-as e defendiam a livre circulação de informações técnicas. Falavam abertamente sobre transformar o mundo através do software, e Stallman assimilou o desprezo instintivo dos hackers por qualquer obstáculo que se interpusesse a essa missão sagrada. No topo dessa lista de entraves figuravam três pecados capitais: software mal escrito, burocracia acadêmica e egoísmo mesquinho.

Stallman também absorveu as lendas da casa: histórias de como os pioneiros, diante de portas trancadas, recorriam a saídas mirabolantes para garantir que as ferramentas de pesquisa continuassem ao alcance de todos. Isso incluía invadir gabinetes de professores catedráticos para "libertar" terminais sequestrados. Ao contrário dos catedráticos mimados de Harvard, os docentes do MIT sabiam muito bem que não podiam tratar as escassas máquinas do laboratório como bens privados. Se algum professor cometia o deslize de trancar um terminal dentro de sua sala antes de ir para casa, os hackers não pensavam duas vezes: arrombavam a fechadura, devolviam o terminal à circulação pública e ainda deixavam um bilhete malcriado repreendendo o docente por ter conspirado contra o bem comum.

Alguns realizavam essas invasões destrancando os miolos das fechaduras com arames finos (*lock hacking*); outros removiam as placas de gesso do forro do teto e rastejavam sobre as divisórias para pular lá dentro; outros ainda, aproveitando o piso elevado do nono andar por onde corriam os cabos de força, esgueiravam-se pelo subsolo como espeleólogos urbanos. "Chegaram a me mostrar uma espécie de aríete improvisado montado sobre um carrinho com um pesado cilindro de aço, usado certa vez para botar a porta do gabinete de um professor abaixo!", conta Stallman[6].

A intransigência dos hackers cumpria uma função social vital: impedia que a vaidade acadêmica sabotasse o andamento das pesquisas. Os hackers não ignoravam as demandas particulares dos professores, mas exigiam que elas fossem atendidas sem prejudicar o restante da comunidade. Por exemplo: quando algum professor alegava que precisava trancar a sala para proteger papéis confidenciais ou objetos pessoais contra furtos, os hackers respondiam de pronto: "Ninguém vai reclamar se você trancar a sua porta — embora isso não seja lá muito simpático —, contanto que você não tranque o terminal do laboratório lá dentro!".

Embora os acadêmicos convencionais superassem os hackers em número no AI Lab, a Ética Hacker reinava soberana. Afinal, eram os hackers da equipe e os estudantes que haviam desenhado os circuitos das máquinas, soldado os componentes e escrito praticamente todo o software que os demais pesquisadores usavam para trabalhar. Eram eles que mantinham o sistema em pé. O trabalho deles era insubstituível, e eles não aceitavam ser pisoteados. Dedicavam-se tanto a projetos demandados pelos usuários quanto a melhorias que brotavam de pura curiosidade técnica; como jovens mecânicos de racha apaixonados por turbinar motores, encaravam o ato de fuçar nas entranhas das máquinas como a mais pura e refinada forma de arte.

Em nenhum lugar esse impulso libertário se materializou de forma mais perfeita do que no sistema operacional que gerenciava o computador central do laboratório: o mítico **ITS** (*Incompatible Timesharing System*). Desenvolvido pelos próprios hackers como um protesto explícito contra a arquitetura engessada do CTSS (*Compatible Time-Sharing System*) do Projeto MAC — daí o batismo irônico de "Sistema Incompatível" —, o ITS trazia a alma da cultura hacker gravada em suas instruções mais fundamentais. Na época, os técnicos consideravam o CTSS excessivamente restritivo, podando a liberdade dos desenvolvedores para alterar e expandir o núcleo do sistema sob demanda. A tradição oral do laboratório assegura ainda que o nascimento do ITS guardava forte tempero político: enquanto o CTSS fora desenhado para os computadores gigantes da IBM, o ITS foi talhado para rodar no novo PDP-6. Ao permitir que os hackers escrevessem seu próprio sistema operacional do zero, a diretoria do AI Lab garantiu que apenas os hackers soubessem pilotar aquela máquina com destreza. Naquele tabuleiro feudal de disputas orçamentárias entre departamentos universitários, o estratagema funcionou com precisão cirúrgica: os outros departamentos desistiram de disputar tempo de processamento, e o laboratório de IA conquistou a posse exclusiva da máquina. Alavancado pelo ITS e pelo PDP-6, o laboratório pôde declarar sua independência formal em relação ao Projeto MAC pouco antes da chegada de Stallman.

Em 1971, o ITS migrou para o novíssimo PDP-10, deixando a máquina antiga para tarefas auxiliares. O PDP-10 do laboratório de inteligência artificial contava com uma memória assombrosa para os padrões de 1971 — o equivalente a pouco mais de um megabyte —, duplicada no final daquela década. Os hackers de hardware chegaram ao cúmulo de desenhar e soldar circuitos próprios de memória virtual paginada — um recurso revolucionário que o fabricante da máquina não oferecia de fábrica.

Na condição de jovem aprendiz, Stallman apaixonou-se perdidamente pelo ITS. Embora causasse estranheza em usuários leigos, o sistema ostentava recursos que o mercado corporativo levaria décadas para implementar (e que muitos sistemas não oferecem até hoje), como multitarefa real, a capacidade de acoplar o depurador de código imediatamente a qualquer programa em execução e edição de texto em tela cheia:

— O ITS contava com um mecanismo interno de uma elegância primorosa para permitir que um programa examinasse as entranhas de outro — derrete-se Stallman. — Você podia monitorar todas as variáveis de estado de outro processo de forma limpa, transparente e perfeitamente documentada.

Outro recurso adorado permitia congelar o processamento de um programa de forma cirúrgica, exatamente entre duas instruções da máquina. Em outros sistemas operacionais, operações desse tipo interrompiam a execução no meio de uma chamada de sistema opaca, deixando variáveis internas em estados indefinidos e inacessíveis ao programador. No ITS, o controle era total:

— Se você ordenasse "pare o processo", ele sempre congelava em modo de usuário, exatamente no intervalo entre duas instruções bem comportadas, com todas as variáveis estáveis e visíveis — explica Stallman. — Se você desse a ordem de retomar, ele continuava o trabalho perfeitamente. E mais: se você alterasse o estado interno do programa antes de mandá-lo prosseguir e depois desfizesse a alteração, tudo permanecia perfeitamente consistente. Não havia segredos nem armadilhas escondidas.

A partir de setembro de 1971, o laboratório do MIT converteu-se no eixo central da rotina semanal de Stallman. De domingo a sexta-feira, ele cumpria expediente como estudante em Harvard; contudo, bastava despontar a tarde de sexta-feira para que ele corresse ao metrô rumo ao Tech Square. Stallman costumava chegar com folga antes do sagrado ritual da caça à comida: juntando-se a cinco ou seis companheiros de terminal, apinhava-se dentro de um carro caindo aos pedaços e cruzava a ponte de Harvard rumo ao bairro chinês de Boston. Pela hora seguinte, debruçados sobre pratos fumegantes, aquele bando de mentes brilhantes debatia de tudo: dos ponteiros do ITS às sutilezas lógicas dos ideogramas da língua chinesa. Terminado o banquete, regressavam ao MIT e viravam a noite programando até o amanhecer — quando não decidiam voltar à Chinatown às três da madrugada para uma nova rodada.

Stallman costumava varar a manhã de sábado codificando sem parar; outras vezes desabava em um sofá qualquer do laboratório. Ao acordar, engatava nova sessão de programação, devorava outro banquete chinês e só então pegava o metrô de volta a Harvard — isso quando não estendia a vigília até a noite de domingo. Aqueles jantares asiáticos não eram apenas deliciosos: eram a salvação biológica contra a comida intragável dos refeitórios de Harvard, onde, em média, apenas uma refeição por dia continha algo que seu estômago tolerava. (O café da manhã sequer entrava no cômputo, já que Stallman detestava as opções matinais convencionais e costumava passar aquelas horas dormindo).

Para o jovem desajustado que passara a adolescência marginalizado pelos colegas de colégio, conviver com pessoas que partilhavam do mesmo amor visceral por computadores, ficção científica e banquetes chineses foi uma experiência inebriante:

— Lembro-me com imenso carinho de tantos e tantos pores do sol e amanheceres vistos de dentro do carro voltando da Chinatown — recordaria Stallman quinze anos mais tarde, comovido, em uma conferência no Instituto Real de Tecnologia da Suécia. — Ver o sol nascer no horizonte era uma das coisas mais belas do mundo, porque aquela é a hora mais serena e tranquila do dia. É o momento perfeito para se recolher para a cama. Caminhar de volta para casa com a luz do dia clareando o asfalto e os passarinhos começando a cantar no parque traz uma sensação indescritível de paz, de doce plenitude pelo trabalho honesto que você realizou ao longo daquela madrugada[7].

Quanto mais tempo Stallman passava ao lado dos pioneiros do AI Lab, mais assimilava aquela cosmovisão comunitária. Já devoto da liberdade pessoal, começou a impregnar seus passos com um profundo senso de **dever comunitário**. Quando alguém feria o código de honra da casa, Stallman era o primeiro a erguer a voz. Em menos de um ano de convivência, era ele próprio quem liderava as incursões noturnas para recuperar terminais confiscados. No melhor espírito hacker, tratou de dar sua contribuição autoral à arte de abrir fechaduras: um dos

truques mais requintados, atribuído a Richard Greenblatt, consistia em dobrar um arame rígido em sucessivos ângulos retos, colar uma fita adesiva na ponta e passá-lo por baixo da fresta da porta até colar na maçaneta interna, girando-a do lado de fora.

Ao tentar replicar a façanha, Stallman achou o método instável demais: a fita nem sempre colava com firmeza e girar o arame sem perder a tração era tarefa ingrata. Outros subiam pelo forro falso do teto, mas saíam de lá cobertos de pó de fibra de vidro, coçando-se inteiros. Stallman bolou uma solução original: em vez de enfiar o arame por baixo da soleira ou se arranhar no teto, por que não levantar apenas duas placas do forro e descer um laço por cima da parede?

Subindo em uma cadeira apoiada em cima de uma escrivaninha, Stallman descia uma longa tira dobrada em "U" feita de fita magnética pesada, trazendo em sua base um pedaço de fita adesiva com a cola virada para cima. Equilibrando-se com destreza, manobrava a fita até laçar a maçaneta do lado de dentro do gabinete. Bastava puxar uma das pontas para que a maçaneta girasse e o trinco se abrisse:

— Às vezes era preciso dar um chute de leve na porta na hora exata em que você girava a maçaneta por cima — lembra ele, detalhando o truque. — Exigia um bocado de equilíbrio para não despencar da cadeira em cima da mesa, mas abria que era uma beleza!

Traquinagens dessa ordem revelavam a crescente prontidão de Stallman para agir em defesa de suas convicções. O espírito de ação direta do laboratório curou de vez a timidez paralisante de sua adolescência. Arrombar a sala de um professor autoritário para devolver um terminal à coletividade não equivalia a marchar em uma manifestação política de massas, mas tinha uma virtude que as passeatas raramente exibiam: **resolvia o problema prático na hora**.

Nos seus últimos anos em Harvard, Stallman passou a transplantar as lições de irreverência e desobediência civil do MIT para dentro de seu próprio campus universitário:

— Ele chegou a contar a história da cobra? — dispara Alice Lippman durante a entrevista. — O Richard e os companheiros de dormitório registraram uma cobra viva como candidata às eleições estudantis! E ela recebeu uma enxurrada de votos!

A serpente concorreu às eleições internas da Currier House — o dormitório onde Stallman residia. Ele recorda o episódio com bom humor:

— Muita gente votou nela achando que estava votando no dono do bicho, já que o réptil e o estudante compartilhavam o mesmo sobrenome. Espalhamos cartazes pela faculdade anunciando que a candidata estava "rastejando rumo à vitória". Como a cobra havia escapado pela grade de ventilação semanas antes e ninguém sabia em que buraco da parede ela estava escondida, fizemos campanha dizendo que ela era uma candidata "em fuga" (*at large*).

Em outra eleição caricata, Stallman e os amigos lançaram a candidatura do filho de três anos do reitor do dormitório: "A principal bandeira da campanha dele era a aposentadoria compulsória aos sete anos de idade!", diverte-se.

Piadas desse calibre, contudo, empalideciam perto dos trotes eleitorais organizados pelos estudantes do MIT. Um dos casos mais emblemáticos foi o do gato Woodstock, felino adotado pelos alunos que conseguiu superar em votos quase todos os candidatos de carne e osso na disputa para o diretório central dos estudantes:

— A reitoria nunca teve a coragem de divulgar o número exato de votos que o gato recebeu; anularam todos os boletins de urna como votos nulos — relata Stallman. — Mas a quantidade absurda de cédulas invalidadas indicava com clareza que o Woodstock havia vencido a eleição com folga! Dois anos depois, o gato foi misteriosamente atropelado por um carro no campus. Até hoje ninguém sabe se o motorista trabalhava a mando da reitoria! Eu não tive nada a ver com o lançamento do Woodstock, mas aplaudi a iniciativa de pé[8].

No front do AI Lab, contudo, os embates políticos de Stallman adquiriram contornos muito mais sérios e combativos. Ao longo da década de 1970, os veteranos travaram uma guerra de trincheiras contra administradores e docentes recém-chegados que tentavam adulterar o desenho aberto do ITS. O sistema fora concebido de modo a permitir que qualquer usuário sentasse diante do console e fizesse o que bem entendesse — inclusive emitir um comando para desligar o computador em cinco minutos. Se alguém abusava do privilégio e ordenava um desligamento arbitrário sem justificativa plausível, qualquer outro usuário cancelava o comando no mesmo instante e a vida seguia. Em meados daquela década, contudo, professores formados em outras escolas começaram a exigir a implementação de sistemas de segurança e senhas para blindar o acesso aos seus arquivos pessoais. Em outros ambientes acadêmicos e comerciais, ferramentas de controle desse tipo eram padrão; aqueles docentes estavam habituados a viver sob a tutela de senhas e à ilusão de proteção que elas proporcionavam. Mas o AI Lab, graças à resistência inflexível de Stallman e de seus pares, permaneceu como uma **zona livre de senhas**.

Stallman empunhava argumentos éticos e técnicos contra a introdução de travas de segurança. Sob o ponto de vista ético, apelava para o pacto histórico de confiança e transparência que unia o laboratório. Sob o prisma técnico, apontava para o esqueleto do ITS: o sistema fora construído de cabo a rabo para estimular a colaboração e a partilha, e não para vigiar ou punir os usuários. Qualquer tentativa de subverter essa arquitetura exigiria reescrever o núcleo inteiro do zero. Para inviabilizar manobras futuras, Stallman tomou uma providência genial: utilizou o último campo livre disponível no descritor de arquivos do sistema para programar uma rotina que registrava automaticamente o nome do último usuário que havia alterado o arquivo. O novo recurso ocupou o espaço que poderia ser usado para guardar permissões de segurança de acesso — e como a ferramenta de auditoria de autoria era incrivelmente útil para todos, ninguém no laboratório teve coragem de propor sua remoção.

— Os hackers que construíram o ITS compreenderam desde cedo que privilégios de controle de arquivos costumam ser usados por administradores de rede autoproclamados para acumular poder e mandar na vida alheia — explicaria Stallman anos mais tarde. — Eles não queriam que ninguém tivesse o poder de tiranizá-los, por isso recusaram-se a programar esse tipo de trava. O resultado maravilhoso disso é que, quando qualquer engrenagem do sistema quebrava, você tinha permissão total para meter a mão e consertar sem burocracia, porque ninguém colocava senhas no seu caminho[9].

Graças a essa obstinação heroica, os hackers mantiveram os computadores centrais do AI Lab livres de senhas por anos a fio. Contudo, em outro laboratório vizinho — o Laboratory for Computer Science (LCS) do MIT —, os entusiastas da segurança venceram a parada. Em 1977, o grupo DM do LCS instalou o primeiro sistema de autenticação por senha do prédio.

Mais uma vez, Stallman chamou para si a responsabilidade de corrigir o que considerava um relaxamento moral inaceitável. Conseguindo acesso ao código-fonte do módulo de senhas, programou uma rotina para decodificar todas as senhas criptografadas guardadas no banco de dados. Em seguida, disparou uma campanha implacável de e-mails para cada um dos usuários do sistema, exortando-os a adotarem uma **cadeia de caracteres nula** (*null string*) como senha. Se o usuário havia cadastrado a palavra "estrela-do-mar", por exemplo, recebia uma mensagem personalizada nestes termos:

> *"Vejo que você escolheu a senha 'estrela-do-mar'. Sugiro fortemente que altere sua senha para um simples 'Enter' (retorno de carro), que é a senha que eu próprio utilizo. É incomparavelmente mais fácil de digitar e expressa a nossa oposição frontal à imposição de senhas e barreiras de segurança."*

Os usuários que acatavam o conselho e cadastravam a tecla Enter — isto é, deixavam o campo de senha em branco — abriam suas contas para o acesso irrestrito do mundo inteiro, exatamente como a computação sempre fora até ali. O objetivo era escancarar o ridículo da situação: ao se recusarem a trancar os cadeados novinhos em folha de suas contas, ridicularizavam a própria existência dos cadeados. Sabiam que aquela segurança capenga jamais barraria um invasor mal-intencionado de verdade — e que isso sequer importava, pois ninguém estava ali para cometer crimes, mas apenas para visitar, aprender e colaborar.

Em entrevista histórica concedida a Steven Levy para o livro *Hackers*, Stallman gabou-se de que nada menos que **um quinto de toda a equipe do LCS** curvou-se aos seus argumentos e adotou a senha em branco em seus computadores[10].

A cruzada da senha nula — e a resistência quixotesca de Stallman contra os mecanismos de segurança de computadores — acabaria sendo derrotada com o tempo. No início dos anos 1980, até as máquinas do laboratório de inteligência artificial ostentariam telas de login obrigatórias. Ainda assim, aquele levante solitário marcou um divisor de águas crucial na maturação pessoal e política de Richard Stallman. Visto em perspectiva histórica, representou o salto definitivo pelo qual aquele adolescente tímido, apavorado com o mundo e incapaz de protestar contra uma guerra sangrenta, converteu-se no ativista destemido que faria da desobediência ética a sua missão em tempo integral.

Ao levantar a voz contra a segurança nos computadores, Stallman canalizou todas as obsessões que moldaram sua infância: a sede voraz por conhecimento, a aversão visceral à autoridade e a dor profunda diante de regras secretas e preconceitos que condenavam pessoas ao ostracismo. Incorporou também os princípios morais que nortearam sua maturidade: a responsabilidade coletiva com a comunidade, a confiança mútua e o compromisso da ação direta. Em termos da ciência da computação, a campanha da senha nula foi nada menos que a **versão 1.0 da cosmovisão política de Richard Stallman** — ainda imatura em alguns detalhes operacionais, mas já plenamente forjada em sua essência ética.

Olhando para trás, Stallman evita inflar o heroísmo daquele episódio:

— Naquela fase inicial, muita gente comungava dos mesmos sentimentos que eu — pondera ele. — A adesão maciça à senha nula comprovou que muitos pesquisadores concordavam que aquilo era o correto a se fazer. A única diferença é que eu tinha a disposição nata para ir à luta e comprar a briga.

Ele não poupa reverências, contudo, ao papel que o laboratório desempenhou no despertar dessa consciência cívica:

— Eu ingressei em uma comunidade que havia construído um modo de vida alicerçado no respeito mútuo à liberdade alheia — reflete Stallman. — Não demorei nada para perceber que aquilo era algo maravilhoso. O que me levou mais tempo foi alcançar a certeza absoluta de que aquilo era, acima de tudo, um imperativo moral inegociável.

Programar no AI Lab não foi a única válvula de escape a turbinar a autoestima de Stallman naquele período. No início do seu terceiro ano em Harvard, começou a frequentar um grupo amador de **dança folclórica internacional** que acabara de nascer na Currier House. A princípio, relutara em pisar no salão, julgando-se irremediavelmente destituído de coordenação para a dança; mas um amigo puxou sua orelha com uma verdade desconcertante: "Você não tem como afirmar que não sabe dançar se nunca tentou na prática!". Para seu próprio assombro, descobriu não apenas que levava jeito para a coisa, mas que adorava a experiência. O que começara como uma experiência tímida transformou-se em paixão arrebatadora, disputando espaço com os estudos e com as madrugadas diante dos computadores. Era também uma rara oportunidade de interagir com mulheres — embora não tenha descolado um único namoro ao longo de toda a sua trajetória universitária.

No salão de dança, Stallman libertava-se do trauma daquele menino desengonçado de dez anos que chorava de raiva nas peladas de futebol americano. Sentia-se leve, ágil, seguro e vivo. Nos anos 1980, levou a paixão tão a sério que ingressou no grupo oficial de dança folclórica do MIT. Apresentando-se diante de plateias lotadas, trajando réplicas de vestimentas tradicionais de camponeses dos Bálcãs, descobriu o prazer de ocupar o centro do palco — uma desenvoltura cênica que se revelaria preciosa anos mais tarde em suas palestras pelo mundo.

Embora a dança e o hacking não tenham curado por completo suas dores de sociabilidade, ajudaram-no a exorcizar o fantasma do isolamento que ensombrara seus anos de colégio. Em 1977, ao pisar pela primeira vez em uma convenção de ficção científica, deparou-se com o estande de uma artesã chamada Nancy, que fabricava botons personalizados escrevendo à mão o que o cliente pedisse. Encantado com a ideia, Stallman encomendou um boton ostentando duas palavras provocativas em letras garrafais: **"IMPEACH GOD"** (*Destituam Deus* / *Impeachment de Deus*).

Para a mente de Stallman, a mensagem do boton operava em múltiplas camadas intelectuais. Ateu convicto desde a mais tenra infância, ele encarava o lema como uma tentativa ousada de abrir uma "segunda frente" no debate secular sobre a religião:

— Naquela época, todo mundo se perdia em discussões intermináveis sobre se Deus existia ou não — recorda Stallman. — O slogan *'Impeach God'* atacava a questão por um flanco completamente novo. Ora, se existisse um deus tão todo-poderoso a ponto de criar o universo inteiro do nada e que, ainda assim, não movesse uma palha para estancar as desgraças e as injustiças da Terra, por que cargas d'água nós deveríamos adorar uma divindade dessas? Não seria infinitamente mais justo levá-la a julgamento e cassar os seus poderes?

Simultaneamente, o boton dialogava de forma direta com o escândalo de **Watergate** — o terremoto político dos anos 1970 que culminara na renúncia desonrosa do presidente Richard Nixon. Watergate marcara a consciência de Stallman com ferro em brasa. Tendo crescido desconfiando de qualquer autoridade, ele vira esse ceticismo transformar-se em convicção filosófica dentro do laboratório do MIT. Para a comunidade hacker, Watergate não passava da versão shakespeariana das picuinhas cotidianas de abuso de poder que tornavam a vida dos homens

comuns um inferno: era a parábola escancarada do preço desastroso que uma sociedade paga quando aceita trocar a transparência e a liberdade civil pela miragem da conveniência e da segurança nacional.

Munido de renovada empáfia juvenil, Stallman desfilava com o boton preso ao peito com orgulho. Aos incautos que se aproximavam para perguntar o sentido daquela frase insólita, desferia um monólogo afiado que trazia na ponta da língua:

— *"Meu nome é Jeová"*, disparava Stallman com solenidade teatral. — *"Tenho um plano infalível e secreto para erradicar a dor e as injustiças da face da Terra, mas, por razões celestiais de segurança nacional, estou proibido de revelar os detalhes desse plano a vocês. Eu enxergo o quadro geral e vocês não entendem nada; e vocês sabem que eu sou infinitamente bom porque fui eu mesmo quem disse isso a vocês! Portanto, depositem toda a sua fé em mim e obedeçam às minhas ordens sem pestanejar. Se vocês desobedecerem a uma única vírgula, isso provará que vocês são a encarnação do mal — e então colocarei o nome de vocês na minha lista negra de inimigos e os atirarei em um fosso eterno onde a Receita Federal do Inferno auditará as suas declarações de renda a cada segundo por toda a eternidade!"*

Aqueles que reduziam a tirada a uma mera paródia das audiências de Watergate captavam apenas metade da mensagem. Para Stallman, a outra metade era algo que apenas os companheiros do AI Lab pareciam alcançar com lucidez: um século após o lorde Acton advertir a humanidade de que o poder absoluto corrompe absolutamente, os cidadãos pareciam ter se esquecido da primeira parte do aforismo — **o poder em si corrompe**. Em vez de perder tempo apontando desvios pontuais de pequenos burocratas, Stallman preferia vociferar sua indignação moral contra o próprio sistema que considerava normal e legítimo conceder tanto poder a quem quer que fosse:

— Pensei comigo: por que parar nos peixes pequenos? — conclui Stallman, rememorando o sentido daquele boton rebelde. — Se tivemos a coragem de caçar o presidente Nixon, por que não ir atrás do Chefão Supremo? Pelo meu modo de ver as coisas, qualquer entidade que acumule poder excessivo e abuse desse poder merece, por uma questão elementar de justiça, ser sumariamente destituída dele!

Ver Michael Gross, *"Richard Stallman: High School Misfit, Symbol of Free Software, MacArthur-certified Genius"* (1999). ↩

Carmine DeSapio detém a duvidosa honra de ter sido o primeiro chefe ítalo-americano do Tammany Hall, a histórica máquina política novaiorquina. ↩

Dan Chess resume os protestos estudantis como mero "ruído de fundo": "Todos nós éramos politizados, mas o programa de ciências era sagrado; jamais mataríamos aula para ir a uma manifestação". ↩

Stallman duvida dessa afirmação: "Uma das razões pelas quais migrei da matemática e da física teórica para a programação de computadores é que eu nunca aprendi a descobrir coisas verdadeiramente novas nas duas primeiras ciências; eu apenas aprendia a decifrar o que os outros já haviam descoberto antes de mim. Na programação, pelo contrário, eu conseguia construir algo útil e palpável todos os dias". ↩

Ver Steven Levy, *Hackers: Heroes of the Computer Revolution* (1984), obra fundamental para compreender a gênese dessa cultura. ↩

Gerald Sussman contesta a veracidade histórica desse causo específico, sustentando que os hackers jamais precisaram derrubar portas fisicamente para reaver máquinas. ↩

Ver Richard Stallman, conferência no Royal Institute of Technology (KTH), Estocolmo (30 de outubro de 1986): http://www.gnu.org/philosophy/stallman-kth.html. ↩

Anos mais tarde, Stallman revelaria que também buscou inspiração política nas aulas de História Chinesa em Harvard: "Li a epopeia da primeira grande revolta popular contra a dinastia Qin — aquele imperador tirano que mandou queimar todos os livros de história e foi sepultado cercado pelos guerreiros de terracota. A narrativa talvez não seja historicamente rigorosa, mas era de uma beleza comovente". ↩

Ver Richard Stallman (1986). ↩

Ver Steven Levy, *Hackers* (1984), p. 417. ↩

Capítulo 5: Uma Poça de Liberdade

> **[RMS:** Neste capítulo, corrigi asserções sobre fatos, incluindo fatos relativos aos meus pensamentos e sentimentos, e removi certa dose de hostilidade gratuita em algumas descrições de eventos. Preservei os relatos das impressões pessoais de Williams, salvo onde expressamente anotado.**]**

Pergunte a qualquer pessoa que já tenha passado mais de um minuto na presença de Richard Stallman e a resposta será rigorosamente a mesma: esqueça a cabeleira desgrenhada. Esqueça o comportamento excêntrico. A primeiríssima coisa que salta aos olhos é o olhar. Basta fitar os olhos verdes de Stallman uma única vez para saber que você está diante de um verdadeiro cruzado.

Dizer que o olhar de Stallman é intenso soa quase como um eufemismo grosseiro. Os olhos dele não apenas encaram você; eles atravessam o seu corpo de lado a lado. Mesmo quando o seu próprio olhar desvia por um segundo — naquele reflexo primata elementar de polidez —, os olhos de Stallman permanecem cravados na sua têmpora, faiscando na lateral da sua cabeça feito feixes duplos de fótons em brasa.

Talvez seja exatamente por isso que a maioria dos jornalistas, ao tentar descrevê-lo, quase sempre apele para metáforas religiosas. Em uma reportagem de 1998 para o site *Salon.com* intitulada "O Santo do Software Livre", o repórter Andrew Leonard retrata os olhos verdes de Stallman como algo que "irradia a cólera de um profeta do Antigo Testamento"[1]. Uma matéria da revista *Wired* de 1999 define sua barba como "digna de um Rasputin moderno"[2], enquanto um perfil do jornal londrino *The Guardian* descreve seu sorriso como o de "um apóstolo vislumbrando Jesus Cristo"[3].

E essas são apenas algumas amostras da lavra mística. O troféu de comparação mais hiperbólica até hoje pertence a Linus Torvalds que, em sua autobiografia (*Just for Fun*), cravou com todas as letras: "Richard Stallman é o Deus do Software Livre".

Menção honrosa deve ser concedida ao jurista Lawrence Lessig que, em uma longa nota de rodapé de seu livro *The Future of Ideas*, compara Stallman a Moisés:

> *"...assim como aconteceu com Moisés, coube a outro líder, Linus Torvalds, a tarefa de conduzir o movimento até a terra prometida ao viabilizar a peça final do quebra-cabeça do sistema operacional. Do mesmo modo que Moisés, Stallman é simultaneamente respeitado e hostilizado por seus próprios aliados de marcha. Ele é um líder intransigente — e, por essa mesmíssima razão, inspirador — de um dos pilares mais decisivos da cultura contemporânea. Guardo um respeito reverente pelos princípios e pela entrega desse homem extraordinário, embora guarde igual respeito por aqueles bravos o bastante para questionar suas ideias e suportar a fúria de sua ira."*

Em uma de nossas últimas conversas, perguntei a Stallman o que ele achava de tantas comparações bíblicas:

— Algumas pessoas de fato me comparam a um profeta do Antigo Testamento — ponderou ele. — E o motivo é muito simples: os profetas bíblicos apontavam o dedo para a sociedade e denunciavam que certas práticas estavam erradas. Eles se recusavam a negociar princípios morais. Não podiam ser comprados pelo ouro do rei. E, invariavelmente, eram tratados com desprezo e escárnio pelas multidões.

Tais analogias cumprem um papel retórico, mas no fundo falham feio. Falham porque ignoram por completo o lado mais vulnerável e humano da personalidade de Stallman. Observe o olhar de Richard por um período contínuo e você começará a notar uma sutil e desconcertante mutação. Aquilo que à primeira vista parece uma tentativa deliberada de hipnotizar ou intimidar o interlocutor revela-se, após o segundo ou terceiro olhar, como um esforço desesperado e frustrado de construir e sustentar contato humano. Se a sua psique guarda algum traço ou "sombra" do espectro autista ou da Síndrome de Asperger — possibilidade que o próprio Stallman já aventou de tempos em tempos —, seus olhos certamente oferecem munição ao diagnóstico. Mesmo no auge de sua intensidade de farol alto, eles costumam de repente ficar turvos, distantes e desamparados, feito os olhos de um animal ferido preste a entregar os pontos.

Meu primeiro encontro com o legendário olhar de Stallman ocorreu em março de 1999, durante a feira LinuxWorld Convention and Expo em San José, Califórnia. Apresentada à imprensa como a grande "festa de debutante" da comunidade de software do "Linux", a convenção marcou também o evento que recolocou Stallman no radar dos grandes veículos de tecnologia. Decidido a reivindicar a fatia justa de crédito histórico que lhe cabia, Stallman aproveitou os holofotes para dar verdadeiras aulas públicas aos repórteres e visitantes sobre a história do Projeto GNU e os objetivos abertamente políticos do movimento.

Como jovem repórter destacado para cobrir o evento, recebi minha primeira lição particular durante a coletiva de imprensa convocada para anunciar o lançamento do GNOME 1.0 — a interface gráfica livre para computadores pessoais. Sem desconfiar do vespeiro em que estava me metendo, apertei todos os botões de alerta de Stallman logo na minha pergunta inaugural:

— O senhor acredita que a maturidade do GNOME poderá alavancar a popularidade comercial do sistema operacional Linux?

— Peço a gentileza de parar de chamar este sistema operacional de Linux — disparou Stallman, com os olhos verdes cravando-se instantaneamente nos meus feito agulhas. — O núcleo Linux é apenas uma fração minúscula de todo o sistema operacional. A esmagadora maioria dos softwares que compõem o sistema que você chama de Linux não foi desenvolvida por Linus Torvalds. Foram programas concebidos e escritos por voluntários do Projeto GNU, dedicando seu tempo e suas vidas para que os usuários pudessem desfrutar de um sistema operacional inteiramente livre como este que temos hoje. Negar o reconhecimento ao esforço histórico desses programadores não é apenas uma tremenda falta de educação: é uma falsificação deliberada da história. É por isso que exijo que, ao se referir ao sistema como um todo, você use o nome correto: **GNU/Linux**.

Enquanto eu rabiscava a advertência no meu bloco com as mãos trêmulas, um silêncio sepulcral engoliu o auditório lotado. Quando finalmente criei coragem para levantar os olhos, dei de cara com o olhar impassível de Stallman ainda esperando por mim. Intimidado com a cena, um segundo jornalista emendou outra pergunta, tomando o cuidado quase religioso de pronunciar "GNU/Linux" em vez de Linux. Miguel de Icaza, líder do projeto GNOME, chamou para si a resposta. Somente quando a fala de Icaza já ia pela metade é que os olhos de Stallman finalmente se desconectaram dos meus. No instante exato em que ele desviou a vista, senti um calafrio gélido escorrer pela espinha. Segundos depois, quando Stallman começou a passar um sabão homérico em outro repórter

por conta de um deslize semântico qualquer, experimentei um alívio culposo: *ao menos ele não está mais olhando para mim*, suspirei para os meus botões.

Para a estratégia de Stallman, embates corpo a corpo daquele tipo cumpriam com louvor o seu papel. Ao término daquela primeira edição da LinuxWorld, os repórteres já pensavam dez vezes antes de pronunciar a palavra "Linux" na presença do homem; e o portal da *Wired* já publicava matérias de capa comparando Stallman a um líder bolchevique pré-stalinista apagado dos anais oficiais da história por empresários e oportunistas ávidos por varrer as metas políticas do Projeto GNU para debaixo do tapete. Outros artigos seguiram a mesma toada: e se pouquíssimos veículos adotaram a expressão "GNU/Linux" no papel impresso, quase todos passaram a creditar Stallman como o pioneiro visionário que deflagrara a batalha pelo software livre quinze anos antes.

Eu só voltaria a encontrar Stallman dezessete meses depois. Nesse intervalo, ele retornara ao Vale do Silício para a edição de agosto de 1999 da LinuxWorld. Embora não tivesse sido incluído no quadro de palestrantes oficiais, conseguiu roubar a cena com a frase mais genial de todo o evento. Ao subir ao palco para receber em nome da Free Software Foundation o troféu *Linus Torvalds Award for Community Service* — uma honraria batizada ironicamente com o nome do criador do Linux —, Stallman disparou ao microfone com um sorriso de canto de boca:

— Entregar o Prêmio Linus Torvalds à Free Software Foundation é mais ou menos como condecorar a Aliança Rebelde com a Medalha Han Solo!

Desta vez, porém, a piada não bastou para sacudir as engrenagens da grande mídia. Naquela mesma semana, a Red Hat Inc. — a mais badalada distribuidora de GNU/Linux do mercado — abriu seu capital na bolsa de valores em uma oferta pública inicial estrondosa. A euforia dos investidores apenas sacramentou o que jornalistas como eu já suspeitavam: "Linux" havia se tornado a palavra de ordem de Wall Street, a nova queridinha do mercado financeiro, exatamente como "comércio eletrônico" e as empresas "ponto-com" antes dela. Com os índices da Nasdaq subindo quase verticalmente rumo à virada do milênio, qualquer reflexão sobre software livre como instrumento de emancipação política foi atropelada pela ganância do mercado financeiro.

Talvez resida aí o motivo de Richard Stallman ter brilhado por sua ausência na terceira edição da LinuxWorld, em agosto de 2000.

Meu segundo encontro com ele e com seu temível olhar de raio laser deu-se logo após esse terceiro evento. Ao saber que ele desembarcaria na Califórnia para uma série de compromissos acadêmicos, agendei uma entrevista durante o almoço na abastada cidade de **Palo Alto**. O local do encontro não deixava de ser uma ironia saborosa: fora de Redmond, pouquíssimas cidades no planeta oferecem um testemunho material tão ostensivo do poder da acumulação capitalista alavancada pelo software proprietário. Curioso para ver como Stallman — um homem que dedicou a existência a denunciar o egoísmo e a venalidade da sociedade de consumo — sobrevivia em uma cidade onde qualquer casebre com tamanho de garagem custava facilmente meio milhão de dólares, peguei a estrada descendo de Oakland.

Segui as coordenadas que Stallman me passara por e-mail até alcançar a sede da Art.net, um coletivo virtual de artistas sem fins lucrativos. Instalada em uma casa envolta por arbustos espessos no extremo norte da cidade, a sede do coletivo exibia um ar decadente e acolhedor. De repente, a figura de Stallman acampado no coração do Vale do Silício não parecia tão absurda assim.

Encontrei-o sentado no fundo de uma sala na penumbra, digitando freneticamente no teclado de um laptop cinzento. Ele ergueu a cabeça no instante em que passei pela porta, disparando em cheio a carga máxima de seu olhar de duzentos watts. Quando soltou um "olá" seco, ensaiei uma saudação de volta; contudo, antes que a primeira sílaba saísse da minha boca, seus olhos já haviam retornado à tela do computador.

— Estou terminando de redigir um ensaio sobre o espírito hacker — disse ele sem parar de teclar. — Dê uma olhada aqui.

Aproximei-me. Na escuridão da sala, o monitor brilhava com letras esverdeadas sobre um fundo preto absoluto — a inversão exata do padrão gráfico utilizado pelos processadores de texto comerciais —, de modo que meus olhos levaram alguns segundos para se acostumar. Quando o foco se estabilizou, vi-me lendo o relato de uma refeição recente de Stallman em um restaurante coreano. Antes de começar a comer, ele reparara que o garçom cometera um equívoco, deixando seis pauzinhos de madeira (*hashis*) diante de seu prato em vez do par habitual. Onde qualquer cliente comum simplesmente empurraria os talheres extras de lado, Stallman enxergou um desafio irresistível de engenharia: descobrir uma maneira de manipular os seis pauzinhos simultaneamente com uma única mão. Como acontece com tantos hacks célebres da informática, a solução encontrada era simultaneamente brilhante e hilária. Daí sua decisão de transformá-la em parábola filosófica.

Enquanto eu lia, sentia os olhos de Stallman me vigiando de soslaio. Olhei para o lado e deparei com um sorriso orgulhoso, quase infantil, desenhado em seu rosto. Quando elogiei a sagacidade da crônica, meu comentário mal arrancou um erguer de sobrancelhas.

— Estarei pronto em um instante — avisou ele.

Stallman voltou à sua sinfonia de cliques. O computador era um aparelho cinza, pesado e quadrado, a anos-luz dos modelos prateados e esguios que faziam a festa dos geeks na LinuxWorld. Sobre o teclado do aparelho repousava um segundo teclado menor e levíssimo — testemunha muda do desgaste físico de suas mãos. Em meados da década de 1990, as dores de LER (lesão por esforço repetitivo) nos pulsos e dedos de Stallman tornaram-se tão incapacitantes que ele precisou contratar uma digitadora profissional para conseguir trabalhar. Hoje, utiliza teclados adaptados cujas teclas exigem uma pressão mínima dos tendões.

Quando está programando ou escrevendo, Stallman desliga-se do mundo exterior com uma blindagem impenetrável. Vendo seus olhos colados aos caracteres e seus dedos dançando pelas teclas, tem-se a impressão límpida de dois velhos amantes engatados em uma conversa íntima e ininterrupta.

A sessão encerrou-se com uma sequência vigorosa de toques secos e a desmontagem meticulosa dos periféricos do laptop.

— Pronto para o almoço? — perguntou ele.

Caminhamos em direção ao meu carro. Queixando-se de dores no tornozelo, Stallman mancava a passos lentos. Ele atribuía o problema a uma lesão crônica no tendão do pé esquerdo, contraída três anos antes e que se agravara a ponto de obrigá-lo a abandonar uma de suas maiores paixões na vida: a dança folclórica.

— Eu amo dança folclórica de corpo e alma — lamentou ele com genuína tristeza. — Ter sido forçado a parar de dançar é uma verdadeira tragédia pessoal para mim.

O corpo de Stallman trazia as marcas visíveis dessa perda. A falta de atividade física cobrara seu preço na forma de bochechas inchadas e uma proeminente barriga saliente, muito mais acentuada do que no ano anterior. A mudança de peso fora drástica: ao caminhar, Stallman arqueava as costas para trás, lembrando uma mulher grávida tentando equilibrar um peso desconhecido no ventre.

A marcha tornava-se ainda mais arrastada pelo hábito de Stallman de parar para cheirar as flores no caminho — literalmente. Ao bater o olho em uma rosa deslumbrante em um canteiro da calçada, ele roçou as pétalas mais profundas diretamente contra as narinas, respirou fundo e deu um passo atrás soltando um suspiro de êxtase:

— Mmm, *rinofitofilia* pura! — exclamou, massageando a base da coluna[4].

O trajeto até o restaurante levou menos de três minutos. Seguindo uma dica de Tim Ney, ex-diretor executivo da FSF, deixei a escolha do local a cargo de Stallman. Embora parte da imprensa adore romantizar sua vida retratando-o como um monge asceta, a verdade pura e simples é que Stallman é um gastrônomo apaixonado por comida de verdade. Um dos benefícios colaterais de atuar como pregador itinerante do software livre pelo planeta é a oportunidade de degustar banquetes refinados em todos os continentes. "Viaje a praticamente qualquer capital do mundo e pode ter certeza de que o Richard conhece o melhor restaurante da cidade", revela Ney. "Ele também tem imenso orgulho em dominar os cardápios e fazer os pedidos para a mesa inteira — desde que os convidados concordem, é claro!".

Para a nossa refeição, Stallman escolheu uma tradicional casa de *dim sum* cantonês a duas quadras da University Avenue, a artéria central de Palo Alto. A escolha decorria de sua recente passagem pela China e por Hong Kong, somada à sua conhecida aversão a pratos apimentados demais, típicos das culinárias de Sichuan ou Hunan: "Não sou chegado a pimenta", confessou ele.

Chegamos pouco depois das 11 horas da manhã e deparamos com uma fila de espera de vinte minutos. Conhecendo a ojeriza visceral dos hackers à perda de tempo, prendi a respiração por um instante, temendo um rompante de fúria. Stallman, para minha surpresa, recebeu a notícia com serenidade impecável.

— Uma pena que não tenhamos conseguido mais ninguém para vir conosco — comentou ele. — Almoçar em grupo é infinitamente mais divertido.

Enquanto esperávamos no saguão, Stallman ensaiou alguns passos discretos de dança folclórica. Seus movimentos eram contidos, mas precisos. Começamos a papear sobre o noticiário. Stallman confidenciou que seu único pesar por não ter comparecido à última LinuxWorld fora ter perdido a cerimônia de lançamento da GNOME Foundation. Patrocinada por pesos-pesados como Sun Microsystems e IBM, a criação da fundação representava, em larga medida, uma consagração tardia para Stallman, que sempre defendeu que o software livre e a economia de mercado não são excludentes. Mesmo assim, não escondeu sua amargura com os discursos da cerimônia:

— Da forma como foi apresentado à imprensa, os executivos só falaram de Linux para cá e Linux para lá, sem dedicar uma única palavra ao Projeto GNU — queixou-se ele.

Desilusões dessa natureza contrastavam fortemente com o entusiasmo caloroso que encontrava em suas viagens internacionais, sobretudo na Ásia e na América Latina. Uma folheada rápida em sua agenda de viagens no ano 2000 revela a dimensão planetária de sua mensagem: entre périplos recentes por Índia, China e Brasil, Stallman

passara míseros doze dos últimos 115 dias em território norte-americano. Suas andanças davam-lhe uma perspectiva privilegiada sobre como o ideal do software livre era traduzido em diferentes línguas e culturas:

— Na Índia, muitos se encantam pelo software livre porque enxergam nele uma maneira formidável de erguer sua infraestrutura de tecnologia sem gastar montanhas de dinheiro que não têm — avalia Stallman. — Na China, a assimilação é muito mais arrastada: comparar "software livre" com "liberdade de expressão" é um exercício bem mais difícil em um país que não tem liberdade de expressão alguma. Ainda assim, o interesse genuíno demonstrado pelos chineses na minha última passagem foi profundo.

A conversa desviou então para o **Napster** — o aplicativo desenvolvido na vizinha San Mateo que se convertera na maior dor de cabeça do planeta para a indústria fonográfica. A empresa distribuía um software pioneiro que permitia aos internautas vasculhar e copiar arquivos de áudio diretamente dos discos rígidos de outros usuários em rede. Turbinada pela arquitetura distribuída da internet, essa tecnologia "ponto a ponto" (*peer-to-peer*) transformara-se em uma colossal vitrola global, permitindo a qualquer fã de música baixar e ouvir canções em MP3 sem pagar um único tostão de direitos autorais às grandes gravadoras.

Embora operasse sob um software estritamente proprietário, a febre do Napster bebia diretamente na fonte daquela velha tese defendida por Stallman: uma vez que uma obra entra no domínio digital — ou seja, no instante em que fazer uma cópia deixa de ser um processo físico de forjar matéria e fundir átomos e passa a ser uma simples duplicação de sequências lógicas de informação —, o impulso natural da humanidade de compartilhar o que ama torna-se impossível de represar. Em vez de impor barreiras artificiais, os fundadores do Napster decidiram surfar a maré desse instinto solidário, apostando que poderiam rentabilizar o tráfego brutal de usuários mais adiante.

O estrondo repentino do Napster fez tremer as bases da indústria da música. Poucos dias antes do nosso almoço, a juíza Marilyn Patel, da Justiça Federal dos Estados Unidos, acolhera um pedido de liminar impetrado pela RIAA (a associação das gravadoras norte-americanas) ordenando a suspensão imediata do serviço. A decisão acabara de ser suspensa em grau de recurso, mas, no início de 2001, o Tribunal Federal de Recursos confirmaria a condenação do Napster por violação em massa de direitos autorais [5] — desfecho que os porta-vozes da RIAA celebrariam na época como "uma vitória retumbante para a comunidade criativa e para o comércio eletrônico legítimo".

Para hackers da linhagem de Stallman, o modelo comercial do Napster era um prato indigesto por razões muito mais sutis. A pressa da empresa em se apropriar de princípios históricos da cultura hacker — como o compartilhamento desinteressado e o saber comunitário — para vender um serviço aprisionado sob um software fechado e proprietário enviava uma mensagem perigosamente confusa à sociedade. Sendo alguém que já sofria o pão que o diabo amassou para fazer sua própria mensagem ser compreendida com clareza pela imprensa, Stallman pisava em ovos ao comentar o caso. Mesmo assim, admitiu que a dimensão social do Napster lhe ensinara lições valiosas:

— Antes do surgimento do Napster, eu imaginava que talvez fosse suficiente assegurar às pessoas o direito de redistribuir obras de entretenimento em caráter estritamente privado, entre amigos — ponderou ele. — Contudo, a multidão assombrosa de cidadãos que encontrou utilidade prática no Napster me convenceu de uma verdade maior: o direito de redistribuir cópias exatas não apenas na base do vizinho para vizinho, mas para o público em geral, é um direito fundamental da era digital que jamais pode ser cassado!

Mal terminou a frase, a porta do restaurante abriu-se e o garçom nos conduziu para o salão interno, acomodando-nos em uma mesa de canto junto a uma imensa parede espelhada.

O cardápio da casa funcionava como uma comanda interativa, e Stallman já preenchia os quadradinhos das opções antes mesmo de o garçom servir os copos de água:

— Rolinho de camarão frito envolto em película de tofu — leu ele em voz alta. — Película de tofu! Tem uma textura absolutamente fascinante na boca; precisamos pedir este!

A escolha abriu caminho para um parêntese apaixonado sobre a gastronomia oriental e suas andanças pela China:

— A culinária na China é de um refinamento sublime — disse Stallman, deixando transparecer na voz uma emoção rara até então. — Havia tantas iguarias deliciosas que eu nunca tinha visto na América: receitas regionais preparadas com cogumelos silvestres e vegetais colhidos na hora. Cheguei ao ponto de começar a escrever um diário gastronômico só para registrar cada banquete memorável.

A conversa deslizou suavemente para os sabores da Coreia. Durante aquele mesmo périplo asiático no ano 2000, Stallman visitara a Coreia do Sul. Sua chegada causou um alvoroço na imprensa de Seul devido a uma conferência de informática que contava com a presença de ninguém menos que Bill Gates naquela mesma semana. Além do deleite de ver sua foto estampada acima do retrato de Gates na primeira página do principal jornal sul-coreano, Stallman revelou que o ponto alto da estada fora a comida de rua:

— Pedi uma tigela de *naengmyeon*, aquela tradicional massa fria coreana. O macarrão tinha uma elasticidade e uma consistência extraordinárias. A maioria dos lugares usa ingredientes banais para fazer a massa do *naengmyeon*, de modo que posso afirmar com total convicção de que foi o prato mais requintado que provei na vida.

A palavra "requintado" é o elogio máximo no vocabulário de Stallman. E eu comprovei isso na prática porque, segundos após ouvi-lo discorrer sobre a massa coreana, senti a miragem de seu olhar incandescente queimar a parte de trás do meu ombro direito.

— Há uma mulher de um requinte deslumbrante sentada logo atrás de você — sussurrou Stallman.

Virei o pescoço discretamente, captando o perfil de uma jovem na casa dos vinte e poucos anos, vestindo um elegante vestido branco de lantejoulas. Ela e o acompanhante estavam no final do acerto da conta. Quando os dois se levantaram da mesa para sair, nem precisei olhar para saber: a intensidade do olhar de Stallman apagou-se no mesmo instante.

— Que pena... — suspirou ele, desolado. — Já se foram. E pensar que muito provavelmente nunca mais porei os olhos nela nesta vida...

Após um breve instante de lamento, Stallman recuperou o prumo. A cena abriu brecha para sondar sua fama histórica em relação ao sexo oposto — um terreno marcado por relatos contraditórios. Vários hackers relatam a predileção de Stallman por saudar mulheres beijando respeitosamente o dorso de suas mãos[6]. Por outro lado, um perfil publicado pelo portal *Salon.com* retratava-o quase como uma espécie de conquistador libertário dos códigos:

explorando as pontes entre software livre e amor livre, a jornalista Annalee Newitz citava uma declaração na qual ele rejeitava os moldes do casamento burguês: "Acredito no amor, mas não na monogamia"[7].

Stallman baixou o cardápio e franziu a testa:

— Bem, a imensa maioria dos homens parece querer apenas sexo e demonstra uma atitude incrivelmente arrogante e desdenhosa em relação às mulheres — ponderou ele. — Inclusive em relação às mulheres com quem se relacionam! Eu simplesmente não consigo compreender essa baixeza.

Mencionei então uma passagem do livro *Open Sources* (1999) na qual ele revela que pensou em batizar o núcleo do sistema GNU com o nome de sua namorada da época. A moça chamava-se Alix — um nome que casava com a tradição dos desenvolvedores Unix de colocar a letra "x" no final dos sistemas operacionais (como em Linux). Alix era administradora de redes e certa vez comentara com amigos: "Alguém deveria batizar um kernel em minha homenagem!". Stallman decidiu então batizar o núcleo do GNU de "Alix" como uma surpresa afetuosa para a amada. O desenvolvedor encarregado do código acabou rebatizando o projeto de "Hurd", mas manteve o nome "Alix" em um dos módulos internos. Quando um colega de Alix bateu o olho no código-fonte e lhe contou o segredo, ela ficou profundamente comovida[8].

Pela primeira vez naquela manhã, Stallman abriu um sorriso sincero. Perguntei-lhe sobre o hábito de beijar a mão das mulheres:

— Sim, eu tenho esse costume — admitiu ele. — Descobri que é uma forma delicada de oferecer afeto que muitas mulheres apreciam de verdade. É uma oportunidade singela de dar carinho e ser correspondido com gentileza.

O afeto é um fio invisível e doído que atravessa toda a biografia de Richard Stallman, e ele demonstra uma franqueza comovente ao tocar na ferida:

— A verdade pura e simples é que quase não houve afeto real na minha vida, a não ser na minha imaginação — confessou ele, baixinho.

A conversa ameaçou pesar. Após algumas respostas monossilábicas, Stallman ergueu o cardápio, encerrando as inconfidências íntimas por ali:

— Você aceita um pouco de *siu mai* de porco? — desconversou ele.

Quando as travessas começaram a chegar, a conversa ziguezagueou no ritmo dos pratos. Falamos sobre a paixão quase unânime dos hackers por comida chinesa, os lendários jantares em Boston nos seus tempos de MIT e a arquitetura tonal da língua falada na China. Para cada provocação que eu lançava, Stallman devolvia uma análise minuciosa:

— Ouvi algumas pessoas conversando no dialeto de Xangai na última vez em que estive por lá — relatou ele. — Fiquei fascinado: a pronúncia é completamente diferente do mandarim padrão! Pedi que me dissessem palavras de mesma raiz em ambos os idiomas. Em alguns casos, a semelhança salta aos olhos; mas o que me intrigava de verdade era saber se a entonação dos tons musicais se mantinha parecida. E não é parecida em nada! Isso é de um interesse científico fascinante, porque reforça a tese linguística de que os tons das línguas sino-tibetanas evoluíram

a partir de sílabas finais que se perderam na história. Se isso for verdade, os dialetos se separaram antes do desaparecimento dessas terminações arcaicas.

O primeiro prato — uma porção de bolinhos de nabo grelhados na chapa — acabara de aterrissar na mesa. Stallman e eu dividimos os pedaços retangulares, que cheiravam a repolho cozido mas tinham um sabor delicioso, lembrando batatas *latkes* douradas no bacon.

Aproveitei a deixa para voltar ao tema do isolamento social, perguntando se as dores de ter sido um pária na adolescência não teriam forjado nele a têmpera necessária para sustentar posições públicas tão impopulares — a começar pela sua incansável batalha solitária desde 1994 para convencer o mundo a dizer "GNU/Linux":

— Acredito piamente que ter sido um excluído me ajudou imensamente a não me curvar às opiniões da maioria — refletiu Stallman, mastigando com calma. — Eu nunca entendi o poder paralisante que a pressão social exerce sobre as outras pessoas. E acho que o motivo é óbvio: eu fui tão profundamente rejeitado e jogado para escanteio na juventude que simplesmente não tinha absolutamente nada a ganhar tentando seguir as modinhas da vez. Não faria a menor diferença prática! Eu continuaria sendo o mesmo pária rejeitado de sempre; então, para que me rebaixar tentando agradar?

Stallman resgatou suas preferências musicais como prova cabal dessa teimosia libertária. Na adolescência, enquanto os colegas de turma suspiravam pela gravadora Motown ou pelo rock ácido, ele refugiava-se na música barroca. A lembrança resgatou um causo impagável dos seus tempos de ginásio: logo após a apresentação bombástica dos Beatles no *The Ed Sullivan Show*, em 1964, a garotada correu às lojas para comprar os compactos dos cabeludos de Liverpool. Ali mesmo, naquele instante, Stallman declarou um boicote pessoal e intransigente contra o quarteto britânico:

— Eu até gostava de algumas músicas populares pré-Beatles — explicou ele. — Mas peguei birra dos Beatles. Detestava, acima de tudo, a histeria irracional com que as pessoas reagiam a eles. Parecia uma competição insana para ver quem se ajoelhava mais baixo para adorar aqueles quatro sujeitos!

Ao ver que seu boicote solitário não abalava o sucesso da Beatlemania, Stallman bolou uma maneira bem mais espirituosa de escancarar o comportamento de manada dos colegas: pensou em fundar sua própria banda de rock cômico exclusivamente para parodiar o grupo inglês:

— Meu plano era batizar o conjunto de *Tokyo Rose and the Japanese Beetles* ("Tokyo Rose e os Besouros Japoneses").

Sabendo de sua paixão por ritmos tradicionais, perguntei se ele não se encantava pelas canções de protesto de Bob Dylan nos anos 1960. Stallman torceu o nariz:

— Eu gostava do trio Peter, Paul and Mary — admitiu. — E isso me faz lembrar de um excelente *filk*!

Diante da minha cara de interrogação, Stallman explicou que a palavra **filk** é o termo consagrado pelos fãs de ficção científica para definir a arte de compor paródias musicais inteligentes, mantendo a melodia original com letras satíricas novas.

Antes que eu pudesse respirar, Stallman começou a entoar a plenos pulmões no meio do restaurante, na melodia lírica do clássico pacifista *"Blowin' in the Wind"*, de Bob Dylan:

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

A cantoria encerrou-se sob um silêncio constrangedor e um novo meio-sorriso infantil de Stallman. Olhei ao redor: as famílias chinesas que almoçavam serenamente no restaurante não deram a menor pelota para o tenor barbudo cantando na mesa ao lado[9]. Depois de um segundo de hesitação, caí na risada.

— Vai querer o último bolinho? — ofereceu ele com os olhos faiscando de humor. Antes que eu pudesse responder, Stallman pinçou o quitute com os pauzinhos e exibiu o troféu no ar. — Pensando bem, acho que fui eu quem fez por merecer!

Com a mesa limpa, nossa conversa assumiu o compasso de uma entrevista formal. Stallman recostou-se no espaldar da cadeira e envolveu a xícara de chá fumegante com as duas palmas das mãos. Retomamos o tema do Napster e suas fronteiras com o software livre: deveriam os princípios de liberdade do software ser aplicados a outros domínios da cultura, como a música ou a literatura?

— É um erro grosseiro transferir respostas prontas de um domínio para outro sem critério — advertiu Stallman, contrastando a natureza de uma canção com a de um programa de computador. — A postura correta é dissecar cada categoria de obra e analisar quais conclusões éticas ela impõe.

No tocante aos direitos autorais, Stallman divide a produção humana em três grandes esferas: 1. **Obras funcionais:** ferramentas práticas que usamos no dia a dia, como softwares de computador, dicionários, manuais técnicos e livros didáticos. 2. **Obras testemunhais e de opinião:** documentos históricos, artigos científicos, memórias e biografias. Essas obras cumprem uma finalidade de registro que seria pervertida se qualquer leitor pudesse adulterar o texto alheio à revelia do autor. 3. **Obras de arte e entretenimento:** romances, poesias, pinturas e músicas.

Para a primeira categoria — as obras funcionais —, a liberdade de alterar e redistribuir versões modificadas deve ser plena e irrestrita. Para as duas últimas, o direito de modificar o conteúdo pode e deve ser regulado conforme a vontade do autor original. Contudo, em todas as três esferas, **a liberdade de copiar e redistribuir cópias exatas sem fins comerciais deve permanecer inviolável**, brada Stallman. Se isso significa garantir aos cidadãos o direito de baixar uma música, um ensaio ou um livro e repassá-lo por e-mail a amigos ou estranhos pela internet, que assim seja:

— É cristalino que a partilha privada e espontânea entre as pessoas deve ser sagrada, até porque a única maneira de impedi-la seria instaurando um Estado policial asfixiante — defende Stallman. — É de uma crueldade antissocial colocar a polícia entre duas pessoas que desejam ser amigas. O Napster provou para mim que a sociedade precisa assegurar, obrigatoriamente, a liberdade de partilhar cópias exatas com o público pelo simples prazer da generosidade, porque isso enriquece a cultura humana!

Quando questionei se os tribunais norte-americanos aceitariam tese tão generosa, Stallman cortou-me na hora:

— Essa é a pergunta errada! — disparou ele. — Você mudou de assunto no meio do caminho, saltando da reflexão sobre a ética para uma discussão sobre hermenêutica jurídica. E essas são duas questões completamente distintas. É inútil misturá-las. A maneira como os tribunais hoje interpretam a legislação é quase sempre mesquinha e punitiva, porque as leis em vigor foram literalmente compradas pelas grandes corporações de mídia!

O desabafo descortina a espinha dorsal de sua filosofia política: o fato de a legislação contemporânea chancelar a pretensão das empresas de tratar direitos autorais como propriedade privada imobiliária não obriga os cidadãos da era digital a curvarem a cabeça passivamente. A liberdade é um imperativo ético, e não uma concessão dos códigos jurídicos:

— Meu olhar vai muito além do que as leis atuais dizem; olho para o que as leis **deveriam dizer**! Não estou aqui redigindo projetos de emenda constitucional; estou pensando sobre qual é a função civilizatória do Direito. Considero uma lei que criminaliza o ato de partilhar conhecimento com um amigo o equivalente moral das leis segregacionistas de Jim Crow no sul dos Estados Unidos. Uma aberração dessas não merece o menor respeito!

A invocação da segregação racial abriu espaço para outra reflexão: até que ponto Stallman se espelhava nas grandes lideranças históricas dos direitos civis? Afinal, sua pregação alicerçava-se nos mesmos pilares universais da luta de Martin Luther King Jr.: liberdade, fraternidade e justiça social.

Stallman dividia sua atenção entre a minha pergunta e uma mecha particularmente embaraçada de seus cabelos compridos. Quando forcei a analogia a ponto de colocá-lo no mesmo patamar de Martin Luther King, ele quebrou uma ponta dupla do cabelo, jogou-a na boca e me interrompeu:

— Não jogo na mesma divisão que ele, mas certamente disputo a mesma partida — disse ele, mastigando o fio.

Arrisquei citar Malcolm X como outro contraponto: a exemplo do líder negro dissidente, Stallman colecionava desafetos por sua recusa intransigente a alianças fáceis e pregava a autossuficiência radical da comunidade hacker em vez da capitulação ao mercado.

Mastigando outra ponta de cabelo, Stallman repudiou o paralelo:

— Minha mensagem é muito mais próxima da de Martin Luther King — rebateu ele. — É uma mensagem universalista. É uma denúncia firme e inflexível contra práticas que humilham e oprimem o semelhante. Não é uma pregação de ódio contra quem quer que seja; não se destina a um nicho restrito. Convido todo e qualquer ser humano a abraçar e defender a liberdade.

Muitos analistas criticam Stallman por queimar pontes e recusar alianças políticas pragmáticas, diagnosticando essa recusa como puro capricho de personalidade. No caso de sua aversão pública ao termo "código aberto", sua

relutância em celebrar acordos comerciais de ocasião é perfeitamente compreensível: como alguém que dedicou a vida a defender o software livre, todo o seu patrimônio moral está amarrado àquela palavra: **liberdade**. Ainda assim, provocações como o episódio da "Medalha Han Solo" renderam-lhe a pecha de velho dinossauro ressentido, incapaz de se adaptar às exigências do marketing moderno.

— Admiro e respeito o Richard por tudo o que ele construiu com as próprias mãos — resume o presidente da Red Hat, Robert Young. — Minha única ressalva é que, às vezes, ele parece tratar seus próprios amigos pior do que trata seus piores inimigos.

> **[RMS:** O rótulo de "amigos" só se aplica de forma muito tênue e parcial a figuras como Young ou a companhias como a Red Hat. Ele se justifica por algumas coisas que fizeram e ainda fazem: a Red Hat, por exemplo, financia o desenvolvimento de softwares livres, inclusive de alguns módulos do Projeto GNU. Contudo, a Red Hat também toma outras atitudes que sabotam diretamente os princípios do movimento do software livre — como rechear suas versões comerciais do GNU/Linux com programas privativos e fechados. Além disso, no plano do discurso, insistir em chamar o sistema de "Linux" apaga injustamente a existência do Projeto GNU, enquanto promover o "código aberto" em detrimento do "software livre" é cuspir sobre os nossos valores éticos. Eu podia perfeitamente cooperar com Young e com a Red Hat quando estávamos remando na mesma direção, mas isso acontecia com pouca frequência para que pudéssemos considerá-los verdadeiros aliados.**]**

A cautela de Stallman em amarrar o movimento do software livre a outras bandeiras políticas não brota do desinteresse pelas dores do mundo. Quem visita seu gabinete no MIT depara com recortes de matérias jornalísticas denunciando violações de direitos humanos pelo globo. Quem acessa seu site pessoal (http://stallman.org) depara com artigos inflamados contra as patentes de software, a repressão militar antidrogas e os desmandos da Organização Mundial do Comércio. Ele explica o motivo dessa blindagem:

— Precisamos tomar imenso cuidado para não atrelar o software livre a causas que dividam os nossos próprios apoiadores. Evitamos, por exemplo, ligar o movimento a qualquer partido político, para não afastar militantes ou parlamentares de outras legendas que defendam a liberdade digital.

Diante dessa veia combativa, perguntei por que ele não utilizava sua projeção mundial como hacker para amplificar sua voz sobre os demais dilemas da humanidade.

> **[RMS:** Mas é exatamente isso o que eu faço, sempre que surge uma boa oportunidade! Foi exatamente por isso que coloquei no ar o site *stallman.org*. Williams registrou uma resposta minha que não combinava com a pergunta que ele fez; a resposta combinava com outra indagação, algo como: "Por que você concentra suas forças no software livre em vez de liderar as outras causas em que acredita?". Suspeito que a pergunta que ele realmente me fez no almoço tenha sido essa.**]**

Stallman soltou a mecha de cabelos e refletiu por um longo instante:

— Eu tenho certo pudor em inflar a relevância desta nossa pequena poça de liberdade — disse ele, com humildade tocante. — Porque as grandes lutas históricas por justiça social e por um mundo melhor são monumentais. Jamais teria a empáfia de dizer que o software livre é tão importante quanto elas. O software livre foi a responsabilidade que assumi nos ombros porque caiu no meu colo, e eu vi com clareza que tinha condições práticas de fazer algo a respeito. Mas pautas como erradicar a violência policial, pôr fim à sangrenta guerra contra as drogas, estancar o racismo estrutural que ainda envenena a sociedade, garantir uma vida digna aos desamparados, proteger o direito das mulheres ao aborto seguro e blindar o Estado contra o fanatismo teocrático... Essas são causas infinitamente mais grandiosas e urgentes do que o que eu faço! Eu apenas gostaria de saber como fazer a diferença nelas.

Mais uma vez, a coragem cívica de Stallman revelava-se indissociável da sua segurança analítica: ciente do esforço titânico de décadas que despendeu para lapidar os alicerces do software livre, ele hesitava em se lançar em batalhas onde não dominava as peças do tabuleiro:

— Eu gostaria do fundo da alma de saber como provocar uma virada histórica nessas causas gigantescas, e me sentiria infinitamente orgulhoso se pudesse; mas elas são incrivelmente complexas, e gerações de pessoas muito mais capazes do que eu vêm lutando nelas há séculos com vitórias parciais. Pelo meu modo de encarar a vida, enquanto os bravos estavam mobilizados defendendo a civilização contra essas grandes ameaças visíveis, eu enxerguei um flanco desguarnecido que ninguém estava vigiando. Então decidi correr para proteger aquela trincheira esquecida. Pode ser que ela não seja a maior das guerras, mas eu era o único sujeito de prontidão que estava ali para lutar.

Mastigando uma última ponta de cabelo, Stallman propôs pedir a conta. Antes que o garçom recolhesse o pratinho com o dinheiro, porém, ele puxou do bolso uma nota branca e soltou-a sobre a pilha de dólares. A cédula era tão descaradamente fictícia que não resisti e a apanhei para examinar de perto: no lugar da efígie de George Washington ou Abraham Lincoln, trazia o desenho satírico de um porco rechonchudo de terno; acima da caricatura, lia-se a inscrição: *"Untied Status of Avarice"* ("Estados Desunidos da Avareza"). O valor impresso na nota era de **zero dólares**[10]. Quando o garçom veio buscar o montante, Stallman deu um puxãozinho na manga do paletó do rapaz:

— Adicionei um zero extra na sua gorjeta! — avisou Stallman, com mais um meio-sorriso zombeteiro no canto da boca.

O rapaz, sem entender patavina ou deslumbrado com a nota insólita, abriu um sorriso amarelo e sumiu em direção ao balcão.

— Acho que isso significa que estamos livres para ir embora — decretou Stallman.

Ver Andrew Leonard, *"The Saint of Free Software"*, *Salon.com* (agosto de 1998). ↩

Ver Leander Kahney, *"Linux's Forgotten Man"*, *Wired News* (5 de março de 1999). ↩

Ver *"Programmer on moral high ground"*, *The Guardian* (6 de novembro de 1999). ↩

Na hora, imaginei tratar-se do nome botânico da planta. Apenas meses depois descobri que "rinofitofilia" era um neologismo galhofeiro cunhado pelo próprio Stallman para descrever o ato de enfiar o nariz numa flor como uma prática fetichista de "sexo nasal com plantas". ↩

Ver Cecily Barnes e Scott Ard, *"Court Grants Stay of Napster Injunction"*, *News.com* (julho de 2000). ↩

Ver Mae Ling Mak, *"A Mae Ling Story"* (1998). Embora tenha confessado um estranhamento inicial com o gesto, Mak superou a reserva e dançou alegremente com Stallman na LinuxWorld de 1999. ↩

Ver Annalee Newitz, *"If Code is Free Why Not Me?"*, *Salon.com* (maio de 2000). ↩

[RMS: Williams interpretou essa passagem sugerindo romanticamente que eu seria uma espécie de sonhador incorrigível e que meus esforços como programador buscavam impressionar alguma donzela imaginária e misteriosa. Nenhum hacker do MIT acreditaria em uma bobagem dessas, já que todos nós aprendemos desde cedo que as mulheres jamais nos notariam — muito menos nos amariam — pela nossa habilidade em escrever código. Nós programávamos porque a programação era fascinante. Além disso, aquele episódio só aconteceu porque eu tinha uma namorada de carne e osso muito bem definida na época. Se eu era romântico, não era um romântico incurável nem esperançoso: era, naquele instante, um romântico temporariamente bem-sucedido. Apoiando-se nessa interpretação ingênua, Williams chegou a me comparar a Dom Quixote.**]** ↩

Para conhecer as paródias musicais compostas pelo próprio Stallman, visite: http://www.stallman.org/doggerel.html. Para ouvi-lo cantando o hino oficial do movimento, consulte: http://www.gnu.org/music/free-software-song.html. ↩

[RMS: Williams cometeu um equívoco ao chamar a cédula de "falsa". Ela é uma nota perfeitamente legal, no valor facial de zero dólares para o pagamento de qualquer dívida. Qualquer repartição pública do Tesouro norte-americano terá o prazer de convertê-la no equivalente a zero dólares em ouro!**]** ↩

Capítulo 6: A Comuna do Emacs

O Laboratório de Inteligência Artificial do MIT nos anos 1970 era, sob qualquer prisma que se olhasse, um lugar fora do comum. Pesquisas de vanguarda e mentes brilhantes de primeira grandeza conferiam ao laboratório uma posição de prestígio invejável no cenário da ciência da computação. A cultura hacker interna e suas práticas libertárias, que beiravam o anarquismo prático, acrescentavam àquele ambiente uma mística irresistível de rebeldia. Só muito mais tarde, quando os grandes cientistas e os magos do software já haviam batido em retirada, é que os veteranos se dariam conta do ecossistema único e efêmero que haviam habitado.

— Era algo muito parecido com o Jardim do Éden — resume Stallman ao evocar a atmosfera do laboratório e a ética espontânea da partilha de programas em um artigo de 1998 para a revista *Forbes*. — Simplesmente não nos passava pela cabeça a ideia de não cooperar[1].

Descrições de tom mitológico como essa, conquanto superlativas, escancaram uma realidade fundamental: para muitos pesquisadores, o nono andar do edifício no número 545 da Tech Square era muito mais do que um simples local de trabalho. Para hackers como Richard Stallman, aquele chão era o seu verdadeiro **lar**.

A palavra "lar" carrega um peso colossal no léxico pessoal de Stallman. Em uma alfinetada deliberada contra os pais, ele recusa-se até hoje a reconhecer qualquer teto anterior à Currier House — o dormitório estudantil onde viveu seus anos de ouro em Harvard. O próprio encerramento daquele ciclo universitário costuma ser narrado por ele em tomadas tragicômicas. Certa vez, ao discorrer sobre os anos na universidade, afirmou com solenidade que seu único arrependimento fora ter sido expulso de lá. Ao ser questionado sobre qual transgressão grave provocara sua ostração, revelou o desfecho de sua típica piada de salão:

— Em Harvard eles têm uma regra draconiana: se você passar em matérias demais, eles convidam você a se retirar! — ironiza ele.

Sem o quarto no dormitório e sem a menor intenção de regressar à casa dos pais em Nova York, Stallman trilhou a mesma vereda desbravada por Richard Greenblatt, Bill Gosper, Gerald Sussman e tantos outros hackers antes dele. Matriculado na pós-graduação do MIT, alugou um quartinho em Cambridge, mas não demorou a adotar o próprio AI Lab como sua morada de fato. Em um discurso histórico em 1986, Stallman resgatou com carinho a rotina daquele período:

> *"Talvez eu tenha morado dentro do laboratório um pouco mais do que os outros: a cada ano ou dois, por um motivo ou por outro, eu ficava sem apartamento e passava alguns meses dormindo direto no laboratório. E eu sempre achei aquilo extremamente confortável, sem falar que era fresquinho e ventilado no calor do verão. Mas era absolutamente rotineiro encontrar pessoas desmaiadas de sono sobre as mesas do laboratório, movidas pelo entusiasmo da criação: você fica acordado programando o máximo de tempo que o seu corpo aguentar, simplesmente porque não quer parar de jeito nenhum! E então, quando você está completamente exausto e à beira do colapso físico, engatinha até a superfície horizontal acolchoada mais próxima. O ambiente era de uma informalidade deliciosa."*[2]

A atmosfera doméstica do laboratório, contudo, despertava reações azedas em certos círculos acadêmicos. O que para os pioneiros funcionava como uma acolhedora república estudantil era visto por outros como um fumadouro eletrônico de ópio. Em seu clássico livro de 1976, *Computer Power and Human Reason*, o pesquisador e professor do MIT Joseph Weizenbaum desferiu uma crítica demolidora contra o que batizou de "vagabundo do computador" (*computer bum*), seu rótulo desdenhoso para os hackers que habitavam as salas de máquinas:

> *"Suas roupas amarrotadas e encardidas, seus cabelos ensebados e desgrenhados, suas barbas por fazer... tudo neles atesta que são alheios aos próprios corpos e ao mundo real ao seu redor. Os vagabundos do computador existem — ao menos enquanto estão compenetrados na tela — unicamente através e para os computadores."*[3]

Passado um quarto de século do lançamento daquela obra, Stallman ainda range os dentes ao ouvir a expressão cunhada por Weizenbaum, rebatendo-a no tempo presente, como se o autor estivesse em carne e osso na sala:

— Ele queria que as pessoas fossem meros burocratas engravatados, trabalhando apenas pelo holerite no final do mês e ansiosos para bater o ponto e esquecer o trabalho o mais rápido possível — ataca Stallman. — O que para ele era a normalidade desejável da vida civilizada, para mim representa a mais pura tragédia humana.

A vida de um hacker, todavia, cobrava o seu quinhão de sofrimento. Stallman descreve sua transição de hacker bissexto de fins de semana para habitante de tempo integral do laboratório como uma sucessão de agruras dolorosas que só encontravam alívio no transe anestésico do código. Como ele próprio relata, a primeira fatalidade foi sua formatura em Harvard. Sedento por aprofundar suas pesquisas na física teórica, matriculou-se na pós-graduação do MIT. A escolha da instituição fora natural: não apenas lhe concedia a chance de seguir as pegadas de ex-alunos monumentais do instituto — gênios como William Shockley, Richard Feynman e Murray Gell-Mann —, mas também o colocava a três quilômetros mais perto do AI Lab e de seu reluzente computador PDP-10: "Minha atenção pendia cada vez mais para a informática, mas eu ainda alimentava a doce ilusão de que conseguiria abraçar as duas ciências ao mesmo tempo", recorda.

Desdobrando-se entre o rigor árido da física teórica durante o dia e a clausura monástica das madrugadas diante do PDP-10, Stallman tentou a duras penas equilibrar os pratos. O fiel da balança de sua sanidade mental era o ensaio semanal com o grupo de dança folclórica — sua única janela de convivência social e a única oportunidade garantida de interagir com mulheres. Perto do final de seu primeiro ano de pós-graduação, contudo, o destino desferiu-lhe um golpe traiçoeiro: uma lesão séria no joelho obrigou-o a parar imediatamente de dançar.

No início, encarou a dor como um revés passageiro: continuava comparecendo aos ensaios para conversar com os amigos e saborear as melodias balcânicas que amava. Ao final do verão, porém, com as dores persistindo e o recomeço das aulas batendo à porta, o desespero bateu:

— Meu joelho simplesmente não melhorava — relata Stallman com voz embargada. — Isso significava que eu teria que me conformar com a perspectiva assustadora de nunca mais poder dançar pelo resto da vida. Fiquei com o coração em pedaços.

Sem o dormitório protetor e sem a dança, o universo afetivo de Stallman ruiu como um castelo de cartas. O salão de dança fora o único território em que ele conseguira se aproximar de mulheres e, ocasionalmente, descolar

alguns encontros românticos. Ficar privado de dançar já era doloroso por si só; mas significava, ao que tudo indicava, ficar privado de afeto feminino para sempre.

— Senti como se todas as minhas forças tivessem sido drenadas do meu corpo — desabafa ele. — Perdi a energia para fazer qualquer coisa que não fosse aquilo que estava imediatamente ao meu alcance e que me dava prazer imediato. A disposição para enfrentar outras batalhas evaporou. Caí em desespero absoluto.

Stallman refugiou-se ainda mais fundo do mundo exterior, canalizando todas as forças que lhe restavam para os computadores do nono andar. Em outubro de 1975, tomou a decisão drástica: abandonou a pós-graduação no MIT, rompeu em definitivo com a física e nunca mais pisou em uma sala de aula como aluno. A programação, que outrora fora refúgio e passatempo, convertera-se formalmente em sua **vocação de vida**.

Olhando para aquele turbilhão em retrospecto, Stallman encara a metamorfose como inevitável. Mais cedo ou mais tarde, pondera, o canto da sereia do hacking teria engolido qualquer outra ambição profissional:

— Na física e na matemática pura, eu nunca vislumbrei uma maneira de dar uma contribuição verdadeiramente original — explica Stallman. — Eu teria imenso orgulho em fazer avançar qualquer um desses dois campos, mas simplesmente não conseguia enxergar por onde começar. Ficava tateando no escuro. No software, pelo contrário, eu enxergava de imediato como escrever linhas de código que rodassem perfeitamente e fossem úteis de verdade. A sensação embriagante de dominar esse poder me impelia a querer programar mais e mais.

Stallman não era o primeiro a associar a programação ao êxtase puro. Muitos dos luminares do AI Lab ostentavam históricos acadêmicos igualmente truncados e incompletos. Quase todos haviam aportado ali em busca de canudos em engenharia elétrica ou matemática, apenas para abandonar diplomas e carreiras convencionais diante do fascínio inebriante de solucionar charadas lógicas que ninguém no mundo jamais resolvera. A exemplo de São Tomás de Aquino — o teólogo medieval que passava tantas horas debruçado sobre suas *sumas* que alcançava transes místicos de exaustão —, os hackers atingiam estados alterados de consciência pela concentração extrema e pelo esgotamento físico. Embora abominasse drogas psicotrópicas, Stallman deliciava-se com a sensação quase etérea que coroava o final de uma maratona de vinte horas ininterruptas de código.

Contudo, a recompensa mais saborosa residia no sentimento de autorrealização. Quando se tratava de escovar bits, Stallman era um fenômeno da natureza. Uma infância inteira de madrugadas solitárias de leitura conferira-lhe a têmpera de varar noites a fio com pouquíssimo sono. Acostumado a ser um pária social desde os dez anos de idade, trabalhar isolado por dias não lhe causava o menor incômodo. E, na condição de matemático nato dotado de rigor lógico implacável, possuía a rara habilidade de antever e contornar armadilhas de arquitetura que deixavam a maioria dos programadores patinando no vazio.

— Ele era fora de série — recorda Gerald Sussman, professor catedrático do MIT e conselheiro fundador da Free Software Foundation. Descrevendo Stallman como um "pensador de uma clareza cristalina e um arquiteto de código impecável", Sussman convidou-o para participar de seus projetos pioneiros de pesquisa em inteligência artificial em 1973 e 1975, ambos dedicados a construir sistemas capazes de analisar diagramas de circuitos elétricos da mesma forma que os engenheiros humanos raciocinam. A empreitada exigia domínio virtuoso da linguagem Lisp — talhada sob medida para projetos de IA — somada à modelagem cognitiva humana desenvolvida por Sussman. A pesquisa de 1975 lançou as bases de uma técnica revolucionária em inteligência artificial conhecida como *dependency-directed backtracking* (ou manutenção de verdade), que consiste em formular

premissas provisórias, rastrear eventuais contradições lógicas e revisar as hipóteses pertinentes na raiz do problema.

Quando não estava envolvido em pesquisas formais dessa envergadura, Stallman dedicava-se de corpo e alma aos seus projetos pessoais. Era do interesse mais direto de qualquer hacker aprimorar as ferramentas do laboratório; e a maior paixão de Stallman naquele período atendia pelo nome de **TECO**.

A saga do aprimoramento do TECO durante a década de 1970 está umbilicalmente amarrada à futura liderança de Stallman à frente do software livre. Trata-se também de um capítulo definidor na história da computação moderna, a ponto de exigir uma breve reconstituição de contexto. Nas décadas de 1950 e 1960, quando os primeiros computadores aportaram nos campi universitários, programar era uma tarefa abstrata e exasperante. Para se comunicar com a máquina, os técnicos perfuravam centenas de cartões de papelão, onde cada perfuração representava uma instrução de máquina. A pilha de cartões perfurados era então entregue em mãos ao operador do centro de processamento de dados, que inseria o lote no leitor e aguardava o processador mastigar as instruções e cuspir um novo maço de cartões perfurados como resposta — que o programador precisava decifrar manualmente. Esse modelo, conhecido como **processamento em lote** (*batch processing*), era lento, burocrático e abria as portas para abusos de poder: um dos motivos da ojeriza congênita dos hackers por hierarquias residia exatamente no poder arbitrário que os operadores dos computadores centrais exerciam ao decidir quais projetos furavam a fila de processamento.

Em 1962, os pioneiros do Projeto MAC deram o primeiro passo para dinamitar esse modelo feudal. O **compartilhamento de tempo** (*time-sharing*) permitiu que múltiplos usuários utilizassem os ciclos de processamento de uma única máquina simultaneamente. Os terminais de teletipo (*teleprinters*) viabilizaram a comunicação com o computador não mais por buracos em cartões, mas através de texto digitado: o programador inseria instruções pelo teclado e lia a resposta impressa linha a linha no rolo de papel.

No final dos anos 1960, a interface homem-máquina deu outro salto quântico. Em uma lendária apresentação pública em 1968 — que a história consagraria como "A Mãe de Todas as Demonstrações" —, Doug Engelbart, pesquisador do Stanford Research Institute, revelou ao mundo o protótipo da interface interativa moderna. Conectando um monitor de raios catódicos ao computador e acoplando-lhe um dispositivo apontador de madeira batizado de **mouse**, Engelbart criou um ambiente infinitamente mais dinâmico do que o sistema do MIT. Tratando a tela de vídeo como uma superfície de impressão instantânea, seu protótipo permitia mover o cursor pela tela e ver o texto ser reposicionado em tempo real.

Inovações dessa magnitude levariam mais de duas décadas para aterrissar nos computadores de consumo em massa. Contudo, no início dos anos 1970, telas de fósforo verde começavam a substituir os teletipos barulhentos no AI Lab do MIT, descortinando a possibilidade de edição de texto em **tela cheia** — superando a velha edição linha por linha.

Um dos primeiros programas a explorar o potencial dos monitores foi justamente o **TECO** do AI Lab. Abreviação de *Text Editor and COrrector*[4], o editor havia sido adaptado pelos hackers a partir de um utilitário rudimentar de fita do velho PDP-6.

O TECO representava um avanço colossal sobre as ferramentas anteriores, mas ainda guardava limitações indigestas. Para redigir ou alterar um documento, o usuário precisava digitar uma longa sequência de comandos cifrados especificando cada modificação antes de mandar executá-los. Era um exercício puramente abstrato: ao

contrário dos editores modernos, que atualizam o texto a cada toque de tecla, o TECO exigia teclar uma sequência hermética de comandos seguida de dois caracteres de escape (*escape-escape*) para aplicar a mudança. Com o tempo, os hackers adquiriram virtuosismo suficiente para disparar cirurgias monumentais em uma única linha de comando; mas, como o próprio Stallman advertiria mais tarde, a ginástica mental exigida pelo TECO assemelhava-se à destreza de **jogar xadrez às cegas**[5].

Para suavizar o suplício, os hackers dividiram o monitor em duas metades: uma exibindo o texto do arquivo e a outra mostrando a linha de comando do TECO. Apesar desse hack inovador, editar com o TECO continuava sendo um esporte para iniciados.

O TECO não era o único editor de tela cheia que circulava pelos laboratórios da época. Durante uma visita ao Laboratório de Inteligência Artificial da Stanford University (SAIL) em 1976, Stallman deparou-se com um programa de edição chamado **E**. O software trazia um recurso interno genial que atualizava a tela instantaneamente a cada tecla pressionada pelo usuário. No jargão da informática dos anos 1970, o E foi um dos primeiros editores rudimentares do tipo **WYSIWYG** — sigla em inglês para *"What You See Is What You Get"* ("O que você vê é o que você tem"), significando que o usuário manipulava o texto diretamente sobre a tela, sem intermediários.

Fascinado pela elegância da solução, Stallman regressou a Cambridge decidido a incorporar dinâmica semelhante ao TECO. Ele encontrou no sistema um remendo recente batizado de **Control-R**, escrito pelo colega Carl Mikkelson e nomeado a partir da combinação de teclas que o acionava. O hack de Mikkelson colocava o TECO em modo de edição em tempo real caractere a caractere, mas trazia dois defeitos graves: usava apenas cinco linhas da tela e era lerdo demais para o trabalho cotidiano.

Stallman reescreveu o código do Control-R do zero para utilizar a tela inteira com máxima eficiência e, em seguida, operou um salto genial: **permitiu aos usuários atrelar sequências inteiras de comandos do TECO — as chamadas "macros" — a qualquer tecla ou atalho do teclado**. Os programadores mais avançados já guardavam suas macros prediletas em arquivos separados; o golpe de mestre de Stallman permitiu invocá-las na ponta dos dedos em milissegundos. O resultado prático foi o primeiro editor WYSIWYG totalmente programável e customizável pelo usuário na história da informática.

— Aquele foi o verdadeiro divisor de águas! — entusiasma-se Guy Steele, então jovem hacker no AI Lab.

Nas palavras do próprio Stallman, a criação das macros deflagrou uma verdadeira explosão de criatividade coletiva no laboratório:

— Todo mundo e a torcida do Flamengo passou a programar sua própria coleção personalizada de comandos no editor: havia uma macro para cada rotina que o sujeito costumava fazer com frequência — relembra Stallman. — Os programadores passavam as rotinas uns para os outros, corrigiam bugs, tornavam os comandos mais genéricos e mais poderosos. Com o tempo, aquelas coleções de macros transformaram-se em verdadeiros sistemas operacionais por mérito próprio!

Tamanha era a utilidade das macros que o TECO em si tornou-se mero detalhe secundário diante da febre de customização que tomou conta do prédio: "Passamos a encarar o editor não mais como um simples formatador de textos, mas como uma linguagem de programação completa!", celebra Stallman.

Dois anos após a euforia inicial, contudo, a explosão de inovação começou a cobrar um preço inconveniente. O crescimento caótico funcionara como uma comprovação brilhante do poder da colaboração hacker, mas gerara uma grave crise de interoperabilidade:

— Nós nos deparamos com uma autêntica **Torre de Babel** — resume Guy Steele.

A Torre de Babel ameaçava implodir o próprio ecossistema fraterno que a gerara. O sistema ITS fora esculpido para permitir que os programadores compartilhassem conhecimento e aprimorassem o código alheio com fluidez. Isso pressupunha sentar diante do terminal do colega de baia, abrir os arquivos dele e fazer reparos ou correções diretamente no código: "Muitas vezes, a forma mais rápida e generosa de ensinar alguém a programar ou caçar um bug era simplesmente sentar na cadeira dele e resolver a charada ali mesmo", explica Steele.

O dilúvio de macros personalizadas implodiu essa camaradagem: de repente, cada hacker havia customizado seu teclado e seus atalhos a tal ponto que, ao se sentar na máquina do vizinho, perdia a primeira hora de trabalho apenas tentando decifrar o que cada tecla fazia.

Incomodado com a paralisia, Guy Steele resolveu botar ordem na casa. Reuniu os quatro principais pacotes de macros que circulavam pelo prédio e começou a desenhar uma tabela unificada, catalogando as instruções mais úteis e consensuais. Enquanto rascunhava a implementação dessa tabela no teclado, chamou a atenção de Stallman:

— O Richard começou a olhar por cima do meu ombro, curioso, querendo saber o que eu estava aprontando — relata Steele.

Para Steele — um jovem tímido e reservado que raramente se misturava com Stallman —, a lembrança é indelével. Espiar o código por cima dos ombros alheios era prática corriqueira no laboratório. Stallman, na qualidade de mantenedor oficial do código do TECO, achou a iniciativa de Steele "extremamente interessante" e tomou para si a missão de levá-la às últimas consequências:

— Como costumo brincar por aí, eu fiz o primeiro zero vírgula zero zero um por cento da implementação; o Stallman fez todo o restante! — diverte-se Steele.

O nome do novo projeto unificado saiu da lavra de Stallman: **Emacs** — contração de *Editing Macros* ("Macros de Edição"). O batismo simbolizava o salto evolutivo operado sobre a base do TECO e preenchia uma lacuna oportuna: constatando que não havia programas no ITS iniciando com a letra "E", Stallman escolheu Emacs para que os usuários pudessem invocar o programa com um simples toque da letra E. A obsessão hacker pela eficiência deixava mais uma vez sua marca indelével.

É evidente que nem todos os usuários migraram para o Emacs de imediato: cada um continuava livre para manter suas velhas rotinas individuais. Mas a esmagadora maioria aderiu com entusiasmo, sobretudo porque a arquitetura do Emacs fora desenhada para permitir que qualquer pessoa substituísse ou acopladas novas funções sem bagunçar a espinha dorsal do sistema:

— Por um lado, estávamos tentando resgatar um conjunto padronizado e universal de atalhos; por outro, queríamos manter o sistema infinitamente extensível, porque a liberdade de programar era essencial — pontua Steele.

Stallman viu-se então diante de um dilema sociológico crucial: se os programadores fizessem alterações individuais no Emacs e não devolvessem essas melhorias à coletividade, a Torre de Babel reapareceria logo adiante, fragmentando a comunidade mais uma vez. Resgatando o mandamento sagrado da partilha hacker, Stallman inseriu uma declaração pioneira no cabeçalho do código-fonte estabelecendo os termos de convivência da ferramenta: os usuários tinham total liberdade para modificar e redistribuir o código, **desde que devolvessem à comunidade todas as melhorias e extensões que realizassem**.

Stallman batizou esse pacto de honra de **"A Comuna do Emacs"** (*The Emacs Commune*).

Assim como o TECO deixara de ser um mero editor para virar uma linguagem, o Emacs transcendera a condição de software para se converter em um **contrato social**. Em um memorando histórico de 1981, Stallman registrou a essência desse pacto coletivo: *"O EMACS foi distribuído sobre uma base de partilha comunitária, o que significa que todas as melhorias devem obrigatoriamente ser entregues de volta a mim para serem incorporadas e redistribuídas a todos"*.

O Emacs original fora projetado sob medida para o computador PDP-10, mas não tardou para que usuários de outras plataformas passassem a sonhar com uma versão para chamar de sua. A febre de inovação alastrou-se ao longo da década, gerando uma constelação de editores inspirados no Emacs, com variados graus de compatibilidade. As regras da Comuna do Emacs não se aplicavam a esses clones externos, visto que seus códigos eram independentes. Alguns reverenciavam o projeto original adotando nomes recursivos e espirituosos: **Sine** (*Sine Is Not Emacs*), **Eine** (*Eine Is Not Emacs*) e **Zwei** (*Zwei Was Eine Initially*). Para ser considerado um "Emacs legítimo", o programa precisava oferecer plena extensibilidade e programabilidade pelo usuário; ferramentas que apenas imitavam seus atalhos de teclado mas careciam de capacidade de programação interna eram desdenhadas pela comunidade como "Emacs fajutos" (*ersatz Emacs*) — caso do **Mince** (*Mince Is Not Complete Emacs*).

Enquanto Stallman cuidava do Emacs no santuário do AI Lab, nuvens pesadas começavam a se formar no horizonte da cultura hacker. A decisão de Brian Reid em 1979 de injetar bombas-relógio no código do Scribe para garantir que a Unilogic cobrasse pelo acesso soou para Stallman como o prenúncio do fim dos tempos: "Ele considerou aquilo a coisa mais nazista que já tinha visto na vida", recorda o próprio Reid. Apesar de ter conquistado a glória anos mais tarde na internet como o criador da hierarquia *alt* da Usenet, Reid confessa que até hoje carrega essa mancha no conceito de Stallman: "Ele dizia que todo software deveria ser livre e que cobrar dinheiro para restringir o acesso a programas de computador era um crime contra a humanidade!" [6].

Embora não tivesse como impedir a venda corporativa do Scribe, Stallman descobriu que tinha poder de fogo suficiente para barrar comportamentos antiéticos dentro de seu próprio quintal. Na condição de mantenedor supremo do código-fonte do Emacs, passou a usar sua caneta com precisão política. No auge do conflito com os burocratas do Laboratory for Computer Science (LCS) sobre a imposição de senhas, deflagrou uma verdadeira **"greve de software"**: recusou-se sumariamente a enviar as atualizações do Emacs para os pesquisadores do LCS enquanto eles não abolissem as senhas dos computadores! Tratava-se de um gesto mais simbólico do que propriamente punitivo — afinal, nada impedia que os próprios cientistas compilassem o código por conta própria —, mas a mensagem ecoou em alto e bom som: pactuar com senhas atrairia o repúdio público da Comuna do Emacs.

— Muita gente ficou furiosa comigo, dizendo que eu estava fazendo chantagem ou tomando-os como reféns — o que, de certa forma, era verdade! — confessaria Stallman anos mais tarde a Steven Levy. — Eu estava praticando

uma violência simbólica contra eles porque acreditava sinceramente que eles estavam praticando uma violência real contra toda a comunidade[7]!

Com o passar dos anos, o Emacs converteu-se na maior vitrine viva da Ética Hacker. A flexibilidade monumental que Stallman desenhou em suas entranhas não apenas estimulava a cooperação desinteressada: **ela a tornava indispensável**. Os desenvolvedores que não acompanhassem as discussões da Comuna ou guardassem suas melhorias para si corriam o risco de ficar para trás das inovações extraordinárias que brotavam semanalmente. Vinte anos depois, os usuários do GNU Emacs (uma segunda implementação lendária iniciada em 1984) haviam adaptado o editor para tantas funções impensáveis — usando-o como planilha eletrônica, calculadora simbólica, cliente de e-mail, gerenciador de arquivos e até navegador web — que a comunidade passou a representá-lo com a imagem icônica de uma **pia de cozinha transbordando de utensílios**: "Essa era exatamente a ideia que queríamos transmitir", diverte-se Stallman. "A quantidade avassaladora de coisas que o Emacs carrega em seu bojo é, simultaneamente, maravilhosa e assustadora!".

Os contemporâneos de Stallman no MIT são pródigos em elogios. Hal Abelson, professor catedrático do MIT e conselheiro histórico da Free Software Foundation, define o Emacs como "uma criação absolutamente genial". Ao fornecer aos programadores uma plataforma aberta para acoplar novas bibliotecas sem implodir o sistema, Abelson ressalta que Stallman inventou o modelo de desenvolvimento colaborativo em rede décadas antes de a internet virar febre:

— A arquitetura que ele concebeu era tão robusta e elegante que permitia a pessoas espalhadas pelos quatro cantos do planeta colaborar espontaneamente e enriquecer o código em tempo real — atesta Abelson. — Sinceramente, duvido que alguém já tivesse realizado algo parecido na história da tecnologia[8].

Guy Steele expressa reverência semelhante. Pesquisador dos laboratórios da Sun Microsystems, ele recorda Stallman antes de tudo como um "programador monumental, dotado da capacidade assombrosa de cuspir montanhas de código praticamente isentas de bugs". Embora seus temperamentos não fossem os mais compatíveis, os dois cooperaram tempo suficiente para que Steele testemunhasse a intensidade demolidora do estilo de trabalho de Stallman. Ele resgata uma tarde inesquecível no final dos anos 1970, quando os dois se uniram para programar o recurso de *pretty print* do editor — uma rotina concebida por Steele para reformatar e indentar automaticamente o código-fonte na tela, tornando-o mais legível e compacto.

— Sentamo-nos juntos em uma manhã ensolarada — relata Steele. — Eu assumi o teclado e o Stallman colou-se ao meu cotovelo. Ele não se importava em me deixar digitar, mas ditava com precisão cirúrgica cada comando que eu devia teclar.

A sessão de programação estendeu-se por **dez horas ininterruptas**. Ao longo de todo aquele tempo, assegura Steele, nenhum dos dois levantou da cadeira, fez uma pausa para tomar água ou trocou uma única palavra de amenidades. Ao término da maratona, haviam condensado a rotina inteira em menos de cem linhas de código puro e deslumbrante:

— Meus dedos permaneceram sobre o teclado o tempo todo — rememora Steele —, mas a sensação nítida era a de que as ideias de nós dois estavam jorrando unificadas diretamente sobre o fósforo da tela. Ele me dizia o que teclar e eu teclava.

A noção real do tempo só bateu quando Steele finalmente deixou o prédio no número 545 da Tech Square: ao pisar na calçada, levou um choque ao ver que a cidade estava mergulhada na escuridão profunda da noite. Como qualquer programador, estava acostumado a varar horas em claro; contudo, aquela sessão fora completamente diferente. Programar colado a Richard Stallman forçara Steele a anestesiar todos os estímulos do universo e focar cem por cento de sua energia neural naquela tarefa. Olhando em retrospecto, Steele confessa que aquela fusão mental de cérebros foi tão sublime quanto aterrorizante:

— Minha primeiríssima reação ao colocar os pés na rua foi pensar: *esta foi a experiência intelectual mais arrebatadora e intensa da minha vida inteira — e eu nunca mais quero passar por isso de novo enquanto viver!*

Ver Josh McHugh, *"For the Love of Hacking"*, *Forbes* (10 de agosto de 1998). ↩

Ver Richard Stallman, conferência no KTH (1986). ↩

Ver Joseph Weizenbaum, *Computer Power and Human Reason: From Judgment to Calculation* (1976), p. 116. ↩

O nome original do TECO derivava de *Tape Editor and Corrector*. ↩

Ver Richard Stallman, *"EMACS: The Extensible, Customizable, Display Editor"*, AI Lab Memo (1979): http://www.gnu.org/software/emacs/emacs-paper.html. ↩

Em entrevista em 1996, Stallman voltou ao tema: "A grande tragédia daquele episódio foi que ninguém repreendeu ou puniu aquele estudante pelo que ele fez. O resultado nefasto dessa omissão foi que outros programadores sentiram-se tentados a seguir o seu mau exemplo". ↩

Ver Steven Levy, *Hackers* (1984), p. 419. ↩

No memorando de 1979, Stallman sintetizou de forma comovente a filosofia dessa criação: *"O EMACS não poderia ter nascido de um processo rígido e burocrático de planejamento antecipado, porque tais processos só alcançam metas que já são visíveis desde o primeiro instante e cuja utilidade financeira está comprovada na largada. Nem eu nem ninguém jamais vislumbrou um editor infinitamente extensível até que eu construísse um protótipo, tampouco compreendemos o seu valor até que o tivéssemos experimentado na prática. O EMACS existe unicamente porque me senti livre para implementar pequenas melhorias cotidianas ao longo de uma trilha cujo destino final ninguém era capaz de enxergar."* ↩

Capítulo 7: Uma Dura Escolha Moral

Na madrugada de 27 de setembro de 1983, os programadores que se conectavam ao grupo de discussão `net.unix-wizards` da rede Usenet depararam-se com uma mensagem insólita. Enviada exatamente à meia-noite e meia, assinada pelo endereço `rms@mit-oz`, o assunto era seco, mas irresistível: *"Nova implementação do UNIX"*. Em vez de anunciar mais uma variante comercial do sistema da AT&T, contudo, o primeiro parágrafo convocava a comunidade para uma verdadeira declaração de guerra:

> *"A partir deste feriado de Ação de Graças, vou começar a escrever um sistema de software completo e compatível com o Unix chamado GNU (de 'GNU Não É Unix'), e vou distribuí-lo livremente para qualquer pessoa que possa utilizá-lo. Contribuições de tempo, dinheiro, programas e equipamentos são extremamente necessárias."*[1]

Para qualquer desenvolvedor Unix experiente, aquele anúncio soava como um misto desconcertante de idealismo pueril e arrogância desmedida. O autor não apenas prometia reconstruir do zero, com as próprias mãos, um sistema operacional maduro e complexo, mas pretendia superá-lo em diversos quesitos. O novo sistema GNU, profetizava a mensagem, traria todas as ferramentas clássicas — um editor de texto de ponta, um interpretador de comandos (*shell*), um compilador de linguagem C, "e mais um punhado de outras coisas". Além disso, incorporaria recursos com os quais os usuários do Unix comercial apenas sonhavam: uma interface gráfica alicerçada na linguagem Lisp, um sistema de arquivos à prova de falhas e protocolos de rede herdados da arquitetura interna do MIT:

> *"O GNU será capaz de rodar programas feitos para o Unix, mas não será idêntico ao Unix", alertava o manifesto. "Implementaremos todas as melhorias e comodidades convenientes, com base na nossa vasta experiência prática com outros sistemas operacionais."*

Antecipando a enxurrada de ceticismo que seu projeto atrairia, o autor arrematava a proposta com um minicurrículo intitulado *"Quem sou eu?"*:

> *"Sou Richard Stallman, criador do editor EMACS original (tão imitado por aí), atualmente trabalhando no Laboratório de Inteligência Artificial do MIT. Trabalhei extensivamente no desenvolvimento de compiladores, editores, depuradores, interpretadores de comandos, no Incompatible Timesharing System e no sistema operacional da Máquina Lisp. Fui o pioneiro no suporte a telas de vídeo independentes de terminal no ITS. Além disso, implementei um sistema de arquivos à prova de travamentos e dois sistemas de gerenciamento de janelas para Máquinas Lisp."*

Por capricho do destino, o grandioso Projeto GNU não conseguiu cumprir o prazo de lançamento no feriado de Ação de Graças. Em janeiro de 1984, porém, Stallman honrou a palavra empenhada e mergulhou de corpo e alma

no ecossistema de desenvolvimento do Unix. Para um arquiteto de software acostumado às liberdades aristocráticas do ITS, aquilo parecia a ingrata tarefa de desenhar galpões pré-fabricados de subúrbio em vez de erguer palácios mouriscos. Ainda assim, construir um clone do Unix trazia trunfos estratégicos invisíveis a olhos leigos. O sistema ITS fora uma obra-prima de poder, mas carregava um calcanhar de Aquiles fatal: fora programado sob medida para a arquitetura proprietária dos computadores PDP-10 da Digital Equipment Corporation (DEC). Quando a cúpula do AI Lab decidiu aposentar o venerável PDP-10 no início dos anos 1980, o sistema que os hackers outrora comparavam a uma metrópole vibrante transformou-se, da noite para o dia, em uma cidade-fantasma deserta.

O Unix, pelo contrário, nascera para ser **portável** — e essa virtude o tornava imune à obsolescência do hardware. Desenvolvido originalmente por jovens pesquisadores nos laboratórios da AT&T, o sistema escorregara pelas brechas da burocracia corporativa e encontrara abrigo fértil nos departamentos acadêmicos de computação que sobreviviam com orçamentos minguados. Contando com recursos incomparavelmente mais modestos do que os faraós do MIT, os programadores do Unix haviam moldado o software para rodar em uma fauna heterogênea de máquinas — a começar pelos modestos minicomputadores PDP-11 de 16 bits (que os hackers do AI Lab desdenhavam como meros brinquedos de laboratório) e, mais tarde, em computadores de grande porte de 32 bits, como o VAX 11/780. Por volta de 1983, empresas emergentes como a Sun Microsystems começavam a projetar uma nova geração de computadores de mesa — as chamadas "estações de trabalho" (*workstations*) —, concebidas sob medida para rodar aquele sistema onipresente em máquinas que rivalizavam em poder de fogo com o velho PDP-10.

Para garantir a portabilidade, os pais do Unix inseriram uma camada crucial de abstração entre o código e a máquina física. Em vez de escrever tudo em linguagem de montagem amarrada aos circuitos de um único processador — como os pioneiros do MIT haviam feito com o ITS e o PDP-10 —, escreveram o Unix em uma linguagem de alto nível chamada **C**. Ao priorizarem as interfaces modulares e os padrões de comunicação que amarravam os diversos utilitários do sistema, em vez de soldar os componentes diretamente à placa-mãe, criaram uma arquitetura capaz de ser portada para qualquer computador em questão de semanas. Se um desenvolvedor detestava um utilitário específico, bastava desparafusá-lo da engrenagem e colocar outro melhor no lugar. Em suma: o modelo Unix privilegiava a flexibilidade, a economia e a modularidade — daí sua expansão vertiginosa[2].

A decisão de Stallman de dar início ao Projeto GNU foi deflagrada diretamente pelo **aniquilamento do sistema ITS**, que os veteranos do AI Lab haviam cultivado com tanto amor ao longo de mais de uma década. O colapso do ITS e o estilhaçamento da irmandade hacker que o sustentava desferiram um golpe traumático na alma de Stallman. Se a celeuma da impressora a laser da Xerox lhe ensinara a enxergar a baixeza moral do software proprietário, a destruição de sua comunidade forçou-o a escolher entre curvar a cabeça à privatização do conhecimento ou dedicar o resto de seus dias a combatê-la na trincheira.

As raízes daquela tragédia vinham de longe. Por volta de 1980, praticamente todo o estado-maior de hackers do laboratório estava mobilizado em uma nova obsessão: a **Máquina Lisp** (*Lisp Machine*) e seu inovador sistema operacional.

Criada no final da década de 1950 pelo pioneiro John McCarthy — um dos pais da inteligência artificial no MIT —, a linguagem Lisp (*LISt Processing*) destacava-se pela elegância matemática incomparável, sendo talhada sob medida para processar dados de estrutura irregular e criar sistemas inteligentes. Após a partida de McCarthy para Stanford, os programadores do MIT refinaram a linguagem em um dialeto local sofisticadíssimo batizado de MACLISP. Sob o comando de Richard Greenblatt, os hackers projetaram nos anos 1970 um computador

revolucionário, desenhado nos circuitos para processar Lisp com velocidade espantosa — a lendária Máquina Lisp — e escreveram para ela um sistema operacional inteiramente concebido nessa linguagem.

Em 1980, contudo, a cobiça comercial partiu a comunidade ao meio. Dois grupos rivais de hackers fundaram empresas concorrentes para fabricar e vender a Máquina Lisp no mercado privado. De um lado, Richard Greenblatt fundou a **Lisp Machines Incorporated (LMI)**: seu plano era recusar aportes de capital de risco e erguer uma "empresa hacker", autossuficiente e gerida pelos próprios técnicos. Do outro lado, Russell Noftsker e a esmagadora maioria dos programadores do laboratório fundaram a **Symbolics** — uma startup convencional, turbinada por milhões de dólares de fundos de investimento agressivos. Em 1982, quase todos haviam batido em retirada do MIT para se dedicar exclusivamente aos seus negócios corporativos.

Sem hackers para cuidar da casa, as máquinas e os softwares do laboratório começaram a apodrecer: falhas levavam semanas para serem reparadas, quando eram consertadas. Para piorar a agonia, o AI Lab sofreu uma mutação demográfica desastrosa: os hackers rebeldes, que outrora ditavam o tom da casa, praticamente desapareceram, enquanto os acadêmicos convencionais e os alunos burocratas — que nunca haviam nutrido amor pelo PDP-10 — mantinham-se firmes em suas cadeiras.

Em 1982, o laboratório recebeu o substituto de seu computador central PDP-10, já com doze anos de serviços prestados. O novo modelo da Digital, o Decsystem 20, até rodava os programas antigos dos usuários, mas exigiria reescrever do zero o sistema operacional ITS para funcionar na nova máquina. Apavorados com a debandada dos gênios da programação, os professores catedráticos do departamento pressionaram a direção para adotar o **Twenex** — o sistema comercial oficial distribuído pela própria fabricante DEC. Em minoria esmagadora, os raros hackers remanescentes não tiveram forças para barrar a capitulação.



— Sem os hackers para manter o sistema em pé, os professores diziam em pânico: "Isso vai ser um desastre completo; precisamos de um software comercial de prateleira!" — desabafaria Stallman anos mais tarde. — Eles diziam ingenuamente: "Podemos contar com o suporte da empresa para cuidar de tudo". O tempo provou que eles estavam redondamente enganados, mas foi exatamente isso o que fizeram[3].

A princípio, os resistentes encararam o novo sistema operacional como mais um símbolo autoritário pronto para ser sabotado. O próprio nome popular do sistema era um deboche: oficialmente batizado pela Digital de **TOPS-20**, o sistema fora lançado como sucessor do velho TOPS-10 proprietário. Na realidade, contudo, não descendia do TOPS-10, mas fora usurpado a partir do sistema Tenex, desenvolvido pela empresa Bolt Beranek and Newman. Stallman, autor do trocadilho, confessou a origem do termo: "O sistema estava a anos-luz de ser *tops* (excelente), de modo que eu jamais o chamaria por aquele nome; resolvi enfiar um 'w' no meio de Tenex e passei a chamá-lo de **Twenex**!".

O computador físico que rodava o Twenex ganhou dos hackers outro apelido impagável: **Oz**. A lenda no laboratório conta que a máquina exigia um minicomputador PDP-11 acoplado apenas para gerenciar o terminal de controle. Ao presenciar aquela parafernália pela primeira vez, um hacker não resistiu e imitou a voz estrondosa do Mágico de Oz no clássico do cinema: *"Eu sou o grande e poderoso Oz! Não prestem atenção no pequeno PDP-11 escondido atrás daquela cortina!"*.

Se a máquina arrancava risadas pela pompa cômica, a graça evaporava diante do Twenex. O sistema trazia segurança de fábrica entranhada em cada linha de comando; e os engenheiros da DEC haviam projetado todas as

ferramentas pressupondo a blindagem por senhas. Aquela guerrilha bem-humorada da década de 1970 contra as senhas convertera-se em uma guerra aberta pela administração dos servidores. Os burocratas sustentavam que, sem senhas, o Oz viveria caindo por imperícia dos usuários; os hackers retrucavam que as quedas seriam evitadas com melhorias no código-fonte. Contudo, como o contingente de hackers com tempo e disposição para escovar bits havia minguado a quase zero, a tese burocrática prevaleceu.

A política inicial do laboratório estabelecia que qualquer pesquisador credenciado tinha direito ao chamado **privilégio da roda** (*wheel privilege*) — a autorização de superusuário que permitia contornar as travas do sistema. Contudo, quem detinha o privilégio da roda podia arbitrariamente revogá-lo dos demais colegas, deixando-os de mãos atadas. Não demorou para que uma facção de administradores tentasse dar um golpe de mão, cancelando o privilégio de todos os outros usuários para concentrar o poder supremo em suas próprias mãos.

Vasculhando senhas e interceptando o sistema com o depurador logo na inicialização da máquina, Stallman frustrou as manobras autoritárias uma a uma. Após desarmar o segundo "golpe de Estado", disparou um comunicado geral a todos os pesquisadores do laboratório:

> *"Houve mais uma tentativa vil de tomada do poder", alertava o texto. "Até o presente momento, as forças aristocráticas foram rechaçadas e derrotadas."*

Para não ser punido na hora, Stallman assinou o panfleto sob o pseudônimo de **"Rádio Oz Livre"** (*Radio Free OZ*).

O disfarce era transparente como água. Em 1982, sua repulsa visceral a senhas e segredos já era tão célebre nos círculos técnicos que usuários de todos os cantos dos Estados Unidos conectavam-se aos computadores do MIT através da **ARPAnet** — a rede pioneira financiada pelos militares que serviu de semente para a internet moderna. Um desses "turistas" digitais foi o programador californiano Don Hopkins que, ao ouvir os boatos na comunidade, descobriu que para explorar as maravilhas do lendário sistema ITS do MIT bastava digitar o login `RMS` e, quando a tela solicitasse a senha, digitar rigorosamente as mesmas três letras: `RMS`.

— Serei eternamente grato ao MIT por ter permitido que eu e centenas de outros jovens usássemos seus computadores de graça — reconhece Hopkins. — Aquilo mudou a vida de muita gente.

Essa acolhedora política de "turistas", tolerada com generosidade durante os anos de ouro do ITS, foi assassinada no instante em que o servidor Oz assumiu o posto de conexão primária do laboratório com a ARPAnet. No início, Stallman bateu o pé e continuou cadastrando suas iniciais como senha para que qualquer forasteiro pudesse acessar a máquina por sua conta. Com o tempo, contudo, a instabilidade crônica do Twenex levou os administradores a proibir a entrada de visitantes. Quando a diretoria enquadrou Stallman exigindo que ele parasse de divulgar sua senha ao público, ele invocou seus princípios morais e tomou uma decisão radical: **recusou-se terminantemente a usar o servidor Oz daquele dia em diante**[4].

A recusa de Stallman em se ajoelhar diante do Mágico de Oz simbolizava a fratura exposta entre o espírito hacker e a nova burocracia universitária. Contudo, essa tensão empalidecia perto da guerra fratricida que estalara dentro da própria comunidade hacker. Com a chegada do Decsystem 20, a irmandade ruiu, cindida em dois exércitos irreconciliáveis: a **LMI** e a **Symbolics**.

A Symbolics, munida de rios de dinheiro de investidores privados, contratou quatorze pesquisadores do laboratório como consultores externos para aprimorar sua versão proprietária da Máquina Lisp fora dos muros acadêmicos. O punhado de técnicos remanescentes fechou com a LMI. Stallman, preferindo a serenidade desimpedida do laboratório e recusando-se a tomar partido no jogo corporativo, optou por não se filiar a nenhuma das duas companhias.

A princípio, os programadores de ambas as empresas continuaram frequentando o nono andar em meio período, colaborando com o sistema operacional da Máquina Lisp mantido pelo MIT. Tanto a LMI quanto a Symbolics haviam licenciado o código básico da universidade sob um contrato que as obrigava a devolver as correções ao MIT, embora sem exigir expressamente que a universidade redistribuísse essas melhorias publicamente. Ao longo de todo o ano de 1981, contudo, as duas rivais honraram um **acordo de cavalheiros**: todas as inovações eram incorporadas à versão oficial do MIT e, por tabela, partilhadas com todos os usuários de Máquinas Lisp do mundo. Esse pacto permitia que quem permanecera no MIT mantivesse uma neutralidade confortável.

No dia **16 de março de 1982** — data que Stallman jamais esqueceria por ser o dia do seu aniversário de 29 anos —, a diretoria da Symbolics rasgou o acordo de cavalheiros. O objetivo da manobra era aniquilar a LMI de vez: dispondo de menos capital e de uma equipe menor de programadores, a LMI dependia umbilicalmente das melhorias compartilhadas do sistema para manter suas máquinas competitivas no mercado. Ao estancar a partilha de código, a Symbolics pretendia estrangular a concorrente financeiramente. Para isso, decidiu cumprir a ferro e fogo a letra fria do contrato: em vez de entregar suas melhorias à versão comunitária do MIT (que a rival LMI podia aproveitar), passou a fornecer à universidade apenas uma cópia compilada de sua própria versão proprietária. Qualquer pesquisador do MIT que usasse o sistema estaria, na prática, trabalhando como testador gratuito e exclusivo para a Symbolics; e qualquer melhoria feita por ele só beneficiaria a corporação.

Na condição de responsável técnico pela manutenção do sistema operacional das Máquinas Lisp do laboratório, Stallman sentiu o sangue ferver nas veias. Para piorar o quadro, os desertores da Symbolics haviam deixado o código comunitário do MIT crivado de centenas de rotinas inacabadas que disparavam erros em cascata. Encarando a manobra da empresa como um ultimato intolerável, Stallman partiu para o revide: cortou fisicamente a conexão de micro-ondas que ligava a sede da Symbolics aos servidores do laboratório. Em seguida, jurou solenemente jamais encostar as mãos em uma máquina da Symbolics e assumiu a missão heroica de manter vivo e atualizado o sistema público do MIT, garantindo que a LMI não fosse massacrada no mercado:

— Pelo meu modo de enxergar as coisas, o AI Lab era um país neutro no meio do tiroteio, exatamente como a Bélgica na Segunda Guerra Mundial — compara Stallman. — Se a Alemanha invade a Bélgica covardemente, a Bélgica declara guerra à Alemanha e fecha fileiras ao lado da França e da Inglaterra!

Quando os executivos da Symbolics perceberam, estarrecidos, que cada um dos seus recursos mais avançados e secretos continuava brotando misteriosamente no sistema público do MIT — e, por extensão, nos computadores da LMI —, o clima azedou. Stallman conhecia as leis de direitos autorais e reescrevia absolutamente todas as funções do zero. Ele aproveitava o direito legítimo de ler o código-fonte que a Symbolics entregara ao MIT para decifrar os gargalos técnicos e as soluções adotadas e, em seguida, redigia uma implementação completamente diferente e original.

A cúpula da Symbolics recusou-se a acreditar naquilo. Convictos de que se tratava de espionagem e plágio descarado, instalaram um programa espião no terminal de Stallman para colher provas de infração de direitos

autorais. Contudo, ao baterem à porta da reitoria do MIT no início de 1983 para exigir a demissão de Stallman, apresentaram evidências ridículas: míseros doze pontos no código-fonte onde as rotinas pareciam semelhantes.

Quando os diretores do laboratório apresentaram as supostas provas a Stallman, ele desmontou a farsa em minutos, provando por A mais B que aquelas similaridades provinham de códigos antigos, escritos antes da cisão entre as duas empresas. Em seguida, inverteu a lógica da acusação com maestria demolidora: se, após dezenas de milhares de linhas de código que ele escrevera sozinho, a Symbolics só conseguira pescar doze trechos vagamente parecidos, aquilo era a prova cabal de que seu esforço para evitar o plágio fora impecável! A direção do MIT deu-lhe ganho de causa integral, e Stallman continuou sua cruzada solitária até o final de 1983.

Para não dar margem a novas intrigas, contudo, ele refinou seu método:

— Por excesso de zelo, parei de ler o código-fonte deles para novos recursos — relata Stallman. — Passei a ler unicamente a documentação das funções e escrevia o programa do zero a partir do manual. Nas inovações mais complexas, eu sequer esperava a Symbolics publicar o manual: desenhava a solução da minha própria cabeça e, quando a documentação deles vinha a público, eu apenas adicionava compatibilidade de interface. Depois, lia os relatos de bugs da Symbolics e corrigia cada um deles por um caminho técnico completamente diferente.

Aquela batalha épica forjou a têmpera definitiva de Richard Stallman. Conforme programava os equivalentes para cada ferramenta da Symbolics, mobilizava os usuários do MIT para testar o sistema comunitário e enviar uma torrente contínua de relatórios de bugs. E a universidade repassava tudo, de mão beijada, para a LMI:

— Eu estava decidido a castigar a Symbolics nem que fosse a última coisa que eu fizesse na vida! — dispara Stallman.

Declarações contundentes como essa não deixam dúvidas: revelam que Stallman nunca foi um pacifista passivo e escancaram a carga emocional avassaladora acumulada naquele embate.

A raiz de tanta fúria brotava da dor profunda diante do que ele definia como o **aniquilamento deliberado de seu lar** — a destruição da cultura fraterna do AI Lab. Em uma célebre troca de e-mails com Steven Levy, Stallman comparou-se a **Ishi** — o último sobrevivente do povo indígena Yahi, exterminado pelos massacres de colonizadores na Califórnia no século XIX[5]. A analogia reveste a solidão de Stallman de um tom épico e trágico. Para os dissidentes que debandaram para a Symbolics, no entanto, a história era outra: em vez de enxergarem a empresa como um rolo compressor assassino, viam nela uma tentativa madura de salvar os princípios da computação hacker das quatro paredes acadêmicas e levá-los para o mercado corporativo. Longe de considerarem Stallman um herói de resistência, muitos o encaravam com desdém, como o fóssil teimoso de uma era extinta.

Rancores mesquinhos também deixaram cicatrizes profundas. Muito antes da cisão formal, Stallman relata que os colegas que fundariam a Symbolics já vinham isolando-o covardemente:

— Deixei de ser convidado para os jantares na Chinatown — recorda ele com mágoa. — O costume inaugurado por Greenblatt determinava que, ao sair para jantar, o sujeito passava de baia em baia ou disparava uma mensagem geral perguntando se mais alguém queria ir junto. Por volta de 1980, pararam de me chamar. E não foi só deixar de convidar: um colega confessou-me mais tarde que sofrera pressão do grupo para mentir descaradamente para mim, escondendo que eles estavam saindo juntos sem me avisar.

Embora o ostracismo pessoal machucasse, a declaração de guerra da Symbolics elevou o conflito do plano das mágoas íntimas para o terreno nobre da **justiça coletiva**. Ao se trancar em seu gabinete no nono andar e programar dia e noite para rivalizar com uma equipe inteira de dezenas de engenheiros de ponta, Stallman garantiu aos clientes da LMI e aos pesquisadores livres as mesmíssimas ferramentas que a gigante do mercado tentara monopolizar.

A façanha cimentou para sempre sua condição de lenda viva da computação. Competir de igual para igual contra o batalhão técnico da Symbolics — que reunia alguns dos cérebros mais brilhantes da história da informática — permanece como um dos feitos individuais mais assombrosos da Era da Informação. Classificando a façanha como um verdadeiro "master hack" e retratando Stallman como uma espécie de **John Henry cibernético** — o mítico operário norte-americano que venceu a britadeira a vapor com o suor dos próprios braços —, Steven Levy registra a admiração envergonhada que até seus rivais mais ferozes nutriam pelo ex-companheiro. Bill Gosper, que acabara migrando para o escritório da Symbolics em Palo Alto, resumiu a perplexidade geral diante daquele furacão:

> *"Eu posso olhar para um código que o Stallman escreveu e até achar que ficou ruim (o que é muito improvável, mas alguém poderia tentar me convencer de que ficou ruim); e mesmo assim eu diria: 'Espere aí um minuto: o Stallman não tem ninguém ao lado dele para varar a madrugada discutindo arquitetura de software! Ele está trabalhando absolutamente sozinho! É humanamente inacreditável que alguém seja capaz de construir tudo isso sozinho!'"*[6]

Para Stallman, as lembranças daqueles meses de guerra solitária misturam orgulho indomável e uma tristeza avassaladora. Em muitos sentidos, a guerra contra a Symbolics funcionou como o rito de passagem definitivo para o qual sua alma caminhava desde que pisara no MIT. Ao mesmo tempo, coincidiu com a agonia fúnebre de seu mundo. Certa tarde, durante uma pausa entre sessões de código, Stallman cruzou o salão de máquinas do laboratório e deu de cara com a carcaça monumental e desligada do velho PDP-10. Chocado diante daquelas lâmpadas mortas no painel — luzes que outrora piscavam freneticamente em código binário anunciando a pulsação viva do sistema operacional —, Stallman sentiu o mesmo impacto visceral de quem se depara com o cadáver inerte de um ente querido:

— Comecei a chorar copiosamente ali mesmo, no meio da sala de máquinas — desabafa ele. — Ver aquela máquina gigantesca morta, sem mais ninguém no mundo disposto a consertá-la, fez cair a ficha definitiva de como a minha comunidade inteira havia sido varrida do mapa.

Stallman não teria tempo para o luto. A Máquina Lisp, a despeito de toda a glória técnica e das paixões que incendiou, não passava de uma escaramuça secundária nas grandes guerras da indústria da informática. A marcha implacável da miniaturização dos chips trazia ao mundo novos microprocessadores que logo engoliriam as funções das caras máquinas dedicadas, exatamente como uma metrópole moderna engole um vilarejo arcaico no deserto.

Sobre as costas dessa onda avassaladora de microprocessadores surfavam milhares de programas de software proprietário — cada um blindado por contratos leoninos de licenciamento e acordos de não divulgação que impediam os programadores de examinar, aprender ou compartilhar código-fonte. As licenças eram toscas e restritivas, mas já tinham força legal para convencer os juízes e afugentar os curiosos. O software, que outrora fora um mero acompanhamento gratuito que os fabricantes de hardware ofereciam para adoçar a venda de seus computadores caros, convertia-se rapidamente no prato principal do banquete financeiro. Sedentos por novos

jogos e aplicativos de escritório, os usuários aceitavam abdicar do direito elementar de conhecer a receita da comida que consumiam.

Em nenhum lugar esse novo cenário era mais visível do que na revolução dos computadores pessoais. Gigantes como a Apple Computer e a Commodore forjavam milionários da noite para o dia vendendo máquinas com sistemas operacionais fechados de fábrica. Alheios à cultura hacker e ao seu amor pelo código-fonte aberto, os novos consumidores não viam razão alguma para protestar contra a ausência dos arquivos de programação. O mercado premiava os empresários rápidos o bastante para lançar programas e espertos o bastante para redigir termos de licença asfixiantes para acorrentar seus usuários.

Um dos pioneiros mais notórios desse novo modelo de negócios chamava-se **Bill Gates** — um ex-estudante de Harvard dois anos mais jovem que Stallman. Embora Stallman não soubesse na época, sete anos antes de seu chamado aos magos do Unix, Gates — então um jovem empreendedor sócio de uma empresa sediada em Albuquerque chamada Micro-Soft — enviara uma carta aberta histórica à comunidade de desenvolvedores. Escrita como desabafo contra os hobbistas que copiavam livremente o interpretador BASIC da empresa, a famosa *"Carta Aberta aos Hobbistas"* de Gates esbravejava contra o ideal do desenvolvimento colaborativo e comunitário:

> *"Quem pode se dar ao luxo de realizar um trabalho profissional de graça?", provocava Gates. "Que hobbista é capaz de dedicar três anos de trabalho humano na programação, na caça a bugs, na documentação de um produto e depois distribuí-lo de graça por aí?"*[7]

Embora a carta de Gates mal tenha repercutido nos corredores do MIT na época, sintetizava a guinada ideológica do setor: por que diabos tratar o software como um bem comum gratuito se o mercado provava que ele podia render fortunas astronômicas? Na virada para a era Reagan, cobrar caro por programas fechados virou mais do que uma escolha comercial: virou uma profissão de fé capitalista. Enquanto a Casa Branca desmantelava as regulações estatais construídas desde o New Deal, muitos programadores passaram a rotular a Ética Hacker como uma heresia anticompetitiva — e, por extensão, antinorte-americana. Como executivos de Wall Street que encontram uma velha camiseta *tie-dye* esquecida no fundo do armário entre ternos caros de risca de giz, a nova geração da informática encarava a partilha de software como a lembrança vergonhosa de uma juventude ingênua que precisava ser sepultada.

Para um homem que passara os anos 1960 inteiros como um anacronismo vivo dos anos 1950, viver em descompasso com as modas de seu tempo não era novidade. Contudo, na condição de programador acostumado a criar as ferramentas mais extraordinárias do mundo, Stallman viu-se encurralado diante do que só conseguia classificar como **uma dura escolha moral**: - Engolir seus escrúpulos éticos, render-se ao software proprietário e passar o resto da existência assinando termos de confidencialidade para privar seus semelhantes do conhecimento; - Ou dedicar cada gota de sua energia vital à construção de uma alternativa livre e comunitária, permitindo que a humanidade escapasse daquele cativeiro digital.

Após sair vitorioso da guerra de trincheiras contra a Symbolics, Stallman sentiu-se finalmente pronto para encarar a segunda opção:

— Suponho que eu pudesse ter abandonado a computação de vez — reflete Stallman. — Eu não dominava muitas habilidades práticas para o mercado convencional, mas tenho certeza de que conseguiria um emprego de garçom.

Dificilmente em um bistrô chique, é verdade, mas poderia muito bem servir mesas em algum restaurante simples por aí.

Virar garçom — isto é, renunciar para sempre à programação — significaria abrir mão da única atividade que lhe trouxera felicidade genuína na vida adulta. Olhando para trás, Stallman reconhece facilmente longos períodos em Cambridge nos quais o ato de programar fora o seu único e exclusivo refúgio de alegria. Em vez de desertar, decidiu lutar:

Ateu irredutível, Stallman rejeita conceitos místicos como destino, carma ou chamado divino. Não obstante, tem a convicção íntima de que sua escolha foi o desfecho natural de sua biografia: fora a combinação ímpar de obstinação cega, visão filosófica e genialidade técnica que o levara àquela encruzilhada que a maioria dos mortais sequer enxergava. Em seu célebre ensaio *"O Projeto GNU"*, resgata as palavras imortais do sábio judeu Hillel para resumir seu dilema:

> *"Se eu não for por mim, quem será por mim?*
> *E se eu for apenas por mim, o que sou eu?*
> *E se não for agora, quando?"*[8]

Ao discursar para o público leigo, contudo, ele prefere enquadrar a decisão em termos estritamente pragmáticos:

— Perguntei a mim mesmo: o que eu, um desenvolvedor experiente de sistemas operacionais, poderia fazer para salvar a situação? Bastou refletir um pouco para me dar conta de uma verdade cristalina: um desenvolvedor de sistemas operacionais era exatamente a peça que faltava para resolver a equação!

Uma vez alcançada essa clareza interior, tudo o mais se encaixou com precisão cirúrgica. No final de 1983, o MIT preparava-se para adquirir a nova geração de Máquinas Lisp da Symbolics, nas quais o sistema operacional comunitário do MIT seria incapaz de rodar. Quando a substituição do maquinário se consumasse, Stallman não teria mais como manter o sistema público por falta de relatórios de bugs dos usuários. Ele teria que parar. Mas ele próprio *queria* parar: o sistema da Máquina Lisp nunca fora um software verdadeiramente livre, já que os usuários tinham acesso ao código mas não podiam redistribuí-lo livremente. Além disso, o objetivo tático fora cumprido com louvor: a LMI sobrevivera e agora caminhava com as próprias pernas.

Stallman não queria gastar o resto de seus dias punindo os algozes que haviam destruído seu lar: ele queria **erguer um novo lar do zero**. Decidiu denunciar publicamente o software que exigia trair a própria consciência e devotar a existência a criar programas que permitissem a ele e a todos os seres humanos viverem em liberdade. Prometendo construir um sistema operacional livre *"ou morrer tentando — de velhice, é evidente!"*, **Richard Stallman pediu demissão formal do MIT em janeiro de 1984** para fundar o Projeto GNU.

A demissão blindava juridicamente o novo projeto contra as garras corporativas do MIT: impedia que a universidade reivindicasse a titularidade dos direitos autorais sobre o GNU ou alegasse conflito de interesses. Ainda assim, Stallman preservava amigos suficientes no laboratório para continuar frequentando as instalações e, tempos depois, manter seu próprio gabinete como pesquisador convidado. Ele financiava os primeiros passos do projeto prestando consultorias pontuais de programação para fora. Aquele mesmo homem cujo medo juvenil da

solidão o empurrara para o colo protetor do AI Lab erguia agora uma barreira jurídica de independência entre ele e o mundo institucional.

Nos primeiros meses de jornada, Stallman trabalhou isolado da própria comunidade Unix. Embora seu anúncio na Usenet tivesse arrancado mensagens simpáticas, quase nenhum voluntário dispôs-se a pegar na enxada na largada:

— A reação geral da comunidade foi praticamente unânime — recorda Rich Morin, líder histórico de grupos de usuários Unix da época. — As pessoas batiam nas costas dele e diziam: "Nossa, que ideia sensacional! Mostre-nos o seu código; prove primeiro que isso é viável na prática!".

Ciente da magnitude homérica da montanha que tinha pela frente, Stallman decidiu reaproveitar ferramentas livres já existentes sempre que possível, convertendo-as para o padrão GNU. Um dos primeiros alvos foi um compilador batizado de **VUCK**, projetado para traduzir a linguagem C em código de máquina. Traduzido do holandês, o acrônimo significava *Free University Compiler Kit* ("Kit de Compilação da Universidade Livre"). Esperançoso, Stallman escreveu ao autor perguntando se o compilador era livre. Quando o professor respondeu que as palavras "Universidade Livre" referiam-se à instituição Vrije Universiteit de Amsterdã, e que o software era estritamente proprietário e pago, o choque foi imediato:

— Ele me respondeu com zombaria e deboche, dizendo que a universidade era livre, mas o compilador custava caro! — relembra Stallman. — Não apenas recusou qualquer ajuda, como ainda teve o desplante de sugerir que eu abandonasse meu plano do GNU e passasse a escrever módulos para ajudá-lo a vender o compilador dele, em troca de uma fatia dos lucros! Foi naquele instante que decidi que o meu primeiro grande programa para o Projeto GNU seria um compilador multilíngue e multiplataforma do zero! [9]

Descartado o VUCK, Stallman encontrou o compilador Pastel (*Pascal desbotado*), escrito por programadores do Laboratório Nacional Lawrence Livermore. Os autores garantiram-lhe que o programa podia ser livremente copiado e alterado. Infelizmente, a ferramenta mostrou-se imprópria para o trabalho: sua exigência de memória era mastodôntica, analisando o arquivo inteiro dentro da memória física antes de gerar o executável. Em computadores de grande porte, aquele desenho era aceitável; no universo do Unix, revelou-se uma barreira intransponível, já que as estações de trabalho não dispunham de tanta RAM. Stallman construiu uma ponta compatível com C e rodou testes no grande VAX, mas, ao tentar portar o compilador para o chip Motorola 68010, o sistema quebrou. Concluiu que teria de construir um compilador completamente novo do zero — o que de fato faria mais tarde, dando à luz o lendário **GCC** (*GNU C Compiler*). Contudo, em 1984, aquele nó parecia intrincado demais, de modo que decidiu deixar o compilador amadurecer na gaveta e focar suas energias em outra frente vital.

Em setembro de 1984, Stallman iniciou o desenvolvimento da versão GNU do **Emacs** — o substituto moderno para o editor que ele liderara por uma década. No mundo Unix, os dois editores nativos dominantes eram o **vi** (escrito por Bill Joy, cofundador da Sun Microsystems) e o **ed** (criado por Ken Thompson nos Bell Labs). Ambos eram utilitários eficientes e populares, mas nenhum deles chegava aos pés da extensibilidade infinita do Emacs.

Olhando para trás, Stallman admite que a escolha não atendeu a nenhum cálculo maquiavélico: "Eu precisava de um Emacs para trabalhar, e tinha a oportunidade perfeita para construir um".

Mais uma vez, tentou economizar tempo reaproveitando código alheio. Na época, James Gosling — estudante de doutorado da Carnegie Mellon e futuro criador da linguagem Java — havia escrito uma versão do editor em linguagem C batizada de **Gosmacs** (*Gosling Emacs*), equipada com um interpretador simplificado de Lisp chamado Mocklisp. Embora Gosling tivesse registrado os direitos autorais e vendido a exclusividade da ferramenta para a empresa UniPress, Stallman recebeu a garantia de outro desenvolvedor de que Gosling lhe concedera formalmente por e-mail a permissão de redistribuir sua própria versão modificada.

A princípio, Stallman planejava alterar apenas a camada de comandos para garantir compatibilidade com o Emacs original do PDP-10. Contudo, ao constatar como a Mocklisp era frágil e limitada perto de um Lisp de verdade, decidiu substituí-la por um motor Lisp completo. Essa cirurgia obrigou-o a reescrever praticamente todo o código de alto nível do Gosmacs, tirando proveito das estruturas dinâmicas do Lisp real. Em meados de 1985, quando o **GNU Emacs** foi formalmente lançado na internet, restavam apenas míseros fragmentos do código original de Gosling.

Foi então que a UniPress descobriu o projeto e negou veementemente que o outro desenvolvedor tivesse autorização para redistribuir o código. Como o rapaz não encontrou a cópia daquele antigo e-mail de Gosling para se defender nos tribunais, Stallman encerrou a controvérsia com um golpe limpo: reescreveu do zero os poucos módulos restantes, expurgando até a última linha de código herdada da UniPress.

Ainda assim, a visão de desenvolvedores leiloando direitos de código para empresas privadas — e o fato de a legislação conferir-lhes o poder de fechar o conhecimento alheio — revirava o estômago de Stallman. Em 1986, ele desabafou na Suécia:

— Às vezes penso que a coisa mais útil que eu poderia fazer da vida seria encontrar uma pilha gigantesca de softwares proprietários guardados a sete chaves como segredo industrial e começar a distribuir cópias na esquina para que deixassem de ser segredos! Talvez essa fosse uma maneira infinitamente mais eficaz de entregar softwares livres à sociedade do que passar os anos escrevendo tudo com as próprias mãos; a desgraça é que as pessoas são covardes demais até para aceitar os discos de graça![10]

Apesar do estresse jurídico, a disputa com a UniPress acelerou o amadurecimento do movimento do software livre. Forçou Stallman a encarar a fragilidade dos acordos informais de cavalheiros da velha Comuna do Emacs e a afiar seus objetivos políticos. Pouco após o lançamento do GNU Emacs, publicou **O Manifesto GNU** (*The GNU Manifesto*) — uma expansão monumental de seu anúncio original de 1983. No texto, rebateu ponto a ponto os chavões usados pelos acadêmicos e empresários para justificar o cerceamento do código. Diante da clássica pergunta: *"Os programadores não merecem uma recompensa por sua criatividade?"*, disparou a resposta que condensava sua cólera:

> *"Se existe algo que realmente merece recompensa nesta vida, é a contribuição social. A criatividade pode perfeitamente ser uma contribuição social, mas apenas e tão somente na medida em que a sociedade for livre para usufruir de seus frutos! Se os programadores merecem ser recompensados por criarem programas inovadores, pela mesmíssima lógica merecem ser punidos exemplarmente se restringirem o uso dessas ferramentas à humanidade!"*[11]

Com a circulação do GNU Emacs, o Projeto GNU finalmente tinha código palpável para mostrar ao mundo. Mas ganhou também as dores de cabeça de qualquer operação de grande porte: conforme programadores Unix do planeta inteiro começaram a testar o editor, doações em dinheiro, presentes e pedidos desesperados de fitas magnéticas começaram a inundar a sua caixa de correio. Para organizar o lado operacional do projeto, Stallman uniu-se a companheiros de confiança e fundou a **Free Software Foundation (FSF)** — uma entidade sem fins lucrativos dedicada a acelerar a conclusão do sistema operacional GNU. Com Stallman na presidência e uma diretoria de amigos e hackers veteranos, a FSF conferiu peso institucional e respeitabilidade jurídica à causa.

Robert Chassell, então programador da LMI, tornou-se um dos cinco diretores fundadores da fundação após um jantar com Stallman. Assumiu também o cargo de primeiro tesoureiro da entidade:

— Acho que em 1985 a nossa receita e despesa total girava em torno de uns 23 mil dólares, mais ou menos — recorda Chassell. — O Richard tinha seu gabinete e nós pegávamos salas emprestadas por aí. Eu guardava os equipamentos e os rolos de fita magnética debaixo da minha própria escrivaninha! Só algum tempo depois a LMI nos cedeu um depósito onde pudemos estocar o material com dignidade.

Além de fornecer um porto seguro, a Free Software Foundation funcionou como um polo de atração para uma legião de programadores desiludidos com os rumos do mercado. O ecossistema Unix, que parecia tão acolhedor na época do primeiro anúncio de Stallman, transformara-se em um ringue impiedoso de monopólios corporativos. Para trancar os clientes em seus currais tecnológicos, as empresas passaram a vedar o acesso ao código-fonte do Unix — uma política asfixiante que só fez explodir o interesse em torno das ferramentas livres do GNU. Aqueles mesmos magos do Unix que outrora desdenhavam de Stallman como um lunático barulhento começavam a enxergá-lo como um profeta da era digital ou como uma Cassandre incompreendida, conforme alimentassem esperanças ou desespero diante dos grilhões da tecnologia.

— Muita gente não faz ideia, até que aconteça consigo, da dor indescritível que é passar anos trabalhando em um software para vê-lo ser confiscado e trancado a chave pela empresa — ensina Robert Chassell, sintetizando o teor das centenas de cartas que a FSF recebia nos primeiros anos. — Quando isso acontece uma ou duas vezes com você, você para e pensa: *"Espere aí um minuto..."*.

Para o próprio Chassell, a decisão de fundar a FSF brotara dessa dor pessoal. Antes de ir para a LMI, passara meses redigindo um livro didático introdutório sobre Unix para uma empresa de software chamada Cadmus. Quando a companhia faliu, arrastou consigo os direitos autorais da obra para o buraco negro da massa falida; e Chassell tentou em vão recomprar os direitos de seu próprio texto:

— Pelo que sei, aquele livro continua mofando em alguma prateleira empoeirada por aí: inutilizado, proibido de ser copiado, varrido do ecossistema — lamenta Chassell. — Era uma introdução primorosa, se me permitem a modéstia. Levaria três ou quatro meses de trabalho para adaptá-lo e transformá-lo em uma apostila perfeita para o GNU/Linux de hoje. Mas todo aquele esforço intelectual, fora o que guardo na memória, foi assassinado e jogado fora.

Obrigado a ver sua obra afundar na lama burocrática da falência alheia, Chassell experimentou na pele aquela fúria sagrada que levava Stallman às raias da apoplexia:

— A grande iluminação moral, para mim, foi a certeza cristalina de que, se você deseja ter uma existência decente e digna, não pode permitir que pedaços da sua vida sejam confiscados e trancados pelo egoísmo de terceiros —

conclui Chassell com comovente lucidez. — A liberdade de poder abrir o capô, consertar o que está quebrado e aprimorar as ferramentas transforma completamente a relação do homem com o trabalho. Faz você olhar para trás após alguns anos e sentir o doce alívio de que a sua passagem pela Terra valeu a pena! Porque, de outro modo, tudo o que você constrói é sumariamente confiscado, jogado no lixo, abandonado em um cofre escuro ou, na melhor das hipóteses, expropriado de você. É exatamente como perder um pedaço irremediável da própria alma.

Ver Richard Stallman, *"Initial GNU Announcement"* (setembro de 1983). ↩

Ver Marshall Kirk McKusick, *"Twenty Years of Berkeley Unix"*, em *Open Sources* (1999), p. 38. ↩

Ver Richard Stallman (1986). ↩

"Quando as senhas aportaram formalmente no AI Lab do MIT, decidi ser fiel à minha convicção de que não deveria haver senhas na computação", explicaria Stallman mais tarde. "Como considero nocivo e destrutivo impor barreiras de segurança a um computador, eu não poderia colaborar ativamente com a sustentação daquele regime opressor." ↩

Steven Levy imortalizaria esse episódio no clássico *Hackers*, chamando Stallman de "o último dos verdadeiros hackers" — rótulo que Stallman nunca usou para si, frisando que Levy apenas queria contrastar o MIT com outras vertentes posteriores da informática. ↩

Ver Steven Levy, *Hackers* (1984), p. 426. ↩

Ver Bill Gates, *"An Open Letter to Hobbyists"* (3 de fevereiro de 1976). ↩

Stallman acrescenta sua própria nota a esse provérbio: "Sendo ateu, não sigo nenhum líder religioso; mas às vezes admiro profundamente a sabedoria expressa por alguns deles". ↩

Ver Richard Stallman, *"The GNU Operating System and the Free Software Movement"*, em *Open Sources* (1999), p. 65. ↩

Ver Richard Stallman (1986). ↩

Ver Richard Stallman, *The GNU Manifesto* (1985): http://www.gnu.org/gnu/manifesto.html. ↩

Capítulo 8: São Ignúcio

O Maui High Performance Computing Center (MHPCC) fica instalado em um discreto edifício térreo sobre as colinas vermelhas e empoeiradas que emolduram a cidade litorânea de Kihei, no Havaí. Cercado por vistas panorâmicas de tirar o fôlego e pelo condomínio multimilionário do Silversword Golf Course, o centro parece, à primeira vista, o mais suntuoso cabide de luxo da comunidade científica norte-americana. A anos-luz dos corredores labirínticos e claustrofóbicos do Tech Square do MIT — ou dos complexos de pesquisas militares de Los Alamos e Argonne —, o MHPCC dá a nítida impressão de ser aquele tipo de lugar paradisíaco onde os cientistas dedicam mais horas ao bronzeamento artificial do que aos seus cálculos de pós-doutorado.

A imagem, contudo, só é verdadeira pela metade. Embora os pesquisadores dali saibam desfrutar das delícias tropicais da ilha, o trabalho técnico é levado a sério. Segundo o ranking mundial do *Top500.org*, o supercomputador IBM SP Power3 abrigado nas entranhas do MHPCC cravava, na época, nada menos que 837 bilhões de operações de ponto flutuante por segundo — credenciando-o como um dos vinte e cinco computadores mais velozes do planeta Terra. Gerida em parceria pela University of Hawaii e pela Força Aérea dos Estados Unidos, a máquina dividia seu colossal poder de fogo entre cálculos logísticos para as Forças Armadas e simulações complexas de física de altas temperaturas.

Em resumo: o MHPCC era um ecossistema singular, onde a cultura hiperintelectualizada da ciência de ponta e a atmosfera descalça e relaxada do Havaí conviviam em perfeito equilíbrio. O slogan estampado no portal do laboratório no ano 2000 resumia o espírito da casa: *"Computação no paraíso"*.

Não é exatamente o tipo de paisagem onde se esperaria encontrar Richard Stallman — um homem que, ao contemplar a deslumbrante vista do canal de Maui através da vidraça panorâmica de uma das salas do instituto, soltou uma avaliação lacônica e desdenhosa: *"Sol demais"*. Ainda assim, na qualidade de embaixador peregrino vindo de outro paraíso computacional, Stallman tinha uma mensagem urgente a entregar aos habitantes locais, mesmo que para isso tivesse de submeter seus olhos sensíveis de hacker à claridade escaldante dos trópicos.

A sala de conferências já estava completamente tomada quando cheguei para acompanhar a palestra. A proporção de gênero mostrava-se ligeiramente mais equilibrada do que na conferência de Nova York — cerca de 85% de homens para 15% de mulheres —, mas ainda assim desproporcional. Metade da plateia vestia bermudas cáqui e camisas polo esportivas bordadas com logotipos de empresas; a outra metade parecia ter adotado os costumes locais de vez, exibindo vistosas camisas floridas estampadas e peles tisnadas de bronzeado caribenho. O único vestígio de seu status nerd residia na parafernália tecnológica em punho: celulares Nokia, computadores de mão Palm Pilot e notebooks reluzentes da linha Sony VAIO.

Escusado dizer que Stallman — plantado no tablado diante de todos, vestindo uma camiseta azul-marinho básica, suas inseparáveis calças marrons de poliéster e meias brancas sem sapatos — destacava-se no ambiente como um marciano caído de paraquedas. A iluminação branca e fria das lâmpadas fluorescentes realçava a palidez fantasmagórica de sua pele privada de sol[1]. Sua vasta cabeleira e a barba espessa eram suficientes para arrancar gotas de suor do mais calmo dos havaianos. A não ser que mandasse tatuar a palavra "forasteiro" na testa, Stallman não conseguiria parecer mais alienígena nem se fizesse força.

Enquanto Stallman arrumava seus papéis no púlpito, alguns rapazes trajando camisetas do grupo de usuários de FreeBSD de Maui (MFUG) corriam contra o tempo montando tripés de câmeras de vídeo e gravadores de áudio. O FreeBSD — variante livre e direta do lendário Unix de Berkeley dos anos 1970 — era, rigorosamente falando, um concorrente tecnológico do sistema GNU/Linux. Ainda assim, no universo hacker, as conferências de Stallman eram registradas com a reverência devota que os fãs do Grateful Dead dedicavam às lendárias fitas cassete dos shows da banda. Cabia àqueles entusiastas locais do software livre garantir que programadores em Hamburgo, Mumbai ou Novosibirsk não perdessem uma única gota da sabedoria destilada por RMS.

A analogia com o **Grateful Dead** não é mero floreio de estilo. Ao explicar a viabilidade econômica do modelo de software livre para plateias de executivos, o próprio Stallman costumava citar o grupo de rock como exemplo acabado de sucesso. Ao autorizar e encorajar abertamente os fãs a gravarem fitas de seus concertos ao vivo e trocarem cópias livremente entre si, o Grateful Dead transcendeu a condição de simples conjunto musical para virar o epicentro de uma gigantesca comunidade tribal. Com o tempo, essa tribo tornou-se tão numerosa, fiel e apaixonada que os músicos mandaram as gravadoras comerciais às favas, sustentando-se exclusivamente das turnês e dos espetáculos de estádio. Em 1994, último ano em que excursionou pelas estradas norte-americanas, o Grateful Dead faturou nada menos que 52 milhões de dólares unicamente com a venda de ingressos de bilheteria[2].

Embora pouquíssimas empresas de tecnologia tenham alcançado patamar financeiro equivalente, a dimensão comunitária e tribal do ecossistema do software livre foi a alavanca que convenceu a indústria, a partir de meados dos anos 1990, de que abrir o código-fonte poderia ser um negócio lucrativo. Cobiçando construir suas próprias legiões de clientes devotos, gigantes como IBM, Sun e Hewlett-Packard passaram a acolher a letra — se não o espírito ético — da pregação de Stallman. Comparando a GNU GPL à célebre *Magna Carta* britânica, o respeitado analista Evan Leibovitch enxergava aquela adesão como uma guinada histórica irreversível: *"Essa revolução está devolvendo aos cidadãos o controle sobre o próprio futuro"*, escreveu Leibovitch. *"Assim como a Magna Carta de 1215 arrancou garantias fundamentais da monarquia em favor dos súditos, a GPL impõe os direitos civis e as liberdades dos usuários de computador diante dos abusos da indústria"*[3].

Foi essa mesma mística que explicava a presença daquelas quatro dezenas de programadores que, em vez de estarem surfando nas ondas de Maui ou checando previsões de vento na praia, espremiam-se na sala para ouvir a pregação de Richard Stallman.

Ao contrário da pomposa solenidade de Nova York, ali Stallman não recebeu nenhuma apresentação formal; tampouco se deu ao trabalho de se apresentar. No segundo exato em que a turma do FreeBSD terminou de ajustar o foco da câmera, ele deu dois passos à frente, pigarreou e passou o trator sobre qualquer resquício de conversa na sala:

— Na maior parte das vezes em que as pessoas debatem sobre quais regras a sociedade deveria adotar para o uso de softwares, as vozes que lideram a discussão pertencem a executivos de corporações privadas, e eles encaram a questão sob uma perspectiva estritamente egoísta — disparou Stallman em sua abertura. — A única pergunta que eles se fazem é: *"Quais amarras legais podemos impor a todo o restante da humanidade para obrigar todo mundo a nos pagar fortunas de dinheiro?"*. Eu tive a imensa felicidade de fazer parte, na década de 1970, de uma comunidade de programadores que compartilhava livremente tudo o que produzia. E, por conta dessa vivência transformadora, aprendi a encarar o dilema por um prisma completamente oposto: **quais regras tornam possível a existência de uma sociedade boa, justa e decente para os seres humanos que vivem nela?** E é por isso que chego a respostas totalmente divergentes das deles!

Sem rodeios, Stallman engatou a clássica parábola da impressora da Xerox, repetindo a encenação do dedo em riste contra a plateia. Em seguida, dedicou alguns minutos ao imbróglio terminológico do nome GNU/Linux:

— Algumas pessoas me dizem: *"Stallman, por que você gasta tanta energia brigando por crédito e reconhecimento para o seu projeto? No fim das contas, o que importa é que o trabalho está pronto, e não quem recebe os louros!"*. Ora, esse conselho soaria muito sábio se fosse verdadeiro. Mas a nossa missão histórica nunca foi apenas construir um sistema operacional: **a nossa missão é difundir e garantir a liberdade a todos os usuários de computadores!** E, para alcançar essa meta, precisamos conscientizar as pessoas de que é possível usar a tecnologia com plena soberania moral e cidadã!

Ele fez uma pausa dramática e arrematou:

— E ainda temos uma montanha colossal de trabalho pela frente[4].

Para parte dos presentes, aquele discurso era chover no molhado; para outros, soava como metafísica pura. Quando um sujeito de camisa polo na terceira fila começou a cabecear de sono, Stallman interrompeu o raciocínio na hora e pediu aos vizinhos que acordassem o homem:

— Certa vez me disseram que o tom da minha voz era tão suave e tranquilizador que me perguntaram se eu não era algum tipo de curandeiro milagroso — brincou Stallman, arrancando gargalhadas do auditório. — Imagino que isso signifique que minhas palavras funcionam como um sedativo perfeito para induzir vocês a um sono relaxante e abençoado! E alguns de vocês talvez estejam precisando dormir mesmo; não serei eu a colocar obstáculos: se o senhor precisa descansar o corpo, por favor, sinta-se em casa!

O sermão encerrou-se com uma reflexão contundente sobre as **patentes de software** — uma das maiores ameaças que pairavam sobre o futuro da computação. Assim como o caso do Napster, as patentes ilustram o desastre de tentar transplantar categorias jurídicas forjadas para a matéria física para o universo imaterial dos algoritmos.

Direito autoral (*copyright*) e patentes funcionam por lógicas completamente diferentes e produzem desfechos radicalmente distintos no universo do software: - O **direito autoral** incide exclusivamente sobre a *expressão literal do código*: protege o texto do programa escrito pelo autor. Contudo, o direito autoral não cobre ideias ou conceitos abstratos. Isso significa que qualquer desenvolvedor é juridicamente livre para recriar, com seu próprio punho e código original, as mesmíssimas funções que viu em outro programa. É igualmente legítimo fazer engenharia reversa para entender a mecânica de um executável binário e reimplementá-lo do zero. - As **patentes de software**, por sua vez, operam como um monopólio avassalador: incidem diretamente sobre a *ideia matemática*, sobre o algoritmo em si. Uma patente confere ao seu titular o poder legal de **proibir qualquer ser humano de escrever códigos independentes** que utilizem aquele princípio computacional, sob pena de processo judicial!

Em uma indústria na qual vinte anos representam a vida útil de gerações inteiras de tecnologias, uma patente de duas décadas equivale a uma sentença de morte para a concorrência. Se outrora Apple e Microsoft degladiavam-se nos tribunais pela estética gráfica de janelas e ícones (*look and feel*), na virada do milênio as corporações passaram a patentear modelos de negócios e cliques banais na internet — como a famigerada tentativa da Amazon de patentear a "compra com um clique" online. Na prática, grandes corporações usavam carteiras gigantescas de patentes como um arsenal nuclear preventivo, forçando acordos mútuos de licenciamento cruzado em uma tensa

paz armada. Contudo, contra desenvolvedores independentes e projetos livres, as patentes funcionavam como bombas devastadoras: bibliotecas inteiras de renderização de fontes e formatos de compressão gráfica precisavam ser mutiladas no software livre devido a ameaças judiciais da Apple e de outros gigantes.

Para Stallman, as patentes expunham a necessidade de vigília ética permanente e escancaravam por que a bandeira da liberdade não podia ser rebaixada a mero argumento comercial de eficiência técnica. A velocidade de processamento e o custo financeiro zero — dois campos nos quais o GNU/Linux e o FreeBSD davam um banho no Windows — eram miudezas secundárias diante do direito inalienável do ser humano de controlar o código que roda em seu computador.

Essa intransigência ética permanecia como o pomo da discórdia com a dissidência do **"Código Aberto"** (*Open Source*). Liderados por teóricos do pragmatismo empresarial como Eric Raymond, os defensores do código aberto optaram deliberadamente por despir o movimento de qualquer conotação moral ou política. Para vender o modelo às diretorias corporativas e aos fundos de investimento do Vale do Silício, Raymond e seus pares sustentavam que o desenvolvimento descentralizado e comunitário era simplesmente uma metodologia de engenharia superior: graças à revisão maciça por pares (*peer review*), gerava programas mais estáveis, seguros e confiáveis.

Para essa corrente, a palavra de ordem era enterrar o termo "software livre" (*free software*), que afastava empresários pelo duplo sentido em inglês (confundido com "gratuito"), e focar na eficiência técnica. Em seu célebre ensaio de 1999, *"Cale a Boca e Mostre o Código"* (*Shut Up and Show Them the Code*), Raymond verbalizou seu rompimento com a linha política de Stallman:

> *"A retórica de RMS é extremamente sedutora para quem é como nós: nós, hackers, somos pensadores e idealistas que vibram naturalmente diante de apelos a 'princípios', 'liberdade' e 'direitos civis'. Mesmo quando discordamos de trechos de seu programa, torcemos para que o estilo inflamado de RMS funcione; achamos que ele deveria funcionar; e ficamos desconcertados e atônitos quando constatamos que ele fracassa fragorosamente diante dos 95% da humanidade que não operam na mesma frequência mental que nós."*[5]

Sem conquistar a simpatia desses 95% — massa composta por executivos de finanças, gerentes de TI e usuários comuns —, Raymond sentenciava que a comunidade estaria condenada à irrelevância perpétua na periferia da sociedade:

> *"Quando RMS insiste em falar sobre 'direitos dos usuários de computador', está nos fazendo um convite perigosamente sedutor para repetir velhos fracassos do passado. É um convite que devemos recusar — não porque seus princípios estejam errados, mas porque esse tipo de linguagem moralizante, quando aplicada ao software, simplesmente não convence ninguém fora da nossa bolha. Na realidade, ela assusta, confunde e afasta a maioria esmagadora das pessoas normais."*

Stallman, contudo, demoliu as premissas de Raymond ponto por ponto:

> *"A tentativa de Raymond de justificar o nosso suposto fracasso é profundamente falaciosa, pela simples razão de que nós não fracassamos! A nossa meta é colossal e ainda temos uma longa estrada a percorrer, mas também já marchamos uma distância extraordinária!*
>
> *A visão pessimista de Raymond sobre a sensibilidade ética dos cidadãos comuns é de um exagero gritante. Milhões de pessoas que não são programadoras se preocupam com a justiça e com a soberania digital muito mais do que com as firulas técnicas que o 'código aberto' tanto bajula. E isso inclui líderes políticos de peso.*
>
> *Foram os princípios éticos e soberanos do software livre — e não a promessa rasteira de 'programas melhores' — que convenceram os governos de nações inteiras, como o Brasil e o Equador, a decretar a migração de seus órgãos públicos para o software livre! Os governantes desses países não são geeks de computador, mas compreendem perfeitamente o valor supremo da liberdade de seu povo!"*

Para além do debate de marketing, Stallman apontava a falha fatal da narrativa utilitária do código aberto: ela não fornece aos cidadãos a armadura ideológica indispensável para defender a liberdade quando a tempestade desaba. Se você convence o usuário a usar programas abertos apenas porque eles são "mais rápidos e quebram menos", o que acontecerá no dia em que uma funcionalidade avançada for bloqueada por uma patente agressiva e o software proprietário oferecer uma solução mais polida no mercado?

— Não é que nos falte talento técnico para programar softwares superiores — desabafou Stallman diante da plateia de Maui. — É que nós não temos o **direito legal** de fazê-lo! Alguém comprou uma lei para nos proibir de servir ao bem comum! O que acontecerá quando o usuário comum se deparar com essas lacunas técnicas no software livre? Se ele foi doutrinado pelo discurso do código aberto, ele dirá com desdém: *"Vocês não entregaram o que prometeram; esse software não é mais potente do que o concorrente, está faltando este recurso vital; vocês mentiram para mim!"*. Mas se esse mesmo cidadão compreendeu a essência ética do software livre — se ele aprendeu que a liberdade é um valor soberano por si só —, ele cerrará os dentes e dirá indignado: *"Como esses canalhas têm a audácia de tentar me chantagear, exigindo que eu abra mão da minha liberdade em troca de uma mera função no computador?!"*. E é unicamente com essa têmpera cívica que seremos capazes de sobreviver aos golpes que a indústria nos desferirá quando essas patentes começarem a explodir por aí!

Testemunhando Stallman falar ao vivo, é difícil enxergar qualquer traço repulsivo ou confuso em sua lógica. Se sua figura destoa do figurino corporativo, suas palavras transbordam coerência moral. Ao ser questionado se, ao boicotar softwares privativos, os militantes do software livre não ficavam para trás das últimas novidades da tecnologia, devolveu a resposta ancorada na sua regra de ouro:

— Considero a liberdade infinitamente mais valiosa do que um avanço técnico passageiro — decretou ele. — Eu sempre preferirei usar um programa livre ligeiramente menos avançado a ter de me curvar a um software proprietário repleto de recursos sofisticados, porque não estou disposto a vender a minha dignidade por um prato de lentilhas. O meu mandamento moral é simples: **se eu não posso partilhar o programa com você, eu me recuso a aceitá-lo para mim!**

Para quem confunde moral com religião, pronunciamentos desse calibre soam quase como fanatismo evangélico. Mas a diferença é abissal: ao contrário do judeu ortodoxo que guarda o sabá ou do mórmon que se abstém de álcool por submissão a um mandamento divino, Stallman não obedece a ordens celestiais: ele apenas se recusa a

ser cúmplice da opressão de seus semelhantes. O software proprietário sequestra a autonomia do usuário; logo, se você preza a sua soberania, a única saída racional é rejeitá-lo.

Como se quisesse exorcizar em público qualquer acusação de arrogância santarrã, Stallman arrematava seus sermões com um ritual teatral impagável. Puxando uma túnica negra surrada do fundo de uma sacola plástica de supermercado, vestiu-a sobre a camiseta azul diante do auditório atônito. Em seguida, pescou de dentro da sacola um disco rígido dourado reluzente e equilibrou-o sobre a cabeça como se fosse uma auréola celestial. A plateia soltou uma gargalhada espontânea.

— **Eu sou São IGNÚCIO, da Igreja do Emacs!** — trovejou Stallman, erguendo a mão direita no ar em sinal de bênção papal. — **Eu abençoo o seu computador, minha filha!**



As risadas converteram-se em aplausos estrondosos. Conforme o público batia palmas, o disco metálico sobre sua vasta cabeleira refletiu a luz das lâmpadas fluorescentes do teto, produzindo um efeito luminoso impecável: por uma fração de segundo, Richard Stallman parecia a encarnação perfeita de um ícone bizantino saído de uma catedral ortodoxa russa.

— O Emacs nasceu originalmente como um simples editor de textos — explicou o profeta em tom professoral. — Com o passar dos anos, converteu-se em um modo de vida para muitos e em uma religião para alguns. Nós batizamos essa fé de **Igreja do Emacs**!

A esquete era uma demonstração genial de autoparódia: uma resposta bem-humorada a todos aqueles que rotulavam sua militância de fanatismo religioso disfarçado. Ao vestir a batina e colocar a auréola na cabeça, Stallman parecia desarmar as resistências da sala, dizendo ao público: *"Tudo bem, podem rir de mim; eu sei que sou esquisito!"*[6].

Ao recordar o nascimento do personagem, Stallman revela que criou São Ignúcio em 1996, muito antes da invenção do termo "código aberto" e das disputas mesquinhas de liderança que se seguiram. Na época, sentia a necessidade de rir de si mesmo e mostrar aos ouvintes que, a despeito de sua teimosia lendária, não era o carrasco desprovido de humor que seus detratores pintavam. Mais tarde, desafetos aproveitariam o figurino para tentar tachá-lo de doido varrido — caso do próprio Eric Raymond, que afirmou certa vez que a esquete era uma encenação semiconsciente de um comunicador manipulador.

Stallman repele a tese conspiratória com desdém: "É apenas o meu jeito brincalhão de rir das minhas próprias manias; enxergar algo além disso reflete a neurose e as segundas intenções dos outros, não as minhas".

Dito isso, ele é o primeiro a reconhecer sua veia de ator e menestrel: "Está brincando? Eu adoro ser o centro das atenções!", confessa. Para aprimorar sua impostação cênica, chegou a frequentar as aulas do clube de oratória *Toastmasters*. Detentor de uma presença de palco que faria inveja a veteranos do teatro de revista, Stallman bebe diretamente na fonte do humor cáustico e anárquico dos Irmãos Marx. Dias após a apresentação em Maui, perguntei-lhe se seu comportamento não refletia o célebre complexo de Groucho Marx — a recusa categórica em fazer parte de qualquer clube que o aceitasse como sócio[7].

— Não — respondeu Stallman —, mas admiro profundamente Groucho Marx em uma infinidade de coisas, e muitas das minhas falas públicas inspiram-se diretamente em seu humor afiado. Embora, para ser sincero, eu também tenha me inspirado bastante no silêncio expressivo de Harpo Marx!

A influência dos Irmãos Marx salta aos olhos em seu amor compulsivo por trocadilhos. Mas a marca registrada de Stallman é a frieza absoluta com que ele solta suas tiradas cômicas: ele as dispara de cara limpa, sem piscar um olho ou esboçar um sorriso, deixando o interlocutor na dúvida se quem está rindo de quem.

Naquele final de tarde em Maui, contudo, a engenharia cômica funcionou com precisão milimétrica:

— Para ser alçado a santo na Igreja do Emacs não é preciso fazer voto de celibato — anunciou ele aos pesquisadores —, mas exige-se solenemente o compromisso inabalável de viver uma vida de pureza moral! Você deve obrigatoriamente exorcizar os malignos sistemas operacionais proprietários de todos os seus computadores e instalar em seu lugar um sistema operacional plenamente [santamente] livre! E, sobre ele, instalar unicamente

softwares que respeitem a sua liberdade. Se você abraçar esse voto e for fiel a ele, você também será sagrado santo da Igreja do Emacs — e você também terá o direito de ostentar uma bela auréola dourada na cabeça!

A pregação encerrou-se com uma provocação impagável contra o arquirrival do Emacs: o editor **vi**, desenvolvido por Bill Joy nos anos de ouro da Universidade de Berkeley. Antes de despir a auréola, São Ignúcio concedeu indulgência aos pecadores:

— Os fiéis frequentemente me procuram em aflição perguntando se utilizar o editor *vi* é considerado pecado mortal na Igreja do Emacs — zombou Stallman. — Eu respondo: usar uma versão livre do *vi* não chega a ser um pecado propriamente dito; **é apenas uma dura penitência**! Portanto, meus irmãos: que tenham todos um santo e abençoado hacking[8]!

Encerrada a sessão de perguntas e respostas, uma fila de admiradores formou-se para pedir autógrafos. Uma pesquisadora estendeu-lhe uma cópia impressa da GNU General Public License:

— Eu assino com muito prazer — avisou Stallman, com a caneta suspensa no ar —, mas apenas sob uma condição inegociável: a senhora precisa me jurar solenemente que passará a chamar este sistema operacional de **GNU/Linux** e que exigirá de todos os seus colegas que façam o mesmo!

O episódio apenas confirmava o que qualquer convivência mais próxima com ele escancara: ao contrário de políticos e estrelas de palco, Richard Stallman não possui o botão "desligar". Fora o personagem de São Ignúcio, o ativista que discursa no palanque é rigorosamente o mesmo homem que senta ao seu lado na mesa do restaurante. Naquela mesma noite, durante o jantar, bastou um programador comentar casualmente que adorava programas de "código aberto" para que Stallman, no meio de uma garfada, passasse-lhe uma reprimenda homérica: "Você quer dizer software livre; esse é o único jeito correto de se referir a ele!".

Em suas palestras, Stallman admite sem rodeios sua vocação professoral e pedagógica:

— Há muita gente bem-intencionada por aí que defende: *"Primeiro vamos atrair as pessoas para a comunidade técnica, e só depois nós as educamos sobre os princípios da liberdade!"*. Essa até poderia ser uma estratégia sensata se funcionasse; mas a realidade nua e crua é que quase todo mundo está empenhado em chamar as massas para dentro da festa, e praticamente ninguém se dá ao trabalho de ensinar-lhes o valor da liberdade depois que elas entram!

O resultado prático dessa omissão, adverte Stallman, assemelha-se à explosão desordenada das grandes metrópoles do Terceiro Mundo:

— Milhões de migrantes chegam todos os dias e erguem imensas favelas precárias nos morros, mas ninguém se preocupa em dar o Passo Dois: urbanizar a cidade e tirar aquelas famílias da miséria! Se você realmente acredita que a liberdade computacional é um imperativo ético, convido-o a se juntar a nós na execução do **Passo Dois**! Há gente de sobra trabalhando no Passo Um. O mundo clama por voluntários dispostos a arregaçar as mangas e executar o Passo Dois!

Executar o "Passo Dois" significa martelar que a liberdade humana — e não a mera conveniência do mercado — é a razão de ser de tudo. Aqueles que sonham em reformar a indústria do software proprietário por dentro estão se iludindo com uma fantasia pueril: "Tentar mudar as corporações por dentro é uma armadilha perigosa", alerta

Stallman. "A não ser que você esteja investido do poder de um Mikhail Gorbachev, você será docemente engolido e neutralizado pelo sistema".

Uma mão se ergueu no fundo do auditório: um engenheiro de camisa polo pediu a palavra:

— Sem patentes, como o senhor sugere que a sociedade combata a espionagem comercial e o roubo de programas?

— Bem, na realidade essas duas coisas não guardam a menor relação entre si — respondeu Stallman com calma.

— Mas e se uma empresa rival decidir *roubar* o software desenvolvido por outra? — insistiu o homem.

Stallman franziu o cenho, como se tivesse sido atingido por um jato de gás tóxico:

— Espere um segundo! — bradou ele. — *Roubar*?! Perdoe-me, meu amigo, mas há tanta carga de preconceito ideológico embutida nessa única palavra que a única resposta honesta que posso lhe dar é que rejeito categoricamente o seu preconceito! — Em seguida, debruçou-se sobre o cerne técnico da pergunta: — Empresas que desenvolvem softwares privativos guardam centenas de segredos industriais sob sete chaves, e isso não vai mudar por causa das patentes. No passado — inclusive nos anos 1980 —, os programadores sequer sabiam da existência de patentes de software e a vida seguia muito bem. O que acontecia era simples: as pessoas publicavam os conceitos científicos inovadores e guardavam em segredo os detalhezinhos obscuros de implementação. Hoje em dia, elas patenteiam a ideia abstrata para proibir o mundo de pensar e continuam guardando os detalhes em segredo da mesma forma! Portanto, para o cenário que o senhor descreveu, a existência de patentes não faz absolutamente a menor diferença prática!

— Mas a patente não garante a publicação da informação para a sociedade? — atalhou outro participante.

— Garante uma mentira! — rebateu Stallman de bate-pronto. — O que o registro da patente publica em letras garrafais é o aviso de que aquela ideia lógica está terminantemente **proibida** de ser usada pela humanidade pelos próximos vinte anos! E que diabos de benefício a sociedade colhe com uma desgraça dessas?! Sem falar que os textos das patentes são redigidos em um jargão jurídico propositalmente hermético e pedante, feito sob medida para obscurecer a essência do invento e esticar o alcance do monopólio o máximo possível, tornando a leitura daquele documento inútil para quem queira aprender algo de verdade! A única razão para alguém consultar uma patente de software é tomar conhecimento da má notícia daquilo que você foi proibido de fazer!

A sala mergulhou em silêncio reverente. O relógio da parede já se aproximava das cinco da tarde, e os participantes começavam a se mexer nas cadeiras, inquietos para dar a largada no fim de semana tropical. Sentindo a exaustão no ar, Stallman olhou ao redor e encerrou a sessão no ritmo ligeiro de um leiloeiro profissional:

— Bem, parece que liquidamos a fatura por hoje — disse ele. — Dou-lhe uma, dou-lhe duas, dou-lhe três... — e, vendo que ninguém mais levantava o braço, despediu-se com a sua bênção tradicional:

— **Happy hacking a todos!**

[RMS: A insinuação de que a pele possa ficar "privada de sol" ou de que a palidez seja "pouco saudável" é uma desinformação perigosa; manter-se longe dos raios solares não traz dano algum à saúde, desde que a pessoa tenha níveis adequados de vitamina D. O que realmente destrói a pele — e pode inclusive levar à morte por câncer — é a exposição excessiva à radiação ultravioleta. Além disso: por acaso há algo de errado em ter uma aparência diferente dos demais?**]** ↩

Ver *"Grateful Dead Time Capsule: 1985-1995 North American Tour Grosses"*. ↩

Ver Evan Leibovitch, *"Who's Afraid of Big Bad Wolves"*, *ZDNet Tech Update* (dezembro de 2000). ↩

O Projeto GNU define a liberdade de software a partir de **quatro liberdades essenciais**: - **Liberdade 0:** A liberdade de executar o programa como você desejar, para qualquer propósito. - **Liberdade 1:** A liberdade de estudar como o programa funciona e alterá-lo para que faça o que você desejar (o que exige acesso irrestrito ao código-fonte). - **Liberdade 2:** A liberdade de redistribuir cópias exatas do programa para ajudar seu próximo. - **Liberdade 3:** A liberdade de distribuir cópias de suas versões modificadas para que toda a comunidade possa se beneficiar de suas melhorias. Para o texto canônico, consulte: http://www.gnu.org/philosophy/free-sw.html. ↩

Ver Eric S. Raymond, *"Shut Up and Show Them the Code"* (junho de 1999). ↩

[RMS: Rir de alguém simplesmente porque a pessoa tem trejeitos incomuns é uma grosseria rasteira, e não é minha intenção justificar esse tipo de falta de educação. Contudo, espero sinceramente que as pessoas dêem boas risadas da minha esquete cômica de São Ignúcio!**]** ↩

[RMS: Williams interpreta erroneamente a famosa tirada de Groucho Marx ao tratá-la como uma neurose psicológica individual. Aquela piada foi formulada como uma bofetada irônica contra o antissemitismo escancarado dos clubes aristocráticos norte-americanos, que barravam judeus em seus quadros associativos. Eu próprio não compreendia isso até que minha mãe me abriu os olhos. Williams e eu crescemos em uma época em que o preconceito racial fora empurrado para a clandestinidade social, não havendo mais a necessidade de mascarar denúncias políticas sob piadas de salão como Groucho fazia.**]** ↩

A liturgia da Igreja do Emacs seguiu evoluindo ao longo dos anos. Os neófitos podem professar a fé recitando o credo sagrado: *"Não há outro sistema além do GNU, e o Linux é um de seus núcleos"*. Stallman também celebra a cerimônia mística do *Foobar Mitzvah*, condena o *Grande Cisma* entre as ramificações do editor e reverencia o culto à *Virgem do Emacs* (termo aplicado a qualquer ser humano que ainda não tenha aprendido a usar o editor). Além disso, a combinação "vi vi vi" (666 em numerais romanos) foi formalmente identificada como o Editor da Besta do Apocalipse! ↩

Capítulo 9: A Licença Pública Geral GNU

Na primavera de 1985, Richard Stallman havia chegado ao primeiro resultado prático e útil do Projeto GNU: uma versão do Emacs baseada em Lisp para sistemas operacionais compatíveis com Unix. Para torná-lo disponível aos outros como software livre, ele precisava conceber uma forma de distribuí-lo — na prática, um sucessor para as regras da antiga Comuna do Emacs.

A tensão entre a liberdade de modificar o código e o privilégio autoral já vinha se acumulando muito antes do Gosmacs. A Lei de Direitos Autorais norte-americana de 1976 (*Copyright Act of 1976*) havia reformulado radicalmente o regime legal, estendendo formalmente o manto dos direitos autorais aos programas de computador. De acordo com o Artigo 102(b) da lei, indivíduos e empresas podiam reivindicar direitos autorais sobre a "expressão" de um programa de software, mas não sobre os "processos ou métodos propriamente ditos incorporados ao programa".[1]

Traduzindo em termos práticos: a lei tratava um programa de forma muito parecida com um livro didático de álgebra. O autor podia reivindicar direitos autorais sobre o texto que escreveu, mas não sobre os conceitos matemáticos da álgebra nem sobre as técnicas pedagógicas empregadas para explicá-los. Assim, pouco importava o que Stallman dissesse a respeito do uso do código do Emacs original: outros programadores tinham todo o direito legal de escrever suas próprias implementações baseadas nas ideias e nos comandos do Emacs — e foi exatamente o que fizeram. O Gosmacs foi apenas uma entre cerca de trinta imitações do Emacs original desenvolvidas para os mais diversos sistemas de computação.

A Comuna do Emacs só se aplicava ao código do programa original escrito pelo próprio Stallman. Mesmo que tivesse valor coercitivo na justiça, ela jamais incidiria sobre imitações desenvolvidas de forma independente, como o Gosmacs. Tornar o Gosmacs um programa não livre foi eticamente condenável sob a perspectiva do movimento de software livre porque, como software proprietário, ele desrespeitava a liberdade de seus usuários; porém, esse problema ético não tinha relação alguma com a origem das ideias do Gosmacs.

Sob o império do direito autoral estrito (*copyright*), qualquer programador que quisesse copiar trechos de código de um programa existente (mesmo que fizesse alterações) precisava da permissão explícita do desenvolvedor original. A nova legislação impunha direitos autorais automáticos mesmo na ausência de qualquer aviso formal de copyright — fato que os hackers, de modo geral, ignoravam por completo —, e esses avisos legais logo começaram a brotar por toda parte.

Stallman enxergava esses avisos de copyright como as bandeiras hasteadas de um exército invasor e ocupante. Era raríssimo encontrar um programa que não tomasse emprestado código-fonte de sistemas anteriores; no entanto, com uma simples canetada do presidente, o governo dos Estados Unidos havia concedido a empresas e programadores o poder legal de proibir sumariamente esse reaproveitamento. Além disso, o copyright injetou uma dose pesada de burocracia e formalismo num ecossistema que sempre funcionara com base na informalidade. Em resumo: disputas que antes eram resolvidas de hacker para hacker agora seriam travadas de advogado para advogado. E, num sistema assim, as corporações — e não os hackers — levavam vantagem absoluta. Alguns até consideravam que colocar o próprio nome num aviso de copyright era uma forma de assumir responsabilidade

pela qualidade do código; todavia, o aviso quase sempre trazia o nome de uma empresa, e havia outros meios bem mais apropriados para que um indivíduo indicasse a autoria daquilo que havia escrito.

Contudo, nos anos que antecederam a fundação do Projeto GNU, Stallman também percebeu que o copyright permitia a um autor autorizar certas atividades protegidas pela lei, estabelecendo inclusive condições para isso. "Eu já tinha visto mensagens de e-mail acompanhadas de avisos de copyright com licenças simples do tipo 'permitida a cópia literal'", relembra ele. "Aquelas mensagens com certeza serviram de inspiração." Tais licenças traziam a exigência explícita de não remover o aviso da licença. A sacada de Stallman foi dar alguns passos além. Por exemplo: um aviso de permissão poderia autorizar os usuários a redistribuir até mesmo versões modificadas, sob a condição estrita de que essas versões trouxessem exatamente as mesmas permissões.

Assim, Stallman concluiu que recorrer à lei de direitos autorais não era algo intrinsecamente imoral. O mal dos direitos autorais no software residia na forma como eles eram tipicamente usados — e no propósito para o qual haviam sido desenhados: negar ao usuário suas liberdades essenciais. A imensa maioria dos autores não conseguia vislumbrar nenhum outro uso possível. Mas o copyright poderia ser virado do avesso: usado para tornar um programa livre e blindar sua liberdade de forma permanente.

No início de 1985, com a versão 16 do GNU Emacs, Stallman redigiu uma licença fundamentada nos próprios direitos autorais que garantia aos usuários o direito irrevogável de fazer e distribuir cópias. Ela também assegurava o direito de criar e distribuir versões modificadas, mas sob uma exigência inegociável: *apenas sob aquela mesma licença*. Ninguém poderia exercer sobre essas modificações o poder restritivo do copyright comum; logo, ninguém poderia transformar suas versões em software proprietário, como havia acontecido com o Gosmacs. E todos eram obrigados a disponibilizar o código-fonte correspondente. Essas condições fecharam com precisão cirúrgica a brecha legal que até então permitia o surgimento de versões restritas e não livres do GNU Emacs.

Apesar de muito útil para formalizar o contrato social da antiga Comuna do Emacs, a primeira licença do GNU Emacs ainda era "informal demais" para as ambições do projeto, diz Stallman. Pouco depois de fundar a Free Software Foundation, ele começou a trabalhar em uma redação juridicamente blindada, consultando os demais diretores e os advogados que haviam ajudado na criação da fundação.

Mark Fischer, advogado especialista em propriedade intelectual de Boston que prestou a primeira consultoria jurídica a Stallman, recorda-se das discussões sobre a licença naquela época: "Richard tinha ideias muito firmes sobre como a licença deveria funcionar", conta Fischer. "Ele se apoiava em dois princípios. O primeiro era tornar o software o mais aberto possível." (Na época em que fez essa declaração, Fischer já parecia influenciado pelo vocabulário dos defensores do código aberto; Stallman jamais buscou tornar o software "aberto", mas sim *livre*.) "O segundo princípio era encorajar os outros a adotarem exatamente as mesmas práticas de licenciamento." As cláusulas da licença foram meticulosamente desenhadas para atingir esse segundo objetivo.

A dimensão revolucionária dessa última exigência demoraria algum tempo para ser plenamente compreendida. Na época, conta Fischer, ele encarava a licença do GNU Emacs como uma simples permuta comercial: ela estabelecia um preço para o uso do GNU Emacs. Em vez de cobrar em dinheiro, Stallman cobrava dos usuários o acesso irrestrito às suas modificações futuras. Ainda assim, Fischer reconhece que os termos da licença eram absolutamente inéditos.

"Exigir que as pessoas aceitassem esse tipo de contrapartida era, se não algo único, algo extremamente incomum para a época", afirma.

Ao estruturar a licença do GNU Emacs, Stallman introduziu uma alteração profunda em relação aos mandamentos informais da antiga Comuna. Enquanto antigamente ele exigia que os membros da comunidade lhe enviassem diretamente todas as alterações que escrevessem, agora Stallman estipulava apenas que eles repassassem o código-fonte e a liberdade *sempre que decidissem redistribuir o programa*. Em outras palavras, quem apenas modificasse o Emacs para seu próprio uso privado não precisava enviar nada de volta para Stallman. Numa rara modulação de sua postura, Stallman barateou o "preço" do software livre. Os usuários ganharam total liberdade para inovar sem ter Stallman fiscalizando por cima de seus ombros, ficando livres para distribuir suas criações apenas quando e se desejassem — desde que todas as cópias distribuídas trouxessem a mesma permissão para que os novos donos pudessem desenvolver e redistribuir adiante.

Stallman explica que essa mudança nasceu de seu próprio incômodo com o viés centralizador do contrato social da Comuna original. Por mais que achasse conveniente receber todas as melhorias, ele concluiu que impor essa obrigatoriedade era uma injustiça:

"Era errado forçar as pessoas a publicarem todas as suas alterações", diz Stallman. "Era errado exigir que elas fossem enviadas a um único desenvolvedor privilegiado. Esse tipo de centralização e privilégio individual não combina com uma sociedade na qual todos gozam de direitos iguais."

A Licença Pública Geral do GNU Emacs (*GNU Emacs General Public License*) estreou em uma versão do GNU Emacs lançada em 1985. Após a publicação, Stallman convidou a comunidade hacker a sugerir melhorias no texto jurídico. Um dos primeiros a aceitar o convite foi o futuro ciberativista John Gilmore, que então prestava consultoria para a Sun Microsystems. Como parte de seu trabalho, Gilmore havia portado o Emacs para o SunOS, a versão interna de Unix da empresa. Ao fazer isso, publicou sua versão modificada sob a licença do GNU Emacs. Em vez de encarar a licença como um entrave, Gilmore a enxergava como uma expressão límpida e concisa da ética hacker. "Até então, quase todas as licenças eram excessivamente informais", lembra Gilmore.

Para ilustrar essa informalidade, Gilmore costuma citar a licença de meados dos anos 1980 do `trn`, um leitor de notícias de Usenet escrito por Larry Wall — hacker que mais tarde alcançaria fama mundial como o criador do utilitário `patch` e da linguagem de programação Perl. Na tentativa de encontrar um meio-termo entre a cortesia comunitária hacker e o direito do autor de controlar a publicação comercial, Wall incluiu no cabeçalho de seu programa uma nota de copyright que soava quase como um desabafo editorial:

> *Copyright (c) 1985, Larry Wall*
> *Você pode copiar o pacote trn no todo ou em parte, desde que não tente ganhar dinheiro com ele, nem finja que foi você quem o escreveu.*[2]

Declarações como essa, embora traduzissem com fidelidade o espírito hacker, também escancaravam a enorme dificuldade de converter a natureza solta e informal dessa ética para a linguagem jurídica engessada dos direitos autorais. Ao redigir a licença do GNU Emacs, Stallman fez muito mais do que vedar a rota de fuga que permitia apropriações proprietárias: ele traduziu a ética hacker em termos perfeitamente compreensíveis tanto para advogados quanto para programadores.

Não demorou, conta Gilmore, para que outros hackers começassem a discutir maneiras de "portar" a licença do GNU Emacs para seus próprios programas. Estimulado por discussões na Usenet, Gilmore enviou um e-mail a

Stallman em novembro de 1986 sugerindo uma generalização do texto:

> *Você provavelmente deveria tirar a palavra "EMACS" da licença e substituí-la por "SOFTWARE" ou algo parecido. Em breve, esperamos, o Emacs não será a maior parte do sistema GNU, e a licença precisa se aplicar a tudo.*[3]

Gilmore não foi o único a sugerir uma abordagem universal. No final de 1986, o próprio Stallman já trabalhava no próximo grande marco do Projeto GNU: o depurador de código-fonte GDB. Para lançá-lo, teve de adaptar a licença do GNU Emacs para que se aplicasse ao GDB. Não era uma tarefa monumental, mas abria margem para erros a cada novo utilitário. Em 1989, Stallman encontrou a solução ideal: removeu do texto todas as referências diretas ao Emacs e amarrou o programa à licença unicamente por meio de uma declaração nos cabeçalhos dos arquivos de código-fonte do próprio programa. Dessa forma, qualquer desenvolvedor poderia aplicar a licença à sua obra sem precisar alterar uma única vírgula do texto legal. Nascia assim a Licença Pública Geral GNU — a famosa **GNU GPL** (*GNU General Public License*). O Projeto GNU rapidamente a transformou na licença padrão de todos os programas oficiais do sistema.

Ao lançar a GPL, Stallman adotou a convenção comum do mundo do software de usar casas decimais para revisões pontuais e números inteiros para grandes atualizações. A edição inaugural, publicada em 1989, foi batizada de Versão 1.0. A licença vinha acompanhada de um preâmbulo que escancarava abertamente seus propósitos políticos:

> *A Licença Pública Geral foi concebida para garantir que você tenha a liberdade de distribuir ou vender cópias de software livre, que receba o código-fonte ou possa obtê-lo caso deseje, que possa modificar o software ou reaproveitar partes dele em novos programas livres; e para assegurar que você esteja ciente de que pode fazer tudo isso.*
>
> *Para proteger seus direitos, precisamos criar restrições que impeçam qualquer pessoa de negar essas liberdades a você ou de exigir que você abra mão delas. Essas restrições se traduzem em determinadas responsabilidades para você, caso decida distribuir cópias do software ou modificá-lo.*[4]

Na galeria dos grandes hacks da história, a GPL figura como uma das maiores obras-primas de Stallman. Ela ergueu um sistema de posse e usufruto comunitário dentro dos limites estritamente proprietários das leis de direito autoral. Mais importante ainda: demonstrou a profunda afinidade intelectual existente entre o código jurídico e o código de software. No preâmbulo da GPL residia uma mensagem revolucionária: em vez de encarar a legislação de direitos autorais com resignação ou mero desdém, os hackers deveriam enxergá-la como um sistema perigoso, sim, mas passível de ser *hackeado*.

"A GPL foi desenvolvida de maneira muito parecida com qualquer outro software livre, com uma vasta comunidade debatendo sua estrutura, apontando o que funcionava e o que gerava atrito, sugerindo ajustes e até concessões pontuais para obter maior aceitação", observa Jerry Cohen, outro advogado que assessorou Stallman após a saída de Fischer. "O processo funcionou extraordinariamente bem, e a GPL, ao longo de suas sucessivas versões, migrou de uma reação inicial cética e muitas vezes hostil para uma consagração universal."

Em uma entrevista concedida à revista *BYTE* em 1986, Stallman resumiu a essência da GPL em termos memoráveis. Além de reafirmar os valores hacker, afirmou ele, os leitores deveriam vê-la como "uma espécie de jiu-jítsu intelectual, usando o próprio sistema jurídico montado pelos acumuladores de software contra eles mesmos".[5]

Essa entrevista oferece um retrato fascinante e surpreendentemente sincero das posturas políticas de Stallman nas origens do Projeto GNU. Ela também serve para traçar a evolução de sua retórica.

Ao descrever o propósito da GPL à época, Stallman disparou: "Estou tentando mudar a maneira como as pessoas lidam com o conhecimento e a informação em geral. Acredito que tentar se apossar do conhecimento, tentar controlar se as pessoas têm permissão para usá-lo ou tentar impedir que outros o compartilhem não passa de sabotagem."

Compare essa declaração com o conselho incisivo dado ao autor deste livro em agosto de 2000: "Peço encarecidamente que você elimine o termo 'propriedade intelectual' do seu pensamento. Ele só serve para confundir as coisas, porque agrupa sob o mesmo rótulo direitos autorais, patentes e marcas registradas. E essas três áreas têm efeitos tão completamente distintos que é uma tolice monumental tentar discuti-las juntas. Se você ouvir alguém falando 'sobre propriedade intelectual' sem usar aspas de distanciamento, saiba que essa pessoa não está raciocinando com clareza, e você não deveria embarcar na conversa dela."

> **[RMS:** Esse contraste revela apenas que aprendi a ter mais cautela com generalizações. Hoje eu provavelmente não usaria a expressão "apossar-se do conhecimento", por se tratar de um conceito amplo demais. No entanto, "apossar-se do conhecimento" não é o mesmo tipo de generalização falaciosa de "propriedade intelectual", e a distinção entre aquelas três leis é crucial para compreender qualquer questão jurídica sobre controle do saber. As patentes são monopólios diretos sobre o uso de um conhecimento específico; isso sim é, literalmente, uma forma de "apossar-se do conhecimento". Os direitos autorais (*copyright*) são apenas um método empregado para barrar o compartilhamento de obras que expressam ou explicam determinado conhecimento, o que é algo inteiramente diferente. Já as marcas registradas têm pouquíssima relação com o tema do conhecimento.**]**

Anos mais tarde, Stallman explicaria a criação da GPL em um tom bem mais sóbrio e reflexivo: "Eu estava ponderando sobre dilemas que eram, a um só tempo, éticos, políticos e legais", diz ele. "Eu precisava conceber algo que se sustentasse de pé perante o sistema jurídico no qual vivemos. No fundo, a missão era equivalente à de legislar as bases de uma nova sociedade; mas, como eu não era o governo, não tinha como mudar as leis vigentes. Minha única saída era construir essa nova realidade sobre a infraestrutura legal existente — uma infraestrutura que jamais havia sido pensada para nada parecido."

Por volta da época em que Stallman quebrava a cabeça com essas encruzilhadas éticas e jurídicas, um hacker da Califórnia chamado Don Hopkins enviou-lhe pelo correio um manual do microprocessador Motorola 68000. Hopkins, profundo conhecedor de Unix e companheiro de gosto por ficção científica, havia pegado o manual emprestado de Stallman tempos antes. Como gesto de agradecimento, Hopkins encheu o envelope de volta com adesivos divertidos que havia garimpado em uma convenção local de ficção científica. Um adesivo específico chamou a atenção de Stallman. Dizia ele: *"Copyleft (L), All Rights Reversed"* (Copyleft, todos os direitos revertidos — um trocadilho com o tradicional *Copyright, All Rights Reserved*).

Inspirado por aquele trocadilho, Stallman resolveu apelidar o mecanismo jurídico concebido na licença do GNU Emacs (e consolidado na GNU GPL) de **"Copyleft"**, simbolizado de forma bem-humorada por uma letra "C" invertida dentro de um círculo. Com o passar dos anos, o termo viraria jargão oficial da Free Software Foundation para definir qualquer licença autoral que "torna um programa software livre e exige que todas as versões modificadas e estendidas desse programa continuem sendo, obrigatoriamente, software livre".

O sociólogo alemão Max Weber certa vez postulou que todas as grandes religiões se sustentam na "rotinização" ou "institucionalização" do carisma. Toda religião bem-sucedida, argumentava Weber, canaliza o carisma ou a mensagem inaugural de seu líder original para dentro de um aparato social, político e ético capaz de transcender diferentes culturas e atravessar gerações.

Embora não tenha natureza religiosa, a GNU GPL certamente se qualifica como um exemplo formidável dessa "rotinização" em pleno funcionamento no universo descentralizado do desenvolvimento moderno de software. Desde o momento de sua apresentação, programadores e empresas que não tinham qualquer simpatia ou lealdade pessoal a Stallman aceitaram de bom grado as regras da GPL pelo seu valor prático. Milhares de desenvolvedores adotaram a licença como um escudo preventivo para proteger seus próprios programas. E mesmo aqueles que rejeitavam suas condições por achá-las excessivamente rígidas reconheciam a magnitude de sua influência.

Um dos hackers desse segundo grupo era Keith Bostic, então funcionário da Universidade da Califórnia na época do lançamento da GPL 1.0. O departamento de Bostic, o Computer Systems Research Group (CSRG), liderava o desenvolvimento do Unix acadêmico desde o final dos anos 1970 e fora pioneiro em avanços cruciais, como a implementação da pilha de rede TCP/IP — a espinha dorsal de toda a internet moderna. No final dos anos 1980, a AT&T, dona original do Unix, passou a focar agressivamente na comercialização do sistema e viu na *Berkeley Software Distribution* (BSD) — a distribuição mantida por Bostic e seus colegas em Berkeley — um celeiro inestimável de inovação técnica.

O problema era que o código produzido pela turma de Berkeley não podia ser compartilhado livremente com quase ninguém, pois estava intrinsecamente misturado ao código proprietário da AT&T. Dessa forma, as distribuições de Berkeley só podiam ser entregues a instituições que já tivessem adquirido uma licença caríssima do código-fonte da AT&T. À medida que a AT&T aumentava suas taxas de licenciamento, esse arranjo — que a princípio parecera inofensivo para quem só pensava no meio acadêmico — tornou-se um fardo insustentável até dentro das universidades. Para aproveitar o código de Berkeley no GNU, Stallman precisaria convencer a universidade a separar seu trabalho do código da AT&T e lançá-lo como software livre. Entre 1984 e 1985, Stallman reuniu-se com as lideranças do projeto BSD e argumentou com franqueza: a AT&T não era uma instituição de caridade, e uma universidade pública doar o resultado de suas pesquisas de graça para alimentar o monopólio da AT&T era um absurdo. Ele implorou para que separassem o código próprio e o liberassem ao mundo como software livre.

Contratado em 1986, Bostic assumiu o desafio pessoal de portar a versão mais recente do BSD para o computador PDP-11. Foi nesse período, relata Bostic, que ele travou contato direto com Stallman em suas visitas regulares à Costa Oeste. "Lembro com muita clareza de debater sobre direitos autorais com Stallman enquanto ele usava estações de trabalho emprestadas no CSRG", diz Bostic. "Depois a gente saía para jantar e continuava a discussão acalorada à mesa."

A insistência de Stallman acabou surtindo efeito, ainda que não da forma que ele gostaria. Em junho de 1989, Berkeley finalmente expurgou todo o código de rede do restante do sistema operacional da AT&T e lançou o pacote sob uma licença livre baseada em copyright. Era uma licença extremamente liberal e permissiva: tudo o que exigia

de quem recebesse o código era dar o devido crédito à universidade em anúncios publicitários de programas derivados.[6] Diferentemente da GPL, essa licença permitia expressamente que terceiros criassem derivações comerciais fechadas. Além disso, havia um obstáculo técnico imediato: o lançamento de rede do BSD (*BSD Networking Release*) não era um sistema operacional completo, mas apenas a camada de conectividade. Embora representasse uma contribuição colossal para qualquer sistema operacional livre, na época ele só conseguia funcionar se acoplado ao restante do código proprietário do Unix da AT&T.

Nos anos seguintes, Bostic e outros funcionários da Universidade da Califórnia trabalharam sem descanso para reescrever todos os componentes proprietários faltantes e transformar o BSD em um sistema operacional completo e livremente redistribuível. Apesar de atrasado por uma pesada batalha jurídica movida pela Unix System Laboratories — subsidiária da AT&T que herdara os direitos do Unix —, o esforço finalmente deu frutos no início da década de 1990. Mas, muito antes disso, vários utilitários de rede de Berkeley já haviam sido incorporados ao sistema GNU de Stallman.

"Acho muito improvável que tivéssemos ido tão longe e com tanta força sem a influência do GNU", reconhece Bostic, ao olhar para trás. "Era evidente que eles estavam pressionando bastante, e nós gostávamos da ideia."

Ao final dos anos 1980, a GPL começou a exercer uma atração quase gravitacional sobre a comunidade de software livre. Um programa não precisava obrigatoriamente estar sob a GPL para ser considerado software livre — como bem provavam os utilitários de rede do BSD —, mas carimbar a GPL em um programa transmitia uma mensagem inequívoca. "A própria existência da GPL inspirou as pessoas a refletirem se estavam realmente criando software livre e como deveriam licenciá-lo", diz Bruce Perens, criador do Electric Fence (um utilitário popular de Unix) e futuro líder do time de desenvolvimento do Debian GNU/Linux. Poucos anos após o lançamento da GPL, Perens decidiu aposentar a licença caseira do Electric Fence e adotar a fórmula jurídica de Stallman: "Foi uma transição incrivelmente fácil", lembra.

Rich Morin, o mesmo programador que havia encarado com ceticismo o anúncio inaugural do GNU em 1983, confessou ter ficado impressionado com o calibre dos softwares que começavam a se abrigar sob o guarda-chuva da GPL. Como líder de um grupo de usuários de SunOS, uma das principais funções de Morin nos anos 1980 era gravar e enviar fitas magnéticas contendo os melhores utilitários freeware ou de software livre. A tarefa frequentemente o obrigava a ligar diretamente para os autores originais para checar se seus códigos tinham copyright ou se haviam sido doados ao domínio público. Por volta de 1989, Morin percebeu que quase todos os melhores programas do ecossistema já vinham carimbados com a GPL. "Como distribuidor de software, no instante em que eu batia o olho na palavra GPL, sabia que estava em terreno seguro", relembra.

Para compensar o trabalho braçal de compilar e gravar as fitas para o grupo de usuários da Sun, Morin cobrava uma taxa de conveniência. Com os programas migrando em massa para a GPL, ele de repente conseguia montar suas fitas na metade do tempo, obtendo um bom lucro no processo. Enxergando ali uma oportunidade de negócio, Morin transformou o hobby em empresa: a *Prime Time Freeware*.

Esse tipo de exploração comercial era perfeitamente legítimo e compatível com os princípios do software livre. "Quando falamos de software livre, estamos nos referindo à liberdade, não ao preço", alertava Stallman logo no preâmbulo da GPL. No fim dos anos 1980, ele sintetizou essa ideia em uma máxima célebre: *"Don't think free as in free beer; think free as in free speech"* — em português: "Não pense em 'livre' como quem pensa em 'cerveja grátis'; pense em 'livre' como em 'liberdade de expressão'".

A maioria das empresas tradicionais ignorava solenemente as pregações de Stallman. Contudo, para um punhado de empreendedores visionários, a liberdade inerente ao software livre coincidia perfeitamente com a liberdade dos mercados abertos. Ao remover a propriedade exclusiva do software da equação comercial, abria-se um cenário em que até a menor das empresas tinha condições reais de concorrer em pé de igualdade contra gigantes do calibre de IBM e DEC.

Um dos primeiros empresários a captar esse potencial foi Michael Tiemann, programador e estudante de pós-graduação na Universidade de Stanford. Durante os anos 1980, Tiemann acompanhou os passos do Projeto GNU com o mesmo fascínio com que um jovem músico de jazz segue seu ídolo. No entanto, foi apenas com o lançamento do GNU C Compiler (GCC) em 1987 que ele compreendeu a verdadeira força do software livre. Classificando o GCC como uma "bomba atômica", Tiemann afirma que a simples existência daquele compilador atestava o vigor sobre-humano de Stallman na programação:

"Assim como todo escritor sonha em escrever o grande romance americano, todo programador dos anos 1980 sonhava em escrever o grande compilador americano", reflete Tiemann. "E de repente Stallman foi lá e fez. Foi uma lição de humildade para todos nós."

"Se você quer falar de pontos únicos de falha na computação, o GCC era o maior deles", faz coro Keith Bostic. "Ninguém tinha um compilador decente naquela época, até o GCC aparecer."

Em vez de competir com Stallman, Tiemann resolveu construir seu futuro sobre a fundação que ele havia criado. A versão inicial do GCC ostentava impressionantes 110 mil linhas de código, mas Tiemann se lembra de ter ficado espantado com a clareza de sua arquitetura. Era tão bem desenhado que ele levou menos de cinco dias para dominá-lo e apenas mais uma semana para portar o compilador para uma nova arquitetura de hardware: o microchip 32032 da National Semiconductor. Ao longo do ano seguinte, Tiemann começou a dissecar o código-fonte, desenvolvendo o primeiro compilador "nativo" (direto) para a linguagem C++, expandindo o GCC para compilar C++ tão perfeitamente quanto compilava C. (A implementação proprietária existente na época funcionava apenas como um conversor tosco, que traduzia código C++ em C puro e depois empurrava o resultado para um compilador C.) Certo dia, ao ministrar uma palestra sobre sua criação nos míticos laboratórios da Bell Labs, Tiemann cruzou com engenheiros da AT&T que vinham sofrendo horrores para conseguir a mesma proeza.

"Havia umas quarenta ou cinquenta pessoas na sala, e eu perguntei quantos deles estavam trabalhando no compilador de código nativo", recorda Tiemann. "O anfitrião me disse que a informação era confidencial, mas sussurrou que, se eu olhasse ao redor da sala, teria uma boa ideia."

Pouco tempo depois, conta Tiemann, acendeu-se uma lâmpada em sua cabeça: "Eu vinha trabalhando naquele projeto havia apenas seis meses. Pensei comigo mesmo: quer o mérito seja meu ou da arquitetura do código, este é um patamar de produtividade e eficiência que o livre mercado certamente estará disposto a remunerar."

Tiemann encontrou inspiração adicional no *Manifesto GNU*: ao mesmo tempo em que fustigava sem dó a ganância dos fornecedores de software proprietário, o texto encorajava expressamente as empresas a usarem e redistribuírem software livre em suas atividades comerciais — contanto que respeitassem escrupulosamente a liberdade de seus clientes. Ao expurgar o monopólio da venda de licenças do modelo de negócios, a GPL permitia que mesmo uma pequena empresa competisse com base em serviços de alta qualidade — desde suporte técnico avançado e treinamento especializado até o desenvolvimento de extensões sob medida para clientes específicos.

Em um ensaio de 1999, Tiemann relembrou o choque provocado pela leitura do texto de Stallman: "Aquilo soava como um panfleto socialista, mas eu enxerguei algo completamente diferente: era um plano de negócios disfarçado."[7]

A rigor, esse plano de negócios não era inteiramente inédito; o próprio Stallman havia garantido seu sustento no final dos anos 1980 prestando consultorias desse tipo em pequena escala. Mas Tiemann pretendia elevar a ideia a uma escala industrial. Ao lado de John Gilmore e David Vinayak Wallace, ele fundou uma empresa de consultoria dedicada exclusivamente a customizar e prestar suporte a programas GNU. Batizada de **Cygnus Support** (informalmente, "Cygnus" era um acrônimo recursivo para *"Cygnus, Your GNU Support"* — Cygnus, Seu Suporte GNU), a companhia assinou seu primeiro contrato de desenvolvimento em fevereiro de 1990. Ao final do mesmo ano, já acumulava mais de 725 mil dólares em contratos de suporte e desenvolvimento.

O sistema operacional GNU completo idealizado por Stallman exigia muito mais do que ferramentas de desenvolvimento. Ao longo dos anos 1990, o Projeto GNU concebeu também um interpretador de linha de comando (ou *shell*), substituto muito aprimorado para o antigo Bourne Shell — escrito pelo funcionário da FSF Brian Fox e batizado com o trocadilho característico de Stallman como **BASH** (*Bourne Again Shell*); o interpretador de PostScript Ghostscript; a plataforma de documentação Texinfo; a biblioteca padrão C (*glibc*), absolutamente indispensável para que qualquer programa escrito em C possa rodar e se comunicar com o núcleo do sistema; a planilha eletrônica Oleo ("melhor para você do que a planilha mais cara do mercado"); e até mesmo um jogo de xadrez bastante sofisticado. Entretanto, eram as ferramentas de programação do GNU que despertavam a maior cobiça da comunidade técnica.

O trio de ouro — GNU Emacs, GDB e GCC — formava o coração desse arsenal, mas esteve longe de ser o único legado do Projeto GNU nos anos 1980. Até 1990, o projeto já havia desenvolvido suas próprias versões do utilitário de compilação `make`, do gerador de analisadores sintáticos `yacc` (rebatizado espirituosamente como `bison`) e do `awk` (rebatizado de `gawk`), além de dezenas de outros programas. Assim como o GCC, os utilitários GNU eram concebidos desde o primeiro dia para rodar em múltiplas plataformas, nunca amarrados ao hardware de um único fabricante. Ao buscar essa portabilidade universal, Stallman e seus colaboradores invariavelmente criavam ferramentas muito superiores e mais úteis do que as originais proprietárias.

Recordando essa postura universalista do GNU, Rich Morin, da Prime Time Freeware, costuma citar um pacote aparentemente inútil, mas fundamental: o **GNU Hello**. Ele servia como molde pedagógico para ensinar os programadores a empacotar seus softwares segundo os padrões oficiais do GNU. "É apenas o clássico programa 'hello world' em cinco linhas de C, empacotado exatamente como se fosse uma distribuição oficial do GNU", explica Morin. "Ele traz toda a estrutura do Texinfo e os scripts de `configure`. Contém toda aquela 'massa cinzenta' de engenharia de software que o Projeto GNU desenvolveu para permitir que um programa seja compilado e portado com facilidade para as mais diversas plataformas. Isso é de uma relevância gigantesca, e beneficiou não apenas os programas de Stallman, mas todo e qualquer software sob o Projeto GNU."

Para Stallman, superar tecnicamente os utilitários do Unix era um objetivo secundário; o primordial era substituí-los por alternativas livres. "A cada peça nova, posso ou não encontrar uma forma de aperfeiçoá-la", declarou Stallman à revista *BYTE*. "Até certo ponto, estou colhendo os frutos de reescrever um sistema do zero, o que naturalmente torna muitos programas melhores. Por outro lado, isso acontece porque atuo nessa área há muito tempo e já trabalhei com inúmeros outros sistemas. Carrego, portanto, uma vasta bagagem de ideias para aplicar."[8]

Ainda assim, à medida que as ferramentas GNU fincavam bandeira no final dos anos 1980, o perfeccionismo cirúrgico de Stallman — lapidado nas madrugadas do Laboratório de IA do MIT — tornou-se lendário em toda a comunidade de desenvolvimento de software.

Jeremy Allison, usuário de sistemas Sun nos anos 1980 e programador que mais tarde fundaria o célebre projeto de software livre Samba, lembra dessa faceta com uma gargalhada. Naquela época, Allison começou a usar o Emacs. Empolgado com o modelo comunitário, enviou uma contribuição de código-fonte — apenas para vê-la friamente rejeitada por Stallman.

"Parecia aquela manchete clássica do jornal satírico *The Onion*", diverte-se Allison: "'Deus responde às preces de uma criança: Não!'."

Enquanto colecionava vitórias expressivas na criação de programas de usuário e bibliotecas essenciais, o Projeto GNU foi adiando o desenvolvimento do *kernel* — o núcleo central do sistema operacional, o "guarda de trânsito" digital encarregado de gerenciar a memória, a CPU e o acesso aos recursos físicos da máquina.

Assim como fizera com outras ferramentas, Stallman buscou ganhar tempo procurando um kernel já existente que pudesse ser adaptado. Uma leitura dos boletins informativos do projeto — as famosas *GNUsletters* do final dos anos 1980 — revela que essa estratégia enfrentou percalços semelhantes aos da tentativa frustrada de usar o compilador Pastel. Em janeiro de 1987, uma *GNUsletter* anunciou a intenção de reformular o TRIX, um kernel desenvolvido no MIT. No entanto, Stallman nunca chegou a colocar as mãos na massa com o TRIX, pois estava totalmente absorvido pelo GCC; mais tarde, concluiu que o TRIX exigiria alterações profundas demais para servir de base. Em fevereiro de 1988, o boletim noticiava uma mudança de rota: o Projeto GNU passaria a apostar no Mach, um "micronúcleo" (*microkernel*) leve desenvolvido na Universidade Carnegie Mellon. O Mach ainda não era software livre, mas seus criadores garantiram reservadamente que iriam libertá-lo. Quando isso finalmente se concretizou, em 1990, o desenvolvimento do kernel do Projeto GNU pôde enfim ter início.[9]

Os atrasos no kernel, entretanto, eram apenas uma entre as muitas preocupações que tiravam o sono de Stallman nesse período. Em 1989, a Lotus Development Corporation abriu um processo judicial contra empresas concorrentes, como a Paperback Software International e a Borland, acusando-as de copiar a estrutura de menus de seu famoso programa de planilhas Lotus 1-2-3. O processo da Lotus, somado à disputa entre Apple e Microsoft em torno do "visual e sensação" (*look and feel*) de suas interfaces gráficas, ameaçava de morte o futuro do Projeto GNU. Embora nenhum desses processos atacasse diretamente o GNU, ambos punham em xeque o direito fundamental de desenvolver software compatível com sistemas existentes — premissa básica da grande maioria dos utilitários GNU. Esses litígios poderiam sufocar de vez a cultura de inovação da programação. Decidido a reagir, Stallman e alguns professores do MIT publicaram um anúncio contundente no *The Tech* (o jornal estudantil do MIT) repudiando as ações judiciais e conclamando um boicote geral contra a Lotus e a Apple. Em seguida, ajudou a fundar um movimento de resistência: a **Liga pela Liberdade de Programação** (*League for Programming Freedom* — LPF), organizando piquetes de protesto diante da sede da Lotus.

As manifestações entraram para o folclore hacker.[10] Segundo um comunicado da LPF, o ato ficou marcado pelo primeiro coro de protesto rimado em código hexadecimal:

> *1-2-3-4, botem os advogados para fora!*
> *5-6-7-8, inovar é nosso negócio, litigar é retrocesso!*

> *9-A-B-C, Lotus 1-2-3 não serve para você!*
> *D-E-F-O, monopólio de interface virou pó!*

Esses protestos refletiam as transformações profundas na indústria de computadores. As aplicações corporativas haviam discretamente substituído os sistemas operacionais como o principal campo de batalha mercantil. Diante de sua jornada ainda inacabada para concluir um sistema operacional livre, o Projeto GNU parecia irremediavelmente ultrapassado para aqueles que mediam valor apenas por sucesso comercial e modismos. Na verdade, o fato de Stallman ter criado uma entidade nova para combater os processos de *look and feel* fez com que alguns críticos decretassem que a FSF havia se tornado obsoleta.

Stallman, contudo, tinha um motivo estratégico cirúrgico para criar uma organização independente: atrair desenvolvedores de software proprietário para a trincheira comum. A expansão desmedida dos direitos autorais sobre interfaces de usuário ameaçava os criadores de programas proprietários tanto quanto os defensores do software livre. Tais desenvolvedores comerciais jamais se filiariam à Free Software Foundation, mas não encontravam na Liga pela Liberdade de Programação nenhum obstáculo ideológico que os afastasse. Pela mesma razão prudente, Stallman transferiu a liderança da LPF para outras mãos assim que a entidade se consolidou.

Em 1990, a prestigiosa Fundação John D. e Catherine T. MacArthur consagrou publicamente a genialidade de Stallman ao conceder-lhe a famosa bolsa MacArthur — o lendário "prêmio para gênios", que no seu caso garantia 240 mil dólares distribuídos ao longo de cinco anos. Embora a instituição não divulgue justificativas detalhadas para seus prêmios, a bolsa foi celebrada como um reconhecimento inequívoco pelo lançamento do Projeto GNU e pela formulação da filosofia do software livre. A verba aliviou uma série de preocupações imediatas de Stallman: permitiu, por exemplo, que ele abandonasse os bicos de consultoria com os quais vinha pagando as contas na década de 1980 e se dedicasse em tempo integral à causa do software livre.

A bolsa trouxe ainda uma consequência inusitada: permitiu que Stallman finalmente conseguisse se registrar como eleitor de forma regular. Em 1985, um incêndio no casarão onde ele morava o deixou sem domicílio oficial. O fogo também cobriu quase todos os seus livros de fuligem, e as tentativas de limpar essas páginas imundas não surtiram muito efeito. Daquele momento em diante, Stallman passou a viver praticamente como um "posseiro" nas salas do prédio 545 da Technology Square, sendo obrigado a votar registrado como "sem-teto". [11] "A comissão eleitoral de Cambridge não queria aceitar o laboratório como meu endereço residencial", recordaria Stallman mais tarde. "Foi só depois que um jornal publicou uma reportagem sobre o prêmio MacArthur mencionando isso que eles finalmente me deixaram registrar." [12]

Acima de tudo, o prêmio MacArthur escancarou para Stallman as portas da imprensa e garantiu-lhe convites para palestras no mundo inteiro — palanques que ele soube utilizar com maestria para divulgar o Projeto GNU, a ética do software livre e denunciar perigos emergentes, como os processos de interface e as patentes de software.

Ironicamente, a peça final que faltava para completar o sistema GNU nasceria exatamente em uma dessas viagens. Em abril de 1991, Stallman viajou a Helsinque, na Finlândia, para proferir uma palestra na Universidade Politécnica. Na plateia estava um jovem de 21 anos chamado Linus Torvalds, que começava a engatinhar no desenvolvimento do kernel Linux — o núcleo livre destinado a preencher a última grande lacuna do sistema GNU.

Estudante da vizinha Universidade de Helsinque, Torvalds observou Stallman com uma ponta de estranheza e curiosidade. "Foi a primeira vez na vida que vi diante de mim o tipo clássico do hacker: cabelos compridos até os

ombros e barba espessa", relembrou Torvalds em sua autobiografia de 2001, *Just for Fun*. "Não tínhamos muitos espécimes desses em Helsinque."[13]

Embora não sintonizasse com a veia "sociopolítica" do discurso de Stallman, Torvalds assimilou com perfeição um dos axiomas fundamentais daquela filosofia: nenhum programador escreve código sem erros. Mesmo quando os usuários não desejam personalizar um programa para atender a caprichos individuais, qualquer software sempre pode ser aprimorado. Ao compartilhar código, os hackers colocam o aperfeiçoamento da ferramenta acima de vaidades pessoais, do apego egoísta e do medo do escrutínio alheio.

Como muitos programadores de sua geração, Torvalds não havia sido forjado em gigantescos computadores de grande porte como o IBM 7094, mas sim em uma colcha de retalhos de computadores caseiros. Como universitário, ele deu o salto da programação básica em computadores pessoais para o ambiente Unix utilizando os computadores MicroVAX da faculdade. Essa trajetória gradual deu a Torvalds uma perspectiva muito particular sobre as barreiras de acesso às máquinas. Para Stallman, os grandes vilões do acesso eram a burocracia e os privilégios institucionais; para Torvalds, o grande inimigo era a geografia e o inverno inclemente da Finlândia. Cansado de ter que caminhar no gelo cortante até a Universidade de Helsinque apenas para se conectar à sua conta no Unix, Torvalds logo começou a maquinar uma forma de trabalhar no conforto aquecido de seu apartamento fora do campus.

Na época, Torvalds utilizava o Minix, um sistema operacional leve e não livre desenvolvido como modelo didático pelo professor holandês Andrew Tanenbaum.[14] O Minix incluía um compilador proprietário e um punhado de utilitários semelhantes àqueles que Tanenbaum havia desafiado arrogantemente Stallman a escrever em 1983.[15]

O Minix cabia perfeitamente na memória acanhada do computador 386 de Torvalds, mas fora desenhado muito mais para estudo acadêmico do que para uso diário em produção. Além disso, faltava ao Minix uma função essencial: um emulador de terminal capaz de imitar os terminais da universidade, permitindo que ele se conectasse ao MicroVAX diretamente de casa.

No início de 1991, Torvalds começou a programar seu próprio emulador de terminal. Ele usou o Minix para desenvolver a ferramenta, mas o emulador não rodava sobre o Minix: era um programa autônomo (*bare-metal*). Com o tempo, acrescentou rotinas para ler e gravar arquivos na partição de disco do Minix. Naquela fase, Torvalds já se referia jocosamente à sua criação como o "GNU Emacs dos emuladores de terminal".[16]

Como o Minix deixava a desejar em recursos elementares, Torvalds começou a expandir seu emulador até transformá-lo em um kernel comparável ao do próprio Minix — com a diferença de que o seu adotava uma arquitetura monolítica. Sentindo-se ambicioso, pediu em um grupo de discussão do Minix cópias das especificações dos padrões POSIX, que definiam a interface universal de um kernel Unix.[17] Poucas semanas depois, após combinar seu kernel com alguns utilitários do Projeto GNU e adaptá-los para rodar juntos, Torvalds enviou à Usenet uma mensagem que ecoava quase que com perfeição o manifesto original de Stallman de 1983:

> *Olá a todos que usam minix por aí-*
>
> *Estou fazendo um sistema operacional (livre) (apenas um hobby, não vai ser grande e profissional como o gnu) para clones de 386 (486) AT. Venho cozinhando isso desde abril e a coisa está quase pronta.*

> *Gostaria de qualquer feedback sobre o que vocês gostam ou não no minix, já que o meu SO se parece um pouco com ele (mesmo arranjo físico de arquivos no sistema de disco, por razões práticas, entre outras coisas). No momento já consegui portar o bash (1.08) e o gcc (1.40)...*[18]

A postagem atraiu um punhado modesto de respostas. Em menos de um mês, Torvalds publicou a versão 0.01 do kernel em um servidor de FTP na internet — a primeira edição minimamente funcional para testes externos. Ao colocá-la na rede, precisou escolher um nome para a criatura. Em seu disco rígido, ele havia salvo o projeto com o nome provisório de **Linux** — uma homenagem à tradição hacker de batizar variantes de Unix com a letra final X. Achando o nome "egocêntrico demais", Torvalds tentou rebatizá-lo para **Freax** (uma mistura de *free*, *freak* e *x*), mas o administrador do servidor FTP achou o apelido terrível e restaurou a pasta original como `linux`.

Torvalds dizia estar escrevendo um "sistema operacional livre", e sua comparação explícita com o GNU demonstra que ele tinha em mente um sistema completo. Contudo, o que ele de fato escreveu foi um kernel, puro e simples. Torvalds não precisou se preocupar em criar nada além do kernel porque, como ele bem sabia, todos os outros blocos fundamentais já haviam sido construídos e disponibilizados ao mundo graças aos desenvolvedores do GNU e de outros projetos livres. Como o Projeto GNU pretendia congregar todas essas ferramentas em um único sistema operacional, ele já havia feito o trabalho pesado de fazê-las interoperar perfeitamente. Enquanto Torvalds aperfeiçoava o kernel, ele (e mais tarde uma legião de colaboradores pelo mundo) garantiu que os utilitários GNU se integrassem perfeitamente a ele.

Nos primeiros meses, o Linux não era formalmente software livre: sua licença original impedia o uso e a distribuição comercial. Torvalds morria de medo de que alguma empresa oportunista se apropriasse de sua criação e tirasse o projeto de suas mãos. Porém, à medida que a combinação GNU/Linux ganhava tração e adeptos, Linus percebeu que a venda de mídias e cópias por empresas seria extremamente benéfica para popularizar o sistema na comunidade, passando a ter menos receio de uma apropriação indevida.[19] Isso o incentivou a rever a licença do Linux.

Nem o fato de ter compilado o Linux com o GCC nem o fato de rodar o GCC sobre o Linux o obrigavam legalmente a licenciar seu kernel sob a GNU GPL. Mas o uso intensivo das ferramentas de Stallman despertou nele uma obrigação moral de retribuição. Como o próprio Torvalds expressaria mais tarde: "Eu havia subido nos ombros de gigantes."[20] Sem surpresa, ele começou a refletir sobre o que aconteceria se outros desenvolvedores decidissem fazer melhorias no seu código. Uma década após essa decisão, Torvalds ecoou os mesmos sentimentos expostos por Robert Chassell, da FSF, ao resumir o dilema daquela época:

> *Você dedica seis meses da sua vida a um projeto desses, quer colocá-lo à disposição de todos e quer tirar algum benefício disso, mas não quer que ninguém se aproveite de você. Eu queria que as pessoas pudessem ver o Linux, modificar e aprimorar tudo o que tivessem vontade. Mas eu também fazia questão de ver o que eles estavam aprontando. Eu queria ter acesso perpétuo ao código-fonte deles, para que, se fizessem melhorias, eu pudesse incorporar essas melhorias no meu próprio código.*[21]

Quando chegou o momento de lançar a versão 0.12 do Linux — a primeira a operar com 100% de integração com o GCC —, Torvalds decidiu abraçar de vez a causa do software livre. Ele rasgou a licença restritiva original e adotou

integralmente a GNU GPL. Três anos depois, a comunidade alcançava o lançamento da versão 1.0 do Linux, o kernel; ele rodava com perfeição acoplado ao sistema operacional GNU quase concluído, formado pelas ferramentas do Projeto GNU e de colaboradores de todo o planeta. Na prática, haviam completado o sistema operacional GNU encaixando o Linux na vaga do kernel. O sistema resultante era, essencialmente, a união de **GNU mais Linux**. Torvalds e a comunidade, contudo, passaram a se referir a tudo aquilo, de forma um tanto confusa, simplesmente como "Linux".

Em 1994, o sistema combinado havia conquistado tanto respeito e notoriedade no meio hacker que observadores do mundo empresarial começaram a questionar se Torvalds não teria "entregado o ouro aos bandidos" ao abraçar a GPL nos primeiros meses do projeto. Na edição de estreia da revista *Linux Journal*, o editor Robert Young entrevistou Torvalds e perguntou à queima-roupa se ele não sentia remorso por ter aberto mão da propriedade comercial exclusiva do código do Linux. Torvalds foi categórico: não. "Mesmo olhando em retrospectiva", afirmou ele, a adoção da GPL havia sido "uma das melhores decisões de projeto" tomadas em toda a história do Linux.[22]

O fato de essa escolha histórica ter sido feita de forma totalmente espontânea, sem nenhuma consulta prévia ou concessão a Richard Stallman e à Free Software Foundation, atesta a impressionante autonomia e solidez da GPL. Embora Stallman tenha levado alguns anos para se dar conta da real magnitude do fenômeno, a explosão de popularidade do Linux despertava lembranças imediatas da epopeia do Emacs. Desta vez, porém, a faísca que desencadeou a reação em cadeia não foi um atalho genial de teclado como o `Control-R`, mas a possibilidade inédita de rodar um sistema completo de nível Unix dentro de computadores pessoais comuns. Os caminhos e as motivações podiam ser diferentes, mas o produto final correspondia com fidelidade matemática ao sonho ético original: um sistema operacional inteiro, robusto, funcional e composto exclusivamente por software livre.

Como atesta seu e-mail despretensioso à Usenet, Linus Torvalds levou meses para enxergar o Linux como algo mais do que um passatempo temporário enquanto os engenheiros do GNU não entregavam o kernel Hurd. Na cabeça do jovem finlandês, ele era apenas mais um garoto curioso desmontando e montando brinquedos por pura diversão. Não obstante, ao avaliar o sucesso estrondoso de um projeto que poderia perfeitamente ter apodrecido esquecido no disco rígido de um quarto de estudante, Torvalds reconhece a sabedoria de seu eu mais jovem por ter tido a coragem de renunciar ao monopólio e aceitar o pacto ético do copyleft.

"Posso não ter tido uma iluminação mística", escreveu Torvalds, refletindo sobre a palestra que ouviu de Stallman em Helsinque em 1991 e sua subsequente conversão à GPL. "Mas acho que alguma coisa daquele discurso realmente entrou na minha cabeça."[23]

Notas

Ver Hal Abelson, Mike Fischer e Joanne Costello, "Software and Copyright Law", versão atualizada (1997). Disponível em: http://groups.csail.mit.edu/mac/classes/6.805/articles/int-prop/software-copyright.html. ↩

Ver arquivo README do pacote Trn, disponível em: http://stuff.mit.edu/afs/sipb/project/trn/src/trn-3.6/README. ↩

Citação de John Gilmore em e-mail enviado ao autor Sam Williams. ↩

Ver Richard Stallman et al., "GNU General Public License: Version 1" (fevereiro de 1989). Disponível em: http://www.gnu.org/licenses/old-licenses/gpl-1.0.html. ↩

Ver David Betz e Jon Edwards, "Richard Stallman discusses his public-domain [*sic*] Unix-compatible software system with *BYTE* editors", *BYTE* (julho de 1986). Republicado no site do Projeto GNU: http://www.gnu.org/gnu/byte-interview.html. ↩

A chamada "cláusula de propaganda ofensiva" (*obnoxious advertising clause*) da Universidade da Califórnia mais tarde se revelaria um tremendo problema. Buscando uma alternativa permissiva à GPL, alguns programadores passaram a copiar a licença BSD original, substituindo o nome "University of California" pelos seus próprios nomes ou pelas instituições onde trabalhavam. O resultado catastrófico foi que qualquer sistema livre composto por muitos desses utilitários era forçado a estampar dezenas de créditos em qualquer material de divulgação. Em 1999, após anos de pressão e conscientização lideradas por Stallman, a Universidade da Califórnia concordou em revogar formalmente essa cláusula. Ver o ensaio "The BSD License Problem" em: http://www.gnu.org/philosophy/bsd.html. ↩

Ver Michael Tiemann, "Future of Cygnus Solutions: An Entrepreneur's Account", *Open Sources* (O'Reilly & Associates, Inc., 1999), p. 139. Disponível em: http://www.oreilly.com/catalog/opensources/book/tiemans.html. ↩

Ver Richard Stallman na revista *BYTE* (1986). ↩

Ver "Hurd History", disponível no site oficial do GNU: http://www.gnu.org/software/hurd/history.html. ↩

De acordo com o comunicado oficial de imprensa da Liga pela Liberdade de Programação (LPF), arquivado em http://progfree.org/Links/prep.ai.mit.edu/demo.final.release, o protesto entrou para a história por contar com o primeiro coro entoado em notação hexadecimal. ↩

Ver Reuven Lerner, "Stallman wins $240,000 MacArthur award", jornal do MIT, *The Tech* (18 de julho de 1990). Disponível em: http://the-tech.mit.edu/V110/N30/rms.30n.html. ↩

Ver Michael Gross, "Richard Stallman: High School Misfit, Symbol of Free Software, MacArthur-certified Genius" (1999). ↩

Ver Linus Torvalds e David Diamond, *Just For Fun: The Story of an Accidental Revolutionary* (HarperCollins Publishers, Inc., 2001), pp. 58-59. Embora presumivelmente fidedigno no que tange à vida pessoal de Torvalds, o livro comete alguns equívocos quando se refere a Stallman. Afirma, por exemplo, que Stallman "quer transformar tudo em código aberto" (*open source*) e que ele "vive reclamando de quem não usa a GPL". Na verdade, Stallman milita pelo *software livre*, e não pelo "código aberto". Além disso, ele recomenda com entusiasmo a adoção da GNU GPL na maioria das situações, mas sempre defendeu que todas as licenças que respeitam as quatro liberdades essenciais são legítimas e éticas. ↩

O Minix era não livre em 1991; hoje, contudo, é software livre. ↩

Tanenbaum descreve o Minix como um "sistema operacional" em seu livro *Operating System Design and Implementation*, mas o que a obra de fato examina é apenas a camada correspondente ao kernel do Unix. Existem dois usos corriqueiros para a expressão "sistema operacional", e um deles refere-se estritamente àquilo que o jargão do Unix chama de "kernel". E essa não é a única complicação de nomenclatura nesse campo: aquela porção do Minix consiste em um micronúcleo somado aos servidores que rodam sobre ele — uma arquitetura similar à do GNU Hurd rodando sobre o Mach. A união do micronúcleo com

esses servidores equivale conceitualmente ao kernel do Unix tradicional; contudo, quando o livro menciona "o kernel", refere-se unicamente ao micronúcleo. Ver Andrew Tanenbaum, *Operating System Design and Implementation* (1987). ↩

Ver Linus Torvalds e David Diamond, *Just For Fun*, op. cit., p. 78. ↩

O padrão POSIX foi posteriormente ampliado para abarcar normas de interface de linha de comando e ferramentas de usuário, mas isso ainda não existia em 1991. ↩

Ver Linus Torvalds e David Diamond, *Just For Fun*, op. cit., p. 85. ↩

Ibid., pp. 94-95. ↩

Ibid., pp. 95-97. ↩

Ver Linus Torvalds e David Diamond, *Just For Fun: The Story of an Accidental Revolutionary* (HarperCollins Publishers, Inc., 2001), pp. 94-95. ↩

Ver Robert Young, "Interview with Linus, the Author of Linux", *Linux Journal* (1º de março de 1994). Disponível em: http://www.linuxjournal.com/article/2736. ↩

Ver Linus Torvalds e David Diamond, *Just For Fun*, op. cit., p. 59. ↩

Capítulo 10: GNU/Linux

Em 1993, o movimento do software livre encontrava-se em uma encruzilhada. Para os de temperamento otimista, todos os sinais indicavam o triunfo iminente da cultura hacker. A revista *Wired*, uma publicação nova e arrojada repleta de reportagens sobre criptografia de dados, Usenet e liberdade de software, esgotava-se nas bancas. A palavra "Internet", antes um jargão restrito a hackers e cientistas de laboratório, entrava em definitivo no vocabulário popular. Até o presidente Bill Clinton já a utilizava. O computador pessoal, outrora visto como brinquedo de hobbistas, alcançara respeitabilidade corporativa plena, abrindo as portas para que uma novíssima geração de usuários tivesse contato com softwares forjados por hackers. E, muito embora o Projeto GNU ainda não houvesse alcançado sua meta original de entregar um sistema operacional GNU livre e completo, os usuários já podiam rodar a variante conhecida como GNU/Linux.

Olhasse-se por onde olhasse, o cenário parecia promissor — ou pelo menos era o que aparentava. Após uma década de trincheiras e sacrifícios, os hackers e seus valores finalmente ganhavam o respeito da sociedade abrangente. As pessoas finalmente estavam entendendo.

Mas estariam mesmo? Para os de índole pessimista, cada indício de aceitação vinha acompanhado de um contra-ataque perturbador. Claro, ser hacker de repente virara algo descolado — mas será que a badalação fazia bem a uma comunidade que sempre florescera no isolamento e na contracultura? Sim, a Casa Branca elogiava a Internet a plenos pulmões, chegando ao ponto de registrar seu próprio domínio, `whitehouse.gov`; nos bastidores, porém, reunia-se com grandes corporações, entusiastas da censura e autoridades policiais dispostas a enquadrar o faroeste digital da rede. Sim, os PCs estavam incrivelmente mais velozes; todavia, ao comoditizar o mercado de computadores com seus microprocessadores baratos, a Intel criava um ambiente no qual os fornecedores de software proprietário concentravam todo o poder. Para cada novo adepto conquistado para a causa do software livre pelo GNU/Linux, centenas — senão milhares — iniciavam o Microsoft Windows pela primeira vez. O GNU/Linux só dispunha de interfaces gráficas rudimentares e passava a anos-luz de ser amigável para o usuário comum. Em 1993, apenas um especialista conseguia fazê-lo funcionar. Para piorar, a primeira tentativa do Projeto GNU de desenvolver um ambiente de desktop gráfico havia fracassado.

No front político, a tempestade não era menor. O registro de direitos autorais sobre interfaces de usuário ainda era uma ameaça real — os tribunais norte-americanos ainda não haviam sepultado a tese. Ao mesmo tempo, a concessão de patentes sobre algoritmos e recursos de software surgia como um perigo galopante, ameaçando contaminar legislações mundo afora.

Por fim, havia a própria natureza enigmática do GNU/Linux. Blindado contra disputas jurídicas paralisantes (como as que asfixiavam o BSD), o ritmo alucinante de evolução do GNU/Linux fora tão desprovido de planejamento central, e seu sucesso tão acidental, que os programadores mais próximos do código não sabiam ao certo como definir a criatura. Mais parecido com uma coletânea musical de "grandes sucessos" do que com um projeto unificado, o sistema era um pot-pourri de clássicos hacker: desde o GCC, o GDB e a glibc (a novíssima biblioteca C do Projeto GNU) até o X Window System (ambiente gráfico para Unix concebido no Laboratório de Ciência da Computação do MIT), passando por ferramentas nascidas no BSD, como o BIND (*Berkeley Internet Naming Daemon*, que permitia substituir endereços numéricos de IP por nomes de domínio fáceis de memorizar) e a pilha de protocolos TCP/IP. E, costurando tudo isso, vinha o kernel Linux — ele próprio projetado como um

substituto alternativo para o Minix. Em vez de desenvolver um sistema operacional a partir do nada, Torvalds e sua legião crescente de desenvolvedores simplesmente plugaram seu trabalho dentro dessa matriz preexistente. Como o próprio Torvalds definiria mais tarde com sua irreverência costumeira ao explicar o segredo de seu sucesso: "No fundo, sou apenas um sujeito muito preguiçoso que adora levar o crédito pelas coisas que os outros realmente fizeram".[1]

Essa preguiça confessa, embora admirável sob o prisma da eficiência prática, soava alarmante do ponto de vista político. Em primeiro lugar, escancarava a total ausência de uma pauta ideológica por parte de Torvalds. Ao contrário dos desenvolvedores do GNU, Torvalds não construíra seu kernel movido pelo ideal de emancipar seus companheiros de ofício; ele o construíra apenas para ter algo com que brincar. Mas o que era, afinal, aquele sistema combinado, e a que filosofia as pessoas iriam associá-lo? Seria ele a materialização da filosofia do software livre delineada pela primeira vez por Stallman no *Manifesto GNU*? Ou seria apenas uma colcha de retalhos de ferramentas engenhosas que qualquer usuário munido de tempo e paciência poderia costurar em seu computador doméstico?

No final de 1993, uma parcela cada vez maior de adeptos do GNU/Linux inclinava-se para a segunda resposta, passando a misturar suas próprias variações caseiras. Começaram a pipocar as primeiras "distribuições" de GNU/Linux, vendidas ou distribuídas em disquetes e CDs — às vezes de graça, às vezes cobrando caro. Os resultados eram de qualidade mais do que duvidosa.

"Isso foi bem antes da Red Hat e das grandes distribuições comerciais", relembra Ian Murdock, então estudante de ciência da computação na Universidade de Purdue. "Você folheava revistas especializadas de Unix e dava de cara com aqueles anúncios minúsculos, do tamanho de cartões de visita, berrando 'Linux!'. A maioria das empresas era formada por aventureiros que não viam problema algum em enfiar uma pitada de código proprietário fechado no meio da receita."

Murdock, programador experiente em Unix, recorda-se de ter ficado "maravilhado" com o GNU/Linux assim que o baixou e instalou em seu computador pessoal: "Era extremamente divertido", conta ele. "Dava uma vontade imediata de participar." Contudo, a enxurrada de distribuições malfeitas e adulteradas começou a azedar seu entusiasmo inicial. Concluindo que a melhor maneira de ajudar era construir uma versão pura e sem aditivos nocivos, Murdock elaborou uma lista com as melhores ferramentas livres disponíveis, decidido a empacotá-las em sua própria distribuição. "Eu queria algo que fizesse jus ao nome do Linux", afirma Murdock.

Buscando atiçar a curiosidade da comunidade, Murdock publicou seus planos na Usenet, no grupo de discussão `comp.os.linux`. Uma das primeiras mensagens a pipocar em sua caixa de entrada veio do endereço `rms@ai.mit.edu`. Como hacker, Murdock reconheceu o remetente de imediato: tratava-se de Richard M. Stallman, criador do Projeto GNU e figura reverenciada na época como "o hacker dos hackers". Ao ver aquele endereço em sua fila de correio, Murdock ficou atônito: por que cargas d'água Stallman, líder de seu próprio projeto de sistema operacional, haveria de se importar com as reclamações de um estudante sobre distribuições de "Linux"?

Murdock abriu o e-mail.

"Ele dizia que a Free Software Foundation começava a olhar com muita atenção para o Linux e que a FSF tinha interesse em criar um sistema Linux [*sic*] também. Basicamente, pareceu a Stallman que nossos objetivos convergiam perfeitamente com a filosofia deles."

Sem exageros dramáticos, aquela mensagem sinalizava uma profunda guinada de estratégia por parte de Stallman. Até 1993, ele preferira manter prudente distância dos assuntos do Linux. Ao tomar conhecimento do novo kernel, Stallman pedira a um amigo que avaliasse sua viabilidade técnica. Stallman recorda o parecer: "Ele me disse que o código havia sido modelado a partir do System V, que era a versão inferior do Unix. E acrescentou que o kernel não tinha qualquer portabilidade."

O diagnóstico do amigo era correto. Desenhado sob medida para máquinas 386, o Linux estava visceralmente preso àquela plataforma barata de hardware. O que o amigo não previu, contudo, foi a imensa vantagem tática do Linux por ser o único kernel livre disponível no mercado. Em suma: enquanto Stallman passava os dezoito meses seguintes ouvindo relatórios frustrantes sobre a lentidão no desenvolvimento do Hurd, Torvalds conquistava para si legiões de programadores que, mais tarde, cuidariam de transplantar o Linux e o GNU para as mais diversas arquiteturas de processador.

Em 1993, o fracasso do Projeto GNU em entregar um kernel funcional já começava a gerar tensões tanto dentro do próprio projeto quanto no movimento do software livre como um todo. Em março de 1993, uma reportagem de Simson Garfinkel na revista *Wired* descrevia o Projeto GNU como "atolado", a despeito do sucesso estrondoso de suas ferramentas de desenvolvimento.[2] Quem convivia dentro do projeto e nos escritórios da FSF lembrava que o clima era ainda mais soturno do que o artigo deixava transparecer. "Estava muito nítido, ao menos para mim, que existia uma janela de oportunidade histórica para lançar um novo sistema operacional", avalia Robert Chassell. "E que, uma vez fechada essa janela, o interesse das pessoas desvaneceria. Foi exatamente o que aconteceu."[3]

Muito já se especulou sobre as agruras do Projeto GNU entre 1990 e 1993. Embora muitos atribuam a culpa a Stallman, Eric Raymond — um velho conhecido de Stallman que apoiava o Projeto GNU sem grande entusiasmo — sustenta que o entrave era de natureza institucional: "A FSF foi tomada pela soberba", dispara Raymond. "Eles abandonaram o objetivo de entregar um sistema operacional pronto para produção e se perderam em pesquisa acadêmica de sistemas." Pior ainda: "Acharam que nada fora dos muros da FSF poderia atingi-los."

Ian Murdock adota uma leitura mais benevolente: "Acho que parte do problema foi que eles foram ambiciosos demais e insistiram no erro", opina ele. "No final dos anos 1980 e início dos 1990, micronúcleos (*microkernels*) eram a última moda da computação. Infelizmente, foi exatamente nesse período que o Projeto GNU começou a desenhar seu kernel. Acabaram acumulando uma bagagem conceitual pesada demais, e teriam de dar muitos passos para trás para se livrar dela."

Stallman rebate: "Embora as motivações psicológicas citadas por Raymond pertençam unicamente à imaginação fértil dele, ele tem razão em relação a uma das causas do atraso do Hurd: o desenvolvedor responsável pelo Hurd por diversas vezes redesenhou e reescreveu do zero grandes blocos de código com base no que ia aprendendo, em vez de focar em colocar o Hurd para rodar o mais rápido possível. Era uma prática admirável de boa engenharia, mas não era a prática adequada para a nossa meta urgente: obter algo funcional quanto antes."

Stallman elenca ainda outros fatores determinantes para a paralisia. As batalhas judiciais contra a Lotus e a Apple exigiram dele uma dedicação quase integral; a isso somaram-se dores crônicas nas mãos que o impediram de digitar por cerca de três anos, afastando-o praticamente por completo da programação ativa. Stallman aponta também falhas de comunicação interna entre equipes do projeto: "Tivemos de suar muito para fazer o ambiente de depuração funcionar", recorda ele. "E as pessoas responsáveis pela manutenção do GDB na época não foram nem

um pouco colaborativas." Elas priorizavam dar suporte às plataformas que os usuários existentes do GDB já utilizavam, relegando a segundo plano a meta maior de concluir o sistema GNU.

No entanto, em um plano mais estrutural, Stallman reconhece que ele e a equipe do Hurd subestimaram dramaticamente a complexidade de implementar as funcionalidades de um kernel Unix sobre o micronúcleo Mach. "Minha lógica foi: bem, a parte do Mach encarregada de dialogar com a máquina física já foi testada e depurada", explicou Stallman em uma palestra em 2000. "Com essa dianteira, seríamos capazes de concluir o trabalho bem mais rápido. No entanto, descobriu-se que depurar programas assíncronos multithread era uma pedreira infernal. Havia erros de sincronismo (*timing errors*) que corrompiam arquivos inteiros, e ninguém acha graça nisso. O resultado prático foi que levamos anos e anos para colocar uma versão de testes de pé."[4]

Com o tempo, o sucesso retumbante do GNU acoplado ao Linux deixou cristalino que o Projeto GNU precisava embarcar no trem em movimento, em vez de ficar esperando o Hurd na estação. Além disso, a comunidade que se aglutinava em torno do GNU/Linux dava sinais claros de fragilidade ideológica. É verdade que o Linux havia sido abraçado pela GPL; mas, como o próprio Murdock já havia testemunhado, o compromisso de tratar o GNU/Linux como um sistema puramente livre estava longe de ser um consenso. No final de 1993, a base total de usuários do sistema havia saltado de uma dúzia de curiosos para algo entre 20 mil e 100 mil pessoas.[5] O que antes fora um passatempo despretensioso transformava-se num mercado suculento para exploração comercial — e muitos desenvolvedores não viam problema algum em empurrar software não livre para os novos convertidos. Como um Winston Churchill observando as tropas soviéticas marcharem sobre Berlim, Stallman sentiu um misto compreensível de euforia e apreensão diante daquela "vitória" do GNU/Linux.[6]

Embora chegasse tarde à festa, Stallman ainda possuía prestígio e autoridade inegáveis. Assim que a FSF anunciou que ofereceria apoio financeiro e moral ao projeto de Ian Murdock, outras ofertas de ajuda começaram a chover. Murdock batizou sua distribuição de **Debian** — uma fusão carinhosa entre o seu nome e o de sua esposa, Deborah — e, em poucas semanas, colocava na rua a primeira edição do sistema. "O apoio de Richard catapultou o Debian quase da noite para o dia: de um projetinho curioso, virou algo que a comunidade inteira era obrigada a levar a sério", recorda Murdock.

Em janeiro de 1994, Murdock publicou o histórico *Manifesto Debian*. Redigido no mesmo espírito combativo que movera o *Manifesto GNU* de Stallman uma década antes, o documento justificava a importância vital de marchar lado a lado com a Free Software Foundation:

> *A Free Software Foundation desempenha um papel fundamental no futuro do Debian. Pelo simples fato de que ela cuidará da distribuição, transmite-se uma mensagem inequívoca ao mundo: o Linux [sic] não é um produto comercial e jamais deveria sê-lo — o que não significa, em hipótese alguma, que o Linux seja incapaz de competir comercialmente. Aos que discordam, desafio-os a justificar o sucesso do GNU Emacs e do GCC, que não são softwares comerciais e, no entanto, causaram um impacto colossal no mercado corporativo.*
>
> *Chegou o momento de concentrarmos forças no futuro do Linux [sic], em vez de nos perdermos na ambição destrutiva de enriquecimento individual à custa de toda a comunidade Linux e de seu futuro. O desenvolvimento e a distribuição do Debian podem não ser a panaceia para todos os males apontados*

> *neste Manifesto, mas tenho a esperança de que, ao menos, trarão visibilidade suficiente a esses dilemas para que possamos superá-los.*[7]

Pouco depois do lançamento do Manifesto, a Free Software Foundation fez seu primeiro pedido de peso: Stallman queria que Murdock batizasse a distribuição de "GNU/Linux". Inicialmente, Stallman chegou a sugerir o neologismo "Lignux" — fundindo as palavras Linux e GNU —, mas a acolhida foi tão fria e zombeteira que Stallman recuou, optando pela fórmula mais longa, porém menos controversa: **GNU/Linux**.

Muitos críticos desdenharam a exigência de Stallman de estampar o prefixo "GNU", encarando-a como uma busca tardia e vaidosa por aplausos. Murdock, contudo, enxergou a atitude com outros olhos. Em retrospecto, viu naquilo uma tentativa consciente de desarmar uma tensão subterrânea que se acumulava entre os mantenedores oficiais do GNU e os hackers que adaptavam as ferramentas para rodar com o kernel Linux: "Estava surgindo uma cisão", recorda Murdock. "E Richard estava profundamente preocupado."

Desde 1990, cada software do Projeto GNU contava com um mantenedor oficial designado. Como os programas GNU precisavam rodar nos mais variados tipos de computadores, era frequente que usuários enviassem modificações para fazê-los funcionar em novas máquinas. Na maioria das vezes, esses voluntários só entendiam do seu próprio sistema e não tinham a menor preocupação em manter o código limpo e compreensível para os demais ambientes. Para incorporar essas novidades sem transformar o código num emaranhado ininteligível, o mantenedor oficial precisava rever com calma cada remendo e pedir aos autores que reescrevessem certas rotinas. Na imensa maioria dos casos, os voluntários colaboravam com gosto para ter seu trabalho incluído na árvore oficial. O mantenedor, então, polia o código e assumia a responsabilidade de mantê-lo para sempre. Em vários utilitários GNU, esse ciclo harmonioso havia se repetido dezenas de vezes para os mais diversos sistemas operacionais.

Os programadores que adaptaram as ferramentas GNU para rodar com o kernel Linux seguiram essa mesma trilha inicial: preocuparam-se exclusivamente com sua própria máquina. Mas quando os mantenedores do GNU pediram ajuda para organizar as alterações e facilitar a manutenção futura, vários programadores do ecossistema Linux deram de ombros. Não queriam ter o trabalho de fazer as coisas do jeito certo nem estavam preocupados com o destino futuro dos pacotes GNU: só se importavam com a versão que rodava no Linux e preferiam manter suas próprias versões paralelas e separadas (*forks*).

No universo hacker, a prática de um *fork* (a bifurcação de um projeto de software) é uma faca de dois gumes. Embora a ética hacker garanta a qualquer programador o direito sagrado de fazer o que bem entender com o código-fonte, a boa convivência dita que o correto é trabalhar em harmonia com o autor original para manter uma única versão unificada. A comunidade sempre considerou mais nobre e produtivo canalizar esforços na versão principal. Toda licença de software livre dá o direito inalienável de bifurcar um projeto — e por vezes isso se faz indispensável —, mas criar uma ruptura sem um motivo grave é visto como um ato de deselegância comunitária.

Como capitão do Projeto GNU, Stallman já sentira na pele as dores de uma divisão em 1991: "A empresa Lucid contratou vários programadores para desenvolverem melhorias para o GNU Emacs, supostamente como contribuições para o projeto", relembra Stallman. "Porém, eles não me avisaram de nada e projetaram vários recursos novos por conta própria. Como era de esperar, concordei com algumas escolhas e discordei de outras. Eles exigiram que eu incorporasse o pacote completo de código; quando respondi que aproveitaria cerca de metade

das funções, eles se recusaram a me ajudar a adaptar essa metade. Tive que fazer tudo sozinho." A disputa deu origem a uma versão paralela rival, o Lucid Emacs, gerando anos de mágoas e ressentimentos mútuos.[8]

Agora, de repente, diversos pacotes essenciais do GNU estavam sendo bifurcados de uma só vez pelo pessoal do Linux. No começo, Stallman julgou que se tratava de mera impaciência juvenil. Em contraste com a agilidade informal e quase anárquica da equipe do Linux, os mantenedores do código GNU costumavam ser mais metódicos, lentos e extremamente cuidadosos ao avaliar alterações que pudessem comprometer a sustentabilidade do sistema a longo prazo. Além disso, não tinham pudor algum em criticar publicamente códigos mal estruturados. Com o tempo, contudo, Stallman percebeu algo muito mais grave ao ler as mensagens dos desenvolvedores do Linux: uma completa alienação quanto à existência do Projeto GNU e às suas causas éticas.

"Descobrimos que a maioria daquelas pessoas que se autointitulavam 'usuários de Linux' não dava a mínima para o Projeto GNU", lamenta Stallman. "Eles diziam: 'Por que eu deveria perder tempo arrumando isso? Não estou nem aí para o Projeto GNU. O programa está funcionando para mim, está funcionando para nós, usuários de Linux, e o resto não interessa'. Foi um choque para nós, pois aquelas pessoas estavam usando, na essência, uma variante do sistema operacional GNU, e tinham tamanho desprezo pelo projeto. Demonstravam menos consideração pelo GNU do que qualquer outro usuário." Iludidos pelo hábito de chamar o conjunto simplesmente de "Linux", eles sequer desconfiavam de que o sistema operacional em suas telas continha infinitamente mais código do GNU do que do Linux.

Em nome da concórdia, Stallman instruiu os mantenedores oficiais do GNU a assumirem eles mesmos o fardo técnico que deveria ter cabido aos colaboradores externos. Na maioria dos pacotes, o arranjo deu certo; mas na **glibc**, o buraco era bem mais embaixo. A GNU C Library é o alicerce absoluto sobre o qual repousam todos os outros programas: é ela que processa todas as chamadas de sistema (*system calls*) direcionadas ao kernel. Em qualquer ambiente tipo Unix, os programas de usuário só conseguem conversar com o kernel através da biblioteca C.

As modificações feitas às pressas pelos voluntários para fazer a glibc servir de ponte entre o Linux e o restante dos utilitários haviam sido feitas de modo grosseiro e pontual, sem qualquer consideração pelo impacto nas outras arquiteturas. Para o mantenedor oficial da biblioteca, organizar aquele remendo era uma tarefa monumental e inglória. A Free Software Foundation foi obrigada a abrir os cofres e bancar o salário do mantenedor durante quase um ano inteiro para que ele reescrevesse do zero todas as integrações, garantindo que a versão 6 da glibc rodasse perfeitamente no GNU/Linux direto da caixa.

Ian Murdock confirma que esse imbróglio da biblioteca C foi a gota d'água que levou Stallman a bater o pé pela adoção do nome composto quando o Debian veio a público: "Aquela ramificação foi finalmente pacificada mais tarde. Ainda assim, na época, havia um temor real de que, se a comunidade Linux passasse a se enxergar como algo divorciado da comunidade GNU, isso viraria um foco de desunião irremediável."

Embora alguns acusassem a FSF de mero oportunismo político por classificar o conjunto GNU/Linux como uma "variante" do GNU, Murdock, já simpático à causa ética do software livre, achou o pedido de Stallman perfeitamente justo e razoável: "Ele agiu muito mais pela união da comunidade do que pela vaidade dos créditos", garante Murdock.

Aos poucos, contudo, os pedidos de natureza técnica começaram a gerar atritos. Se Murdock fora maleável no campo político, mostrou-se irredutível quanto à engenharia de software e ao modelo operacional do sistema. O que

nascera como uma parceria harmoniosa logo virou um cabo de guerra constante.

"Posso dizer por experiência própria que tive minha cota de discussões com ele", conta Murdock com um sorriso resignado. "Com toda a franqueza, Richard pode ser uma pessoa bastante difícil de lidar." O pomo da discórdia principal girava em torno dos símbolos de depuração (*debug symbols*). Stallman insistia que todos os programas executáveis distribuídos pelo Debian deveriam ser compilados com todas as informações de depuração embutidas, para que qualquer usuário pudesse dissecar e investigar imediatamente as falhas que encontrasse. Murdock contra-argumentava que isso inflaria os arquivos de forma monstruosa, inviabilizando o download pela internet e a distribuição em mídias físicas da época. Nenhum dos dois estava disposto a ceder.

Em 1996, recém-formado pela Universidade de Purdue, Murdock decidiu passar o bastão do projeto Debian, que não parava de crescer. Ele já vinha delegando tarefas administrativas a Bruce Perens — o hacker célebre pela criação do Electric Fence. Perens, assim como Murdock, era um engenheiro de Unix que se apaixonara pelo GNU/Linux tão logo as potencialidades do sistema se tornaram palpáveis. E, a exemplo de Murdock, acompanhava com admiração a militância da FSF, embora mantivesse uma distância cautelosa de seu líder.

"Lembro que depois de Stallman já ter publicado o *Manifesto GNU*, o GNU Emacs e o GCC, li uma matéria dizendo que ele prestava consultoria para a Intel", relata Perens, recordando seu primeiro contato com o fundador da FSF no fim dos anos 1980. "Escrevi para ele perguntando como ele conseguia defender com tanta veemência o software livre de um lado e prestar serviços para a Intel de outro. Ele me respondeu com toda a serenidade: 'Trabalho como consultor com o objetivo expresso de produzir software livre'. Ele foi impecavelmente educado, e a resposta fez todo o sentido do mundo."

Contudo, como desenvolvedor de ponta do Debian, Perens via com profunda aflição o desgaste entre Murdock e Stallman. Ao assumir o comando da equipe de desenvolvimento, Perens tomou uma decisão executiva firme: afastar formalmente o Debian da tutela da Free Software Foundation. "Concluí que não queríamos o estilo de microgerenciamento de Richard", dispara ele.

Segundo Perens, Stallman foi pego de surpresa pela decisão, mas teve a sabedoria de não transformar aquilo em uma guerra: "Ele esperou a poeira baixar e enviou uma mensagem dizendo que era fundamental mantermos um bom relacionamento. Ele nos pediu apenas que chamássemos a distribuição de GNU/Linux e não tocou mais em outros assuntos. Decidi que aquilo era justo e aceitei unilateralmente. Todo mundo respirou aliviado."

Com o passar dos anos, o Debian consolidou sua reputação como a distribuição definitiva para hackers exigentes, ao lado do lendário Slackware — outra distribuição pioneira surgida no mesmo biênio de 1993-1994. Contudo, o Slackware sempre incluiu programas proprietários em seus discos, e o Debian, após seu divórcio da FSF, também passou a distribuir programas não livres em repositórios separados.[9] Embora o Debian os rotulasse expressamente como *non-free* e deixasse claro que eles não faziam parte oficial do sistema, o simples fato de oferecê-los ao usuário funcionava como um endosso implícito. Conforme o Projeto GNU tomava ciência dessas práticas, concluiu com amargura que nem o Slackware nem o Debian eram distribuições de GNU/Linux que ele pudesse recomendar de sã consciência ao público amante da liberdade.

Para além dos círculos estritamente técnicos, o GNU/Linux começava a ganhar uma força avassaladora no mercado comercial de Unix. No estado da Carolina do Norte, uma empresa fundada em torno de soluções Unix e batizada de **Red Hat** reformulou radicalmente seu modelo para apostar todas as fichas no GNU/Linux. O diretor-executivo era Robert Young — o mesmo editor da *Linux Journal* que, em 1994, havia colocado Linus Torvalds

contra a parede ao perguntar se ele não se arrependia de ter submetido o kernel à GPL. Para Young, a resposta de Torvalds tivera um impacto "profundo". Em vez de buscar fórmulas para encurralar o mercado de software livre por meio de artimanhas proprietárias, Young começou a cogitar o que aconteceria se uma corporação comercial adotasse a mesma premissa do Debian: construir um sistema operacional composto integralmente por componentes de software livre. A Cygnus Solutions, fundada por Michael Tiemann e John Gilmore em 1990, já provara ser plenamente possível comercializar software livre faturando alto com qualidade, confiabilidade e serviços especializados. O que impediria a Red Hat de replicar a mesma fórmula mágica em torno do GNU/Linux?

"Na tradição científica ocidental, nós nos erguemos sobre os ombros de gigantes", filosofa Young, parafraseando tanto Torvalds quanto Sir Isaac Newton. "No mundo dos negócios, isso significa que não precisamos reinventar a roda a cada novo passo. A beleza desse modelo é que você devolve seu código para o domínio público.[10] Se você é uma desenvolvedora independente de software tentando criar um aplicativo e precisa de um discador de modem, por que raios reinventar a roda? Basta pegar o código do PPP do [GNU/]Linux da Red Hat e usá-lo como núcleo da sua ferramenta. Se precisa de um conjunto de interfaces gráficas, não precisa reescrever uma biblioteca gráfica inteira: basta baixar o GTK. De repente, você ganha o superpoder de reaproveitar o melhor do que já foi feito na história. E seu foco deixa de ser gerenciar bibliotecas genéricas para se concentrar exclusivamente em resolver as dores reais do seu cliente." Contudo, Young passava longe de ser um ativista ético do software livre, e não hesitou em embutir softwares proprietários fechados dentro do sistema comercial da Red Hat.

Young não era o único executivo fascinado com os ganhos de produtividade gerados pelo software livre. No final de 1996, praticamente todas as grandes corporações do mundo Unix começavam a acordar para o aroma irresistível do código compartilhado. O mercado comercial em torno do GNU/Linux ainda levaria um ano ou dois para explodir em definitivo, mas qualquer pessoa sintonizada com a cultura hacker já pressentia no ar: algo histórico estava prestes a acontecer. O processador Intel 386, a Internet e a World Wide Web haviam atingido a sociedade como uma sucessão de ondas gigantescas; o software livre despontava no horizonte como o tsunami mais poderoso de todos.

Para Ian Murdock, aquela imensa onda soava simultaneamente como uma homenagem justa e um castigo irônico para o homem que dedicara a vida a conferir uma identidade moral ao software livre. Como muitos entusiastas do Linux, Murdock lera as mensagens fundadoras na Usenet. Vira a declaração despretensiosa de Linus de que o Linux era "apenas um hobby". Lera também o desabafo sincero de Torvalds ao criador do Minix, Andrew Tanenbaum: *"Se o kernel do GNU estivesse pronto na primavera passada, eu nem teria me dado ao trabalho de começar o meu projeto"*.[11] Assim como tantos outros, Murdock lamentava as oportunidades históricas perdidas; mas compartilhava da eletricidade irresistível de ver novos horizontes brotando a cada segundo do próprio tecido da Internet.

"Participar do Linux naqueles primeiros tempos era uma delícia", recorda Murdock. "Ao mesmo tempo, era apenas uma distração, um jeito bacana de passar o tempo. Se você voltar e reler aquelas discussões antigas na Usenet, vai captar exatamente o espírito da época: era algo com que podíamos nos divertir enquanto o Hurd não ficava pronto. O pessoal estava impaciente. É curioso, mas, sob vários aspectos, suspeito que o Linux jamais teria acontecido se o Hurd tivesse chegado um pouco mais depressa."

Ao final de 1996, entretanto, quaisquer conjecturas do tipo "e se..." já não tinham a menor importância prática: o kernel de Linus Torvalds havia conquistado uma massa crítica insuperável de adeptos. A fatídica janela histórica de 36 meses havia se fechado para sempre. Mesmo que o Projeto GNU finalmente colocasse o kernel Hurd para

rodar perfeitamente, as chances de que alguém fora do círculo mais radical da cultura hacker viesse a se importar eram mínimas. O Linux, ao preencher o último grande vazio do ecossistema GNU, havia consumado com maestria o sonho dourado do Projeto GNU: erigir um sistema operacional livre, tipo Unix, plenamente funcional.

No entanto, a esmagadora maioria dos usuários não tinha a mais vaga ideia do que realmente acontecera: acreditavam piamente que o sistema inteiro se chamava "Linux" e que o mérito por aquela revolução pertencia exclusivamente ao jovem Linus Torvalds. Quase todos instalavam distribuições empestadas de softwares proprietários; tendo Torvalds — e não Stallman — como bússola de comportamento, não viam qualquer razão de princípio para recusar as algemas digitais do código fechado. Ainda assim, para aqueles poucos que sabiam valorizá-la, uma liberdade precária e luminosa encontrava-se finalmente ao alcance das mãos.

Notas

Torvalds fez essa declaração bem-humorada em diversas ocasiões. A mais célebre aparição da frase ocorreu no ensaio de Eric Raymond, "A Catedral e o Bazar" (*The Cathedral and the Bazaar*, maio de 1997). Disponível em: http://www.catb.org/~esr/writings/cathedral-bazaar/. ↩

Ver Simson Garfinkel, "Is Stallman Stalled?", *Wired* (março de 1993). ↩

A tese de Robert Chassell sobre a existência de uma "janela de oportunidade" de aproximadamente 36 meses para emplacar um novo sistema operacional não se aplica exclusivamente ao Projeto GNU. No início da década de 1990, as versões livres do BSD sofreram atrasos paralisantes provocados pela ação judicial da Unix System Laboratories, que bloqueou judicialmente a distribuição de derivados do código do BSD. Embora muitos especialistas considerem que derivados modernos do BSD, como FreeBSD e OpenBSD, superavam com folga o GNU/Linux em estabilidade, segurança e performance, o contingente de usuários do BSD permaneceu uma fração diminuta perante a legião massiva conquistada pelo GNU/Linux. Para uma análise detalhada dessa dinâmica, ver o ensaio do hacker neozelandês Liam Greenwood, "Why is Linux Successful" (1999), disponível em: http://www.freebsddiary.org/linux.php. ↩

Ver palestra proferida por Richard Stallman no Maui High Performance Computing Center (2000). Em correspondências com o autor Sam Williams, Stallman esclareceu tecnicamente o que caracterizava esses erros: "Erros de sincronismo (*timing errors*) ocorrem em sistemas assíncronos nos quais tarefas executadas em paralelo podem, teoricamente, ocorrer em qualquer sequência temporal, e uma ordem específica e desfavorável gera falhas graves. Imagine que o programa A execute uma rotina X e o programa B execute uma rotina Y, sendo ambas pequenas tarefas encarregadas de consultar e atualizar a mesma estrutura de dados na memória. Na esmagadora maioria das vezes, o computador executará X antes de Y, ou Y antes de X, e nada de errado acontecerá. Em raras ocasiões casuais, o escalonador do processador permitirá que o programa A rode até a metade da rotina X e, de repente, chaveará para o programa B, que executará Y inteira. Assim, Y será processada enquanto X estava pela metade. Como ambas alteram a mesmíssima estrutura, haverá uma colisão destrutiva. O resultado é uma falha imprevisível e impossível de reproduzir com facilidade em testes, pois depende do acaso do escalonador. A solução é usar travas (*locks*), mas quando um programador se esquece de aplicar a trava num ponto específico, o sistema padece desses erros crônicos." ↩

As estimativas de usuários de GNU/Linux no período são imprecisas. O número de 100 mil provém do histórico oficial corporativo da Red Hat (*Red Hat Milestones*), disponível na época em: http://www.redhat.com/about/corporate/milestones.html. ↩

O autor Sam Williams escreveu essa analogia com Winston Churchill antes de receber o seguinte comentário espontâneo de Richard Stallman: "A Segunda Guerra Mundial e a determinação implacável necessária para vencê-la foram lembranças fortíssimas durante a minha infância. Discursos como o de Churchill — 'Lutaremos nas praias, lutaremos nos campos... jamais nos renderemos' — sempre ecoaram profundamente dentro de mim." ↩

Ver Ian Murdock, *A Brief History of Debian* (6 de janeiro de 1994), Apêndice A: "The Debian Manifesto". Disponível em: http://www.debian.org/doc/manuals/project-history/ap-manifesto.en.html. ↩

Jamie Zawinski, ex-programador da Lucid que mais tarde lideraria o desenvolvimento do navegador Mozilla, documentou essa ruptura no ensaio histórico "The Lemacs/FSFmacs Schism", disponível em: http://www.jwz.org/doc/lemacs.html. A resposta detalhada de Stallman àquelas alegações pode ser lida em: http://stallman.org/articles/xemacs.origin. ↩

A versão Debian Buzz, lançada em junho de 1996, já trazia o navegador proprietário Netscape 3.01 em sua seção `contrib`. ↩

Robert Young utiliza aqui a expressão "domínio público" de maneira descuidada e leiga. Em termos jurídicos estritos, código em domínio público é aquele desprovido de direitos autorais. O código sob a GNU GPL jamais está em domínio público, pois depende visceralmente do direito autoral (*copyright*) para que suas cláusulas de proteção e copyleft tenham validade jurídica. ↩

Declaração extraída do célebre debate ("flame war") entre Linus Torvalds e Andrew Tanenbaum logo após o anúncio inicial do Linux. Ao defender sua decisão de criar um kernel monolítico não portável, Torvalds explicou que começara o Linux apenas como uma experiência pessoal para dominar seu novo PC 386: "Se o kernel do GNU estivesse pronto na primavera passada, eu nem teria me dado ao trabalho de começar o meu projeto". Ver Chris DiBona et al., *Open Sources* (O'Reilly & Associates, Inc., 1999), p. 224. ↩

Capítulo 11: Código Aberto

> **[RMS:** Exclusivamente neste capítulo, decidi suprimir algumas citações. O material suprimido tratava de código aberto e em nada dizia respeito à minha vida ou ao meu trabalho.**]**

Em novembro de 1995, Peter Salus — membro da Free Software Foundation e autor da obra clássica de 1994, *A Quarter Century of Unix* — enviou uma chamada de artigos aos membros da lista de discussão `system-discuss` do Projeto GNU. Como presidente do comitê organizador, Salus queria avisar seus companheiros hackers sobre a iminente Conferência sobre Software Livremente Redistribuível (*Conference on Freely Redistributable Software*), marcada para fevereiro de 1996 em Cambridge, Massachusetts. Patrocinado pela Free Software Foundation, o evento prometia ser a primeira conferência de engenharia exclusivamente dedicada ao software livre e, em uma demonstração expressa de união com outras frentes comunitárias, recebia de braços abertos artigos sobre "qualquer faceta de GNU, Linux, NetBSD, 386BSD, FreeBSD, Perl, Tcl/Tk e outras ferramentas cujo código-fonte fosse acessível e redistribuível". Escreveu Salus:

> *Ao longo dos últimos 15 anos, o software livre e de baixo custo tornou-se onipresente. Esta conferência reunirá desenvolvedores de diversas vertentes do software livremente redistribuível e distribuidores dessas ferramentas (nas mais variadas mídias). Teremos tutoriais, artigos avaliados por pares e palestras magnas proferidas por Linus Torvalds e Richard Stallman.*[1]

Entre os destinatários da mensagem de Salus estava Eric S. Raymond, membro do comitê da conferência. Embora não chefiasse um projeto de ponta nem dirigisse uma empresa como outros integrantes da lista, Raymond gozava de sólida reputação na comunidade hacker por alguns projetos de software e, sobretudo, por ter editado *The New Hacker's Dictionary* — uma versão substancialmente ampliada do lendário *The Hacker's Dictionary*, publicado uma década antes por Guy Steele.

Para Raymond, a conferência de 1996 era uma excelente notícia. Muito embora não encampasse integralmente a filosofia do movimento de software livre, ele já havia contribuído para vários utilitários do GNU, com destaque para o GNU Emacs. Essas contribuições, contudo, haviam sido interrompidas de forma abrupta em 1992, quando Raymond exigiu carta branca para fazer alterações na versão oficial do GNU Emacs sem precisar consultar Stallman — que liderava pessoalmente o desenvolvimento do editor. Stallman rejeitou a exigência, e Raymond o acusou de praticar "microgerenciamento". "Richard armou um escândalo por causa de modificações não autorizadas que fiz quando estava organizando as bibliotecas Lisp do Emacs", queixou-se Raymond. "Fiquei tão frustrado que decidi que não trabalharia mais com ele."

Apesar do rompimento, Raymond continuou atuante na comunidade de software livre. A tal ponto que, quando Salus cogitou colocar Stallman e Torvalds lado a lado como os grandes oradores do evento, Raymond apoiou a ideia com entusiasmo. Com Stallman representando a velha guarda sábia e veterana dos hackers de ITS e Unix, e Torvalds encarnando a safra jovem e efervescente dos hackers de Linux, a dobradinha soava como um pacto de

solidariedade que só traria benefícios — em especial para hackers ambiciosos com menos de 40 anos, como o próprio Raymond. "Eu estava com um pé em cada canoa", lembra ele.

Quando a conferência começou, a tensão velada entre as duas correntes era palpável. Ambos os lados, porém, tinham uma curiosidade comum: aquela seria a primeira oportunidade de ver o prodígio finlandês de carne e osso. Para a surpresa geral, Torvalds revelou-se um orador carismático e afável. Falando inglês com um sotaque sueco quase imperceptível, ele desarmou o público com um humor ágil e autodepreciativo.[2]

Mais surpreendente ainda, conta Raymond, foi a desenvoltura com que Torvalds disparava alfinetadas contra outros figurões da comunidade — a começar pelo mais imponente de todos, Richard Stallman. Ao final do encontro, o estilo descontraído de Linus — metade hacker brilhante, metade garoto descompromissado — havia conquistado veteranos e novatos.

"Foi um divisor de águas", relembra Raymond. "Até 1996, Richard era o único pretendente legítimo à liderança ideológica de toda a cultura hacker. Quem discordava dele não ousava fazê-lo em público. Quem quebrou esse tabu foi Torvalds."

A quebra definitiva do tabu deu-se quase no encerramento da conferência. Durante um debate sobre a hegemonia avassaladora do Microsoft Windows ou algum tema afim, Torvalds confessou candidamente que era fã incondicional do PowerPoint, o programa proprietário de apresentações da Microsoft. Para os puristas da velha guarda do software livre, aquilo soou como se alguém se gabasse de possuir escravos em plena conferência abolicionista. Já para Torvalds e sua legião crescente de seguidores, era puro bom senso: por que diabos abrir mão da conveniência de um software prático apenas para fazer uma declaração de princípios? Eles sequer concordavam com o princípio. Quando a liberdade exige algum sacrifício, aqueles que não se importam com ela enxergam a renúncia como autoflagelação tola, e não como o preço a pagar por algo infinitamente maior. Para eles, ser hacker não tinha nada a ver com martírio moral: tratava-se de resolver problemas práticos e colocar a máquina para funcionar.

"Foi uma declaração chocante para a época", relembra Raymond. "Mas ele tinha cacife para bancar aquilo porque, entre 1995 e 1996, seu prestígio explodia a cada dia."

Stallman, por seu turno, não guarda lembranças de qualquer animosidade durante o evento de 1996 — é provável até que nem estivesse presente na sala quando Linus fez o comentário sobre o PowerPoint. Mas recorda-se com nitidez de ter sentido na pele o veneno da famosa "irreverência" de Torvalds: "Havia um trecho na documentação do Linux que dizia explicitamente: 'imprima os padrões de código do GNU e depois rasgue-os em pedaços'", conta Stallman. "Quando você ia ver a fundo, a única coisa de que ele discordava era a parte menos importante de todas: a recomendação de indentação do código C."

"Tudo bem ele discordar de algumas convenções nossas. Não vejo problema nisso. Mas ele escolheu uma maneira deliberadamente grosseira de expressar sua opinião. Ele poderia simplesmente ter dito: 'Esta é a forma como prefiro que vocês identem o código'. Ponto. Não havia razão alguma para hostilidade."

Para Raymond, a simpatia com que outros hackers acolheram os gracejos de Torvalds confirmava uma suspeita: a trincheira que separava os desenvolvedores do Linux dos programadores do GNU era essencialmente geracional. Muitos hackers do Linux, assim como o próprio Linus, haviam crescido num ecossistema dominado por programas comerciais fechados. Haviam começado a colaborar com software livre sem enxergar qualquer injustiça

ética no software proprietário. Para a grande maioria, a conveniência prática era o único critério que importava. A menos que um programa fosse tecnicamente porco, não viam motivo algum para rejeitá-lo unicamente por causa de sua licença jurídica. Algum dia, quem sabe, os hackers criariam um substituto livre para o PowerPoint; até lá, por que perder tempo criticando o PowerPoint ou a Microsoft, em vez de simplesmente usá-los?

Tratava-se da manifestação explícita de uma fenda ideológica cada vez mais profunda dentro da comunidade: de um lado, os que defendiam a liberdade como um valor ético inegociável; de outro, os que valorizavam primariamente programas poderosos, rápidos e confiáveis. Stallman costumava se referir a esses dois polos como "partidos políticos" da comunidade, apelidando a primeira corrente de "partido da liberdade". Como os defensores da segunda vertente não se preocupavam em se rotular, Stallman os chamava ironicamente de "partido da onda" (*bandwagon party*) ou "partido do sucesso", já que viviam repetindo que "conquistar mais usuários" era a única meta primordial.

Na década decorrida desde a fundação do Projeto GNU, Stallman consolidara uma reputação formidável como programador genial. Mas construíra também a fama de ser uma figura intransigente e inflexível, tanto na arquitetura dos softwares quanto na gestão de pessoas. Havia um fundo de verdade nisso, mas essa fama também virou um álibi conveniente para que qualquer desafeto o culpasse sempre que suas próprias vontades eram contrariadas. Com o tempo, a lenda foi sendo inflada por boatos infundados.

Pouco antes da conferência de 1996, por exemplo, a Free Software Foundation enfrentou uma debandada em massa de seus funcionários. Brian Youmans, contratado por Peter Salus logo após a crise, descreveu o cenário: "Houve um momento em que Peter [Salus] era o único funcionário dando expediente no escritório." Os colaboradores anteriores haviam se desentendido feio com a diretora-executiva da época. Como Bryt Bradley relatou a amigos em dezembro de 1995:

> *[Nome omitido] (a diretora-executiva da FSF) decidiu retornar de sua licença médica/política na semana passada. A equipe do escritório (Gena Bean, Mike Drain e eu) concluiu que não tinha condições de trabalhar sob a supervisão dela devido aos inúmeros erros cometidos em suas funções antes de se afastar. Além disso, presenciamos repetidas ameaças indevidas de demissão e fomos alvo de constantes agressões verbais direcionadas a TODOS os funcionários do escritório. Pedimos reiteradas vezes que ela não voltasse como nossa chefe direta, esclarecendo que estávamos dispostos a trabalhar com ela em pé de igualdade, como colega. Nossos apelos foram ignorados. Pedimos demissão coletiva.*

A diretora em questão, então, deu um ultimato a Stallman: ou ele lhe concedia autonomia irrestrita sobre o escritório, ou ela renunciaria. Como presidente da FSF, Stallman recusou-se a entregar as rédeas da entidade, a diretora cumpriu a ameaça e pediu as contas, e ele chamou Peter Salus para gerenciar o escritório.

Quando Eric Raymond, que acompanhava os bastidores de fora, soube que a equipe havia debandado da FSF, presumiu de imediato que a culpa era do temperamento de Stallman. Foi o pretexto perfeito para alimentar sua tese favorita: a de que a personalidade de Stallman era a causa de todo e qualquer problema no Projeto GNU.

Raymond acalentava outra tese de peso: atrasos crônicos como o do Hurd e cismas amargos como a divisão do Lucid Emacs refletiam deficiências de gestão de projetos, e não dificuldades de codificação propriamente dita.

Pouco tempo após a conferência, Raymond começou a programar um projeto pessoal: um utilitário de e-mail chamado `fetchmail`. Inspirando-se no método informal de Torvalds, Raymond lançou o programa com a promessa expressa de atualizar o código-fonte o mais cedo e com a maior frequência possível (*release early, release often*). Quando uma enxurrada de voluntários começou a inundar sua caixa de entrada com relatórios de falhas e sugestões de melhoria, Raymond — que a princípio temia ver o código virar uma colcha de retalhos confusa — ficou pasmo ao notar que o programa se tornava incrivelmente robusto. Analisando o sucesso da fórmula de Torvalds, Raymond publicou um diagnóstico certeiro: ao usar a Internet como sua "placa de Petri" e o escrutínio inclemente da comunidade de hackers como um mecanismo de seleção natural, Torvalds criara um modelo evolutivo inteiramente livre de planejamento centralizado.

Mais do que isso, concluiu Raymond: Torvalds havia encontrado uma brecha genial para contornar a célebre **Lei de Brooks**. Formulada por Fred P. Brooks, lendário gerente do projeto OS/360 da IBM e autor do clássico de 1975, *The Mythical Man-Month* (O Mítico Homem-Mês), a lei sentenciou que *"acrescentar desenvolvedores a um projeto de software atrasado apenas o torna ainda mais atrasado"*. Compartilhando a crença ancestral dos hackers de que o código, assim como uma boa sopa, fica melhor quando feito por poucos cozinheiros, Raymond enxergou ali algo revolucionário: ao convidar cada vez mais cozinheiros para a cozinha, Torvalds havia descoberto como tornar a sopa *infinitamente melhor*.[3]

Raymond colocou suas observações no papel. Transformou-as em uma palestra e apresentou-a diante de um grupo de amigos no condado de Chester, na Pensilvânia. Intitulada **"A Catedral e o Bazar"** (*The Cathedral and the Bazaar*), a conferência confrontava o estilo descentralizado do "Bazar" (inaugurado por Torvalds) com o modelo hierárquico da "Catedral" (adotado pelo restante da indústria).

Segundo Raymond, a reação foi empolgada — mas nada comparada à apoteose que se seguiu durante o *Linux Kongress* de 1997, um encontro de entusiastas de GNU/Linux realizado na Alemanha na primavera seguinte:

"No Kongress, o público me aplaudiu de pé ao final da apresentação", orgulha-se Raymond. "Encarei aquilo como algo maiúsculo por duas razões. Primeiro, provava que as pessoas estavam eletrizadas com a tese. Segundo, mostrava que o entusiasmo superara até mesmo as barreiras do idioma."

Com o tempo, a palestra foi burilada e publicada na forma de ensaio com o mesmo título. O nome nascia da metáfora central: até então, os grandes softwares eram construídos como "catedrais" — monumentos solenes, caros e meticulosamente planejados por mestres de obras para resistir à passagem dos séculos. O Linux, por outro lado, assemelhava-se a um "vasto e ruidoso bazar", um ecossistema pulsante e caótico impulsionado pela dinâmica livre da Internet.

O texto de Raymond cometeu, contudo, uma armadilha retórica: associou o modelo da Catedral — amplamente praticado por toda a indústria e pelo próprio Raymond no passado — especificamente ao Projeto GNU e à figura de Richard Stallman, transformando o embate metodológico em um confronto de personalidades entre Stallman e Torvalds. Enquanto Stallman era retratado como o arquiteto da catedral — o mago solitário capaz de sumir por dezoito meses numa masmorra e retornar empunhando o compilador GCC —, Torvalds era pintado como o anfitrião caloroso de um jantar entre amigos. Ao deixar que os convidados guiassem o debate sobre o design do Linux e intervir apenas para apaziguar os ânimos da mesa, Torvalds forjara um modelo de desenvolvimento à imagem e semelhança de seu próprio temperamento relaxado. Na visão de Torvalds, a missão suprema da liderança não era policiar o código alheio, mas manter o fluxo de ideias em movimento perpétuo.

Sentenciou Raymond em seu ensaio: "Acredito que o hack mais genial e de maiores consequências concebido por Linus não foi a construção do kernel Linux em si, mas a invenção do modelo de desenvolvimento do Linux."[4]

Se a caracterização dos dois estilos operacionais era perspicaz, a tentativa de colar o rótulo de "Catedral" exclusivamente na testa de Stallman (ignorando todos os outros engenheiros que o utilizavam, inclusive o próprio Raymond) não passava de pura difamação. Na realidade, desenvolvedores de diversos utilitários do GNU — a começar pela própria equipe do Hurd — já conheciam e aplicavam os métodos colaborativos de Torvalds muito antes de Raymond testá-los no `fetchmail`, embora sem a afetação de criar manifestos para transformá-los em bandeira pessoal. Milhares de jovens programadores, ao lerem o ensaio de Raymond, foram induzidos a nutrir uma aversão injusta contra o Projeto GNU por conta dessa caricatura distorcida.

Ao decodificar as supostas virtudes da gestão de Torvalds, Raymond atraiu a simpatia de outros expoentes da comunidade para os quais a liberdade não passava de um detalhe supérfluo. Esse grupo sonhava em seduzir o grande capital corporativo para o ecossistema do software livre e, para atingir esse objetivo, decidiu traduzir a mensagem na única linguagem que os executivos compreendiam: eficiência, confiabilidade, preço baixo e vantagem tecnológica. Raymond tornou-se o principal porta-voz dessa cruzada pragmática, e seus ecos chegaram à diretoria da Netscape — cujo navegador comercial perdia mercado a olhos vistos para o Internet Explorer, distribuído gratuitamente pela Microsoft. Fascinados por uma palestra de Raymond, os executivos levaram a proposta à cúpula da empresa. Poucos meses depois, em janeiro de 1998, a Netscape chocou o mundo ao anunciar que liberaria o código-fonte de seu principal produto, o navegador Navigator, na esperança de atrair os hackers globais para acelerar seu desenvolvimento sob o projeto Mozilla.

Quando o CEO da Netscape, Jim Barksdale, declarou publicamente que o ensaio "A Catedral e o Bazar" fora a grande fagulha por trás de sua decisão histórica, Raymond foi alçado quase que instantaneamente ao status de celebridade pop do mundo hacker. Ele convidou um punhado de figuras estratégicas para uma conversa a portas fechadas: Larry Augustin, fundador da VA Research (que vendia estações de trabalho pré-instaladas com GNU/Linux); Tim O'Reilly, fundador da prestigiada editora técnica O'Reilly & Associates; e Christine Peterson, presidente do Foresight Institute, um respeitado centro de estudos do Vale do Silício especializado em nanotecnologia. A pauta do encontro resumia-se a uma pergunta crucial: como surfar a onda da decisão da Netscape para convencer outras gigantes da tecnologia a abrirem seus códigos?

Raymond confessa não se lembrar de todas as minúcias da conversa, mas recorda-se do primeiro gargalo colocado na mesa. A despeito de todos os esforços homéricos de Stallman para esclarecer que a palavra *free* em *free software* significava liberdade e não preço, a mensagem teimava em não entrar na cabeça dos empresários. A maioria esmagadora dos executivos, ao ouvir o termo pela primeira vez, traduzia a expressão mentalmente como "software de graça" e descartava qualquer diálogo subsequente. Enquanto a comunidade não superasse esse mal-entendido linguístico, o movimento enfrentaria uma batalha inglória no mundo dos negócios, mesmo após o precedente da Netscape.

Christine Peterson, cuja instituição acompanhava com muito interesse a evolução do software, ofereceu a cartada de ouro: **"código aberto"** (*open source*).

Olhando para trás, Peterson relata que o termo surgiu enquanto debatia o caso da Netscape com um amigo do ramo de relações públicas. Ela não se recorda de onde pinçou a expressão ou se a tomou emprestada de outra área, mas lembra com clareza de que seu amigo achou o nome péssimo.[5]

Naquela reunião na Califórnia, contudo, a recepção foi apoteótica. "Fiquei com receio de sugerir", relembra Peterson. "Eu não tinha autoridade técnica no grupo, então soltei o termo de maneira casual na conversa, sem alarde." Para surpresa dela, o termo colou como chiclete. Ao fim do encontro, quase todos os presentes, inclusive Eric Raymond, estavam radiantes com a novidade.

Raymond diz que hesitou um ou dois dias antes de adotar publicamente "código aberto" como substituto para "software livre", guardando o termo para o encontro organizado por Tim O'Reilly logo após a festa de lançamento do Mozilla. Batizado de "Encontro do Freeware" (*Freeware Summit*), o evento de O'Reilly visava atrair os holofotes da mídia e da comunidade para outros projetos valorosos que haviam tornado possível a liberação do navegador. "Todos aqueles pioneiros tinham muito em comum e, no entanto, muitos deles sequer se conheciam pessoalmente", conta O'Reilly. "Eu também queria escancarar para o mundo o impacto colossal que a cultura do software livre já exercia no planeta."

Ao fechar a lista de convidados, contudo, O'Reilly tomou uma decisão que deixaria sequelas políticas profundas: resolveu convidar quase que exclusivamente desenvolvedores baseados na Costa Oeste dos Estados Unidos — como Larry Wall (Perl), Eric Allman (Sendmail) e Paul Vixie (BIND). Houve exceções pontuais, é claro: o próprio Raymond, que morava na Pensilvânia mas já estava pela área por conta do evento da Netscape, descolou um convite imediato; Guido van Rossum, criador do Python e morador da Virgínia, também foi incluído. "Frank Willison, meu editor-chefe e grande entusiasta do Python, o convidou sem falar comigo antes", recorda O'Reilly. "Fiquei feliz com a presença dele, mas a princípio era para ser apenas um encontro regional."

Para muitos observadores atentos, a recusa tácita em convidar Richard Stallman para a mesa pareceu uma afronta deliberada. "Decidi não comparecer ao evento exatamente por causa dessa exclusão", revelou Bruce Perens. Raymond, que compareceu, alega ter defendido a presença de Stallman, mas sem sucesso. A suspeita de boicote ganhou ainda mais força pelo fato de O'Reilly e Stallman terem travado debates acalorados em público no passado por conta dos direitos autorais sobre manuais técnicos de computador. Stallman defendia com unhas e dentes que a documentação de um software livre deveria ser tão livre e modificável quanto o próprio código; O'Reilly, por sua vez, sustentava que a comercialização de livros impressos protegidos por copyright gerava receita para expandir o ecossistema e popularizava o software entre o público leigo. Ambos já haviam se estranhado até pelo título do evento, com Stallman rejeitando terminantemente a palavra "freeware" — termo que quase sempre definia programas proprietários distribuídos gratuitamente em formato binário, sem o código-fonte.

Em retrospecto, O'Reilly nega qualquer intenção de vingança: "Na época, eu nunca havia me encontrado pessoalmente com Richard, mas nossas trocas de e-mail haviam sido marcadas por uma tremenda intransigência da parte dele. Eu queria garantir que a tradição do GNU estivesse presente, por isso convidei John Gilmore e Michael Tiemann, que eu já conhecia e sabia que defendiam a GPL com paixão, mas que se mostravam bem mais dispostos a um debate aberto sobre os pontos fortes e fracos das diversas licenças. Diante de todo o barulho que se seguiu, reconheço que teria sido melhor tê-lo convidado, mas a ausência dele jamais deveria ser interpretada como desrespeito ao Projeto GNU ou à pessoa de Richard."

Desfeita deliberada ou não, o fato é que o neologismo "código aberto" conquistou a ampla maioria dos participantes do encontro. Os pioneiros compartilharam dores e traçaram planos conjuntos para polir a imagem pública de suas ferramentas. A obsessão central era como exaltar o triunfo do software livre na infraestrutura invisível da Internet — em vez de ficar apenas martelando a guerra aberta do GNU/Linux contra o monopólio da Microsoft. Mas, assim como na reunião anterior, a conversa logo desaguou nas dores de cabeça causadas pela palavra "livre" (*free*). Tim O'Reilly relembra um comentário feito por Linus Torvalds durante a mesa:

"Linus tinha acabado de se mudar para o Vale do Silício e nos contou que só recentemente havia descoberto que, em inglês, a palavra *free* tinha dois significados completamente distintos: *free* como liberdade (*libre*) e *free* como gratuidade (*gratis*)."

Michael Tiemann, fundador da pioneira Cygnus, sugeriu uma denominação alternativa para aplacar a confusão: *sourceware*. "Ninguém se animou muito", diverte-se O'Reilly. "Foi aí que Eric Raymond colocou o termo 'código aberto' na roda."

Embora o termo agradasse a muitos, o abandono da terminologia tradicional esteve longe de ser uma unanimidade pacífica. Ao término daquele dia histórico, os quinze convidados submeteram as três alternativas — *software livre*, *código aberto* e *sourceware* — a uma votação direta. Segundo O'Reilly, nove dos quinze presentes votaram em **"código aberto"** (*open source*). Embora alguns guardassem ressalvas íntimas, todos os presentes concordaram em adotar a nova nomenclatura dali em diante em suas entrevistas com a imprensa. "Queríamos transmitir uma mensagem monolítica de união para o mercado", explica O'Reilly.

O termo não demorou a explodir no vocabulário corporativo global. Pouco depois da cúpula, O'Reilly reuniu os líderes para uma concorrida entrevista coletiva com repórteres de gigantes como *The New York Times* e *The Wall Street Journal*. Em questão de meses, a foto de Linus Torvalds estampava a capa da revista *Forbes*, acompanhado nas páginas internas por fotos de Stallman, Larry Wall (Perl) e Brian Behlendorf (Apache). O código aberto estava oficialmente aberto para os negócios.

Para empresários presentes à reunião, como Michael Tiemann, a demonstração de união corporativa era tudo o que importava. Embora sua empresa acumulasse anos de sucesso prestando suporte e desenvolvendo ferramentas sob o GNU, ele testemunhava a cada dia as portas que se fechavam para os jovens empreendedores por causa de entraves conceituais:

"Não resta a menor dúvida de que a palavra *free* gerava confusão monumental no mercado", reflete Tiemann. "O código aberto se posicionou desde o primeiro minuto como algo amigável para os negócios, pragmático e corporativamente viável. O software livre insistia em se posicionar a partir de uma retórica de retidão moral. Para o bem ou para o mal, concluímos que seria infinitamente mais vantajoso pegar carona no trem do código aberto."

Logo após o encontro, Raymond ligou para Stallman para dar a notícia sobre a nova bandeira e sondar se ele aceitaria adotá-la. Raymond afirma que Stallman chegou a cogitar o termo por alguns instantes antes de descartá-lo: "Sei disso porque conversei diretamente com ele por telefone", sustenta Raymond.

A reação imediata de Stallman fora prudente: "Preciso pensar a respeito." No dia seguinte, porém, ele já havia decifrado o jogo: percebeu com clareza que a cartilha puramente pragmática de Raymond e O'Reilly ditaria os rumos do discurso do "código aberto", e que a única maneira de manter viva a chama ética do software livre perante a opinião pública era fincar o pé na denominação histórica.

Meses depois, Stallman anunciou sua posição definitiva: a expressão "código aberto", embora útil para vender os benefícios técnicos do software perante diretores financeiros, servia sobretudo para varrer para debaixo do tapete o tema sagrado da liberdade do usuário. O novo rótulo esquivava-se tanto da conotação indesejada de "programa de graça" quanto da conotação essencial de "programa que liberta o cidadão". Como ferramenta para promover a emancipação humana, era uma casca vazia. Na prática, Raymond e O'Reilly haviam apenas fundado uma sigla

partidária para aglutinar a ala anti-idealista da comunidade — aquela mesma com a qual Stallman jamais compactuaria.

Além disso, Stallman anteviu com lucidez que o discurso sedutor do código aberto faria a comunidade se ajoelhar perante as exigências corporativas. Embora o apoio do setor privado não fosse algo necessariamente danoso em si, a sede desesperada por aprovação empresarial fatalmente culminaria em concessões trágicas. "A lição número um de qualquer negociação ensina: se você está desesperado para agradar a outra parte, você está implorando por um péssimo acordo", sentencia ele. "É preciso ter a coragem de dizer não." Sintetizando sua postura na histórica conferência *LinuxWorld* de 1999 — celebrada pelo próprio Linus Torvalds como o baile de debutante da comunidade Linux —, Stallman implorou aos hackers que resistissem ao canto da sereia do conformismo:

> *Porque já provamos o que somos capazes de realizar, não precisamos nos humilhar para fechar acordos com corporações nem rebaixar nossas bandeiras. Deixem que eles venham até nós com propostas e nós as avaliaremos. Não precisamos mutilar o que fazemos apenas para conseguir a bênção deles. Você pode dar um passo consciente em direção ao seu objetivo supremo, depois outro, mais outro, e finalmente alcançar o destino. Ou pode aceitar uma meia-medida covarde que o impedirá de dar o passo seguinte, condenando-se a nunca chegar a lugar algum.*

Muito antes da *LinuxWorld*, contudo, Stallman já demonstrava disposição inabalável para comprar briga aberta com os defensores do código aberto. Poucos meses após a cúpula dos executivos, a O'Reilly promoveu sua segunda conferência anual sobre a linguagem Perl. Desta vez, Stallman estava na plateia. Durante uma mesa-redonda festiva que celebrava a decisão da IBM de adotar o servidor web livre Apache em seus produtos comerciais, Stallman aproveitou o microfone do auditório para desferir um ataque frontal e devastador contra um dos debatedores: John Ousterhout, criador da festejada linguagem Tcl. Stallman chamou Ousterhout abertamente de "parasita" da comunidade de software livre por comercializar extensões proprietárias e fechadas para a linguagem por meio de sua nova empresa, a Scriptics. Ousterhout já havia declarado com desfaçatez que a Scriptics devolveria apenas o mínimo indispensável de código para a versão livre do Tcl, usando essa migalha como escudo moral para faturar alto vendendo softwares proprietários. Stallman repudiou a manobra com indignação: "Não considero a existência da Scriptics necessária para a sobrevivência do Tcl", disparou Stallman, debaixo de vaias estrondosas da plateia estarrecida.[6]

"Foi uma cena horrível", relembrou Rich Morin, da Prime Time Freeware. "John havia feito contribuições gigantescas para a computação: o Tcl, o Tk, o sistema de arquivos Sprite. Ele é um programador de verdade." Apesar de nutrir profunda admiração pelas bandeiras de Stallman, Morin sentiu constrangimento diante da agressividade das palavras.

Stallman, fiel a si mesmo, recusou-se a pedir desculpas: "Criticar o software proprietário não é feio — o que é feio de verdade é o software proprietário. Ousterhout de fato dera contribuições admiráveis no passado, mas o fato concreto é que a Scriptics nascera para ser uma empresa quase 100% proprietária. Naquele auditório, defender a liberdade significava confrontar quase todo mundo. Falando do chão da plateia, eu só dispunha de poucas frases. A única forma de impedir que o problema fosse solenemente ignorado era dizê-lo com termos contundentes."

"Se as pessoas me reprovam por 'armar um barraco' quando faço uma crítica grave à conduta ética de alguém, ao mesmo tempo em que acham 'irreverente e genial' quando Torvalds destila grosserias por ninharias técnicas, isso

me parece uma clássica e deslavada política de dois pesos e duas medidas."

A polêmica contra Ousterhout custou a Stallman o afastamento temporário de um aliado de peso: Bruce Perens. Em 1998, Eric Raymond propôs a fundação da *Open Source Initiative* (OSI), uma entidade destinada a patentear e policiar o uso do selo "código aberto", além de estabelecer critérios formais para certificar licenças corporativas. Raymond recrutou Perens para redigir a famosa **Definição de Código Aberto** (*Open Source Definition*).[7]

Tempos depois, Perens renunciaria publicamente à OSI, expressando profundo remorso por ver a entidade ser instrumentalizada como trincheira anti-Stallman e anti-FSF. Ainda assim, olhando para trás, Perens compreende perfeitamente por que tantos programadores sentiam necessidade de manter uma distância profilática do líder do GNU: "Gosto imensamente de Richard e o admiro do fundo do coração", pondera Perens. "Mas tenho convicção de que ele faria um trabalho infinitamente melhor se tivesse mais equilíbrio na vida. Isso inclui tirar férias e passar alguns meses longe do software livre."

A resistência solitária de Stallman pouco pôde fazer diante do rolo compressor mercadológico erguido pelos entusiastas do código aberto. Em agosto de 1998, quando a gigante Intel adquiriu uma participação acionária na distribuidora Red Hat, uma reportagem do *The New York Times* descreveu a companhia como fruto de um movimento "conhecido indistintamente como software livre ou código aberto".[8] Seis meses mais tarde, uma matéria assinada por John Markoff no mesmo jornal noticiava em letras garrafais que a Apple Computer passaria a utilizar o servidor "código aberto" Apache em suas máquinas.[9]

Essa consagração na imprensa alimentou uma das maiores bolhas financeiras da história recente de Wall Street. Em agosto de 1999, a Red Hat — vendendo-se com orgulho como pioneira do "código aberto" — abriu seu capital na Nasdaq. Em dezembro, a VA Linux (antiga VA Research) protagonizou a estreia mais espetacular da história da bolsa: com ações ofertadas a 30 dólares no início do pregão, o papel explodiu até superar os 300 dólares, acomodando-se ao final da sessão na casa dos 239 dólares. Acionistas afortunados que compraram na abertura acumularam uma valorização inacreditável de 698% em um único dia — recorde absoluto na história da Nasdaq. Eric Raymond, como conselheiro de administração da empresa, viu seu patrimônio em ações saltar da noite para o dia para estarrecedores 36 milhões de dólares. O delírio, contudo, teve vida efêmera: os papéis desabaram em queda livre logo após o estouro da bolha das pontocom.

A cantilena dos apóstolos do código aberto era sedutoramente simples: para emplacar o software livre no mundo real, bastava torná-lo palatável aos apetites do mercado financeiro. Eles encaravam a militância de Stallman como uma birra infantil contra o mercado; em vez de bater de frente com o sistema, preferiam surfá-lo. Em vez de encarnarem o papel de nerds marginalizados e rebeldes, abraçaram a tietagem corporativa e viraram garotos-propaganda do grande capital, alavancando fortunas astronômicas no processo.

Tais expedientes garantiram um sucesso estrondoso para a grife do código aberto — mas não para as liberdades fundamentais do software livre. Para "espalhar a palavra", os novos profetas simplesmente omitiram o mandamento mais precioso de todos: a defesa da liberdade como um imperativo ético da dignidade humana. Os efeitos dessa amputação moral estão à vista de todos: a esmagadora maioria das distribuições modernas de GNU/Linux vem entupida de programas e bibliotecas proprietárias; a árvore oficial do kernel Linux mantida por Torvalds contém microcódigo e firmwares fechados (*blobs*); e a própria empresa antes conhecida como VA Linux abandonou suas raízes para lucrar com software proprietário. Mais da metade dos servidores de internet do

planeta rodam variações do Apache — que é um programa livre em sua matriz —, mas uma imensidão deles opera versões modificadas proprietárias distribuídas pela IBM.

"Nos seus piores dias, Richard acredita piamente que Linus Torvalds e eu conspiramos na calada da noite para sequestrar a revolução dele", ironiza Raymond. "A rejeição visceral de Richard à expressão código aberto e a cisão deliberada que ele cavou decorrem de uma mistura bizarra de idealismo cego e disputa de território. Há quem jure que tudo não passa de pura vaidade pessoal dele; eu não penso assim. O ponto é que ele se identifica de forma tão visceral e messiânica com a ideia do software livre que enxerga qualquer sombra de discordância como um atentado à sua própria existência."

Stallman rebate sem meias-palavras: "Raymond distorce deliberadamente meu pensamento: jamais achei que Torvalds tivesse 'conspirado' com quem quer que seja, até porque a dissimulação não condiz com o feitio de Linus. Por outro lado, a baixeza do caráter de Raymond transparece com clareza em suas próprias declarações. Em vez de combater meus argumentos no mérito das ideias, ele se põe a tecer conjecturas pseudopsicológicas baratas sobre a minha personalidade. Ele atribui as interpretações mais cruéis a terceiros anônimos para, em seguida, posar de bom moço e me 'defender' sugerindo uma caricatura apenas ligeiramente menos ofensiva. Ele já se valeu dessa covardia para me 'defender' inúmeras vezes."

Ironicamente, o sucesso do código aberto e de propagandistas como Raymond não destruiu o papel histórico de Stallman como líder comunitário — mas provocou uma confusão colossal sobre o que, afinal de contas, ele liderava. Como o movimento de software livre não conta com as verbas de publicidade corporativa nem com a adulação bajuladora da imprensa econômica, grande parte dos usuários de GNU/Linux sequer desconfia de sua existência, quanto mais de suas bandeiras éticas. Bombardeados diuturnamente pelas pregações mercantilistas do código aberto, esses recém-chegados não conseguem sequer conceber que Stallman defenda algo radicalmente diferente. Com frequência, ele recebe mensagens enternecidas de admiradores agradecendo por sua "luta abnegada em prol do código aberto" — ocasiões que aproveita para explicar, com didatismo e paciência monástica, que jamais compactuou com tal corrente, utilizando o contato para semear as verdades do software livre.

Muitos autores e pesquisadores tentam escapar da saia-justa recorrendo ao acrônimo guarda-chuva **FLOSS** (*Free/Libre and Open Source Software*). Todavia, incorrem no erro crasso de afirmar que existiria um "movimento unificado de FLOSS" — o que equivale à tolice de decretar que os Estados Unidos são governados por um único "movimento Liberal-Conservador". E, para surpresa de ninguém, as teses que costumam atribuir a essa suposta entidade unificada são, invariavelmente, os mesmos clichês comerciais do código aberto que já dominam os microfones.

Apesar de todas essas barreiras, o movimento de software livre continua fazendo sua voz ressoar e segue crescendo em termos absolutos. Ao fincar o pé em seus princípios fundadores e contrapor com clareza suas exigências éticas às concessões do código aberto, a causa preserva sua integridade. "Um dos traços mais marcantes da personalidade de Stallman é que ele simplesmente não arreda o pé", observa com admiração Ian Murdock. "Ele é capaz de esperar dez anos sentado até que as pessoas finalmente enxerguem a verdade pelo próprio discernimento, se for preciso."

Murdock, particularmente, enxerga nessa firmeza inabalável uma virtude preciosa e rara. Stallman pode não ser mais o general inconteste e solitário de um exército em marcha, mas permanece como a estrela-guia inegociável da consciência moral dos hackers: "Você sempre sabe que a postura dele será 100% coerente e fiel aos seus ideais",

arremata Murdock. "A esmagadora maioria das pessoas não tem essa fibra. Quer você concorde com ele, quer não, é impossível não se curvar em respeito a uma integridade dessas."

Notas

Ver Peter Salus, "FYI-Conference on Freely Redistributable Software, 2/2, Cambridge" (1995). Arquivado por Terry Winograd em: http://bat8.inria.fr/~lang/hotlist/free/licence/fsf96/call-for-papers.html. ↩

Embora cidadão finlandês, Linus Torvalds pertence à minoria linguística sueca da Finlândia, tendo o sueco como língua materna. O documento "The Rampantly Unofficial Linus FAQ", arquivado por Eric Raymond em http://catb.org/~esr/faqs/linus/, elucida: "A Finlândia possui uma expressiva minoria de fala sueca (cerca de 6% da população). Eles se autodenominam *finlandssvensk* e consideram-se genuinamente finlandeses, com raízes históricas fincadas no país há séculos. O sueco é uma das duas línguas oficiais da Finlândia." ↩

A chamada Lei de Brooks sintetiza a famosa passagem de Fred P. Brooks em *The Mythical Man-Month* (Addison-Wesley, 1995): "Sendo a construção de software inerentemente um esforço sistêmico — um exercício de inter-relações complexas —, o esforço despendido em comunicação é colossal e rapidamente supera o ganho de tempo obtido com a divisão de tarefas. Portanto, acrescentar homens a um projeto de software atrasado apenas o atrasa ainda mais." ↩

Ver Eric S. Raymond, *The Cathedral and the Bazaar* (1997). ↩

Ver Malcolm Maclachlan, "Profit Motive Splits Open Source Movement", *TechWeb News* (26 de agosto de 1998). ↩

Ibid. ↩

Ver Bruce Perens et al., "The Open Source Definition", *The Open Source Initiative* (1998). Disponível em: http://www.opensource.org/docs/definition.html. ↩

Ver Amy Harmon, "For Sale: Free Operating System", *The New York Times* (28 de setembro de 1998). ↩

Ver John Markoff, "Apple Adopts 'Open Source' for its Server Computers", *The New York Times* (17 de março de 1999). ↩

Capítulo 12: Uma Breve Jornada pelo Inferno Hacker

> **[RMS:** Neste capítulo, minha única intervenção foi acrescentar algumas poucas notas identificadas como esta.**]**

Richard Stallman fita fixamente, sem piscar, através do para-brisa do carro alugado, esperando o semáforo abrir enquanto nos arrastamos pelo centro de Kihei.

Nós dois estamos a caminho da cidade vizinha de Pa'ia, onde temos um jantar marcado com alguns programadores e suas esposas dali a cerca de uma hora.

Faz aproximadamente duas horas desde o término da palestra de Stallman no Centro de Computação de Alto Desempenho de Maui (*Maui High Performance Computing Center*), e Kihei — uma cidade litorânea que parecera tão acolhedora antes do evento — agora se mostrava profundamente exasperante. Como a maioria das cidades praianas americanas, Kihei é um exercício unidimensional de expansão suburbana desordenada. Dirigir pela sua avenida principal, diante de uma sucessão infindável de lanchonetes de hambúrguer, imobiliárias e lojinhas de biquínis, dá a incômoda sensação de ser um bolo alimentar blindado de metal deslizando pelo trato digestivo de uma solitária comercial gigantesca. O desconforto é agravado pela ausência de vias secundárias. Sem nenhuma alternativa a não ser seguir em linha reta, o trânsito avança em solavancos mecânicos: duzentos metros à frente, o sinal fica verde; no instante em que finalmente engatamos a primeira marcha, ele já amarelou de novo.

Para Stallman, que viveu a vida inteira na Costa Leste dos Estados Unidos, a perspectiva de passar a melhor parte de uma tarde ensolarada no Havaí preso num engarrafamento arrastado era o bastante para desencadear um colapso nervoso.

> **[RMS:** Como eu estava ao volante, também perdia um tempo precioso que poderia estar usando para responder aos meus e-mails — o que é uma aflição terrível, já que mal consigo dar conta da minha caixa de entrada de qualquer maneira.**]**

Pior ainda era a constatação de que, com duas ou três curvas rápidas à direita cerca de quatrocentos metros atrás, todo aquele tormento teria sido facilmente evitado. Infelizmente, estávamos à mercê do motorista do carro da frente: um programador do laboratório que conhecia o caminho e resolvera nos guiar até Pa'ia pelo trajeto "cênico", em vez de pegar a autoestrada Pilani Highway, que corria logo ali ao lado.

— Isso é terrível — desabafa Stallman, entre suspiros de pura frustração. — Por que não pegamos o outro caminho?

Lá na frente, o sinal volta a abrir. Nosso carro rasteja mais alguns metros. O ritual se repete por dez minutos angustiantes, até que finalmente alcançamos um cruzamento de grande porte que promete acesso direto à autoestrada vizinha.

O motorista da frente ignora a placa olímpicamente e segue reto pelo cruzamento.

— Por que ele não virou? — geme Stallman, jogando as mãos para o alto em desespero. — Dá para acreditar numa coisa dessas?

Prefiro ficar em silêncio. A simples constatação de que estou sentado no banco do passageiro de um carro conduzido por Stallman — e em Maui, nada menos — já me parecia inacreditável o suficiente. Até duas horas atrás, eu sequer sabia que Stallman sabia dirigir. Agora, ouvindo o violoncelo virtuoso de Yo-Yo Ma desfiar os acordes melancólicos de "Appalachian Journey" no rádio do carro e vendo o pôr do sol escorrer pelo horizonte à nossa esquerda, faço o possível para me camuflar no estofamento do banco.

Quando surge a oportunidade seguinte de conversão, Stallman aciona a seta para a direita, numa tentativa desesperada de sinalizar para o carro da frente. Em vão. Mais uma vez, rastejamos lentamente pelo cruzamento, estancando duzentos metros antes do semáforo seguinte. A essa altura, Stallman está lívido de raiva.

— Parece que ele está nos ignorando de propósito! — dispara, gesticulando freneticamente como um sinalizador de pista de porta-aviões na tentativa inútil de atrair a atenção do nosso guia pelo retrovisor. O homem da frente não demonstra a menor perturbação; durante os cinco minutos seguintes, tudo o que conseguimos ver dele é um pedaço da sua nuca refletida no espelho.

Olho pela janela de Stallman. Ao longe, as ilhas de Kahoolawe e Lanai emolduram o sol poente num cenário deslumbrante. É uma vista de tirar o fôlego — o tipo de paisagem que, imagino eu, torna engarrafamentos como este perfeitamente toleráveis para qualquer nativo do Havaí. Tento chamar a atenção de Stallman para a beleza do crepúsculo, mas ele, inteiramente obcecado pela displicência do motorista à nossa frente, me ignora por completo.

Quando o guia atravessa outro semáforo verde ignorando friamente uma placa enorme que anuncia "Pilani Highway — Próxima Direita", cerro os dentes. Lembro-me na hora de um aviso que o programador de BSD Keith Bostic me dera tempos antes: "Stallman não tem paciência com idiotice", alertara Bostic. "Se alguém disser ou fizer uma estupidez, ele olha no fundo dos olhos da pessoa e diz na cara: 'Isso é idiota'."

Olhando para aquele motorista alheio à nossa frente, compreendo que não é o atraso ou o desconforto que está torturando Stallman naquele momento: é a estupidez da situação.

— É como se ele tivesse escolhido essa rota sem pensar nem por um segundo em como chegar lá de forma eficiente — murmura Stallman.

A palavra "eficiente" fica pairando no ar do automóvel como um odor rançoso. Pouquíssimas coisas enfurecem mais a mente de um hacker do que a ineficiência. Fora a ineficiência de ter que descer até a impressora a laser da Xerox duas ou três vezes por dia que deflagrara sua investigação inaugural sobre o código-fonte da máquina. Fora a ineficiência criminosa de ter que reescrever ferramentas de software confiscadas por corporações gananciosas que o levara a combater a Symbolics e a erguer o Projeto GNU. Se, como sentenciou Jean-Paul Sartre, o inferno são os outros, o inferno dos hackers é ser obrigado a repetir os erros estúpidos dos outros. E não há o menor

exagero em afirmar que a vida inteira de Stallman tem sido uma cruzada obstinada para salvar a humanidade desse purgatório.

A metáfora do inferno ganha contornos palpáveis enquanto observamos o cenário se arrastar lá fora. Com sua fartura desordenada de estacionamentos, galerias comerciais e semáforos pessimamente sincronizados, Kihei parece menos uma cidade e mais um programa de computador monstruosamente mal projetado. Em vez de escoar o tráfego e distribuir os carros por vias marginais e corredores expressos, os planejadores urbanos canalizaram tudo para um único gargalo central. Da perspectiva de um hacker, ficar encurralado naquele labirinto é o equivalente sensorial a ouvir uma gravação em volume máximo de unhas arranhando um quadro-negro.

"Sistemas imperfeitos enfurecem os hackers", já observara o escritor Steven Levy em clássico estudo sobre a categoria — outro conselho que eu deveria ter lembrado antes de entrar naquele carro com Stallman. "Essa é uma das razões pelas quais hackers geralmente detestam dirigir: o sistema de sinais vermelhos programados aleatoriamente e ruas de mão única desenhadas de modo bizarro provoca atrasos tão irritantemente *desnecessários* que o impulso é sair do carro, arrancar as placas, abrir as caixas de controle dos semáforos... e redesenhar o sistema inteiro."[1]

A gota d'água, porém, é a traição de nosso guia de confiança. Em vez de caçar um atalho engenhoso — como qualquer hacker autêntico faria por puro instinto —, o motorista da frente preferiu ser cúmplice da burrice dos engenheiros de trânsito. Como Virgílio no *Inferno* de Dante, nosso guia parecia obstinado a nos oferecer o tour completo por aquele inferno hacker, quer quiséssemos, quer não.

Antes que eu consiga compartilhar essa reflexão poética, o motorista da frente finalmente liga a seta para a direita. Os ombros curvados de Stallman relaxam ligeiramente e, por um instante fugaz, a atmosfera sufocante dentro do carro se dissipa. A trégua, contudo, dura pouco: o carro da frente reduz subitamente a marcha. Cavaletes laranjas com avisos de "Obras à Frente" ladeiam os dois lados da rua e, muito embora a cobiçada autoestrada esteja a menos de trezentos metros de distância, a pistinha que nos levaria até ela encontra-se brutalmente interditada por uma retroescavadeira desligada e dois montes gigantescos de terra.

Stallman leva alguns segundos para processar o absurdo da cena, enquanto o nosso guia engata a ré para manobrar em cinco tempos no meio da rua estreita. Ao bater o olho na máquina amarela e na placa zombeteira de "Tráfego Interrompido", Stallman finalmente explode:

— Por que, por que, por quê?! — grita ele, jogando a cabeça contra o encosto do banco. — Você tinha obrigação de saber que a rua estava bloqueada! Você devia saber que não dava para passar por aqui! Você fez isso de propósito!

> **[RMS:** O que eu quis dizer foi que ele escolheu o caminho mais lento de propósito. Como explico adiante, considero que essas citações não reproduzem com exatidão o que eu disse.**]**

O motorista conclui a manobra e emparelha conosco no retorno em direção à avenida principal. Ao passar ao nosso lado, ele balança a cabeça e dá de ombros com um sorrisinho amarelo e constrangido. O gesto, misturado a uma expressão dócil, revela uma pitada de frustração típica de quem veio do continente, mas temperada com uma dose generosa de resignação insular. Através do vidro fechado do nosso carro alugado, a mensagem silenciosa soava cristalina: *"Poxa, cara, estamos em Maui... fazer o que, né?"*.

Stallman perde as estribeiras.

— Não dê esse sorrisinho de merda! — berra ele, embaçando o vidro da janela com o calor da respiração. — A culpa é toda sua, porra! Tudo teria sido infinitamente mais fácil se a gente tivesse feito do meu jeito!

> **[RMS:** Essas falas parecem nitidamente imprecisas, uma vez que não utilizo palavrões como advérbios no meu vocabulário. Aquilo não era uma entrevista gravada, de modo que Williams não estava com um gravador ligado. Tenho certeza de que os fatos se desenrolaram em linhas gerais como ele descreve, mas essas citações literais refletem a impressão pessoal dele, e não as minhas palavras exatas.**]**

Stallman enfatiza a expressão "do meu jeito" cravando as mãos com força no volante e puxando o próprio corpo contra a direção por duas vezes consecutivas. A imagem de sua silhueta robusta sacolejando daquele jeito lembra a de uma criança tendo um acesso de fúria na cadeirinha do banco de trás — impressão reforçada pelo tom agudo de sua voz. A meio caminho entre a fúria e o desespero, Stallman parece na iminência de cair no choro.

Felizmente, as lágrimas não vêm. Como uma tempestade repentina de verão, o acesso de raiva evapora quase na mesma velocidade em que começou. Depois de um par de respirações entrecortadas, Stallman engata a marcha a ré e executa sua própria manobra com habilidade cirúrgica. Quando finalmente desembocamos de volta na avenida principal, seu rosto recuperou a mesma impassibilidade de esfinge com que deixara o hotel meia hora antes.

Levamos menos de cinco minutos para encontrar a próxima transversal. Esta, por sorte, oferece acesso livre e direto à autoestrada; em questão de segundos, já estamos rodando suavemente rumo a Pa'ia a uma velocidade confortável e constante. O sol que antes castigava o ombro esquerdo de Stallman agora queima num tom avermelhado e sereno no retrovisor, tingindo as árvores *wiliwili* que desfilam velozes dos dois lados da pista.

Durante os vinte minutos seguintes, o único som perceptível dentro daquele automóvel — além do murmúrio suave do motor e do atrito macio dos pneus contra o asfalto — são as cordas melancólicas de um trio de violoncelo e violinos dedilhando uma balada folclórica das montanhas Apalaches.

Notas

Ver Steven Levy, *Hackers: Heroes of the Computer Revolution* (Penguin Books, 1984), p. 40. ↩

Capítulo 13: A Luta Continua

Para Richard Stallman, o tempo pode não curar todas as feridas, mas se revela um aliado conveniente.

Quatro anos após a publicação de "A Catedral e o Bazar", Stallman ainda se ressente das críticas de Raymond. E ainda resmunga com a consagração de Linus Torvalds ao posto de hacker mais famoso do planeta. Ele se lembra, com visível irritação, de uma camiseta popular que começou a circular em feiras e congressos de Linux por volta de 1999: inspirada no pôster promocional do primeiro filme de *Star Wars*, a estampa mostrava Torvalds empunhando um sabre de luz como Luke Skywalker, enquanto o rosto barbudo de Stallman aparecia encaixado sobre o corpo do robozinho R2-D2. A estampa ainda atinge os nervos de Stallman, não apenas por rebaixá-lo a mero coadjuvante cômico de Torvalds, mas por alçar o jovem finlandês ao papel de líder supremo da comunidade de software livre — um posto que o próprio Torvalds sempre rejeitou categoricamente assumir:

— É uma tremenda ironia — desabafa Stallman, em tom amargo. — Empunhar a espada para lutar é exatamente aquilo que Linus se recusa a fazer. Ele deixa todo mundo colocá-lo no pedestal de símbolo do movimento, mas na hora de combater pelos princípios, ele pula fora. Para que serve isso?

Por outro lado, foi justamente essa recusa obstinada de Torvalds em "empunhar a espada" que deixou o caminho aberto para que Stallman consolidasse sua autoridade como o grande árbitro moral da comunidade hacker. A despeito de suas mágoas, Stallman é forçado a admitir que a virada do milênio foi extraordinariamente generosa para ele e para a sua fundação. Empurrado para as margens pelo sucesso irônico de um sistema operacional GNU/Linux que os recém-chegados insistiam em chamar apenas de "Linux", Stallman soube virar o jogo e retomar a iniciativa política. Sua agenda de viagens entre janeiro de 2000 e dezembro de 2001 acumulou palestras nos seis continentes, incluindo paradas estratégicas em países onde o conceito de autonomia tecnológica e liberdade de software ressoava com peso monumental — como Índia e China.

Longe dos palanques, Stallman fez valer todo o peso institucional da Licença Pública Geral GNU (GPL), da qual continuava sendo o guardião supremo. Durante o verão boreal de 2000, enquanto o ar vazava rapidamente da bolha especulativa das pontocom em Wall Street, Stallman e a Free Software Foundation conquistaram duas vitórias históricas maiúsculas. Em julho de 2000, a Trolltech — empresa norueguesa responsável pela biblioteca de interfaces gráficas Qt, amplamente utilizada no GNU/Linux — anunciou que passaria a licenciar seu código sob a GNU GPL. Poucas semanas mais tarde, a Sun Microsystems — gigante corporativa que até então tentava pegar carona no oba-oba do código aberto sem entregar nada em troca — finalmente cedeu e anunciou que licenciaria sua suíte de escritório OpenOffice sob uma licença dupla: a GNU LGPL (*Lesser General Public License*) e a SISSL (*Sun Industry Standards Source License*).[1]

No caso da Trolltech, o triunfo foi o desfecho de uma cruzada paciente e calculada travada pelo Projeto GNU. O fato de a biblioteca Qt ser proprietária representava um perigo mortal para a comunidade de software livre porque o KDE — um ambiente de desktop gráfico livre que se tornara uma febre entre os usuários — dependia visceralmente dela para funcionar. A biblioteca Qt era código fechado, mas a Trolltech havia autorizado generosamente que projetos livres (como o KDE) a utilizassem sem pagar royalties. Embora o código do KDE em si fosse 100% livre, qualquer usuário que exigisse viver em liberdade era impedido de rodá-lo, pois teria que engolir a Qt proprietária. Stallman anteviu a armadilha: milhares de novos usuários queriam uma interface bonita

no GNU/Linux e a maioria não hesitaria em cair na tentação do KDE, com a Qt fechada escondida em suas entranhas. O risco iminente era ver o GNU/Linux ser instrumentalizado como um mero cavalo de Troia para disseminar a biblioteca proprietária da Trolltech, destruindo na raiz o propósito emancipatório do GNU.

Para neutralizar a ameaça, Stallman mobilizou a comunidade em duas frentes de contra-ataque simultâneas. A primeira foi o lançamento do **GNOME** — um ambiente de trabalho gráfico inteiramente livre concebido sob o manto oficial do GNU. A segunda foi o projeto **Harmony** — uma biblioteca livre projetada para ser compatível pino a pino com a Qt. Se o GNOME triunfasse, o KDE deixaria de ser necessário; se o Harmony vingasse, o KDE poderia se livrar da Qt proprietária. Em qualquer dos cenários, os usuários ganhariam um ambiente gráfico sofisticado no GNU/Linux sem precisar abrir mão de um milímetro de sua liberdade.

Em 1999, esses dois projetos avançavam a passos largos, e a diretoria da Trolltech começou a sentir o chão tremer sob os pés. Diante da ameaça real de ver seu ecossistema canibalizado por clones livres, a empresa resolveu liberar a Qt sob uma licença livre própria: a QPL. A QPL atendia aos requisitos básicos do software livre, mas Stallman prontamente apontou seu defeito fatal: incompatibilidade jurídica com a GPL. Era impossível misturar código da GPL com bibliotecas da QPL num único programa sem violar uma das duas licenças. Por fim, os executivos da Trolltech deram o braço a torcer, reconheceram que a GPL atendia com perfeição a seus interesses comerciais e publicaram a Qt sob licenciamento duplo: o mesmo código passava a ser oferecido tanto sob a GNU GPL quanto sob a QPL. Após três anos de cabo de guerra, a vitória do software livre era absoluta.

Com a Qt finalmente libertada, a razão de ser do projeto Harmony (que ainda não estava maduro para uso em produção) evaporou, e seus voluntários abandonaram o desenvolvimento. O GNOME, por sua vez, já havia conquistado tamanho fôlego comunitário que seguiu seu curso triunfal, consolidando-se como o ambiente gráfico oficial do Projeto GNU.

A Sun Microsystems também se viu forçada a dançar conforme a música da Free Software Foundation. Na conferência *O'Reilly Open Source* de 1999, o lendário cofundador e cientista-chefe da Sun, Bill Joy, subiu ao palco para defender a licença de "código comunitário" (*community source*) de sua empresa — na prática, um arremedo de compromisso que permitia aos usuários inspecionar e modificar o código da Sun, mas os proibia terminantemente de comercializar cópias sem antes pagar royalties à matriz. (Com essa restrição draconiana, o texto não se qualificava nem como software livre nem como código aberto.) Um ano após o discurso de Joy, o vice-presidente da Sun, Marco Boerries, pisava no mesmo palco para anunciar a nova postura da gigante tecnológica no lançamento do OpenOffice, suíte de escritório desenhada especialmente para GNU/Linux:

— Posso resumir nossa nova estratégia em apenas três letras — disparou Boerries. — **G-P-L**.

Na época, Boerries declarou que a capitulação da Sun tinha pouca relação com carinho por Stallman e tudo a ver com o poder gravitacional dos projetos protegidos pela GPL: "Basicamente, reconhecemos que produtos diferentes atraem comunidades diferentes, e a licença que você escolhe determina o tipo de talento que você consegue atrair", explicou Boerries. "Com o OpenOffice, ficou cristalino que nossa maior sintonia técnica se dava com a comunidade da GPL."[2] Ainda assim, Stallman considerou a vitória agridoce e incompleta, já que o OpenOffice incentivava abertamente a instalação de extensões e plug-ins proprietários.

Essas reviravoltas escancaram a força formidável da GPL e consagram, indiretamente, a genialidade política do homem que a concebeu. "Não existe um único advogado sobre a face da Terra que redigiria a GPL daquele jeito", avalia Eben Moglen, professor titular da Faculdade de Direito da Universidade de Columbia e consultor jurídico

geral da Free Software Foundation. "E, no entanto, ela funciona com perfeição matemática. E funciona justamente por causa da filosofia de design de Richard."

Ex-programador profissional em seus dias de juventude, Moglen começou a atuar *pro bono* ao lado de Stallman em 1990, quando o fundador da FSF lhe pediu assistência jurídica para uma questão pessoal. Moglen, que à época assessorava o criptógrafo Phil Zimmermann em sua célebre batalha contra o governo federal americano por conta do programa de criptografia PGP, sentiu-se honrado com o chamado:[3]

— Respondi a Richard que usava o Emacs todos os dias da minha vida e que precisaria advogar de graça para ele durante anos a fio só para começar a pagar essa dívida.

Desde então, Moglen teve a oportunidade ímpar de acompanhar de perto como os axiomas de engenharia hacker de Stallman eram transplantados para as ciências jurídicas. Segundo Moglen, a abordagem de Stallman ao código da lei é idêntica ao seu método de programar computadores: "Preciso confessar que, como advogado, a ideia de que a função de um texto jurídico é 'eliminar todos os bugs' soa quase absurda", pondera Moglen. "Toda engrenagem legal opera com uma margem inerente de incerteza, e o trabalho de noventa por cento dos advogados é justamente capturar os benefícios dessa imprecisão a favor de seus clientes. A meta de Richard é o exato oposto: ele quer erradicar qualquer ambiguidade, o que é teoricamente impossível. É impossível redigir uma única licença capaz de prever todas as circunstâncias em todos os sistemas jurídicos do planeta. Mas se alguém tentasse essa loucura, teria que fazê-lo exatamente do jeito que ele fez. E a elegância resultante, a simplicidade de design do texto, atinge quase à perfeição aquilo a que se propõe. A partir daí, basta um toque leve de advocacia tradicional para levá-la muito longe."

Como porta-voz encarregado de defender os interesses de Stallman nos tribunais, Moglen compreende com perfeição o desgaste dos aliados: "Richard é um homem incapaz de contemporizar em temas que considera fundamentais", explica ele. "Ele tem alergia a malabarismos verbais e a ambiguidades calculadas — virtudes que a diplomacia humana com frequência exige de quase todo mundo."

Além de blindar a FSF, Moglen emprestou sua combatividade à defesa de outros alvos célebres da indústria de copyright, como o programador russo Dmitry Sklyarov e os distribuidores do utilitário de descriptografia de DVDs `deCSS`.

Sklyarov havia escrito e disponibilizado um programa para quebrar a trava anticópia de livros digitais da Adobe na Rússia — onde tal prática era perfeitamente legal —, como funcionário de uma companhia russa. Ao desembarcar nos Estados Unidos para apresentar uma palestra técnica sobre suas descobertas, foi sumariamente preso pela polícia federal. Stallman participou ativamente dos protestos públicos repudiando a truculência da Adobe, e a Free Software Foundation denunciou a famigerada lei de direitos autorais DMCA (*Digital Millennium Copyright Act*) como pura "censura de software"; contudo, a entidade foi impedida de defender o utilitário de Sklyarov nos tribunais porque o código do programa era proprietário. Assim, Moglen assumiu a defesa de Sklyarov através da *Electronic Frontier Foundation* (EFF). Da mesma forma, a FSF evitou envolver-se diretamente na distribuição do `deCSS` para não incorrer em ilegalidade flagrante, mas Stallman condenou com veemência o governo americano por proibir o algoritmo, enquanto Moglen atuou na linha de frente como advogado dos réus.

Acompanhando as recusas calculadas da FSF em se desviar de sua missão institucional, Moglen aprendeu a respeitar a obstinação de Stallman: "Houve inúmeras ocasiões ao longo dos anos em que cheguei afobado para Richard e disse: 'Precisamos agir agora! A conjuntura é esta, a próxima jogada é aquela, temos que fazer isso

imediatamente!'. E a resposta de Richard era sempre uma só: 'Não precisamos fazer nada. Apenas espere. Aquilo que precisa ser feito se fará sozinho'."

— E quer saber de uma coisa? — conclui Moglen com um sorriso. — Na maioria das vezes, ele estava coberto de razão.

Esses relatos desmentem a autoimagem simplória que Stallman faz de si mesmo: "Não sou bom em jogos de estratégia", alega ele, rebatendo os analistas que o acusam de ser um maquiavélico jogador de xadrez político. "Não tenho o menor talento para antecipar mentalmente o que o adversário vai fazer dali a cinco lances. Minha estratégia sempre foi focar nas fundações das ideias. Dizer a mim mesmo: 'Vamos construir esse alicerce com a maior solidez que formos capazes'."

A força gravitacional inabalável da GPL é o melhor tributo a essa solidez erguida por Stallman e seus parceiros do GNU. Embora não tenha sido o primeiro a compartilhar código no mundo, Stallman é o único responsável pela edificação de todo o arcabouço ético do software livre. O fato de os programadores modernos sentirem-se à vontade ou não dentro desse templo moral é irrelevante: o simples fato de eles terem o direito de fazer uma escolha ética é o maior legado de Richard Stallman para a humanidade.

Avaliar o legado histórico de Stallman neste momento da narrativa pode parecer prematuro: aos 48 anos de idade, ele ainda tem muito por construir ou retificar. Não obstante, o impacto titânico do software livre torna tentador analisar sua trajetória para além das rusgas corporativas diárias, situando-o em um panorama histórico de maior envergadura.

Para seu crédito pessoal, Stallman recusa-se categoricamente a especular sobre o futuro: "Nunca tive capacidade de traçar planos meticulosos sobre como o amanhã haveria de ser", esquiva-se ele, oferecendo um epitáfio antecipado. "Eu apenas disse a mim mesmo: 'Vou lutar. Quem sabe onde isso vai dar?'."

É inegável que, ao escolher suas guerras de forma tão intransigente, Stallman afastou de si pessoas brilhantes que poderiam ter sido seus maiores generais, caso ele houvesse aceitado negociar seus ideais pelos deles. É prova de sua estatura moral que mesmo seus mais amargos oponentes políticos reconheçam sua grandeza quando instados a opinar. Contudo, a tensão perene entre o ideólogo inflexível e o hacker genial provoca uma indagação inevitável para qualquer biógrafo: como a posteridade julgará Richard Stallman quando sua personalidade tempestuosa não estiver mais aqui para atrapalhar a visão?

Nas primeiras versões deste manuscrito, batizei esse enigma de a "pergunta dos cem anos": pedi a diversos expoentes da revolução tecnológica que se despissem do imediatismo presente e assumissem a postura de um historiador examinando o software livre no ano de 2102. Sob essa perspectiva secular, é fascinante notar as semelhanças entre Stallman e figuras míticas da história americana que, marginalizadas em sua própria época, ganharam contornos gigantescos com o passar das décadas — luminares como Henry David Thoreau, filósofo transcendentalista e autor de *A Desobediência Civil*, ou John Muir, criador do Sierra Club e patrono do preservacionismo ecológico moderno. Ou, por outro lado, se ele terminará assemelhando-se a políticos como William Jennings Bryan — "O Grande Plebeu", herói do populismo do século XIX e algoz implacável dos monopólios que, a despeito de seu imenso poder em vida, acabou relegado a uma nota de rodapé no ensino fundamental.

Garantido nos anais da história pela invenção da GPL, valeria a pena indagar: será a GPL uma ferramenta ainda utilizada pelos programadores em 2102, ou terá se transformado em um fóssil curioso de um período transitório da computação? Soará a expressão "software livre" tão anacrônica e ingênua quanto soa hoje o movimento pelo "livre curso da prata" do século XIX, ou será ela reverenciada como uma profecia brilhante diante dos descalabros do autoritarismo digital?

Fazer previsões é um esporte perigoso. Stallman desdenha da charada, argumentando que especular sobre o que as pessoas pensarão daqui a cem anos parte do pressuposto conformista de que não temos como intervir na realidade presente. A pergunta que ele prefere fazer é outra: "O que devemos fazer hoje para construir um futuro melhor?". A maioria dos analistas, contudo, mordeu a isca sem hesitar.

— Daqui a cem anos, Richard e um punhado de outras pessoas merecerão muito mais do que uma nota de rodapé — sentencia Eben Moglen. — Eles serão estudados como o fio condutor de toda esta epopeia.

Esse "punhado de outras pessoas" citado por Moglen inclui nomes como John Gilmore — que além de suas contribuições técnicas ajudou a fundar a Electronic Frontier Foundation — e Ted Nelson, visionário autor da obra *Literary Machines* (1982). Para Moglen, Stallman, Nelson e Gilmore despontam como profetas complementares: Nelson, por ter criado o conceito de hipertexto e antevisto o nó górdio da propriedade da informação no mundo binário; Gilmore e Stallman, por terem diagnosticado os efeitos liberticidas da censura digital e erguido as fortalezas institucionais (EFF e FSF) para combatê-los. Entre os dois, porém, Moglen enxerga na batalha de Stallman uma natureza muito mais íntima e moral.

— Richard foi uma figura única porque a dimensão ética do software não livre tornou-se translúcida para ele num momento em que ninguém mais enxergava nada — afirma Moglen. — Isso decorre visceralmente da personalidade de Richard, que tantos biógrafos tentam pintar como um defeito acidental ou um entrave à sua obra.

John Gilmore corrobora a análise de Moglen:

> *Suspeito que os escritos de Stallman resistirão ao teste dos séculos com a mesma dignidade das cartas de Thomas Jefferson; ele é um autor de admirável clareza conceitual e inabalável na fidelidade a seus princípios. Se Richard exercerá no futuro uma influência comparável à de Jefferson, dependerá de uma questão essencial: se a abstração que chamamos de "direitos civis" será considerada mais preciosa daqui a cem anos do que as abstrações que chamamos de "software" ou "travas tecnicamente impostas".*

Outro pilar colossal do legado de Stallman que jamais deveria ser subestimado, lembra Gilmore, foi a invenção do modelo descentralizado de colaboração comunitária forjado no Projeto GNU. Com todos os seus percalços, esse molde de trabalho tornou-se a espinha dorsal de toda a engenharia de computação moderna. No fim das contas, diz Gilmore, essa fórmula social de produção coletiva talvez ressoe mais alto na história do que o Projeto GNU, a licença GPL ou qualquer compilador isolado concebido por Stallman:

> *Antes da Internet, era uma tarefa quase insuperável coordenar equipes de desenvolvimento à distância, mesmo quando os programadores se conheciam e confiavam mutuamente. Richard foi o grande desbravador da engenharia colaborativa de software, arregimentando voluntários anárquicos que*

> *jamais haviam se olhado nos olhos. Richard não inventou as ferramentas básicas desse ecossistema (o protocolo TCP, listas de e-mail, diff e patch, arquivos tar, ou utilitários como RCS e CVS), mas soube utilizar com maestria as ferramentas disponíveis para forjar comunidades humanas capazes de produzir obras-primas coletivas.*

Stallman, fiel ao seu rigor, pondera que essa avaliação, embora elogiosa, comete um equívoco de foco: "Ela valoriza o método produtivo acima da liberdade humana, refletindo as prioridades do código aberto e não as do software livre. Se for assim que os usuários do futuro enxergarem o Projeto GNU, temo que caminhemos para um mundo sombrio onde corporações mantêm os usuários sob cativeiro permanente, concedendo-lhes de vez em quando a migalha de colaborar em seus sistemas como prêmio de bom comportamento, mas sem jamais retirar as correntes de seus pulsos."

Lawrence Lessig, festejado jurista de Stanford e autor do clássico *The Future of Ideas*, é igualmente categórico. Como grande parte dos constitucionalistas do ciberespaço, Lessig enxerga a GPL como o principal pilar dos chamados "bens comuns digitais" (*digital commons*) — o vasto repositório de softwares comunitários e protocolos abertos que viabilizaram a expansão prodigiosa da Internet nas últimas três décadas. Em vez de emparelhar Stallman com luminares como Vannevar Bush, Vinton Cerf ou J.C.R. Licklider, Lessig enxerga a revolução de Stallman sob um prisma estritamente filosófico:

> *[Stallman] mudou o eixo do debate de "o que as coisas são" para "o que as coisas deveriam ser". Ele fez a sociedade enxergar a dimensão do que estava em jogo e forjou o instrumento jurídico perfeito para carregar esse estandarte adiante. Dito isso, não sei exatamente como enquadrá-lo ao lado de pioneiros como Cerf ou Licklider: a inovação dele foi de outra natureza. Não se resumia a escrever código ou a pavimentar estradas para a Internet. Tratava-se de abrir os olhos das pessoas para o valor intrínseco de um determinado tipo de Internet. Não vejo ninguém comparável a ele nessa categoria, nem antes, nem depois.*

Nem todos encaram o futuro de Stallman sob cores tão generosas. Eric Raymond, que considera a liderança moral de RMS superada desde 1996, enxerga contradições incontornáveis ao consultar sua bola de cristal para o ano 2102:

> *Creio que os artefatos erguidos por Stallman (a GPL, o Emacs, o GCC) serão reverenciados como obras revolucionárias, verdadeiras pedras fundamentais do mundo da informação. Acredito, contudo, que a história será muito menos benevolente com algumas das premissas teóricas das quais ele partiu, e implacável com sua inclinação para o territorialismo e comportamentos messiânicos de líder de seita.*

Quanto a Richard Stallman, ele próprio também vislumbra um cenário de incertezas:

> *O que a história dirá a respeito do Projeto GNU daqui a vinte anos dependerá exclusivamente de quem vencer a guerra sagrada pela liberdade do uso do conhecimento público. Se formos derrotados, não*

> *passaremos de uma nota de rodapé esquecida. Se triunfarmos, não há certeza alguma de que as pessoas compreenderão o papel do sistema operacional GNU — caso continuem achando que tudo não passou de "Linux", construirão uma imagem inteiramente falsa do que aconteceu e dos porquês da nossa luta.*
>
> *Mas, mesmo que vençamos, a versão da história ensinada nas escolas daqui a cem anos dependerá, como sempre, de quem detiver o poder político.*

Buscando sua própria metáfora nas brumas do século XIX americano, Stallman evoca a figura de **John Brown** — o incendiário líder abolicionista reverenciado como santo mártir ao norte da fronteira da escravidão e demonizado como louco sanguinário nas terras do Sul:

— O levante armado de escravos liderado por John Brown nunca chegou a triunfar de fato, mas seu julgamento e sua coragem na forca incendiaram a indignação moral da nação inteira pelo fim da escravatura. Durante a Guerra Civil, ele era um herói; cem anos mais tarde, durante a maior parte do século XX, os livros didáticos ensinavam que ele não passava de um fanático perturbado. Ao longo do regime de segregação racial, enquanto o preconceito imperava descarado, os Estados Unidos compraram alegremente a versão que os fazendeiros derrotados do Sul gostavam de contar a respeito de si mesmos, e as enciclopédias escolares reproduziram mentiras deslavadas sobre a Guerra Civil e seus mártires.

A comparação documenta tanto a consciência lúcida que Stallman tem do isolamento a que foi relegado quanto a polarização extrema que seu nome evoca. É improvável que a posteridade manche a biografia de Stallman com a pecha violenta que desabou sobre John Brown; apesar de suas metáforas belicosas, suas armas sempre foram puramente lógicas e jurídicas. Não obstante, é perfeitamente plausível conceber um futuro distópico no qual suas advertências éticas sejam sepultadas sob o entulho do esquecimento corporativo.[4]

Contudo, é precisamente essa têmpera de aço que poderá consagrar Stallman aos olhos da eternidade. Eben Moglen adverte os críticos que cometem a leviandade de tratar a personalidade tempestuosa de Stallman como um estorvo supérfluo perante os "grandes artefatos" de sua vida: sem essa mesma personalidade de ferro, alerta Moglen, não haveria artefato algum para ser celebrado:

— O maior ser humano com quem já tive a honra de trabalhar na vida foi o juiz Thurgood Marshall [primeiro ministro negro da Suprema Corte dos EUA]. Eu compreendia com perfeição o que fazia dele um gigante; compreendia por que ele fora capaz de redesenhar o destino do mundo. Sei que estou cruzando uma fronteira arriscada ao fazer essa analogia, até porque eles não poderiam ser mais diferentes: Marshall era um homem plenamente inserido na vida pública, operando com maestria as convenções sociais para defender uma comunidade marginalizada diante da elite dominante. Seu grande gênio residia na diplomacia interpessoal. Mas Marshall era forjado de uma só peça maciça. E, por mais distintos que sejam em tudo o mais, a única pessoa que consigo colocar no mesmo patamar de solidez ética — talhado de um só bloco de rocha, compacto, feito da mesmíssima matéria estelar de ponta a ponta — é Richard Stallman.

Para ilustrar essa imagem imorredoura, Moglen relembra um almoço informal que compartilharam na primavera de 2000. O delírio financeiro da abertura de capital da VA Linux ainda dominava as manchetes dos cadernos de economia, e uma enxurrada de polêmicas sobre o destino da tecnologia inundava o noticiário. Cercado por um turbilhão de jornalistas implorando por uma declaração, Moglen recorda-se de sentar à mesa com Stallman e ter a

sensação de estar no olho tranquilo de um furacão: durante a hora seguinte, a conversa girou mansa e serenamente em torno de uma única obsessão: blindar as cláusulas jurídicas da GPL contra os ataques que viriam.

— Estávamos ali sentados debatendo o que faríamos com certos litígios na Europa Oriental e como reagiríamos quando o monopólio corporativo sobre o conteúdo digital começasse a asfixiar o software livre — relembra Moglen. — No meio da conversa, peguei-me pensando em como aquela cena deveria parecer bizarra para quem nos olhasse de fora. Ali estávamos nós, dois sujeitos barbudos com ar de anarquistas, tramando os passos seguintes da história. E, claro, com Richard arrancando fiapos dos nós da própria barba, jogando-os na tigela de sopa e agindo com seu jeitão desajeitado de sempre. Qualquer um que ouvisse nossa prosa acharia que éramos dois lunáticos varridos; mas eu tinha a mais absoluta certeza íntima: a revolução inteira estava acontecendo naquele exato restaurante, em cima daquela mesa de fórmica. É isso o que faz a história se mover. E aquele homem sentado diante de mim era a força motriz que mantinha tudo de pé.

Para Moglen, aquela cena resume com fidelidade a pureza quase franciscana do estilo de Stallman:

— Foi muito engraçado — conclui Moglen. — Olhei nos olhos dele e disse: 'Richard, você já reparou que nós dois fomos os únicos sujeitos neste planeta inteiro que não ganharam um único centavo com essa revolução?'. E em seguida paguei a conta do restaurante, porque eu sabia que ele não tinha dinheiro no bolso para pagar o próprio almoço.[5]

Notas

A Sun Microsystems foi forçada por uma notificação judicial de marca registrada a adotar o nome oficial mais rebuscado de `OpenOffice.org`. ↩

Entrevista de Marco Boerries concedida ao biógrafo Sam Williams em julho de 2000. ↩

Para maiores detalhes sobre as agruras jurídicas de Philip Zimmermann, ver Steven Levy, *Crypto* (Penguin Books, 2001), pp. 287-288. Na primeira edição impressa de *Free as in Freedom*, o autor Sam Williams relatou incorretamente que Moglen havia defendido Zimmermann contra a Agência de Segurança Nacional (NSA); segundo a narrativa de Levy, as investigações foram conduzidas pela Procuradoria-Geral dos EUA e pela Receita/Alfândega americana, e não pela NSA. ↩

[RMS: As palavras adicionais de Sam Williams aqui no manuscrito original, alegando que eu havia 'construído a causa do software livre não como um movimento de massas, mas como uma coleção de batalhas individuais contra as tentações proprietárias', não correspondem aos fatos históricos. Desde o primeiro manifesto público do Projeto GNU, convoquei a sociedade inteira a apoiar a causa. O movimento de software livre sempre almejou ser um movimento de massas, e a única questão legítima é saber se já acumulamos adeptos suficientes para ostentar esse título. Em 2009, a Free Software Foundation contava com mais de 3.000 membros contribuintes ativos pagando anuidades generosas e mais de 20.000 assinantes em seu boletim informativo mensal.**]** ↩

[RMS: Jamais recuso quando alguém de boa vontade se oferece para me pagar uma refeição, pois meu amor-próprio não depende da vaidade tola de disputar quem assina a comanda. No entanto, é óbvio que eu tinha dinheiro suficiente para pagar aquele almoço. Minha renda pessoal, proveniente de cerca de metade das palestras remuneradas que profiro pelo mundo, é bastante inferior aos honorários de um professor titular da Faculdade de Direito de Columbia, mas estou bem longe de ser um homem pobre.**]** ↩

Epílogo de Sam Williams: Solidão Esmagadora

> **[RMS:** Por se tratar de um capítulo com um teor tão visceralmente pessoal de Sam Williams, assinalei todas as minhas alterações no texto original entre colchetes ou reticências, e limitei tais intervenções exclusivamente a esclarecer pontos técnicos ou jurídicos e a suprimir passagens que considerei gratuitas, hostis e desprovidas de informação relevante. Acrescentei também notas identificadas como "RMS:" para responder a determinados pontos. O próprio Williams também fez alterações no texto deste capítulo; as mudanças feitas por ele não estão explicitamente demarcadas.**]**

Escrever a biografia de uma pessoa viva assemelha-se muito a produzir uma peça de teatro: o drama encenado na frente das cortinas frequentemente empalidece perto das tempestades dos bastidores.

Na obra *A Autobiografia de Malcolm X*, o escritor Alex Haley brinda os leitores com um vislumbre raro desse drama de camarim. Despindo-se do papel de *ghostwriter*, Haley assume a narrativa em primeira pessoa no epílogo. Ali, ele relata como um repórter *freelancer*, inicialmente rotulado de "marionete" e "espião" pelos porta-vozes da Nação do Islã, conseguiu vencer barreiras políticas e idiossincrasias pessoais para colocar no papel a trajetória brutal de Malcolm X.

Embora hesite em comparar este livro ao clássico de Alex Haley, guardo uma dívida de gratidão para com a sinceridade desarmada de seu epílogo. Ao longo dos últimos doze meses, aquele relato serviu como uma espécie de manual de sobrevivência ensinando como lidar com um biografado que construiu uma carreira inteira em cima da arte de ser desagradável.

> **[RMS:** Construí minha carreira em cima do hábito de dizer *não* a coisas que os outros aceitam sem pestanejar; mas se por vezes pareço ou sou desagradável, não é por deliberada intenção.**]**

Desde o primeiro minuto, planejei encerrar esta biografia com um epílogo semelhante: tanto em homenagem a Haley quanto para revelar aos leitores os bastidores improváveis de como este livro veio ao mundo.

A história por trás desta história começa em um apartamento em Oakland e serpenteia pelos diversos cenários retratados no livro: o Vale do Silício, Maui, Boston e Cambridge. No fundo, porém, trata-se de um conto de duas cidades: Nova York, a capital editorial do mundo, e Sebastopol, na Califórnia, a capital editorial do condado de Sonoma.

A trama tem início em abril de 2000. Na época, eu escrevia reportagens para um portal de tecnologia de vida curta chamado `BeOpen.com`. Uma das minhas primeiras pautas foi uma entrevista por telefone com Richard M. Stallman. O bate-papo rendeu tão bem que o *Slashdot* (`slashdot.org`) — o lendário site de "notícias para nerds" pertencente à VA Software (antiga VA Linux Systems e, antes disso, VA Research) — destacou o link da matéria em

sua página principal. Em poucas horas, os servidores do BeOpen superaqueceram diante da enxurrada de cliques dos leitores.

Para todos os efeitos práticos, a história deveria ter se encerrado ali. Três meses após a entrevista, contudo, enquanto eu cobria a conferência *O'Reilly Open Source* em Monterey, na Califórnia, recebi na caixa de entrada a seguinte mensagem de Tracy Pattison, gerente de direitos internacionais de uma grande editora de Nova York:

> ***Para:*** `sam@BeOpen.com`
> ***Assunto:*** *Entrevista com RMS*
> ***Data:*** *Segunda-feira, 10 de julho de 2000, 15:56*
>
> *Prezado Sr. Williams,*
>
> *Li sua entrevista com Richard Stallman no BeOpen com enorme interesse. Venho acompanhando a trajetória e as ideias de RMS há algum tempo e fiquei encantada com seu texto; acho que você fez um trabalho admirável capturando a alma daquilo que Stallman está tentando construir com o GNU-Linux e a Free Software Foundation.*
>
> *O que eu adoraria fazer, no entanto, é ler mais a respeito — e tenho certeza de que não estou sozinha. Você acha que existem mais informações e fontes para expandir e atualizar sua entrevista, transformando-a em um perfil aprofundado de Stallman? Quem sabe incluindo mais detalhes anedóticos sobre a personalidade e o passado dele que pudessem interessar e cativar leitores de fora da bolha dos programadores mais radicais?*

Tracy encerrava o e-mail sugerindo que eu lhe desse um telefonema para amadurecer a ideia. Foi o que fiz. Tracy me explicou que sua editora estava inaugurando um selo de livros digitais (*e-books*) voltado para o público pioneiro de tecnologia. O formato padrão girava em torno de trinta mil palavras — cerca de cem páginas —, e ela havia vendido aos chefes a proposta de traçar o perfil de um ícone da cultura hacker. A cúpula aprovou a sugestão e, na garimpagem por personagens fascinantes, Tracy esbarrou na minha matéria com Stallman no BeOpen. Daí o contato.

Foi quando Tracy disparou a pergunta: eu estaria disposto a transformar aquela entrevista num livro de perfil biográfico?

Minha resposta foi instantânea: sim. Antes de bater o martelo, Tracy pediu que eu redigisse uma proposta estruturada para apresentar à diretoria. Dois dias depois, enviei o projeto polido. Uma semana mais tarde, veio a resposta triunfal: a cúpula dera sinal verde.

Preciso confessar: conseguir o aval de Stallman para participar de um projeto de livro digital parecia, na minha cabeça, um mero detalhe burocrático. Como repórter que cobria a cena do código aberto, eu sabia perfeitamente que Stallman era rigoroso ao extremo: àquela altura, eu já havia levado pelo menos meia dúzia de puxões de orelha por e-mail por ter cometido o pecado de escrever "Linux" em vez de "GNU/Linux".

Por outro lado, eu também sabia que Stallman vivia à caça de palanques para disseminar sua mensagem ao grande público leigo. Se eu apresentasse o projeto sob esse prisma generoso, imaginei que ele seria receptivo. Caso batesse

o pé, eu sempre poderia recorrer à montanha de documentos, palestras gravadas e entrevistas públicas que ele espalhara pela Internet e publicar uma biografia não autorizada.

Ao aprofundar minha pesquisa preliminar, deparei-me com um artigo intitulado "Liberdade — Ou Copyright?". Assinado pelo próprio Stallman e publicado na edição de junho de 2000 da *Technology Review* do MIT, o texto desancava os livros eletrônicos por uma sucessão de crimes contra a liberdade dos usuários. Não bastasse a exigência absurda de obrigar o leitor a rodar programas proprietários fechados para abrir as páginas, lamentava Stallman, os mecanismos de proteção contra cópias não autorizadas eram de uma brutalidade desmedida. Em vez de baixar um arquivo aberto e transferível em formato HTML ou PDF, o leitor baixava um pacote criptografado. Comprar um e-book, no fundo, resumia-se a adquirir uma chave intransferível para decodificar um conteúdo trancafiado. Qualquer tentativa de abrir o livro sem a bendita chave configurava crime federal sob a DMCA (*Digital Millennium Copyright Act*), a famigerada lei de 1998 desenhada para militarizar os direitos autorais no ciberespaço. Punições idênticas ameaçavam qualquer leitor que ousasse converter o texto em um formato aberto, mesmo que sua única intenção fosse lê-lo em outro computador de sua própria casa. Ao contrário do livro de papel tradicional, o leitor de e-books perdia o direito sagrado de emprestar, copiar ou revender a obra. Restava-lhe apenas a licença precária de lê-la em uma máquina autorizada pela corporação, alertava Stallman:

> *Ainda preservamos as velhas liberdades no uso de livros de papel. Mas se os e-books substituírem os volumes impressos, essa exceção de nada servirá. Com a "tinta eletrônica", que tornará possível baixar novos textos sobre uma folha aparentemente de papel, até os jornais do dia poderão virar efêmeros. Imagine só: o fim dos sebos; nunca mais poder emprestar um livro a um amigo; nunca mais pegar um exemplar emprestado na biblioteca pública — nenhuma brecha que permita a alguém ler sem pagar. (E, a julgar pelas propagandas do Microsoft Reader, nada de comprar livros anonimamente com dinheiro vivo). Esse é o admirável mundo novo que os barões editoriais planejam para todos nós.*[1]

Escusado dizer que a leitura do ensaio acendeu todas as minhas luzes de emergência. Nem Tracy nem eu havíamos discutido qual software a editora utilizaria, tampouco havíamos conversado sobre os termos de copyright e licenciamento que regeriam o e-book. Enviei o artigo da *Technology Review* a Tracy e perguntei quais seriam as diretrizes da editora. Ela prometeu consultar o departamento jurídico.

Ansioso para engatar o trabalho, resolvi ligar para Stallman assim mesmo e colocar as cartas na mesa. Ao ouvir a proposta, ele demonstrou interesse imediato e, no mesmo fôlego, uma preocupação aguda:

— Você leu o meu ensaio sobre livros digitais? — indagou ele.

Quando respondi que sim, que havia lido e que aguardava o parecer da editora, Stallman estabeleceu duas condições inegociáveis: ele não aceitaria dar suporte a um modelo de licenciamento de e-books ao qual se opunha por princípios fundamentais, nem aceitaria que seu nome fosse usado para chancelar tal mecanismo.

— Não participarei de nada que me faça parecer um hipócrita — sentenciou.

Para Stallman, a questão do software leitor era secundária perante o contrato de direitos autorais. Ele me disse que estava disposto a relevar o software que a editora viesse a usar, desde que ficasse explicitado no aviso de copyright que os leitores tinham a liberdade inalienável de produzir e distribuir cópias literais do conteúdo do livro. Stallman

citou o experimento recente de Stephen King com a novela seriada *The Plant* como modelo viável. Em junho de 2000, King anunciara em seu site oficial que publicaria a obra em capítulos digitais independentes: o custo total seria de 13 dólares, divididos em parcelas de 1 dólar por capítulo. Desde que pelo menos 75% dos leitores pagassem por cada episódio baixado, King prometia continuar escrevendo e liberando as continuações. Em agosto, a estratégia parecia dar certo, com os dois primeiros capítulos já no ar e o terceiro a caminho.

— Eu aceitaria algo desse tipo — ponderou Stallman. — Desde que também permitisse a cópia literal sem restrições.

> **[RMS:** Pelo que me recordo, também levantei expressamente a exigência de não haver criptografia proprietária; o texto dois parágrafos abaixo confirma isso. Eu jamais teria aceitado publicar um livro de forma que *exigisse* o uso de um programa proprietário para a leitura.**]**

Repassei as condições a Tracy. Confiante de que seríamos capazes de costurar um meio-termo aceitável, liguei para Stallman e agendei a primeira entrevista para a biografia. Ele concordou prontamente, sem exigir novas confirmações sobre o contrato. Empolgado com o resultado, corri para marcar a segunda rodada — desta vez presencial, em Kihei —, encaixando a conversa no Havaí antes de Stallman embarcar para catorze dias de férias no Taiti.

> **[RMS:** Não foram puramente férias; proferi uma palestra oficial lá também.**]**

Foi durante a viagem de Stallman ao Pacífico que a bomba estourou na caixa postal: o departamento jurídico da editora de Nova York recusava-se terminantemente a mexer numa única vírgula da licença de seus e-books. Leitores que quisessem transferir o texto teriam obrigatoriamente de quebrar a criptografia do arquivo para convertê-lo a um formato aberto e livre como HTML — ato taxado como crime federal passível de processo e cadeia pela DMCA.

Com duas longas entrevistas já gravadas, eu não via como escrever o livro sem utilizar aquele material inédito. Voei às pressas para Nova York a fim de me reunir com meu agente literário e com Tracy em busca de uma saída de emergência.

Ao desembarcar em Manhattan, encontrei meu agente, Henning Guttman. Era nosso primeiro encontro cara a cara, e Henning mostrou-se cético quanto à chance de dobrar a editora. Os grandes conglomerados do mercado já olhavam para o formato de e-books com profunda desconfiança e não estavam nem um pouco a fim de brincar com termos de copyright que estimulassem o público a não pagar. Como agente especializado em livros de tecnologia, no entanto, Henning achou a minha sinuca de bico fascinante. Contei-lhe sobre as gravações e a promessa moral feita a Stallman de não fazê-lo "parecer um hipócrita". Concordando que eu me encontrava preso num dilema ético, Henning sugeriu que usássemos esse argumento como ponta de lança na mesa de negociação.

Caso a editora batesse o pé, disse Henning, restaria apelar para a clássica política do morde-e-assopra. O agrado seria a publicidade positiva de publicar um e-book pioneiro que respeitava a ética sagrada da comunidade hacker; a bordoada seria o risco colossal de lançar um e-book que violava essa mesma ética. Nove meses antes de o

programador russo Dmitry Sklyarov virar o mártir supremo do ciberespaço, nós já tínhamos a certeza matemática de que seria apenas uma questão de semanas até que um hacker talentoso descobrisse como quebrar a trava daquele e-book. E sabíamos perfeitamente que uma grande editora nova-iorquina lançando um livro digital criptografado sobre a vida de Richard M. Stallman equivalia, na prática, a estampar um imenso adesivo de *"Roube Este Livro"* na capa.

Após a conversa com meu agente, liguei para Stallman. Na tentativa de tornar a proposta mais palatável, sugeri uma série de concessões: e se a editora publicasse o conteúdo sob um licenciamento duplo, algo similar ao que a Sun Microsystems fizera com o OpenOffice? A editora comercializaria a versão travada por DRM nos formatos fechados habituais, recheada de firulas interativas de design, e paralelamente disponibilizaria a versão de texto puro em formato HTML livre para cópia.

Stallman respondeu que não se opunha à ideia de licenciamento duplo, mas abominava a premissa de rebaixar a versão livre a um formato esteticamente inferior à versão proprietária. Além disso, acrescentou ele, aquele caso era muito diferente porque ele tinha o poder de controlar o desfecho: bastava recusar-se a colaborar.

> **[RMS:** A questão essencial era saber se seria eticamente errado de minha parte compactuar com a versão restrita. Eu posso aplaudir a versão livre do OpenOffice da Sun, porque ela é software livre e infinitamente melhor do que nada, ao mesmo tempo em que repúdio sua versão proprietária. Não há qualquer contradição nisso, porque a Sun não precisava nem me pediu autorização para lançar a versão fechada; eu não tinha qualquer responsabilidade pela existência dela. Neste caso do livro, porém, se eu dissesse "sim" para a versão não livre com travas, o fardo moral recairia inteiramente sobre as minhas costas.**]**

Arrisquei outras sugestões, sem sucesso. O máximo que consegui arrancar de Stallman foi uma concessão — uma nova flexibilização de sua parte — aceitando que a licença restringisse o compartilhamento exclusivamente à "redistribuição não comercial".

Antes de desligar, Stallman sugeriu que eu avisasse à editora que eu havia me comprometido com ele a lançar a obra como um texto livremente compartilhável. Respondi que não podia endossar essa declaração taxativa — muito embora ela fosse a expressão da verdade, uma vez que eu aceitara suas condições desde o início —, mas assegurei que considerava a biografia inviável sem o seu consentimento. Aparentemente apaziguado, Stallman despediu-se com seu jargão característico: *"Happy hacking"*.

Henning e eu nos sentamos com Tracy no dia seguinte. Tracy explicou que sua chefia aceitava publicar trechos livres em formato aberto e desprotegido, mas impunha um teto máximo de quinhentas palavras por citação. Henning rebateu de imediato que aquela esmola não bastava para honrar minha obrigação moral perante o biografado. Tracy colocou na mesa as amarras contratuais de sua editora com redes varejistas online, a começar pela gigante Amazon.com: mesmo que a editora quisesse abrir uma exceção lírica para Stallman, corria o risco de ser acionada judicialmente pelos distribuidores por quebra de contrato. Salvo uma súbita iluminação espiritual da diretoria da editora ou um recuo milagroso de Stallman, a decisão final repousava em minhas mãos: eu podia usar o material das entrevistas e rasgar o pacto com Stallman, ou invocar escrúpulos de ética jornalística e abortar a publicação do livro.

Terminada a reunião infrutífera, meu agente e eu nos refugiamos num pub na Terceira Avenida. Usei o celular dele para ligar para Stallman, deixando recado na secretária eletrônica. Henning afastou-se até o balcão por alguns instantes, deixando-me sozinho com meus pensamentos. Quando voltou, estendia o aparelho na minha direção:

— É o Stallman — sussurrou.

A conversa começou desastrosa desde o primeiro segundo. Transmiti a ponderação de Tracy sobre os contratos comerciais com a Amazon.

— E daí? — disparou Stallman, com a delicadeza de uma marreta. — Por que raios eu deveria dar a mínima para os contratos comerciais deles?

— Porque exigir que um dos maiores conglomerados editoriais do país encare uma batalha jurídica contra seus distribuidores por causa de um livro digital de trinta mil palavras é pedir a lua — argumentei.

> **[RMS:** A premissa velada de Williams era a de que seria impensável eu recusar um contrato daqueles por mero apego a princípios éticos.**]**

— Você não percebe? — rebateu Stallman. — É exatamente por isso que estou fazendo isso! Eu quero uma vitória emblemática. Quero forçá-los a escolher entre a liberdade humana e a ganância corporativa habitual!

Enquanto as palavras "vitória emblemática" ecoavam na minha cabeça, meus olhos se perderam no movimento dos pedestres na calçada lá fora. Ao entrar no bar, eu havia notado com um sorriso nostálgico que o pub ficava a menos de meia quadra da lendária esquina imortalizada na canção dos Ramones de 1976, "53rd and 3rd" — um clássico do punk que eu adorava tocar na guitarra nos meus tempos de músico. Assim como o garoto de programa angustiado daquela letra, sentia meu chão desmoronar na mesma velocidade em que tudo havia se erguido. A ironia da situação era avassaladora: após semanas registrando com gosto as agruras alheias, eu me encontrava na posição inglória de tentar arrancar a mais rara das façanhas da história da computação: uma concessão política de Richard Stallman. Quando comecei a titubear, gaguejando justificativas em favor da editora e deixando transparecer minha crescente simpatia pela posição da empresa, Stallman, farejando sangue no ar, partiu para o ataque:

— Então é isso? Você simplesmente vai me ferrar? Vai se curvar covardemente à vontade deles?

> **[RMS:** As citações mostram que a interpretação de Williams sobre este diálogo foi inteiramente equivocada. Ele me compara a um predador sanguinário, quando tudo o que eu fazia era dizer "não" a um acordo com o qual ele vinha me acossando para aceitar. Eu já havia feito diversas concessões importantes, descritas acima; o que recusei foi rasgar integralmente meus princípios. Costumo agir assim com frequência: as pessoas que não conseguem o que querem saem dizendo que "me recusei a ceder em qualquer coisa", mas isso é um manifesto exagero. O que temi naquele momento foi que ele ignorasse as condições prévias com as quais havia concordado e publicasse a minha biografia trancafiada com DRM à minha revelia. O que farejei não foi o "sangue" dele, mas o cheiro podre de uma traição ética.**]**

Tentei colocar novamente na mesa a tese do copyright duplo.

— Você quer dizer licença dupla — corrigiu Stallman, rispidamente.

— É, licença, copyright, tanto faz! — soltei, sentindo-me subitamente como um atum arpoado sangrando nas águas profundas.

— Droga, por que diabos você não fez o que eu mandei você fazer?! — berrou ele.

> **[RMS:** Considero essa fala totalmente distorcida: em primeiro lugar, porque praguejar dessa forma nunca fez parte dos meus hábitos linguísticos; em segundo, porque tais palavras não condizem com a realidade da situação. As palavras que ele cita soam como a bronca de um chefe furioso contra um subordinado desobediente. Eu achava que ele tinha um dever moral a honrar, mas ele não era meu funcionário, e eu jamais me dirigiria a ele nesses termos. Baseando-se em meras anotações mentais em vez de uma fita gravada, ele não tinha como reproduzir com fidelidade o que foi dito.**]**

Eu devia estar advogando em favor da editora até o último suspiro, pois guardei em meu bloco de notas o tiro de misericórdia desferido por Stallman: "Não me interessa. O que eles estão fazendo é abominável. Não posso ser cúmplice do mal. Adeus."

> **[RMS:** Parece evidente que percebi que ele jamais aceitaria um não como resposta, e a única forma de encerrar a conversa sem ceder à proposta indecente era bater o telefone na cara dele.**]**

No instante em que pousei o aparelho sobre a mesa, meu agente deslizou uma caneca transbordante de chope Guinness na minha direção:

— Achei que você fosse precisar disto — riu Henning. — Dava para ver suas mãos tremendo daqui.

Eu estava tremendo, e muito. A tremedeira só começou a passar quando a cerveja escura chegou à metade da caneca. Ouvir-me rotulado como emissário do "mal" provocava uma sensação esquisita no estômago.

> **[RMS:** Minhas palavras, tais como citadas, criticavam a editora, não o Williams pessoalmente. Se ele vestiu a carapuça, talvez seja sinal de que sua consciência pesou diante do pacto indecente que tentava me empurrar goela abaixo.**]**

Mais bizarro ainda era lembrar que, três meses antes, eu estava pacatamente sentado em meu apartamento em Oakland pensando em qual seria minha próxima pauta de jornalismo. Agora, encontrava-me encurralado num pedaço de Manhattan que eu só conhecia pelas letras de rock, participando de reuniões executivas com tubarões do mercado editorial e bebendo cerveja com um agente literário que eu acabara de conhecer na véspera. O roteiro era surreal demais, quase como assistir à minha própria vida desdobrar-se num clipe de cinema.

Foi nessa hora que meu detector interno de absurdo disparou: o tremor nas mãos cedeu lugar a uma crise incontrolável de riso. Para o meu agente, devo ter parecido mais um autor desequilibrado à beira de um colapso emocional. Para mim, tratava-se apenas de apreciar a beleza cínica da minha encruzilhada. Com contrato ou sem contrato, eu já tinha em mãos uma história fascinante; a única questão era encontrar a vitrine certa para contá-la. Quando as gargalhadas finalmente cessaram, ergui minha caneca num brinde triunfal:

— Bem-vindo ao front de batalha, meu amigo — disse, estalando o vidro no copo de Henning. — O melhor que podemos fazer é nos divertir.

Se esta crônica fosse de fato uma peça de teatro, aqui começaria um breve e doce interlúdio romântico. Comovida com a tensão desgastante da nossa reunião matinal, Tracy convidou Henning e eu para bebermos algo com ela e alguns colegas de trabalho no fim daquela tarde. Deixamos o bar da Terceira Avenida, pegamos o metrô rumo ao East Village e nos juntamos à turma de Tracy.

Lá chegando, engatei uma conversa longa com ela, tomando o cuidado meticuloso de não tocar em uma única palavra sobre livros ou contratos. O papo fluiu leve, descontraído e gostoso. Antes de nos despedirmos na madrugada nova-iorquina, combinamos um jantar a dois para a noite seguinte. Mais uma vez, a química foi perfeita — tão perfeita que o projeto de biografia de Stallman virou quase uma lembrança remota no retrovisor.

Ao aterrissar de volta em Oakland, telefonei para amigos e conhecidos do meio jornalístico relatando meu dilema ético. A maioria desancou minha ingenuidade por ter cedido terreno demais a Stallman antes das entrevistas.

> **[RMS:** Aqueles que leram o livro inteiro sabem muito bem que eu jamais teria aberto mão daquelas condições.**]**

Um antigo professor da faculdade de jornalismo aconselhou-me a ignorar o chilique de Stallman sobre "hipocrisia" e simplesmente publicar a matéria. Colegas veteranos que conheciam a malícia midiática de Stallman solidarizaram-se com a minha situação, mas cravaram a mesma resposta: a decisão era exclusivamente minha.

Decidi engavetar o livro. Mesmo com as fitas gravadas, eu não conseguia avançar uma única linha no texto. Além do mais, a trégua me dava o pretexto perfeito para conversar com Tracy sem precisar filtrar as palavras pelo meu agente literário. No Natal, já havíamos trocado visitas: ela voou para a Costa Oeste uma vez; eu voei de volta para Nova York em seguida. Na véspera do Ano-Novo, pedi sua mão em casamento. Na hora de escolher em qual costa iríamos morar, escolhi Nova York. Em fevereiro, guardei meu laptop e todos os cadernos de anotações sobre Stallman na mala e pousei de mala e cuia no aeroporto JFK. Tracy e eu subimos ao altar no dia 11 de maio. Nada mal para uma negociação fracassada de livro.

No verão de 2001, comecei a amadurecer a ideia de transformar minhas anotações em uma reportagem de fôlego para alguma revista. Sob o prisma ético, minha consciência estava limpa, pois o trato original não dizia uma linha sobre publicações tradicionais de papel. Para ser sincero, eu também me sentia infinitamente mais seguro para escrever sobre Stallman após oito meses de silêncio absoluto. Desde o bate-boca no pub em setembro anterior, eu só havia recebido duas mensagens dele — ambas cobrando-me asperamente por ter grafado "Linux" em vez de "GNU/Linux" em matérias para a revista online *Upside Today*. Fora isso, aquele silêncio fora uma dádiva. Em junho, dias após a conferência de Stallman na Universidade de Nova York, sentei-me à máquina e rascunhei um

perfil de cinco mil palavras. Desta vez, o texto fluiu com facilidade assombrosa: o distanciamento me devolvera a perspectiva emocional necessária.

Em julho de 2001 — cravando exatamente um ano desde aquele primeiro e-mail de Tracy —, recebi um telefonema de Henning: a editora O'Reilly & Associates, sediada em Sebastopol, na Califórnia, estava vivamente interessada em lançar o projeto como uma biografia completa em livro.

> **[RMS:** Guardo uma lembrança vaga de que fui eu quem sugeriu procurar a O'Reilly, mas não tenho certeza absoluta depois de tantos anos.**]**

A notícia me encheu de entusiasmo. De todas as editoras do planeta, a O'Reilly — a mesma casa editorial que publicara o clássico *The Cathedral and the Bazaar* de Eric Raymond — parecia ser a mais sensível às bandeiras éticas que haviam implodido o projeto original. Como jornalista, eu usara fartamente a antologia *Open Sources* da O'Reilly como manual de referência histórica. E sabia muito bem que diversos capítulos daquele livro — inclusive o assinado pelo próprio Stallman — haviam sido publicados com avisos explícitos de licenciamento garantindo o direito de livre redistribuição. Esse precedente seria a nossa tábua de salvação caso o fantasma da publicação digital voltasse a assombrar a mesa.

E o fantasma, com efeito, não tardou a reaparecer. Soube através de Henning que a O'Reilly pretendia publicar a biografia tanto no formato impresso tradicional quanto dentro de seu recém-lançado serviço de assinaturas digitais, o *Safari Tech Books Online*. A licença de uso do Safari embutia restrições específicas,[2] advertiu Henning, mas a O'Reilly estava perfeitamente disposta a carimbar na obra uma licença que garantisse ao leitor a liberdade de copiar e compartilhar o texto integral em qualquer meio. Na prática, como autor, eu tinha a liberdade de escolher entre duas licenças livres: a *Open Publication License* (OPL) ou a *GNU Free Documentation License* (GFDL).

Fui pesquisar a fundo o histórico e as entrelinhas de cada documento. A *Open Publication License* concedia ao público o direito de reproduzir e distribuir a obra, no todo ou em parte, em qualquer formato ("físico ou eletrônico"), desde que a cópia retivesse a mesma licença OPL. Permitia também a criação de obras derivadas, sob condições razoáveis. Por fim, a OPL trazia cláusulas opcionais que, se assinaladas pelo autor original, podiam proibir a circulação de versões "substancialmente modificadas" ou impedir a comercialização de livros impressos derivados sem anuência prévia.[3]

A *GNU Free Documentation License*, por sua vez, assegurava o direito irrevogável de copiar e distribuir o documento em qualquer mídia, desde que a nova obra trouxesse a mesmíssima licença.[4] Permitia igualmente a modificação do texto, sob parâmetros claros. Diferentemente da OPL, contudo, a licença do GNU não dava ao autor a prerrogativa de vetar modificações substantivas nem o direito de barrar versões modificadas que concorressem comercialmente com o original em formato de livro impresso. Exigia apenas a preservação de determinadas informações de capa e quarta-capa caso terceiros decidissem imprimir mais de cem cópias de uma versão modificada.

Nessa busca minuciosa, fiz questão de visitar a página do Projeto GNU dedicada à análise de licenças públicas.[5] Ali, encontrei uma crítica incisiva de Stallman à Open Publication License: ele alertava que a possibilidade de o

autor marcar caixinhas vetando alterações feria a integridade da liberdade. Caso o escritor não desejasse restringir tais modificações, o mais prudente e honesto era adotar de uma vez a GFDL, eliminando o risco de que restrições autoritárias brotassem sorrateiramente em derivações futuras.

A prioridade concedida à liberdade de modificação em ambas as licenças decorria de seu propósito primordial: assegurar que os leitores de manuais técnicos de software tivessem o poder de corrigir e aprimorar a documentação, compartilhando esses ganhos com a coletividade. Como meu livro era uma biografia narrativa e não um manual de compilador, a questão de alterações de terceiros pouco me tirava o sono: meu único compromisso era assegurar aos leitores a liberdade sagrada de emprestar, copiar e repassar o texto — a mesmíssima liberdade secular que eles sempre desfrutaram ao comprar um livro físico numa livraria. Julgando ambas as licenças perfeitamente adequadas para esse fim, assinei o contrato da O'Reilly assim que ele bateu na minha mesa.

Ainda assim, a perspectiva de permitir modificações irrestritas atiçou minha veia imaginativa. Nas conversas iniciais com Tracy, eu já havia defendido o brilhantismo de adotar uma licença estilo GPL para o conteúdo. Na pior das hipóteses, argumentara eu, o gesto garantiria uma propaganda monumental para o lançamento; no melhor dos cenários, encorajaria o público a participar ativamente da feitura da obra. Como autor, eu não via problema algum em ver outros complementarem meu trabalho, desde que meu nome permanecesse no topo dos créditos. Além disso, seria instigante ver o texto se transformar ao longo dos anos: vislumbrei edições futuras parecidas com o *Talmud*, com o meu texto original ocupando uma coluna central cercado por iluminuras e comentários de terceiros nas margens.

Minha inspiração bebia na fonte do lendário *Projeto Xanadu*, o conceito precursor de hipertexto concebido por Ted Nelson em 1960. Durante a conferência da O'Reilly de 1999, eu assistira à primeira demonstração pública do *Udanax* — a versão livre do Xanadu — e ficara boquiaberto. Numa das telas, o programa exibia um documento-matriz e uma obra derivada lado a lado; com um clique do mouse, o software desenhava linhas dinâmicas conectando cada frase do original ao seu desdobramento conceitual no texto derivado.[6] Uma biografia digital de Richard M. Stallman não precisava necessariamente rodar no Udanax, mas diante de horizontes tecnológicos tão fascinantes, por que não convidar os leitores a experimentar?

Quando Laurie Petrycki, minha editora na O'Reilly, me ofereceu a escolha formal entre a OPL e a GFDL, dei asas novamente à fantasia. Em setembro de 2001, mês em que firmei o contrato, o mercado de e-books havia praticamente virado cinzas: quase todas as grandes editoras — inclusive a de Tracy — estavam sepultando seus selos digitais por absoluta falta de público. Fiquei me perguntando: se aquelas corporações tivessem encarado os livros eletrônicos não como mercadorias enjauladas para cobrança automática, mas como instrumentos vivos de construção comunitária, teriam seus projetos sucumbido de forma tão melancólica?

Com o contrato devidamente assinado, avisei a Stallman que a biografia estava de pé novamente. Contei sobre a encruzilhada oferecida pela O'Reilly entre a OPL e a GNU Free Documentation License. Disse que me inclinava pela OPL, até porque não via motivo para dar de bandeja a concorrentes da O'Reilly a chance de imprimir o mesmo livro sob outra capa. Stallman respondeu de volta, rebatendo meus argumentos e defendendo a adoção límpida da GFDL, lembrando que a própria O'Reilly já a utilizara diversas vezes no passado. Apesar do histórico de atritos, propus um trato: eu escolheria a GFDL se ele me concedesse novas entrevistas presenciais e concordasse em ajudar a O'Reilly nos eventos de divulgação da obra. Stallman aceitou as entrevistas de bom grado, mas ponderou com honestidade que seu engajamento na promoção do livro dependeria visceralmente daquilo que eu

escrevesse nas páginas. Encarando a resposta como justa, agendei a rodada decisiva de entrevistas para o dia 17 de dezembro de 2001, em Cambridge.

Organizei a viagem para coincidir com compromissos de trabalho que minha esposa Tracy tinha agendados em Boston naquela mesma semana. Dois dias antes de pegarmos a estrada, Tracy sugeriu que convidássemos Stallman para um jantar:

— Afinal de contas — brincou ela —, foi ele quem nos uniu.

Enviei um e-mail para Stallman, que aceitou o convite quase que instantaneamente. Quando cheguei a Boston, encontrei Tracy em seu hotel e pegamos a linha vermelha do metrô em direção ao campus do MIT. Ao batermos à porta de sua sala no complexo da Technology Square, encontramos Stallman no meio de uma conversa animada:

— Espero que vocês não se importem — disse ele, entreabrindo a porta o suficiente para entrevermos sua interlocutora: uma moça jovem, na casa dos 20 anos, chamada Sarah.

— Tomei a liberdade de convidar mais uma pessoa para jantar conosco — anunciou Stallman com toda a naturalidade do mundo, estampando o mesmo sorrisinho enigmático de gato que me mostrara tempos atrás naquele restaurante em Palo Alto.

Para ser sincero, a surpresa não foi total: o boato de que Stallman tinha uma nova companhia feminina havia chegado aos meus ouvidos semanas antes, cortesia de sua mãe: "Inclusive viajaram juntos para o Japão no mês passado, quando Richard foi receber o Prêmio Takeda", contara-me Alice Lippman na época.[7]

No caminho até o restaurante, soube como Sarah e Richard haviam se conhecido — e o roteiro me pareceu curiosamente familiar. Escritora em início de carreira trabalhando em seu primeiro romance de ficção, Sarah ouvira falar da figura exótica de Stallman e decidira criar um personagem inspirado nele. Em nome da pesquisa de laboratório, agendou uma entrevista com ele. O papo engrenou, a faísca acendeu e os dois estavam saindo juntos desde o início de 2001.

— Fiquei profundamente fascinada com a forma como Richard foi capaz de construir um movimento político mundial gigantesco para enfrentar um dilema de angústia puramente pessoal — revelou Sarah, explicando o que a atraíra em Stallman.

Minha esposa disparou a pergunta na lata:

— E qual era o dilema?

— Uma solidão esmagadora.

Durante o jantar, deixei as mulheres conduzirem a conversa, gastando a maior parte do tempo tentando captar se os últimos doze meses haviam amolecido a carapaça de Stallman de alguma maneira relevante. Não encontrei o menor indício de amolecimento: embora demonstrasse um galanteio tímido inédito, Stallman exibia a mesmíssima aspereza de sempre. Em dado momento, quando minha esposa soltou um espontâneo "Deus me livre!", recebeu na hora uma correção típica de RMS:

— Odeio ter que estragar a sua ilusão, mas Deus não existe — cortou ele, impassível.

> **[RMS:** Eu devia estar com uma expressão séria demais. Williams poderia com justiça me acusar de estar bancando o engraçadinho metido a sabichão, mas nunca de estar repreendendo a esposa dele.**]**

Mais tarde, encerrado o jantar e após Sarah se despedir, Stallman pareceu desarmar ligeiramente a guarda. Enquanto caminhávamos rumo a uma livraria próxima, confessou que os últimos doze meses haviam transformado radicalmente sua visão sobre a vida:

— Eu tinha certeza absoluta de que passaria o resto dos meus dias sozinho — murmurou ele. — Estou muito feliz por ter errado.

Antes de nos despedirmos na esquina, Stallman entregou-me seu curioso "cartão de visitas para prazeres": um cartãozinho pessoal onde figuravam seu endereço, telefone e sua lista de passatempos favoritos (*"compartilhar bons livros, boa comida, dança e músicas exóticas"*), para combinarmos nossa última sessão de entrevistas.

Na manhã seguinte, diante de mais uma farta rodada de *dim sum*, Stallman parecia ainda mais apaixonado do que na véspera. Relembrando os debates acalorados nos alojamentos da Currier House sobre os prós e contras de um suposto soro da imortalidade, ele confessou o desejo de que a ciência um dia decifrasse o enigma do rejuvenescimento biológico:

— Agora que finalmente comecei a encontrar a felicidade na vida, tenho vontade de viver muito mais tempo — desabafou.

Quando toquei no comentário de Sarah sobre a "solidão esmagadora", Stallman a princípio não enxergou qualquer ligação entre a carência afetiva no plano individual e a solidão vivenciada na esfera hacker: "O impulso de compartilhar código tem a ver com camaradagem, mas é uma camaradagem num patamar bem mais abstrato", desconversou. Minutos depois, contudo, quando o tema voltou à tona, Stallman admitiu que o isolamento social — ou o pavor do isolamento perpétuo no universo comunitário dos programadores — fora o grande combustível por trás de sua obstinação de ferro nos primeiros dias do Projeto GNU:

— Meu fascínio por computadores não foi consequência de carência alguma — ponderou ele. — Eu não teria sido menos fascinado por programação se tivesse sido o garoto mais popular do colégio ou se as mulheres fizessem fila atrás de mim. No entanto, é rigorosamente verdade que a dor de não ter um lar, de finalmente encontrar um e perdê-lo logo em seguida, e depois encontrar outro e vê-lo destruído diante dos meus olhos me marcou profundamente. O lar que perdi foi o dormitório da faculdade; o lar que vi ser exterminado foi o Laboratório de Inteligência Artificial do MIT. A fragilidade de se ver repentinamente desprovido de qualquer chão ou comunidade foi uma experiência avassaladora. Foi ela que me deu forças para lutar sem trégua para reconquistar esse solo.

Ao término daquela última entrevista, foi impossível não sentir uma tocante sensação de harmonia e simetria emocional. Ouvindo Sarah decodificar a centelha que a atraíra até Richard, e ouvindo o próprio Stallman desnudar as feridas íntimas que o impeliram a abraçar a bandeira do software livre, reencontrei as motivações essenciais que me levaram a escrever este livro. Desde julho de 2000, aprendi a conviver tanto com os encantos sedutores quanto com as arestas cortantes da personalidade de Richard Stallman. Como bem sentenciou Eben Moglen antes de

mim, cometeríamos um erro crasso e imperdoável se descartássemos essas arestas como meras excentricidades supérfluas diante da epopeia do software livre: sob múltiplos aspectos, o homem e a causa se entrelaçam de forma tão indissociável que é absolutamente impossível separar um do outro.

> **[RMS:** Williams projeta suas próprias reações emocionais, tanto as simpatias quanto as antipatias, como se fossem facetas intrínsecas da minha personalidade; contudo, elas são reflexos diretos de seus próprios valores sobre padrões de aparência física, convenções sociais e sucesso empresarial.**]**

Embora saiba perfeitamente que nem todos os leitores nutrirão simpatia instantânea por Stallman, tenho a convicção inabalável de que quase todos concordarão que pouquíssimos seres humanos oferecem um retrato tão singular, comovente e grandioso quanto Richard M. Stallman. É minha sincera esperança que, com este retrato inaugural concluído e com o escudo libertador da GFDL, outros companheiros sintam o mesmo chamado irresistível para acrescentar seus próprios traços, cores e perspectivas a esta tela viva.

Notas

Ver Richard M. Stallman, "Freedom — Or Copyright?", *Technology Review* (MIT, maio/junho de 2000). Disponível em: http://www.technologyreview.com/articles/stallman0500.asp. ↩

Ver "Safari Tech Books Online; Subscriber Agreement: Terms of Service", arquivado em http://my.safaribooksonline.com/termsofservice. Em dezembro de 2009, esses livros exigiam softwares leitores proprietários restritivos, razão pela qual qualquer usuário consciente deveria recusar-se terminantemente a utilizá-los. ↩

Ver "The Open Publication License: Draft v1.0" (8 de junho de 1999), disponível em: http://opencontent.org/openpub/. ↩

Ver "The GNU Free Documentation License: Version 1.3" (novembro de 2008), disponível em: http://www.gnu.org/copyleft/fdl.html. ↩

Ver a seção oficial "Various Licenses and Comments About Them" no portal do Projeto GNU: http://www.gnu.org/philosophy/license-list.html. ↩

Qualquer programador disposto a portar este livro para o formato do Udanax — a versão livre do Xanadu — contará com meu apoio irrestrito e entusiasmado. Para conhecer mais sobre essa intrigante tecnologia, visite: http://www.udanax.com. ↩

Infelizmente, o autor só tomou conhecimento da decisão da Fundação Takeda de conceder a Stallman, ao lado de Linus Torvalds e Ken Sakamura, seu primeiro prêmio por "Conquistas Tecno-Empreendedoras para o Bem-Estar Social/Econômico" após o retorno da comitiva do Japão. O prêmio distribuía 1 milhão de dólares entre os laureados. Mais informações em: http://www.takeda-foundation.jp. ↩

Apêndice A: Hack, Hackers e Hacking

Para compreender a plenitude do significado da palavra "hacker", faz-se necessário examinar a etimologia do termo ao longo das décadas.

The New Hacker's Dictionary, um célebre compêndio online do jargão dos programadores de computador, lista oficialmente nove conotações distintas para a palavra *hack* e um número semelhante para *hacker*. Não obstante, a mesma publicação traz um ensaio assinado pelo veterano do MIT Phil Agre, que adverte os leitores a não se deixarem iludir por essa aparente maleabilidade semântica: "*Hack* tem um único e exclusivo significado", argumenta Agre. "Um sentido sutil e profundo que desafia qualquer articulação verbal." Richard Stallman procura sintetizá-lo em uma fórmula lapidar: *"esperteza travessa"* (*playful cleverness*).

Independentemente de a definição ser ampla ou restrita, a esmagadora maioria dos hackers modernos traça a certidão de nascimento do termo até o MIT, onde a palavra emergiu como gíria estudantil no início dos anos 1950. Em 1990, o Museu do MIT publicou uma revista comemorativa documentando o fenômeno do *hacking*. Segundo os registros históricos, os estudantes que frequentavam o instituto na década de 1950 usavam o termo *hack* da mesma forma que um universitário de hoje usaria palavras como "molecagem", "tirada engenhosa" ou "zoeira saudável". Pendurar uma carcaça de automóvel enferrujada pela janela do dormitório era um *hack*; já qualquer ato destrutivo ou maldoso — como atirar ovos nas janelas de um prédio rival ou pichar uma estátua do campus — era liminarmente excluído dessa categoria. Embutido no conceito de *hack* residia, desde o berço, um espírito inegociável de diversão criativa e inofensiva.

Esse espírito logo inspirou a forma verbal no gerúndio: *hacking*. Um estudante dos anos 1950 que passasse a maior parte da tarde proseando ao telefone ou desmontando pacientemente as entranhas de um aparelho de rádio descreveria seu passatempo como *hacking* — o equivalente ao que hoje se chamaria de "fuçar", "mexericar" ou "passar o tempo aprontando".

Com o avanço dos anos 1950, o vocábulo ganhou contornos mais agudos e rebeldes. O ambiente acadêmico do MIT daquela época era asfixiantemente competitivo, e o *hacking* surgiu tanto como reação quanto como válvula de escape contra aquela pressão. Travessuras elaboradas e trotes geniais viraram a melhor forma de extravasar o estresse, desafiar as autoridades do campus e exercitar um pensamento criativo e não convencional, frequentemente sufocado pelo currículo rígido de graduação. Com seu labirinto de corredores e túneis subterrâneos de vapor, o instituto oferecia um campo fértil de exploração para qualquer aluno que não se deixasse intimidar por portas trancadas e placas de "Proibida a Entrada". Os estudantes passaram a batizar essas incursões clandestinas de *tunnel hacking*. Na superfície, o sistema de telefonia interna da universidade abria oportunidades semelhantes: com experimentos informais e muita observação meticulosa, os alunos aprenderam a aplicar truques e comandos inusitados na rede telefônica. Inspirando-se no desbravamento dos túneis subterrâneos, logo apelidaram essa novidade de *phone hacking*.

Essa dobradinha entre brincadeira criativa e exploração livre de amarras serviu como alicerce para as mutações semânticas seguintes. Os primeiros pioneiros a se autointitularem "hackers de computador" no campus do MIT no início dos anos 1960 originaram-se de um clube de ferromodelismo fundado no fim dos anos 1950: o *Tech Model Railroad Club* (TMRC). Dentro do clube operava uma confraria fechada: o Comitê de Sinais e Energia (*Signals*

and Power Committee — S&P), o grupo de gênios responsáveis pelo complexo sistema de circuitos elétricos da ferrovia em miniatura. A maquete era controlada por uma teia sofisticada de relés e chaves eletromecânicas semelhante à central telefônica do campus. Para acionar os trilhos, o membro da equipe simplesmente discava códigos num telefone acoplado ao painel e via as locomotivas obedecerem instantaneamente às suas ordens.

Aqueles jovens aspirantes a engenheiros eletricistas encaravam a fiação da ferrovia com o mesmíssimo espírito do *phone hacking*. Ao adotarem a expressão para si, deram-lhe um polimento adicional: do ponto de vista de um hacker do S&P, conseguir economizar um relé na automação de um determinado trecho da linha significava ter um relé a mais sobrando para inventar brincadeiras novas no futuro. Assim, a palavra *hacking* migrou sutilmente de mero passatempo desocupado para uma atividade lúdica que, ao mesmo tempo, aperfeiçoava o desempenho e a eficiência do sistema como um todo. Não demorou para que os integrantes do comitê passassem a se referir com indisfarçável orgulho a qualquer melhoria ou remodelagem dos circuitos como um *hack*, chamando a si mesmos de *hackers*.

Dada a sua atração irresistível por circuitos sofisticados — somada à tradicional indiferença dos alunos do MIT por portas trancadas e avisos burocráticos —, bastou pouco tempo para que os hackers sentissem o cheiro de uma máquina inédita que acabava de aterrissar no campus: o TX-0, um dos primeiros computadores transistorizados comercializados no mundo. Ao término da década de 1950, quase toda a turma do S&P migrou em bloco para a sala de controle do TX-0, levando consigo todo o espírito da travessura criativa. O vasto e inexplorado oceano da programação de computadores estimulou uma nova mutação etimológica: "hackear" já não significava soldar componentes exóticos numa placa, mas sim costurar programas de software ignorando métodos acadêmicos engessados e procedimentos formais enfadonhos. Significava também espremer até a última gota de velocidade e eficiência de rotinas preexistentes que teimavam em desperdiçar os preciosos recursos da máquina. E, sendo fiel às raízes da palavra, significava também conceber programas sem nenhuma utilidade prática além de divertir, encantar e arrancar gargalhadas.

Um monumento clássico dessa ampliação semântica é o lendário jogo *Spacewar!*, o primeiro videogame de computador da história. Desenvolvido por hackers do MIT no início dos anos 1960, o *Spacewar!* reunia todos os predicados de um *hack* autêntico: era lúdico, imprevisível e desprovido de qualquer propósito utilitário, servindo apenas para entreter as madrugadas de uma dúzia de aficionados. Sob a ótica da engenharia de software, contudo, representava um prodígio formidável de virtuosismo e inovação na programação. Além disso, era inteiramente livre: tendo criado o jogo por pura diversão, os hackers não viam sentido algum em trancafiar sua criação, compartilhando o código abertamente com colegas de outros centros de pesquisa. Ao final da década de 1960, o *Spacewar!* já era o passatempo favorito de programadores em diversos cantos do planeta que tivessem o privilégio de contar com um display gráfico (uma raridade naqueles tempos).

Essa consagração da autoria coletiva e da posse comunitária do software distanciou o *hacking* de computadores dos anos 1960 das aventuras individuais de exploração de túneis e telefones da década anterior. Estas últimas eram atividades solitárias ou praticadas em grupelhos conspiratórios; dependiam de segredos transmitidos aos sussurros e, por sua natureza proibida, desencorajavam a circulação pública das descobertas. Já os hackers de computador exerciam seu ofício no coração de uma disciplina científica efervescente, vocacionada para a colaboração acadêmica e o reconhecimento do mérito. Hackers e cientistas da computação formais nem sempre foram os aliados mais pacíficos, mas, diante da expansão vertiginosa da informática, ambas as espécies estabeleceram uma relação de simbiose e cooperação mútua.

Os hackers tinham profundo desprezo pelos regulamentos arbitrários dos burocratas. Encaravam as travas de segurança e as senhas de computadores que impediam o acesso livre às máquinas simplesmente como mais um defeito no sistema — um bug incômodo a ser contornado ou corrigido. Por essa razão, quebrar travas de segurança (nunca com intuitos nocivos) já era uma prática reconhecida do *hacking* em 1970, utilizada tanto para trotes engraçados (quando a vítima exclamava com bom humor: "Acho que alguém está me hackeando!") quanto para abrir caminho até o computador. Mas isso nunca foi o coração do *hacking*. Quando havia um obstáculo autoritário de segurança, os hackers tinham orgulho de exibir sua astúcia para superá-lo; contudo, quando tinham a chance de escolher — como ocorria no Laboratório de Inteligência Artificial do MIT —, preferiam simplesmente não ter trava alguma e gastar suas energias em hacks muito mais nobres e criativos. Onde não há fechaduras arbitrárias, ninguém precisa perder tempo arrombando portas.

É prova irrefutável do talento descomunal dos hackers pioneiros que as gerações seguintes de programadores — a começar pelo jovem Richard M. Stallman — almejassem com ardor envergar o mesmo manto de honra. Entre meados e o final dos anos 1970, o substantivo "hacker" já adquirira uma conotação aristocrática de virtuosismo. Em sentido amplo, hacker de computador era qualquer pessoa que escrevesse código pelo simples prazer visceral de programar. Em sentido estrito, contudo, o termo atestava a excelência suprema do ofício. Assim como o vocábulo "artista", a designação carregava um sentido tribal de consagração: chamar um colega de programador de "hacker" era a maior deferência e demonstração de respeito que se podia prestar; definir a si mesmo como hacker era prova de coragem e imensa autoconfiança técnica. De todo modo, a informalidade original do termo foi se afunilando à medida que os computadores deixavam de ser raridades de laboratório.

Conforme a definição se refinava, o universo hacker incorporou outros traços culturais bastante específicos. Os programadores do Laboratório de IA do MIT compartilhavam hábitos peculiares: adoração por comida chinesa, repulsa visceral à fumaça de tabaco e aversão a bebidas alcoólicas, cigarros e substâncias entorpecentes que pudessem turvar a lucidez do raciocínio lógico. Tais idiossincrasias passaram a fazer parte do imaginário coletivo sobre o que significava ser um hacker, e a comunidade exercia uma influência silenciosa sobre os recém-chegados, muito embora jamais exigisse conformismo de ninguém. Essas peculiaridades comportamentais, no entanto, diluíram-se com a extinção da velha comunidade do laboratório; hoje, a maioria dos hackers reflete os mesmos costumes da sociedade em que vive.

À medida que os hackers de templos de excelência como MIT, Stanford e Carnegie Mellon debatiam os hacks que mais admiravam, passaram também a refletir sobre a ética de suas próprias ações. Começou-se a falar abertamente de uma **"ética hacker"**: o código de conduta não escrito que norteava o dia a dia daqueles artesãos do silício. Em seu célebre livro de 1984, *Hackers: Heroes of the Computer Revolution*, o escritor Steven Levy organizou e formalizou essas diretrizes éticas em princípios fundamentais que regiam a vida comunitária.

Na década de 1980, com a popularização maciça dos computadores pessoais, os incidentes de invasão de sistemas de segurança explodiram. Na maioria esmagadora das vezes, essas invasões eram perpetradas por criminosos comuns ou funcionários mal-intencionados de empresas que nada tinham a ver com a cultura hacker. Contudo, em algumas ocasiões, a polícia e administradores burocráticos — que enxergavam qualquer desobediência como um crime hediondo — rastrearam invasões digitais até jovens entusiastas cuja única regra de conduta era "não prejudicar ninguém". Repórteres de jornais e revistas passaram a estampar manchetes sensacionalistas em que "hackear" virou sinônimo de invadir computadores alheios, encampando acriticamente a versão autoritária das corporações. Embora obras como a de Steven Levy tenham documentado com maestria o espírito científico pioneiro que deu origem à contracultura dos computadores, na cabeça da maioria dos repórteres e leitores de jornal a expressão "hacker de computador" tornou-se erroneamente sinônimo de "arrombador digital".

Ao final da década de 1980, legiões de adolescentes americanos já tinham computadores em seus quartos. Muitos deles sentiam-se marginalizados ou entediados; inspirados pela imagem distorcida e glamourizada veiculada pela imprensa, canalizaram sua rebeldia quebrando senhas e invadindo redes alheias — do mesmo modo que outros jovens rebeldes atiravam pedras em vidraças. Esses garotos passaram a se autodenominar "hackers", sem jamais terem tomado contato com o princípio basilar dos veteranos do MIT de jamais causar dano deliberado a quem quer que seja. Conforme essa ala jovem passou a utilizar suas habilidades técnicas para fins destrutivos — disseminando vírus digitais, desfigurando servidores por travessura ou derrubando sistemas corporativos —, a palavra "hacker" ganhou uma aura niilista e rebelde que passou a atrair cada vez mais pessoas com a mesma mentalidade inconsequente.

Os hackers autênticos vêm protestando veementemente contra esse sequestro semântico há décadas. Richard Stallman, sempre pronto a lutar pela integridade conceitual das ideias, cunhou o termo **"cracking"** para definir especificamente o ato de quebrar travas de segurança de sistemas (*security breaking*), permitindo que o público pudesse abandonar de vez o uso indevido da palavra *hacking*.

Contudo, a fronteira entre *hacking* e *cracking* costuma ser mal compreendida. Esses dois termos não foram desenhados como categorias mutuamente excludentes, como se fossem dois polos estanques num mapa. *Hacking* e *cracking* são atributos descritivos de atividades distintas, do mesmo modo que "jovem" e "alto" são características independentes de uma pessoa.

A esmagadora maioria das atividades de *hacking* nada tem a ver com segurança de sistemas; portanto, não constitui *cracking*. Por outro lado, a maior parte do *cracking* moderno é praticada com intuitos de lucro fácil, espionagem corporativa ou maldade deliberada, inteiramente desprovida de qualquer espírito de curiosidade lúdica; logo, não se qualifica como *hacking*. Em raras ocasiões históricas, um ato singular pode merecer simultaneamente os dois rótulos, mas isso está longe de ser a regra. O espírito hacker traz em seu DNA uma irreverência inata contra regras arbitrárias, mas quase todos os hacks brilhantes jamais violam qualquer lei. O *cracking*, por definição, é um ato de desobediência a um sistema, mas nem todo *cracking* é necessariamente perverso ou destrutivo: a própria indústria de segurança da informação divide seus especialistas entre crackers de "chapéu preto" (*black hat*, os criminosos invasores) e crackers de "chapéu branco" (*white hat*, os pesquisadores éticos que testam falhas justamente para que elas possam ser consertadas).

O mandamento sagrado de jamais ser mal-intencionado permanece como o cordão umbilical definitivo que conecta a cultura hacker do século XXI às suas origens na década de 1950. É fundamental frisar que, ao longo de todas as mutações e revoluções pelas quais o conceito de computador passou nos últimos cinquenta anos, o significado inaugural do termo — a execução de uma travessura genial, inofensiva e inspiradora — sobreviveu intacto. No outono de 2000, o Museu do MIT inaugurou uma exposição permanente para homenagear essa secular tradição: o *Hall of Hacks*. A galeria reúne fotografias históricas que remontam aos anos 1920, incluindo a imagem lendária de uma réplica perfeita de viatura de polícia. Em 1993, os estudantes prestaram seu mais célebre tributo à tradição hacker ao colocar essa viatura policial inteira, com as luzes de emergência piscando alegremente, em cima da cúpula do grande domo central do MIT. A placa do carro exibia as letras `IHTFP` — uma sigla clássica do campus com múltiplos significados satíricos. O mais famoso deles, forjado na panela de pressão das provas do MIT nos anos 1950, era a frase *"I hate this fucking place"* ("Detesto este raio de lugar"); contudo, em 1990, o próprio museu utilizou a sigla para batizar sua revista comemorativa: *The Journal of the Institute for Hacks, Tomfoolery, and Pranks* ("Revista do Instituto de Hacks, Travessuras e Molecagens").

"Na cultura do *hacking*, uma criação elegante e simples é tão reverenciada quanto nas ciências puras", resumiu com precisão cirúrgica o jornalista Randolph Ryan no *The Boston Globe*, em 1993, ao cobrir a exibição do carro de polícia no domo do MIT. "Um *hack* distingue-se do trote estudantil vulgar na medida em que exige planejamento meticuloso, engenharia de ponta, virtuosismo técnico e uma inteligência artística subjacente. A regra de ouro dita que todo hack deve ser bem-humorado, seguro e não destrutivo. Na verdade, os próprios hackers com frequência ajudam a desmontar pacientemente suas próprias instalações para não causar transtornos à comunidade."

A tentação de enclausurar o mundo da computação dentro desses mesmos limites românticos é nobre, mas inexequível. Embora os grandes hacks de software compartilhem dessa mesma busca obsessiva pela elegância e pela simplicidade, o universo dos algoritmos tem uma natureza indelével: é muito mais fácil desmontar e descer uma viatura de polícia do alto de uma cúpula do que desmantelar uma ideia no mundo real — especialmente uma ideia cujo tempo histórico de triunfar finalmente chegou.

Outrora um dialeto obscuro sussurrado nos corredores de um dormitório acadêmico, a palavra "hacker" transformou-se numa bola de bilhar sociológica, sujeita a manobras políticas, interesses corporativos e nuances éticas apaixonadas. Talvez seja por isso que tantos programadores e repórteres tenham tamanho fascínio por ela. Não temos como prever quais rumos o vocábulo tomará na boca das gerações futuras. Podemos, no entanto, escolher com responsabilidade como nós mesmos o pronunciaremos a cada dia. Utilizar o termo **"cracking"** sempre que nos referirmos à violação indevida de segurança digital — reservando o nobre estandarte do **"hacking"** para a esperteza engenhosa, generosa e livre da criação humana — é prestar o devido tributo a Richard Stallman e a todos os pioneiros retratados nesta epopeia. E é, acima de tudo, a forma mais digna de manter viva a chama do espírito hacker para todo o sempre.

Apêndice B: Licença de Documentação Livre GNU (GFDL)

> ***Nota sobre a tradução:*** *Esta é uma tradução não oficial para o português brasileiro da GNU Free Documentation License, Versão 1.3. A Free Software Foundation não publica traduções oficiais da GFDL como documentos juridicamente vinculantes; apenas o texto original em inglês possui validade legal estrita. A tradução a seguir é fornecida para fins informativos e de compreensão ao leitor de língua portuguesa. O texto original e autêntico em inglês encontra-se reproduzido integralmente ao final deste apêndice, em estrito cumprimento à Seção 8 da licença.*

Licença de Documentação Livre GNU

Versão 1.3, 3 de novembro de 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.
http://fsf.org/

É permitido a qualquer pessoa copiar e distribuir cópias literais deste documento de licença, mas não é permitida sua modificação.

Preâmbulo

O propósito desta Licença é tornar um manual, livro didático ou outro documento funcional e útil "livre" no sentido de liberdade: assegurar a todos a liberdade efetiva de copiá-lo e redistribuí-lo, com ou sem modificações, seja comercial ou não comercialmente. Em segundo lugar, esta Licença preserva para o autor e para a editora uma forma de receberem crédito por seu trabalho, sem que sejam considerados responsáveis pelas modificações feitas por terceiros.

Esta Licença é uma modalidade de "*copyleft*", o que significa que as obras derivadas do documento devem, elas próprias, ser livres no mesmo sentido. Ela complementa a Licença Pública Geral GNU (*GNU GPL*), que é uma licença com *copyleft* projetada para software livre.

Nós concebemos esta Licença com o intuito de utilizá-la para manuais de software livre, porque software livre precisa de documentação livre: um programa livre deve vir acompanhado de manuais que forneçam as mesmas liberdades que o software provê. No entanto, esta Licença não se limita a manuais de software; ela pode ser

utilizada para qualquer obra textual, independentemente do tema ou de ser publicada como livro impresso. Recomendamos esta Licença principalmente para obras cujo objetivo seja didático, instrucional ou de referência.

---

1. APLICABILIDADE E DEFINIÇÕES

Esta Licença se aplica a qualquer manual ou outra obra, em qualquer meio, que contenha um aviso inserido pelo detentor dos direitos autorais afirmando que ela pode ser distribuída sob os termos desta Licença. Tal aviso concede uma licença mundial, livre de royalties e de duração ilimitada, para utilizar aquela obra sob as condições aqui estabelecidas. O termo "**Documento**", adiante, refere-se a qualquer manual ou obra desse tipo. Qualquer membro do público é um licenciado e é tratado como "**você**". Você aceita a licença se copiar, modificar ou distribuir a obra de uma forma que exija permissão sob a lei de direitos autorais (*copyright*).

Uma "**Versão Modificada**" do Documento significa qualquer trabalho contendo o Documento ou uma porção dele, seja copiado literalmente, seja com modificações e/ou traduzido para outro idioma.

Uma "**Seção Secundária**" é um apêndice nomeado ou uma seção preliminar do Documento que trata exclusivamente do relacionamento dos editores ou autores do Documento com o tema geral do Documento (ou com assuntos afins) e que não contém nada que possa se enquadrar diretamente dentro desse tema geral. (Assim, se o Documento for em parte um livro-texto de matemática, uma Seção Secundária não poderá explicar matemática). O relacionamento pode ser uma questão de conexão histórica com o assunto ou com assuntos relacionados, ou de posicionamento legal, comercial, filosófico, ético ou político a respeito deles.

As "**Seções Invariantes**" são certas Seções Secundárias cujos títulos são expressamente designados como sendo os de Seções Invariantes no aviso que declara que o Documento é lançado sob esta Licença. Se uma seção não se enquadrar na definição acima de Seção Secundária, não será permitido designá-la como Invariante. O Documento pode conter zero Seções Invariantes. Se o Documento não identificar nenhuma Seção Invariante, então não há nenhuma.

Os "**Textos de Capa**" são certas passagens curtas de texto listadas como Textos de Capa Frontal ou Textos de Quarta-Capa no aviso que afirma que o Documento é lançado sob esta Licença. Um Texto de Capa Frontal pode conter no máximo 5 palavras, e um Texto de Quarta-Capa pode conter no máximo 25 palavras.

Uma cópia "**Transparente**" do Documento significa uma cópia legível por máquina, representada em um formato cuja especificação esteja disponível ao público em geral, que seja adequada para revisar o documento de maneira direta com editores de texto genéricos ou (para imagens compostas de pixels) programas de pintura genéricos ou (para desenhos) algum editor de ilustrações amplamente disponível, e que seja adequada para alimentar formatadores de texto ou para tradução automática para uma variedade de formatos adequados para alimentar formatadores de texto. Uma cópia feita em um formato de arquivo de outro modo Transparente, mas cuja marcação (ou ausência de marcação) tenha sido planejada para frustrar ou desestimular a modificação subsequente pelos leitores, não é Transparente. Um formato de imagem não é Transparente se for utilizado para qualquer quantidade substancial de texto. Uma cópia que não é "Transparente" é denominada "**Opaca**".

Exemplos de formatos adequados para cópias Transparentes incluem ASCII puro sem marcação, formato de entrada Texinfo, formato de entrada LaTeX, SGML ou XML utilizando uma DTD publicamente disponível, e

HTML simples em conformidade com os padrões, PostScript ou PDF projetados para modificação humana. Exemplos de formatos de imagem transparentes incluem PNG, XCF e JPG. Formatos Opacos incluem formatos proprietários que podem ser lidos e editados apenas por processadores de texto proprietários, SGML ou XML para os quais a DTD e/ou ferramentas de processamento não estejam geralmente disponíveis, e HTML, PostScript ou PDF gerados por máquina produzidos por alguns processadores de texto apenas para fins de saída visual.

A "**Página de Título**" significa, para um livro impresso, a própria página de rosto, acrescida das páginas subsequentes necessárias para conter, de forma legível, o material que esta Licença exige que figure na folha de rosto. Para obras em formatos que não possuem uma página de rosto propriamente dita, "Página de Título" significa o texto próximo à aparição mais proeminente do título da obra, antes do início do corpo do texto.

O "**editor**" significa qualquer pessoa física ou jurídica que distribua cópias do Documento ao público.

Uma seção "**Intitulada XYZ**" significa uma subunidade nomeada do Documento cujo título seja precisamente XYZ ou contenha XYZ entre parênteses após texto que traduza XYZ em outro idioma. (Aqui XYZ representa um nome de seção específico mencionado adiante, como "**Agradecimentos**", "**Dedicatórias**", "**Endossos**" ou "**Histórico**"). "**Preservar o Título**" de tal seção quando você modificar o Documento significa que ela continuará sendo uma seção "Intitulada XYZ", de acordo com esta definição.

O Documento pode incluir Renúncias de Garantia ao lado do aviso que afirma que esta Licença se aplica ao Documento. Considera-se que essas Renúncias de Garantia estão incluídas por referência nesta Licença, mas apenas no que tange à renúncia de garantias: qualquer outra implicação que essas Renúncias possam ter é nula e não tem efeito sobre o significado desta Licença.

---

2. CÓPIA LITERAL

Você pode copiar e distribuir o Documento em qualquer meio, comercial ou não comercialmente, desde que esta Licença, os avisos de direitos autorais e o aviso de licença afirmando que esta Licença se aplica ao Documento sejam reproduzidos em todas as cópias, e que você não adicione nenhuma outra condição àquelas desta Licença. Você não pode usar medidas técnicas para obstruir ou controlar a leitura ou a cópia subsequente das cópias que você fizer ou distribuir. Contudo, você pode aceitar compensação em troca de cópias. Se você distribuir um número suficientemente grande de cópias, deverá também seguir as condições da seção 3.

Você também pode emprestar cópias, sob as mesmas condições declaradas acima, e pode exibir cópias publicamente.

---

3. CÓPIA EM QUANTIDADE

Se você publicar cópias impressas (ou cópias em mídias que habitualmente possuam capas impressas) do Documento, em número superior a 100, e o aviso de licença do Documento exigir Textos de Capa, você deverá acondicionar as cópias em capas que tragam, de forma clara e legível, todos esses Textos de Capa: Textos de Capa

Frontal na capa frontal, e Textos de Quarta-Capa na quarta-capa. Ambas as capas devem também identificar você, de forma clara e legível, como o editor dessas cópias. A capa frontal deve apresentar o título completo com todas as palavras do título igualmente visíveis e proeminentes. Você pode adicionar outros materiais nas capas adicionalmente. A cópia com alterações limitadas às capas, desde que preservem o título do Documento e satisfaçam essas condições, pode ser tratada como cópia literal em outros aspectos.

Se os textos exigidos para qualquer uma das capas forem volumosos demais para caber de forma legível, você deverá colocar os primeiros listados (tantos quantos couberem razoavelmente) na capa propriamente dita, e continuar o restante nas páginas adjacentes.

Se você publicar ou distribuir cópias Opacas do Documento em número superior a 100, deverá incluir uma cópia Transparente legível por máquina junto com cada cópia Opaca, ou declarar em cada cópia Opaca (ou junto a ela) uma localização na rede de computadores a partir da qual o público geral tenha acesso para baixar, usando protocolos padrão públicos de rede, uma cópia Transparente completa do Documento, livre de material adicional. Se você optar por esta última opção, deverá tomar medidas razoavelmente prudentes, ao iniciar a distribuição de cópias Opacas em quantidade, para garantir que essa cópia Transparente permaneça acessível no local especificado até pelo menos um ano após a última vez que você distribuir uma cópia Opaca (diretamente ou por meio de seus agentes ou revendedores) dessa edição ao público.

Solicita-se, mas não se exige, que você entre em contato com os autores do Documento bem antes de redistribuir qualquer grande quantidade de cópias, para dar a eles a oportunidade de lhe fornecer uma versão atualizada do Documento.

---

4. MODIFICAÇÕES

Você pode copiar e distribuir uma Versão Modificada do Documento sob as condições das seções 2 e 3 acima, desde que lance a Versão Modificada sob precisamente esta Licença, com a Versão Modificada assumindo o papel do Documento, licenciando assim a distribuição e a modificação da Versão Modificada a quem quer que possua uma cópia dela. Além disso, você deve fazer o seguinte na Versão Modificada:

A. Utilizar na Página de Título (e nas capas, se houver) um título distinto daquele do Documento e dos títulos de versões anteriores (que devem, se existirem, constar na seção de Histórico do Documento). Você pode usar o mesmo título de uma versão anterior se o editor original dessa versão der permissão.
B. Listar na Página de Título, como autores, uma ou mais pessoas ou entidades responsáveis pela autoria das modificações na Versão Modificada, juntamente com pelo menos cinco dos autores principais do Documento (todos os autores principais, se houver menos de cinco), a menos que eles o liberem dessa exigência.
C. Indicar na Página de Título o nome do editor da Versão Modificada, como editor.
D. Preservar todos os avisos de direitos autorais do Documento.
E. Adicionar um aviso de direitos autorais apropriado para as suas modificações adjacente aos outros avisos de direitos autorais.
F. Incluir, imediatamente após os avisos de direitos autorais, um aviso de licença concedendo ao público permissão para utilizar a Versão Modificada sob os termos desta Licença, no formato demonstrado no Adendo abaixo.

G. Preservar nesse aviso de licença as listas completas de Seções Invariantes e Textos de Capa exigidos fornecidas no aviso de licença do Documento.
H. Incluir uma cópia inalterada desta Licença.
I. Preservar a seção Intitulada "Histórico", Preservar seu Título e adicionar a ela um item declarando pelo menos o título, o ano, os novos autores e o editor da Versão Modificada conforme constar na Página de Título. Se não houver uma seção Intitulada "Histórico" no Documento, crie uma declarando o título, o ano, os autores e o editor do Documento conforme constar em sua Página de Título, e adicione então um item descrevendo a Versão Modificada como exposto na frase anterior.
J. Preservar a localização de rede, se houver, fornecida no Documento para acesso público a uma cópia Transparente do Documento e, da mesma forma, as localizações de rede fornecidas no Documento para versões anteriores em que ele se baseou. Estas podem ser colocadas na seção "Histórico". Você pode omitir a localização de rede de uma obra que tenha sido publicada pelo menos quatro anos antes do próprio Documento, ou se o editor original da versão a que ela se refere conceder permissão.
K. Para qualquer seção Intitulada "Agradecimentos" ou "Dedicatórias", Preservar o Título da seção e preservar nela toda a substância e o tom de cada um dos agradecimentos e/ou dedicatórias de colaboradores nela constantes.
L. Preservar todas as Seções Invariantes do Documento, inalteradas em seu texto e em seus títulos. Números de seção ou equivalentes não são considerados parte dos títulos das seções.
M. Excluir qualquer seção Intitulada "Endossos". Essa seção não pode constar na Versão Modificada.
N. Não renomear nenhuma seção existente para "Endossos" nem para conflitar em título com qualquer Seção Invariante.
O. Preservar quaisquer Renúncias de Garantia.
Se a Versão Modificada incluir novas seções preliminares ou apêndices que se qualifiquem como Seções Secundárias e não contenham material copiado do Documento, você poderá, a seu critério, designar algumas ou todas essas seções como invariantes. Para fazer isso, adicione seus títulos à lista de Seções Invariantes no aviso de licença da Versão Modificada. Esses títulos devem ser distintos de quaisquer outros títulos de seção.

Você pode adicionar uma seção Intitulada "Endossos", desde que ela contenha nada além de endossos de sua Versão Modificada por várias partes — por exemplo, declarações de revisão por pares ou de que o texto foi aprovado por uma organização como a definição abalizada de um padrão.

Você pode adicionar uma passagem de até 5 palavras como Texto de Capa Frontal e uma passagem de até 25 palavras como Texto de Quarta-Capa ao final da lista de Textos de Capa na Versão Modificada. Apenas uma passagem de Texto de Capa Frontal e uma de Texto de Quarta-Capa podem ser adicionadas por (ou por meio de acordos feitos por) qualquer entidade. Se o Documento já incluir um texto de capa para a mesma capa, previamente adicionado por você ou por acordo feito pela mesma entidade em cujo nome você atua, você não poderá adicionar outro; mas poderá substituir o antigo, mediante autorização explícita do editor anterior que adicionou o texto antigo.

O(s) autor(es) e editor(es) do Documento não concedem, por esta Licença, permissão para utilizar seus nomes para publicidade ou para afirmar ou insinuar endosso de qualquer Versão Modificada.

---

5. COMBINAÇÃO DE DOCUMENTOS

Você pode combinar o Documento com outros documentos lançados sob esta Licença, sob os termos definidos na seção 4 acima para versões modificadas, desde que inclua na combinação todas as Seções Invariantes de todos os documentos originais, sem modificação, e as liste como Seções Invariantes de sua obra combinada no respectivo aviso de licença, e que preserve todas as suas Renúncias de Garantia.

A obra combinada precisa conter apenas uma cópia desta Licença, e múltiplas Seções Invariantes idênticas podem ser substituídas por uma única cópia. Se houver múltiplas Seções Invariantes com o mesmo nome, mas conteúdos diferentes, torne o título de cada seção único adicionando ao final, entre parênteses, o nome do autor ou editor original daquela seção se conhecido, ou então um número exclusivo. Faça o mesmo ajuste nos títulos das seções na lista de Seções Invariantes no aviso de licença da obra combinada.

Na combinação, você deve unificar quaisquer seções Intituladas "Histórico" nos vários documentos originais, formando uma única seção Intitulada "Histórico"; da mesma forma, unifique quaisquer seções Intituladas "Agradecimentos" e quaisquer seções Intituladas "Dedicatórias". Você deve excluir todas as seções Intituladas "Endossos".

---

6. COLEÇÕES DE DOCUMENTOS

Você pode fazer uma coleção consistindo no Documento e em outros documentos lançados sob esta Licença, e substituir as cópias individuais desta Licença nos vários documentos por uma única cópia que seja incluída na coleção, desde que siga as regras desta Licença para cópia literal de cada um dos documentos em todos os outros aspectos.

Você pode extrair um único documento de tal coleção e distribuí-lo individualmente sob esta Licença, desde que insira uma cópia desta Licença no documento extraído e siga esta Licença em todos os outros aspectos relativos à cópia literal daquele documento.

---

7. AGREGAÇÃO COM OBRAS INDEPENDENTES

Uma compilação do Documento ou de seus derivados com outros documentos ou obras separados e independentes, em ou sobre um volume de um meio de armazenamento ou distribuição, é denominada um "agregado" se os direitos autorais resultantes da compilação não forem usados para limitar os direitos legais dos usuários da compilação além do que as obras individuais permitem. Quando o Documento é incluído em um agregado, esta Licença não se aplica às outras obras no agregado que não sejam elas próprias obras derivadas do Documento.

Se a exigência de Texto de Capa da seção 3 for aplicável a essas cópias do Documento, então, se o Documento for inferior a metade de todo o agregado, os Textos de Capa do Documento poderão ser colocados em capas que circundem o Documento dentro do agregado, ou o equivalente eletrônico de capas se o Documento estiver em formato eletrônico. Caso contrário, eles devem constar nas capas impressas que circundam todo o agregado.

8. TRADUÇÃO

A tradução é considerada uma modalidade de modificação; portanto, você pode distribuir traduções do Documento sob os termos da seção 4. Substituir Seções Invariantes por traduções exige permissão especial dos detentores de seus direitos autorais, mas você pode incluir traduções de algumas ou de todas as Seções Invariantes em adição às versões originais dessas Seções Invariantes. Você pode incluir uma tradução desta Licença, de todos os avisos de licença no Documento e de quaisquer Renúncias de Garantia, desde que inclua também a versão original em inglês desta Licença e as versões originais desses avisos e renúncias. Em caso de divergência entre a tradução e a versão original em inglês desta Licença ou de um aviso ou renúncia, a versão original prevalecerá.

Se uma seção no Documento for Intitulada "Agradecimentos", "Dedicatórias" ou "Histórico", a exigência (seção 4) de Preservar seu Título (seção 1) normalmente exigirá a alteração do título real.

---

9. RESCISÃO

Você não pode copiar, modificar, sublicenciar ou distribuir o Documento, exceto conforme expressamente previsto sob esta Licença. Qualquer tentativa em contrário de copiar, modificar, sublicenciar ou distribuir o Documento é nula e rescindirá automaticamente os seus direitos sob esta Licença.

No entanto, se você cessar todas as violações desta Licença, a sua licença concedida por um detentor de direitos autorais específico será restabelecida: (a) provisoriamente, a menos e até que o detentor de direitos autorais rescinda explícita e definitivamente a sua licença; e (b) permanentemente, se o detentor de direitos autorais não o notificar da violação por algum meio razoável em até 60 dias após a cessação.

Além disso, a sua licença concedida por um detentor de direitos autorais específico será restabelecida permanentemente se o detentor de direitos autorais o notificar da violação por algum meio razoável, esta for a primeira vez que você recebeu notificação de violação desta Licença (para qualquer obra) daquele detentor de direitos autorais, e você sanar a violação em até 30 dias após o recebimento da notificação.

A rescisão de seus direitos sob esta seção não rescinde as licenças de terceiros que receberam cópias ou direitos de você sob esta Licença. Se seus direitos tiverem sido rescindidos e não restabelecidos permanentemente, o recebimento de uma cópia de parte ou de todo o mesmo material não lhe confere nenhum direito de usá-lo.

---

10. REVISÕES FUTURAS DESTA LICENÇA

A Free Software Foundation pode publicar versões novas e revistas da Licença de Documentação Livre GNU periodicamente. Tais versões novas serão semelhantes em espírito à presente versão, mas poderão diferir em detalhes para abordar novos problemas ou preocupações. Consulte http://www.gnu.org/copyleft/.

A cada versão da Licença é atribuído um número de versão distinto. Se o Documento especificar que se aplica a ele uma versão numerada específica desta Licença "ou qualquer versão posterior", você terá a opção de seguir os termos e condições daquela versão especificada ou de qualquer versão posterior que tenha sido publicada (não como rascunho) pela Free Software Foundation. Se o Documento não especificar um número de versão desta Licença, você poderá escolher qualquer versão já publicada (não como rascunho) pela Free Software Foundation. Se o Documento especificar que um procurador pode decidir quais versões futuras desta Licença podem ser utilizadas, a declaração pública de aceitação de uma versão por parte desse procurador autoriza você permanentemente a escolher essa versão para o Documento.

---

11. RELICENCIAMENTO

"Site de Colaboração Maciça Multiautoral" (ou "Site CMM") significa qualquer servidor da World Wide Web que publique obras sujeitas a direitos autorais e também ofereça facilidades proeminentes para que qualquer pessoa edite essas obras. Um wiki público que qualquer pessoa possa editar é um exemplo de tal servidor. Uma "Colaboração Maciça Multiautoral" (ou "CMM") contida no site significa qualquer conjunto de obras protegidas por direitos autorais assim publicadas no site CMM.

"CC-BY-SA" significa a licença Creative Commons Atribuição-CompartilhaIgual 3.0 publicada pela Creative Commons Corporation, uma corporação sem fins lucrativos com sede em São Francisco, Califórnia, bem como versões futuras com *copyleft* dessa licença publicadas pela mesma organização.

"Incorporar" significa publicar ou republicar um Documento, no todo ou em parte, como parte de outro Documento.

Uma CMM é "elegível para relicenciamento" se estiver licenciada sob esta Licença e se todas as obras que foram publicadas pela primeira vez sob esta Licença em outro local que não essa CMM, e posteriormente incorporadas no todo ou em parte à CMM: (1) não tiverem textos de capa ou seções invariantes; e (2) tiverem sido assim incorporadas antes de 1º de novembro de 2008.

O operador de um Site CMM pode republicar uma CMM contida no site sob a licença CC-BY-SA no mesmo site a qualquer momento antes de 1º de agosto de 2009, desde que a CMM seja elegível para relicenciamento.

ADENDO: Como utilizar esta Licença para os seus documentos

Para usar esta Licença em um documento que você escreveu, inclua uma cópia da Licença no documento e insira os seguintes avisos de direitos autorais e de licença logo após a página de rosto:

```
 Copyright (c)  ANO  SEU NOME.
É concedida permissão para copiar, distribuir e/ou modificar este documento
sob os termos da Licença de Documentação Livre GNU, Versão 1.3
```

```
    ou qualquer versão posterior publicada pela Free Software Foundation;
    sem Seções Invariantes, sem Textos de Capa Frontal e sem Textos de Quarta-Capa.
    Uma cópia da licença está incluída na seção intitulada "GNU
    Free Documentation License".
```

Se você possuir Seções Invariantes, Textos de Capa Frontal e Textos de Quarta-Capa, substitua a linha "sem Seções... Quarta-Capa." pelo seguinte:

```
     com as Seções Invariantes sendo LISTE SEUS TÍTULOS, com os
    Textos de Capa Frontal sendo LISTE, e com os Textos de Quarta-Capa sendo LISTE.
```

Se o seu documento contiver exemplos substanciais de código de programação, recomendamos lançar esses exemplos em paralelo sob uma licença de software livre de sua escolha, como a GNU General Public License, para permitir seu uso em software livre.

GNU Free Documentation License (Official English Text)

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.
http://fsf.org/

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

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.