

TWO BITS

CHRISTOPHER M. KELTY

TWO BITS

LA TRASCENDENCIA CULTURAL
DEL SOFTWARE LIBRE

Edición a cargo de
FLORENCIO CABELLO

Icaria ♫ editorial

Usted es libre de:

Compartir: copiar y redistribuir el material en cualquier medio o formato.

Adaptar: remezclar, transformar y construir a partir del material.

Atribución-NoComercial-CompartirIgual 3.0 (CC BY-NC-SA 3.0)

Título original: *Two Bits*

© 2008, Duke University Press

© Edición: Florencio Cabello

© Traducción del inglés: Ana M. Alconchel, María Anaya, Ana Balbuena, Aleksandra Bogdanska, Hermes Cañizares, Teresa de Andrés, Carlos de las Heras, César de Pablo, Julieta Fernández de Landa, Leticia Fuentes, Noelia García, Marifé García Braun, Nerea García Garmendia, María García Perulero, Isis Herrero, Elisa Lois, Pedro J. López Quintana, Manuel Martínez, Francisco J. Maya Rodríguez, Lara Moyano, Natalia Parrado, Laura Pérez, Fabián Prieto, Mayte Provencio, Emma Ramos, María Teresa Rascón, Paloma Ruiz, Rubén Salinero, María del Pilar C. Sancosme, Antonio Santo, Fátima Solera, Alejandro Torres, Victoria Trillo, Andrea Vidal y José Antonio Villalobos.

© De esta edición:

Icaria editorial, s. a.

Arc de Sant Cristòfol, 11-23

08003 Barcelona

www.icariaeditorial.com

(Este libro está disponible gratuitamente en la página de Icaria

<http://icariaeditorial.com/>) Edición 2019

ISBN: 978-84-9888-881-2

Fotocomposición: Text Gràfic

A mis padres, Anne y Ted

ÍNDICE

Introducción

Modos de hacer, procomún y públicos recursivos, *Florencio Cabello* 9

Prefacio 53

Agradecimientos 57

Introducción 61

PARTE I INTERNET

I. *Geeks* y públicos recursivos 93

II. Reformadores protestantes, polímatas y transhumanistas 137

PARTE II SOFTWARE LIBRE

III. El movimiento 177

IV. Compartición de código fuente 201

V. Concepción de sistemas abiertos 233

VI. Redacción de licencias de *copyright* 277

VII. Coordinación de colaboraciones 317

PARTE III MODULACIONES

VIII. «Si triunfamos, desapareceremos» 357

IX. Reutilización, modificación y la inexistencia de normas 387

Conclusión 425

Bibliografía 437

INTRODUCCIÓN

MODOS DE HACER, PROCOMÚN Y PÚBLICOS RECURSIVOS

Florencio Cabello

El 25 de marzo de 2017 la Casa Invisible de Málaga acogía la inauguración de las jornadas «Picasso en la institución monstruo. Arte, industria cultural y derecho a la ciudad», organizadas por el eipcp (*European Institute for Progressive Cultural Policies*), el Museo Reina Sofía y la propia Invisible con motivo del octogésimo aniversario de la creación del *Guernica*. Enclavados en un inmueble histórico municipal en pleno centro de una ciudad «donde la cultura es capital»¹ y centrifuga vecinos, se trataba de indagar de forma situada las resonancias contemporáneas del insigne mural de un pintor *renacido* en Málaga (por mor de su Plan Estratégico de 1992) como estandarte de su nueva «marca urbana» cultural. Así, en el texto de presentación del ciclo podía leerse:

La obra de Picasso, dado su reconocimiento, es uno de los mejores ejemplos para analizar las contradicciones y complejidad de estas lógicas. En la actualidad, el trabajo del artista malagueño es ampliamente apreciado y divulgado; al mismo tiempo, tal recepción consolida estereotipos asociados a una idea heroica y fetichizada de las vanguardias artísticas y de sus mitos más persistentes, como la originalidad, la genialidad y la autonomía

1. Título del video promocional para la feria Fitur 2015 disponible en: <https://www.youtube.com/watch?v=fJK0iJOG1XQ> . Esta y otras manifestaciones de la marca «Málaga, ciudad genial» las (re)descubrimos a través de la deconstrucción minuciosa que de tales campañas institucionales (y de sus delatores lapsus) vienen realizando Rogelio López Cuenca y Elo Vega desde hace años. Tras deleitarnos con una muestra de dicha deconstrucción en las jornadas Picasso en la institución monstruo, ambos artistas la volcaron en un monumental (en el buen sentido) repositorio (en formato periódico y web) titulado Málaga 2026 (<http://malaga2026.net/>)..

del arte. Valores que desdibujan una de las obras artísticas más complejas del siglo XX. ¿Cómo puede la ciudad contemporánea resistirse a esta integración del arte, sus públicos e instituciones en la economía de servicios y recuperar el potencial de uno de los trabajos artísticos más desafiantes de las vanguardias en su dimensión histórica? ¿Cuáles son los dispositivos e instituciones anómalas, o «monstruo», capaces de situar a la obra en un espacio ajeno a dicha integración?²

Con estas premisas Gerald Raunig y quien firma estas líneas abrimos las jornadas ofreciendo algunas pistas teóricas sobre el papel de las prácticas e instituciones culturales en la construcción cotidiana del «derecho a la ciudad». En mi caso, la organización me invitó a partir del proyecto Traducciones Procomún (www.traduccionesprocomun.org) que vengo desarrollando desde 2008 con mis estudiantes de la Universidad de Málaga (UMA) y otros colaboradores coordinados a través de Internet.³ De este modo, acordamos orientar mi conferencia hacia la cuarta de las obras sobre las que hemos trabajado en este proyecto. Se trataba de *Two Bits*, del antropólogo y profesor de UCLA Chris Kelty,⁴ que nos vino recomendada por el director de Medialab-Prado (Madrid) Marcos García en una visita a la Casa Invisible en verano de 2010. En efecto, la organización y yo entendíamos que la originalidad de la exploración histórica y etnográfica del software libre contenida en *Two Bits*, y en concreto su concepción de los «públicos recursivos», ayudarían a cuestionar tanto aquellos «mitos persistentes» (¿dónde quedan el «folio en blanco» o la inviolabilidad de la obra

2. Fragmento extraído de: <http://www.museoreinasofia.es/actividades/picasso-institucion-monstruo>

3. En el artículo de 2012 «La riqueza de las redes en la educación universitaria» sintetizé las claves conceptuales, pedagógicas y metodológicas del proyecto Traducciones Procomún definiéndolo como «traducción entre iguales basada en el procomún»: <http://www.redalyc.org/pdf/2010/201024390011.pdf>. Para una exposición más específica de los procesos desarrollados en las traducciones cooperativas de cada una de las obras previas, pueden leerse las respectivas introducciones que escribimos para ellas: «Introducción», en *El Código 2.0* (Madrid, Traficantes de Sueños, 2009) e «Introducción» (junto a María García Perulero), en *Remix* (Barcelona Icaria, 2012), ambos de Lawrence Lessig; e «Introducción» (junto a Andoni Alonso), en *La riqueza de las redes* (Barcelona, Icaria, 2015), de Yochai Benkler. Como *Two Bits*, todas estas obras son libremente accesibles en Internet, además de estar disponibles en formato físico.

4. Véase: <https://www.medialab-prado.es/noticias/proyecto-de-traducion-al-castellano-de-two-bits-de-chris-kelty>

Agradezco a Patricia Domínguez Larrondo y Gabriel Lucas su incansable apoyo para esta convocatoria y tantas otras tareas.

acabada⁵ en una dinámica (auto)sostenida sobre las renovadas apertura y remezcla culturales?) como aquel riesgo de «integración» de los públicos en una *audiencia cautiva*, tanto más impotente cuanto más explotada en las cadenas de valor del *marketing* (digital o urbano). He aquí la suerte de «*milieu*»⁶ a caballo entre Internet y la ciudad (con el trasfondo de la defensa de la Casa Invisible como *procomún urbano*) por el que decidimos que transitara una intervención que titulamos «Nociones de público en la inteligencia colectiva». Lo que sigue es, pues, una versión revisada de la misma con la que aportar mis *two bits* a la conversación sobre esta obra que la presente edición desea abrir al ámbito castellanoparlante.

«¿Por qué otra contribución más al debate sobre el público y las esferas públicas?». He aquí uno de los interrogantes de partida que nos presenta Chris Kelty al comienzo de su obra *Two Bits*, tan explícitamente consagrada al software libre como consciente de su «trascendencia cultural» más allá de las aplicaciones y redes informáticas. Más abajo retornaremos a la respuesta que el autor da a la pregunta, pero por ahora ella me da pie a introducir mi particular visión del alcance de dicha trascendencia en los debates contemporáneos sobre la intersección (ya ineludiblemente *multicapa*⁷) entre prácticas críticas artísticas y políticas, instituciones culturales y políticas urbanas.

Para abrir dicha introducción me valgo de una anécdota que refiere Jesús Martín-Barbero sobre Radio Sutatenza, célebre emisora

5. Ingeniosamente parodiada por López Cuenca con sus instalaciones «*Art scene. Do not cross*» de resonancias criminalísticas: http://www.lopezcuenca.com/do_not_cross.html

6. Véase Gerald Raunig, «Tecnecologías. Enmedios, Midstreams, Territorios Subsistenciales», trad. por Raúl Sánchez Cedillo con revisiones de Kike España, en *transversal*, marzo de 2018, disponible en: <http://eipcp.net/transversal/0318/raunig/es>

7. La formulación más articulada (sobre todo metodológicamente) de la «perspectiva multicapa» la he leído en Javier Toret, *Tecnopolítica y 15M: la potencia de las multitudes conectadas*. UOC, Barcelona, 2015. Para una exhaustiva formulación teórica de las funciones básicas de los sistemas de comunicación como estructuradas por «capas», sigo estimando indispensable la representación en 3 capas (física, lógica y de contenidos) que Yochai Benkler propone en *La riqueza de las redes* (Barcelona, Icaria, 2015). Más concretamente, la indagación de Benkler sobre «el surgimiento de la esfera pública en red» y sobre «la batalla en torno a la ecología institucional del entorno digital» (*ibid.*, pp. 253-346; 437-516) demuestran una decidida voluntad de exploración de lo que Kelty denomina aquí «la estratificación históricamente constituida de poder y control en el seno de las infraestructuras informáticas y comunicativas».

cristiana de Colombia consagrada a la «acción cultural popular». Según parece, los directivos de dicha cadena quisieron evaluar en un momento dado el resultado de dicha acción y para ello encargaron una encuesta entre la población campesina a la que se dirigía, encuesta de la que emergió una conclusión desconcertante: la abrumadora mayoría de campesinos afirmaba que, entre toda la parrilla de Radio Sutatenza (cargada de emisiones formativas, de información agrícola, de entretenimiento, etc.), su programa preferido era *el rezo del rosario*. Convencidos de que todo se debía a un error metodológico, los directivos exigieron una repetición de la encuesta que, no obstante, volvió a contradecir sus expectativas, consagrando el rezo del rosario como el programa de mayor calado. La confusión acabó despejándose cuando uno de los encuestadores inquirió el motivo de tal preferencia, a lo que le repusieron: «porque es el único programa en que podemos contestar a los de Bogotá; en el rezo del rosario ellos dicen una parte del avemaría y nosotros la otra, es el único programa en que no hablan ellos solos».⁸

A mi juicio, el surgimiento mano a mano del software libre y de Internet ha encarnado, y a la vez ha propiciado, unas posibilidades culturales que poseen (todavía, cabría apostillar) un potencial de comunicación y creación autónomas que provoca hoy una inquietud muy similar a la de los directivos de Radio Sutatenza. Las implicaciones de estas transformaciones *in nuce* entrañan para Kelty una «reorientación del saber y el poder» cuyas claves expone de esta manera:

El gobierno y el control de la creación y diseminación del saber han cambiado considerablemente en el contexto de Internet a lo largo de las tres últimas décadas. Casi todos los tipos de productos mediáticos son más fáciles de producir, publicar, difundir, modificar, remezclar o reutilizar. El número de tales creaciones, difusiones y préstamos se ha disparado, y las herramientas para generar y divulgar conocimiento —software y redes— también están cada vez más ampliamente disponibles. Los resultados también han sido explosivos e incluyen desasosiegos acerca de la validez, calidad, propiedad y control, amén de pánicos morales en abundancia y nuevas inquietudes acerca de la forma y legitimidad

8. Jesús Martín-Barbero, *Procesos de comunicación y matrices de cultura. Itinerarios para salir de la razón dualista*, Ediciones G. Gili, Naucalpan, México, 1989, pp. 96-97.

de los sistemas mundiales de «propiedad intelectual». Todas estas cuestiones llegan a configurar una *reorientación del saber y el poder* aún incompleta y emergente, cuyas implicaciones alcanzan de lleno el núcleo de la legitimidad, certeza, fiabilidad y, especialmente, de la integridad y temporalidad del saber y las infraestructuras que creamos colectivamente. Se trata de una reorientación a la vez más específica y más general que los grandiosos alegatos diagnósticos sobre la «sociedad de la información» o la «sociedad red», o sobre el auge del trabajo cognitivo o de las economías basadas en el conocimiento: más específica porque concierne a prácticas técnicas y legales precisas y detalladas, y más general porque constituye una reorientación *cultural*, no solo económica o legal.

Al dejar de lado la senil retórica de la innovación, esta reorientación nos devuelve a la fractura fundamental entre las (pre)concepciones (culturales, jurídicas, políticas, técnicas...) de los públicos como consumidores disciplinados de productos precocinados y la obstinada constatación de su disposición a la réplica (y ello más allá de los salmos responoriales). De este modo, queda delineado el marco para abordar críticamente dos nociones clave de estos debates: *inteligencia colectiva* y *público*.

Por un lado, al subrayar la imbricación entre autoridad epistémica y política (apuntalada por los *derechos de autor* como piedra angular del modelo industrial basado en monopolios intelectuales), la reorientación del saber y el poder permite trascender aquellas visiones de la *inteligencia colectiva* idílicas y deliberadamente confusas («web 2.0» —y 3.0...—, «economía de la compartición», «economía colaborativa», etc.) para identificarla como presa codiciada por grandes corporaciones y Estados. En este sentido, Rodríguez y Sánchez Cedillo emparentan dicha inteligencia colectiva con la noción marxiana de *general intellect* para concluir que, más allá de la creciente aplicación de la ciencia al proceso productivo que Marx constataba, su centralidad hoy se basa en el hecho de que «la ‘cooperación entre cerebros’ deviene principal recurso económico y auténtico capital fijo del tejido empresarial». Ello les lleva igualmente a emparentar los conflictos en torno a los citados monopolios intelectuales (con sus declinaciones específicas en el ámbito alimentario, sanitario, cultural, biotecnológico, farmacéutico...) con un proceso clave en la explicación marxiana de la *acumulación originaria* de capital, a saber, los cercamientos de tierras desplegados en Inglaterra a partir del siglo XVI:

«hablar de cercamientos de la inteligencia colectiva o de nuevas *enclosures* [...] es hablar de las campañas militares de expropiación y subordinación a la producción bajo mando de esas nuevas tierras comunes que continuamente genera y reproduce la cooperación entre cerebros».⁹

Por otro lado, la no menos problemática noción de *público* queda deslindada por Kelty de cualquier visión reduccionista de las audiencias o de la opinión pública como comparsas de un espectáculo que les separa de la acción, que les priva de agencia. Es así cómo el antropólogo de UCLA llega a una definición de «público» como:

Un colectivo que se reivindica como un mecanismo de control sobre otras formas de poder constituidas —como los Estados, la Iglesia y las corporaciones— pero que se mantiene independiente respecto de dichos ámbitos de poder. [...] Un público cuyos participantes no tienen fe en su carácter autotélico y autónomo es poco más que una charada destinada a aplacar la oposición a la autoridad.

Vida cotidiana y modos de hacer

He aquí, pues, las dos claves conceptuales cuya trabazón pretendo explorar en este texto, explicitando de partida una cuestión relativa al delineamiento del campo de estudio, porque ¿adónde acudir a buscar a los públicos así entendidos? A mi juicio, la respuesta, preñada a su vez de nuevos interrogantes, remite ineludiblemente a la *vida cotidiana* como ámbito donde arraigan las tramas relationales, la experiencia vital y social y la creatividad que fluyen bajo la superficie de las tecnologías y los mecanismos mediáticos. En otras palabras, se trata de abordar la cotidianeidad, no como tema que rescatar nostálgicamente o que aislar etnológicamente para descifrarlo, sino como terreno, por más que móvedizo y esquivo, que explorar.

Para emprender tal exploración un autor que estimo imprescindible es Michel de Certeau, historiador francés cuyo devenir intelectual y vital queda indeleblemente marcado por el Mayo del 68 francés. A partir de ese momento, en efecto, algo (mucho) de la forma de concebir la cultura y de actuar en ella es dejado atrás para siempre, lo cual abre una aguda crisis que este pensador refleja así en *La prise de parole*, publicada el mismo 1968:

9. Emmanuel Rodríguez y Raúl Sánchez Cedillo, «Prólogo», en Olivier Blondepou et al. *Capitalismo cognitivo, propiedad intelectual y creación colectiva*, Traficantes de Sueños, Madrid, 2003, pp. 15-16.

Un acontecimiento: la toma de la palabra. El pasado mayo, la palabra fue tomada como se tomó La Bastilla en 1789. La plaza fuerte que se ocupó es un saber detentado por los dispensadores de la cultura y destinado a mantener la integración o el encierro de los trabajadores, estudiantes y obreros en un sistema que les fija cómo funcionar. De la toma de la Bastilla a la toma de la Sorbona, entre estos dos símbolos, una diferencia esencial caracteriza el acontecimiento del 13 de mayo de 1968: hoy es la palabra prisionera la que ha sido liberada.¹⁰

Dichos interrogantes conducen a de Certeau a una apuesta apasionada por la vida cotidiana, donde detecta una multiplicidad de signos que apuntan, en una dimensión *micro* y diseminada, a la resistencia y la creatividad de gente ordinaria que porfiaba por (re)conquistar una libertad que siempre es provisional. En esta línea, ya entrados los 70 de Certeau publica *La culture au pluriel*, donde sostiene que los mecanismos disciplinarios de encasillamiento social examinados por Michel Foucault son incesantemente desafiados por prácticas cotidianas que desbaratan los cálculos de la planificación autoritaria, y ello tanto en el urbanismo como en los *media*, la educación formal, el trabajo asalariado e incluso la propia religiosidad. Así, frente a una «cultura en singular» que ansía sepultar bajo un horizonte de unificación las prolíficas divergencias culturales que no se conforman al signo del poder, la cultura *en plural* «apela incesantemente a un combate»,¹¹ horadando en lo cotidiano las brechas por donde alcancen a expresarse voces insospechadas, creaciones desbordantes.

En 1980 de Certeau retoma estas cuestiones en el primer volumen de *L'invention du quotidien*, subtitulado «*Arts de faire*» («modos de hacer», según la traducción de Jordi Claramonte).¹² En él, el autor sintetiza el viraje investigador previamente esbozado en la noción de «antidisciplina»,¹³ que implica desplazar el foco del disciplinamiento a su puesta en jaque por unos *usos* cotidianos plenamente reconocidos como *producción*, como *poiesis* (creación, invención):

10. M. de Certeau, *La prise de parole et autres écrits politiques*, Éditions du Seuil, París, 1994, p. 40.

11. M. de Certeau, *La culture au pluriel*, Éditions du Seuil, París, 1993, p. 213.

12. Véase P. Blanco, J. Carrillo, J. Claramonte y M. Expósito (eds.), *Modos de Hacer: Arte crítico, esfera pública y acción directa*, Ed. Universidad de Salamanca, Salamanca, 2001, pp. 383-425.

13. M. de Certeau, *L'invention du quotidien, vol. 1. Arts de faire*, Éditions Gallimard, París, 1990, p. XL.

La presencia y la circulación de una representación [...] no indican en modo alguno lo que ella constituye para sus usuarios. Aún queda por analizar su manipulación por los practicantes que no son sus fabricantes. Solo entonces se puede apreciar el desvío o la similitud entre la producción de la imagen y la producción secundaria que se esconde en el proceso de su utilización.¹⁴

A esta «producción secundaria» (contrapuesta a la primaria, «centralizada, ruidosa y espectacular»)¹⁵ la denominará de Certeau «consumo», sin soslayar cuán eufemístico resulta llamar así a una actividad que, en última instancia, comporta una relación de *dominación*. A tal actividad le atribuye el autor unas cualidades *escurridizas* (disimulo, silencio, diseminación, furtividad...) a las que se ven empujados los consumidores por el expansionismo de los sistemas dominantes de comercio, cultura, urbanismo, etc. Dicho consumo tiene igualmente que ver con la creatividad de una fabricación cultural que trabaja «con un vocabulario y una sintaxis *recibidos*»,¹⁶ pero que a la vez se articula como un «*arts*», esto es, con arreglo a una «*ratio* ‘popular’», una manera de pensar investida en una manera de actuar, un arte de combinar indisociable de un arte de utilizar».¹⁷

Con vistas a acabar proponiendo una imbricación entre «públicos recursivos» y «modos de hacer» que reinterpreté las ideas de Michel de Certeau a la luz de la «reorientación del saber y el poder» que plantea Kelty, dedicaré los dos siguientes epígrafes a los dos ejes teóricos con que el autor francés caracteriza dichos *arts de faire*: en primer lugar, a las «combinatorias de operaciones»¹⁸ que, en clave de *uso*, despliegan estas prácticas; y en segundo lugar, a la perspectiva «polemológica» que da cuenta de «las operaciones casi microbianas que proliferan en el interior de las estructuras tecnocráticas, cuyo funcionamiento desvían mediante una multitud de ‘tácticas’ articuladas sobre los ‘detalles’ de lo cotidiano».¹⁹

14. *Ibid.*, p. XXXVIII.

15. *Ibid.*, p. XXXVII.

16. *Ibid.*, p. XXXVIII.

17. *Ibid.*, p. XLI.

18. *Ibid.*, p. XXXVI.

19. *Ibid.*, p. XL.

Modos de hacer y *usos*: sacando los pies del texto

Un primer rasgo esencial de los modos de hacer en cuanto *poiesis* es su cotidiana impugnación de la pretensión monopolística de aquella producción primaria ruidosa y espectacular. Para analizar la lógica de estos *usos* el historiador francés los asimila a los actos de *enunciación*, por los que los locutores inmersos en el campo definido de un sistema lingüístico *lo realizan* en el *acto de habla*, apropiándose para sus propósitos e instaurando en él un presente relativo a un espacio/tiempo concretos y una específica relación con sus interlocutores:

Es ahí donde nos confundimos acerca del acto de «consumir». Pues suponemos que «asimilar» significa necesariamente «volverse similar a» aquello que se absorbe, y no «volverlo similar» a lo que uno es, hacerlo suyo, apropiárselo o reapropiárselo.²⁰

En la misma línea, de Certeau recurre al análisis de la *lectura* por reconocer en ella los rasgos definitorios de los procesos de consumo involucrados en los modos de hacer. Para empezar, el pensador francés constata que la frecuente subestimación de la lectura, tachada como mera recepción de textos privada de capacidad de inscribir nada propio en ellos, resulta inseparable de una desigual relación de fuerzas, esto es, de la proyección en los textos de la segregación entre clases sociales. Solo a partir de tal segregación se hace posible para de Certeau la aparición del «sentido literal», puesto en circulación por las clases privilegiadas para desautorizar y abolir toda producción de sentido *desviada*:

Esta ficción condena a los consumidores a la sujeción al ser desde entonces siempre culpables de infidelidad o ignorancia ante la «riqueza» muda del tesoro así separado. [...] La utilización del libro por parte de los privilegiados [...] erige entre el texto y sus lectores una frontera para la cual sólo estos intérpretes oficiales expedien pasaportes.²¹

A tal visión contrapone el historiador francés una perspectiva de la lectura que bebe tanto de la teoría de Lévi-Strauss acerca del *bricolage* como de la poética de los romanceros medievales, del *Erwartungsho-*

20. *Ibid.*, p. 240-241.

21. *Ibid.*, p. 247-248.

rizont («horizonte de expectativas») por el que la Escuela de Bochum plantea la evolución del texto de «monumento» a «partitura», y también de la distinción que Barthes establece entre tres tipos de lectura: erótica, que se detiene a disfrutar del placer de las palabras; cazadora, que se precipita hacia el desenlace sin poder esperar; e iniciática, en la que se proyecta ya el deseo de escribir.²² A partir de todas estas referencias se llega a una definición de la *actividad lectora* como *peregrinaje*:

En efecto, leer es peregrinar en un sistema impuesto (el del libro, análogo al orden construido de una ciudad o de un supermercado). [El lector] inventa en los textos otra cosa distinta de la que era su «intención», los despegue de su origen (perdido o accesorio) y combina sus fragmentos creando algo imprevisto en el espacio que organiza su capacidad de permitir una pluralidad indefinida de significaciones. Esta actividad «lectora», ¿está reservada al crítico literario [...], es decir, a una nueva categoría de expertos, o puede extenderse a todo el consumo cultural?²³

Un último elemento imprescindible de esta concepción de la lectura es su apelación al *placer* que provoca en los lectores esta *posibilidad* (pues no es otra cosa) de *asimilación* de los textos originales en función de sus propios intereses, perspectivas y deseos. La relevancia de esta reivindicación deriva del rechazo que hacia dicho placer venían demostrando hasta la fecha buena parte de los estudios mediáticos, algo que Umberto Eco hacía notar ya en 1968: «Bastaba la sospecha de que uno se interesaba por la *estructura de la fruición* en lugar de la estructura del código o del mensaje para ser reo de toda clase de excomuniones».²⁴ Frente a esta visión del placer como un *tabú* que cualquier investigación crítica debía evitar so pena de ser tachada de inconsciente o directamente cómplice respecto de la dominación, de Certeau traza un inspirador *vínculo entre creación y cuidados colectivos*:

Estas maneras de reapropiarse del sistema producido, estas creaciones de consumidores, apuntan a una *terapéutica de las socialidades*

22. Roland Barthes, «Sur la lecture», en *Le Français aujourd'hui*, nº 32, enero de 1976, pp. 15–16, citado en *ibid.*, p. 254.

23. *Ibid.*, p. 245.

24. Umberto Eco, *La estructura ausente*, trad. por Francisco Serra Cantarell, Lumen, Barcelona, 1998, p. 395.

deterioradas. Queda por elaborar una política de estas astucias [que] debe interrogarse también sobre lo que puede constituir hoy en día la representación pública («democrática») de las alianzas microscópicas, multiformes e innombrables entre *manipular* y *disfrutar*.²⁵

En este punto se antoja inexcusable apostillar que, si de elaborar una política que ponga los cuidados en el centro se trata, y ello reconociendo cuanto de «invención de lo cotidiano» hay en ellos, la contribución de la literatura feminista a la tarea resulta crucial. De ahí que, para ilustrar lo explicado acerca de la actividad lectora, opte por rescatar la historia de Ada Byron tal y como Remedios Zafra nos la presentó en la Casa Invisible el 2 de junio de 2016 en el ciclo «Ciencia situada» organizado por Eduardo Serrano. Hija de la baronesa de Wentworth, Anne Isabella Noel Byron (pionera estudiosa de las matemáticas y defensora de la abolición de la esclavitud y otras causas sociales), y del barón Lord Byron (que la dejó al poco de nacer), Ada Byron (o Ada Lovelace) es considerada una visionaria de la computación a raíz de su colaboración en el diseño de la *máquina analítica* propuesta por Charles Babbage, «uno de los padres de la computación moderna»²⁶. De hecho, de ella se afirmó ya en 1953 que poseía «una comprensión excepcional de las ideas de Babbage, las cuales explicaba mucho más claramente de lo que el propio Babbage parece haber hecho jamás».²⁷

Cabe destacar de inicio que para Zafra la reciente reivindicación de Ada Byron comporta la reivindicación paralela de lo que Sadie Plant denomina «la urdimbre feminizadora»²⁸ entre la tradicionalmente femenina labor textil (preñada ya de hilos, nudos-nodos, tramas, en definitiva, de *redes*) y la concepción de las primeras computadoras. No en vano, «el ancestro común» que Plant identifica tanto para la mecanización textil desarrollada a lo largo del siglo XIX como para «el proyecto de la máquina ‘pensante’» auspiciado por Babbage es el telar automático lanzado en 1801 por el industrial francés Joseph Marie Jacquard.²⁹ Se trata de un sistema pionero en su capacidad de

25. M. de Certeau, *L'invention du quotidien*, vol. 1, p. LIII.

26. Remedios Zafra, (*h*)adas. *Mujeres que crean, programan, prosumen, teclean*, Páginas de Espuma, Madrid, 2013, p. 100.

27. B. V. Bowden (ed.). *Faster than thought. A Symposium on Digital Computing Machines*. Sir Isaac Pitman & Sons, Londres, 1953, p. 20. Disponible en: <https://archive.org/details/FasterThanThought>

28. Zafra, *op. cit.*, p. 55.

29. *Ibid.*, pp. 54-55.

programar una máquina de tejer a partir de tarjetas perforadas donde se grababan los distintos patrones y tramas que se quisieran obtener (de forma que cada agujero de la tarjeta marcaba el movimiento de los hilos) para luego introducirlas en el telar, que reconocía tales marcas como instrucciones y las procesaba para ejecutar la tarea. Sería de hecho Ada Byron la que desarrollaría con mayor detalle la idea de Babbage de transponer a su máquina analítica este sistema de tarjetas perforadas con el cual, según ella, se podrían llegar a construir «molinos de cifras [...] que tejen modelos algebraicos lo mismo que el telar de Jacquard teje flores y hojas».³⁰

Imbuida de interés hacia las matemáticas por su madre y tutorizada por los científicos Mary Somerville y Augustus de Morgan, Ada Byron se entusiasmó por la mecanización del cálculo proyectada por Babbage desde que a ambos los presentara la propia Somerville en 1833. Sophie Frend describía así uno de los primeros encuentros de Ada con las incipientes calculadoras de Babbage: «Mientras el resto de la fiesta contemplaba este bello instrumento con el mismo tipo de expresión y sentimiento que se dice que mostraron algunos salvajes la primera vez que miraron un espejo u oyeron una pistola, la señorita Byron, siendo tan joven, comprendió su funcionamiento y vio la gran belleza de la invención».³¹

A partir de aquí, Ada mantuvo el contacto con Babbage para seguir los progresos de su diseño de la llamada Máquina Diferencial y, sobre todo, de la más compleja Máquina Analítica, considerada predecesora de los ordenadores modernos. El culmen de esta relación llegó tras la celebración en Turín en 1840 de un congreso científico al que Babbage acudió a presentar sus diseños y buscar financiación para construirlos. De esta presentación tomó buena nota el relator Luigi Federico, conde de Menabrea, quien en 1842 publicó su informe titulado «*Notions sur la machine analytique*», el cual llegó a Babbage y Ada a través de un amigo común, Charles Wheatstone, el cual animó a esta última a traducirlo al inglés para su mayor difusión.³²

Pues bien, si la traducción supone en sí una forma particularmente atenta de *lectura* (a la que se reconoce su dimensión productiva cuando hablamos de *obra derivada*), la *actividad lectora* desplegada por Ada Byron fue mucho más allá representando una simpar demostración de

30. Citada en Armand Mattelart, *Historia de la sociedad de la información*, Paidós Ibérica, Barcelona, 2002, p. 42.

31. Citada en Bowden, *op. cit.*, p. 20.

32. Zafra, *op. cit.*, p. 101.

lo defendido por Michel de Certeau. De hecho, sería el propio Babbage quien, tras recibir de Ada Byron la traducción inglesa que esta había culminado junto con Wheatstone en 1843, le preguntaría por qué no había *traducido* su propio conocimiento detallado sobre la máquina analítica en un artículo propio o, al menos, en notas y comentarios al de Menabrea.

Es así cómo Ada Byron inició un intenso intercambio epistolar con Babbage³³ de cuya mutua inspiración va dando cuenta en prolíficas anotaciones al informe de relatoría del congreso de Turín. Lo más significativo de tal labor es la posición subalterna reservada para Ada en esta suerte de *remix*³⁴ a tres bandas, donde sus *inscripciones* como autora-traductora se supeditan tanto al texto original del autor-relator Menabrea como a las palabras originales del autor-ponente Babbage (por no hablar del resto de participantes en el coloquio de Turín). Dicha posición subalterna, reflejada tanto en el formato diferido de *notas al pie* como en el modo diferido de *réplicas* a un debate científico copado por una élite masculina, contrasta con su trato de tú a tú con Babbage y sobre todo con la extensión y originalidad de sus aportaciones, entre las cuales aparece ya la idea de *recursividad* (la «reversibilidad» *generativa* a la que alude Zafra) aplicada a la programación informática:

Ada terminó las «Notas» a mediados de 1843. El resultado fue un complejo y extensísimo trabajo que podríamos definir como hiper-textual y descentralizado. [...] una tupida red de notas ordenadas y numeradas [...] que en conjunto triplicaba el tamaño del texto original elaborado por Menabrea. [...] Ada continuó algunas de las líneas abiertas en el artículo respondiendo de manera compleja y extensa a muchas preguntas sugeridas y proponiendo otras nuevas. Entre ellas, una de las sugerencias que la historia de la computación suele destacar de las notas de Ada es la propuesta de que las tarjetas perforadas [...] se adaptaran a la máquina de forma que el motor pudiera repetir determinadas operaciones y hacer uso de esos bucles

33. No me resisto a señalar (por lo que tiene de *recursividad*) que entre las muchas aplicaciones de la ciencia a las que Babbage se entregó durante su vida estuvo también la del estudio y propuesta de reforma del sistema postal, que ya en el siglo XIX permitía en una gran ciudad como Londres una frecuencia diaria de intercambio de mensajes nada desmedible. Para una reflexión sobre el caso paradigmático del servicio postal alemán en la segunda mitad del siglo XIX, véase David Graeber, *La utopía de las normas. De la tecnología, la estupidez y los secretos placeres de la burocracia*, trad. por Joan Andreano Weyland. Ariel, Barcelona, 2015, pp. 154 y ss. En esta obra, Graeber llega a afirmar: «En Alemania uno podría argumentar que la propia nación había sido creada, más que ninguna otra cosa, por el propio servicio postal» (p. 155).

34. Sobre lo que Lessig llama «cultura de la lectura/escritura», véase *Remix*, pp. 85-140.

para operaciones más complejas. [...] Ada resaltó la idea de cómo las combinaciones probables mejorarían las opciones si se incluían repeticiones, es decir una suerte de reversibilidad prevista que rompía la linealidad y diversificaba las opciones entre las planificadas, pero también dejando espacio para las imaginables.³⁵

Con todo, la subalternidad *ex ante* de las contribuciones de Ada Byron aún tenía reservada otra vuelta de tuerca *ex post* por parte de Babbage. Y es que, en su afán por *tener la última palabra* para condicionar la recepción en el Reino Unido de la edición anotada de la obra de Menabrea, el ingeniero inglés exigió incluir un preámbulo donde despotricaba contra el Gobierno británico por retirarle la financiación para sus inventos. Ante la negativa de Ada y de su editor a dar pábulo al *bucle* (nada recursivo) de filípicas que durante toda su carrera cultivó el matemático inglés, este trató de frustrar la publicación y, cuando vio que pese a todo la traducción inglesa aparecía en agosto de 1843,³⁶ rompió su relación con la condesa de Lovelace para siempre.

Cabría leer este episodio como un contraste entre el discurso político *explícitamente crítico* que Babbage, situándose fuera del texto, desea anteponer a este, y la «criticabilidad»³⁷ de las implementaciones que Ada Byron incorpora como «llamadas periféricas»³⁸ desde el mismo texto, implementaciones propiamente *infraestructurales* tanto por lidiar con infraestructuras como por su posición de enunciación social y formalmente «infra». De este modo, adquiere resonancia aquella idea de Rogoff de que «en un desplazamiento reflexivo, desde la función analítica a la función performativa de la observación y la participación, podemos estar de acuerdo en que el sentido no es algo a lo que se llega excavando sino algo que *tiene lugar* en el presente».³⁹ Más allá de esta sugerencia, Remedios Zafra ofrece una acabada interpretación de esta intrincada reelaboración en clave de «extextualidad»:

35. Zafra, *op. cit.*, pp. 102-104.

36. L.F. Menabrea, «Sketch of the Analytical Engine invented by Charles Babbage», A. A. Lovelace (trad. y notas), *Taylor's Scientific Memoirs*, vol. III, 1843, pp. 666-731 (1^a ed. en francés, sin notas: *Bibliothèque Universelle de Genève*, n.º 82, 1842).

37. Irit Rogoff, «Smuggling'—An Embodied Criticality», *transform.eipcp.net*, agosto de 2006. Disponible en: http://xenopraxis.net/readings/rogoff_smuggling.pdf

38. Zafra, *op. cit.*, p. 113.

39. Rogoff, *op. cit.*, p. 2.

...me commueve descubrir en ellas un ejercicio subversivo donde el discurso central es también transgredido por la propia escritura [...]. Indudablemente ese trabajo exige un cambio de perspectiva, dejar de mirar al cuerpo central de la escritura para descubrir todo un entramado de asociaciones, conceptos y críticas perimetrales, un reino de vida en los márgenes, un ejercicio de crítica reflexiva y de desjerarquización de los principios de linealidad que hasta entonces habían predominado en los textos científicos, es decir, una demostración de crítica logocéntrica en toda regla. [...] No, no fue solo lo dicho en los textos de Ada, fue también [...] el lugar en que la escritura fue situada, como metáfora que habla de las relaciones de las mujeres con la ciencia y la tecnología a partir de una historia y un contexto, la extextualidad.⁴⁰

En suma, la reivindicación por Michel de Certeau de lo que de dialógico y (re)creativo se despliega en la actividad lectora se traduce en el caso de Ada Byron en unas «notas» tan exuberantes que acaban *sacando los pies del texto* para, cual boa constrictor, enroscar su inferior cuerpo (de letra) alrededor del cuerpo principal de la obra hasta acabar devorándola.⁴¹

Modos de hacer y *tácticas*: revanchas de gorilas adiestrados

Por más sugerente que pueda resultar la reivindicación de los usos cotidianos, e incluso la ilustración vinculada a la *atención a la letra pequeña* en la historia de la computación, lo cierto es que Michel de Certeau concibe dichos usos como indisociables de una perspectiva polemológica que permita captar en toda su *conflictividad* la naturaleza de la producción secundaria que nos ocupa. Bajo esta luz, y sin dudar de su valor científico y carácter pionero, el episodio de Ada Byron podría ser releído a su vez como parte de las disquisiciones ingenieriles de la élite académica y política de la Inglaterra victoriana (con ecos en Italia, de la que Menabrea llegaría a ser Primer Minis-

40. *Ibid.*, pp. 104-105 y 113.

41. De hecho, el propio Menabrea escribiría a la revista *Cosmos* en 1855 con el propósito explícito de redirigir la atención desde su propia obra a las notas y comentarios añadidos luego por Byron, afirmando: «Las observaciones que acompañaban mi breve memoria eran absolutamente notables y revelaban a un autor de sagacidad fuera de lo común». Citado en Bruce Sterling, «Luigi Federico Menabrea paying tribute to Ada Lovelace», *Wired*, 14 de mayo de 2017: <https://www.wired.com/beyond-the-beyond/2017/05/luigi-federico-menabrea-paying-tribute-ada-lovelace/>

tro). Ante ello cabe preguntarse: ¿de qué modo podría el relato al uso del maridaje entre *madre* y *padre de la computación* ilustrar el crucial componente táctico de los modos de hacer?

Este interrogante me lleva a invertir el orden del apartado anterior para anteponer a la explicación teórica unas páginas donde *sacar punta* a la historia decimonónica de las máquinas calculadoras. Lo hago convencido de que si trascendemos la (relevante) reivindicación genérica de Ada Byron encontraremos *lapsus* muy elocuentes sobre aquellas «estructuras tecnocráticas» en cuyo interior afirmaba de Certeau que se despliega «una multitud de ‘tácticas’ articuladas sobre los ‘detalles’ de lo cotidiano». ⁴²

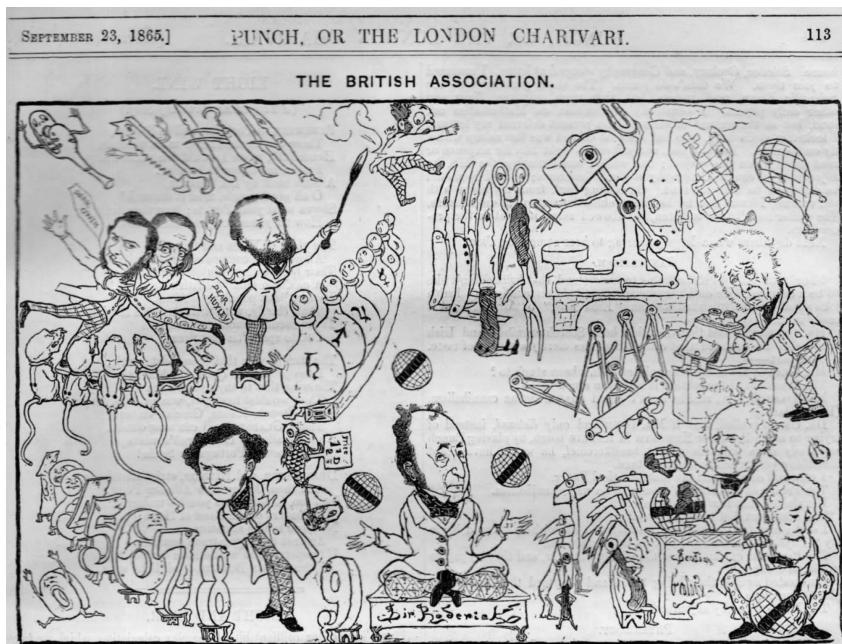
Para empezar, sorprende que la inspiración que la condesa de Lovelace halló en el patrón de mecanización textil pasara por alto los graves disturbios que provocó su implantación a comienzos del siglo XIX. Me refiero especialmente a la (eminente táctica) destrucción de bastidores [frames] emprendida por los autodenominados ludditas entre 1811 y 1812, así como a la feroz represión ordenada por el Gobierno, incluida la aprobación en el mismo 1812 de una *Frame Work Bill* que penaba tales actos con la muerte. Y la sorpresa se torna mayúscula al constatar que la única *voz autorizada* que se alzó contra dicha ley fue justamente la del padre de Ada, quien en su primer discurso ante la Cámara de los Lores defendió así a este incipiente movimiento contra la degradación del trabajo fabril:

Mediante la adopción de una especie de Bastidor en particular, un hombre desempeñaba el trabajo de muchos, y los trabajadores superfluos eran arrojados al desempleo. Mas ha de observarse que la labor así ejecutada era de inferior calidad, no comercializable en el país y meramente apremiada con vistas a la exportación. [...] Los trabajadores expulsados, en la ceguera de su ignorancia, en vez de regocijarse de estas mejoras en artes tan beneficiosas para la humanidad, se concebían como sacrificados ante las mejoras en los mecanismos. [...] Estos hombres nunca habían destruido sus telares hasta que se volvieron inservibles, peor aún, hasta que se volvieron impedimentos reales para sus esfuerzos por obtener su sustento diario. ⁴³

42. M. de Certeau, *L'invention du quotidien*, vol. 1, p. XL.

43. Discurso de Lord Byron ante la Cámara de los Lores de 27 de febrero de 1812, disponible en: <http://ludditebicentenary.blogspot.com/2012/02/27th-february-1812-lord-byrons-maiden.html> La referencia de este episodio procede de Andoni Alonso, quien lo recoge en su libro *El desencanto del porvenir*, 2019.

Tras leer esto, cabría sostener provocativamente que el resarcimiento histórico de Ada Byron ha supuesto el paradójico ensombrecimiento de Charles Babbage, y más concretamente del ambicioso proyecto en que se enmarcaban sus diseños maquínicos y al que consagró gran parte de su carrera. Para acercarnos a la figura de Babbage comienzo rescatando una viñeta satírica sobre el encuentro de la Asociación Británica por el Avance de la Ciencia de 1865, viñeta en la que el matemático inglés es caricaturizado tasando severamente unas piezas de pescado ante una audiencia de números naturales de la que sale huyendo despavorido el cero (véase en la página siguiente, esquina inferior izquierda).



Viñeta de Charles Henry Bennett en el semanario de humor *Punch* (1865). Fuente: JF Ptak Science Books⁴⁴

44. La referencia de esta viñeta (extraída de <http://longstreet.typepad.com/thescience-bookstore/2012/02/british-science-in-cartoon-1865.html>) procede de R. MacLeod y P. Collins (eds.), *The Parliament of Science: The British Association for the Advancement of Science 1831-1981*, Science Reviews, Northwood, Middlesex, 1981, p. 88, citado en Timothy L. Alborn, «Victorian actuaries among the statisticians», en Michael Power (ed.), *Accounting and Science. Natural inquiry and commercial reason*, Cambridge University Press, Cambridge, 1996, p. 96. Para más información sobre la viñeta y, más allá, sobre el número de la revista *Punch* donde aparece, véase el índice del *Science in the Nineteenth-Century Periodical*: https://www.sciper.org/browse/display.jsp?mode=sciper&file=PU1-49.html&reveal=issue_PU1-49-1263&sec

Más allá de reflejar lo que Alborn considera «una reputación de oportunista avaricioso de larga data»,⁴⁵ esta caricatura alude a las ideas de Babbage sobre economía política que sustentan teórica y estratégicamente sus inventos mecánicos. No en vano, cuando Ada Byron conoció a Babbage en 1833, el por entonces Catedrático Lucasiano de Matemáticas en Cambridge gozaba ya de notable éxito por el «tratado de mecánica práctica y economía política» que había publicado un año antes.⁴⁶ En esta obra el ingeniero plasmaba su particular declinación del proyecto leibniziano de «automatización de la razón»,⁴⁷ por la cual aspiraba a extender al ámbito de las operaciones intelectuales el concepto de *división del trabajo* que Adam Smith había aplicado a las operaciones mecánicas de la naciente industria de fines del siglo XVIII. En este punto conviene subrayar que si este original principio de división del trabajo mental constituye, como afirma Mattelart, «la base del proyecto de Babbage de construcción de un prototipo de máquina de calcular»⁴⁸ es justamente porque precede a las máquinas, como atestigua esta crónica histórica sobre Babbage:

En 1812 estaba sentado en los salones de la Analytical Society observando una tabla de logaritmos, que sabía llena de errores, cuando se le ocurrió la idea de computar todas las funciones tabulares mediante maquinaria. El gobierno francés había producido varias tablas mediante un nuevo método. Tres o cuatro de sus matemáticos decidieron cómo computar las tablas, media docena más descompusieron las operaciones en fases simples, y el trabajo en sí, que se restringía a adiciones y sustracciones, era realizado por unos ochenta computadores que solo conocían estos dos procesos aritméticos. He aquí cómo, por vez primera, se aplicó la producción

tion=PU1-49-12-1 Para una amplia información sobre la revista, véase: <https://www.punch.co.uk/about/index> (especialmente recomendable resulta su galería de viñetas sobre ciencia y tecnología: <https://punch.photoshelter.com/gallery/Science-and-Technology-Cartoons/G0000KjR4mk3nljE/>).

45. Alborn, *op. cit.*, p. 96.

46. Charles Babbage, *On the economy of machinery and manufactures*. Charles Knight, Londres, 1832. Aquí nos apoyaremos en la edición castellana aparecida en 1835 a partir de la 3^a edición en inglés: Charles Babbage, *Tratado de mecánica práctica y economía política que con el título de Economía de máquinas y manufacturas / escribió en inglés C. Babbage* (ed. por José Díez Imbrechts). Litografía Martínez, Madrid, 1835. Disponible en: <http://fama2.us.es/fde/tratadoDeMecanicaPractica.pdf>.

47. Mattelart, *op. cit.*, pp. 16 y ss.

48. *Ibid.*, p. 42.

en masa a la aritmética, y Babbage quedó cautivado por la idea de que las operaciones de los computadores descualificados pudieran ser asumidas completamente por máquinas que serían más rápidas y mucho más fiables.⁴⁹

Veinte años después, es esta mecanización de la aplicación de la producción en masa a la aritmética para poder prescindir de los computadores *humanos* descualificados la que Babbage tenía en mente cuando, tras citar a Adam Smith exponiendo las causas que explican las ventajas de la división del trabajo, propuso la siguiente enmienda:

[...] creo, sin embargo, que sería imperfecta la esplicación [sic] del enlace que existe [sic] entre la economía de los productos manufacturados, i [sic] la división del trabajo si se omitiera el principio siguiente.

Al dividir la obra en distintas operaciones de las cuales cada una requiere diferentes grados de habilidad i [sic] fuerza, el maestro director de la fábrica puede adquirir exactamente [sic] la cantidad precisa de fuerza i [sic] habilidad para cada operación, en tanto que si la obra total hubiese de ser ejecutada por un solo artífice, este necesitaría indispensablemente reunir a un propio tiempo bastante habilidad para ejecutar las más delicadas operaciones, i [sic] bastante fuerza para realizar las más penosas.⁵⁰

De la trascendencia de esta reformulación de la división del trabajo (físico y mental) da cuenta su atenta lectura por parte de relevantes autores posteriores. Así, ya en 1867 Karl Marx partió del fragmento en cursiva de Babbage para introducir en el Libro Primero de *El Capital* su análisis de las implicaciones de la división del trabajo en las manufacturas:

La manufactura, pues, desarrolla una jerarquía de las fuerzas de trabajo, a la que corresponde una escala de salarios. [...] Junto a la gradación jerárquica entra en escena la simple separación de los obreros en calificados y no calificados. En el caso de los últimos los costos de aprendizaje desaparecen totalmente; en el de los primeros se reducen, si se los compara con el artesano, porque se ha simplifi-

49. Bowden, *op. cit.*, p. 8.

50. Babbage, *op. cit.*, pp. 159-160.

cado la función. [...] La división manufacturera del trabajo supone la *autoridad* incondicional del capitalista sobre hombres reducidos a meros miembros de un mecanismo colectivo, propiedad de aquél.⁵¹

Pasadas poco más de dos décadas, Alfred Marshall retomó las ideas de Babbage en su (neo)clásico *Principles of Economics* de 1890, cuyo capítulo dedicado a la división del trabajo y a la «especialización en trabajo mental» citaba el fragmento en cursiva incluido arriba como colofón de su explicación de las economías de escala internas.⁵² Si ya entonces, y a diferencia de Marx, Marshall apenas encontraba objeciones a la propuesta del matemático inglés, en 1920 aludía a ella como «el gran principio de Babbage de la producción económica»,⁵³ puntualizando que si no cuajó en el momento de su formulación se debió a que requería tanto una escala de producción como una extensión de los mercados que solo tras la Primera Guerra Mundial comienzan a vislumbrarse.

Igualmente remarcable resulta la observación de Marshall de que dicho «principio de Babbage» anticipaba algunas de las ideas clave de la llamada *organización científica del trabajo* propugnada por Taylor en 1911, incluyendo su estudio pormenorizado de cada una de las tareas desempeñadas por los obreros.⁵⁴ No es de extrañar, pues, que en sus *Principles of Scientific Management* el ingeniero estadounidense ofreciera una de las síntesis más descarnadas de las profundas implicaciones de la división del trabajo surgida del maridaje entre administración industrial (en su caso, centrada en la industria del acero) y método científico: «Este trabajo es de una naturaleza tan

51. Karl Marx. (2009). *El Capital. Crítica de la Economía Política. Libro Primero. El proceso de producción de capital. Vol. II*, trad. por Pedro Scarón. Siglo XXI editores, Madrid, pp. 425-426, 433.

52. Alfred Marshall, *Principles of Economics* (8^a ed.). Palgrave MacMillan, Basingstoke (Inglaterra), 2013, p. 220.

53. Alfred Marshall, *Industry and trade. A study of industrial technique and business organization; and of their influences on the conditions of various classes and nations*. MacMillan and co., Londres, 1920, p. 149.

54. En este sentido, un pasaje de Babbage de significativa resonancia contemporánea es aquel donde, tras ejemplificar con el acto de cavar tierra (luego retomado por Taylor) la importancia de descomponer una tarea en sus componentes menores, afirma: «Se dice que Napoleón comentó de Laplace, cuando este era Ministro del Interior, que estaba demasiado ocupado considerando *les infiniment petites* [«las cosas infinitamente pequeñas»]. El explotarse en asuntos pequeños, que están aislados, no es el dominio de un estadista: pero el integrar el efecto de su recurrencia constante es digno del más grande». Véase Charles Babbage, *The Exposition of 1851, or Views of the industry, the science, and the government, of England*. Alfred Murray, Londres, p. 3.

primitiva y elemental que el que esto escribe tiene la firme creencia de que sería posible adiestrar a un gorila inteligente para convertirlo en un manipulador de lingotes de hierro más eficiente de lo que cualquier hombre podría llegar a ser»⁵⁵.

Por último, la temprana crítica marxiana al principio de Babbage fue actualizada en 1974 por Harry Braverman con el acento puesto en «la degradación del trabajo en el siglo XX». De entrada, el autor de *Labor and monopoly capital* constataba cómo el propio Babbage reconocía que su concepción de la división del trabajo apuntaba a un sustancial abaratamiento de los salarios. A partir de aquí, el economista estadounidense refutaba su visión tecnocéntrica sosteniendo que lo que subyace a dicha división es una sistemática descualificación de las labores productivas y que, como vimos, este «aspecto social» precedía al «aspecto técnico» (mecanización incluida):

El principio de Babbage [...] da expresión, no a un aspecto técnico de la división del trabajo, sino a su aspecto social. [...] En la mitología del capitalismo, el principio de Babbage se presenta como una respuesta a la «escasez» de trabajadores cualificados [...], cuyo tiempo se usa así de modo más «eficiente» para beneficio de la «sociedad». [...] El modo de producción capitalista sistemáticamente destruye las capacidades circundantes allá donde existan. [...] La generalizada distribución del conocimiento del proceso productivo entre todos sus participantes deviene, a partir de este momento, no meramente «innecesaria», sino un obstáculo directo al funcionamiento del modo capitalista de producción.⁵⁶

En suma, por más que el papel de Ada Byron dentro de las *maquinaciones* de Babbage se ciña al de «encantadora de números»⁵⁷ para su pionero dispositivo analítico,⁵⁸ la reivindicación de su figura en clave

55. Frederick Winslow Taylor, *Principles of Scientific Management*, Harper & Brothers, Nueva York, 1911, p. 40.

56. Harry Braverman. *Labor and monopoly capital. The Degradation of Work in the Twentieth Century*. Monthly Review Press Taylor, Nueva York, 1998, pp. 56-57.

57. Zafra, *op. cit.*, p. 107.

58. Hasta donde sé, la única colaboración ajena a la Máquina Analítica entre Babbage y Byron giró en torno al diseño de un sistema «infallible» (y a la postre ruinoso) de apuestas en carreras de caballos (véase Bowden, *op. cit.*, pp. 20-21), en lo que podría verse como una vuelta a los orígenes de la teoría de probabilidades con la obra *De ratiociniis in ludo aleae* [«Sobre el raciocinio en los juegos de azar»], publicada en 1657 por Christiaan Huygens.

de género no debería eclipsar el proyecto científico-industrial en que se enmarca históricamente,⁵⁹ resumido así por Silvia Federici:

[...] desde el punto de vista del proceso de abstracción por el que pasa el individuo en la transición al capitalismo, el desarrollo de la ‘máquina humana’ fue el principal salto tecnológico [...]. Podemos observar, en otras palabras, que la primera máquina desarrollada por el capitalismo fue el cuerpo humano y no la máquina de vapor, ni tampoco el reloj.⁶⁰

Sirva, pues, esta breve revisión de los albores de la computación mecánica como trasfondo para examinar la referida dimensión *táctica* de los modos de hacer, que de este modo se revelan inequívocamente como *revanchas de gorilas adiestrados*.⁶¹ Y es que Michel de Certeau arraiga su análisis en la constatación, a menudo escamoteada, de que por más capacidad de contestación desviada y creativa que pueda atribuirse a los lectores, ella va a estar supeditada a las correlaciones de fuerzas y a las condiciones hegemónicas que circunscriben el terreno en el que se despliegan los *arts de faire*. En relación con ello, el historiador francés hace hincapié en las *asimetrías* que marcan la segregación entre las posiciones de lectura y escritura (o, si se prefiere, de *consumo* y *producción*) y, dentro de ellas, en los distintos márgenes de maniobra delimitados por las coyunturas en que se hallan los usuarios y por los recursos culturales que tengan a su alcance:

La cultura articula conflictos y ora legitima, ora desplaza, ora controla la razón del más fuerte. Ella se desarrolla en un elemento de tensiones, y a menudo de violencias, al cual proporciona equilibrios

59. Con todo, el proyecto de *transferencia* universitaria *avant la lettre* de Babbage no se limitó al capitalismo fabril, sino que abarcó también su vertiente más financiera a través del estudio de las florecientes compañías aseguradoras (véase *A Comparative View of the Various Institutions for the Assurance of Lives*, J. Mawman, Londres, 1826) y del asesoramiento matemático al emergente gremio de actuarios de seguros (véase Timothy L. Alborn, *op. cit.*, pp. 85–96).

60. Silvia Federici, *Calibán y la bruja. Mujeres, cuerpo y acumulación originaria*, trad. por Verónica Hendel y Leopoldo Sebastián Touza. Traficantes de Sueños, Madrid, 2010, p. 201.

61. En esta línea, Michel de Certeau presenta la llamada *perruque* como un modo de hacer propiamente fabril por el que «el trabajador [...] sustrae a la fábrica tiempo (más que bienes, pues él solo emplea desechos) con la perspectiva de un trabajo libre, creativo y precisamente no lucrativo» (*L'invention du quotidien*, vol. 1, p. 45). Para un análisis propio al respecto, véase Florencio Cabello, «Distorsión comunicativa: La *perruque* a través de las industrias culturales en red», en Félix Moral, Ana Masedo y Margarita Bravo (eds.), II Jornadas sobre movimientos sociales, UMA/Casa Invisible, Málaga, 2010, pp. 60–66.

simbólicos, contratos de compatibilidad y compromisos más o menos temporales. Las tácticas de consumo, ingenios del débil para sacar partido del fuerte, desembocan así en una *politización de las prácticas cotidianas*.⁶²

Para examinar este conflicto esencial del consumo cotidiano el autor galo recurre a la distinción entre «estrategia» y «táctica». Según él, la primera consiste en aquel «cálculo (o manipulación) de las relaciones de fuerzas» que deriva de la posibilidad de aislar a una determinada instancia dotada de voluntad y poder (una empresa, un ejército, una institución científica o urbana) con respecto a una *exterioridad*, con la cual interactuará desde el *lugar propio* circunscrito mediante dicho aislamiento. Las implicaciones de tal cesura son sintetizadas por de Certeau en tres proposiciones fundamentales: primero, la estrategia supone «una victoria del lugar sobre el tiempo» que otorga la potestad de capitalizar las ventajas logradas y de sustraerse en buena medida a los imperativos coyunturales; segundo, la estrategia permite una práctica *panóptica* que busca controlar el tiempo (de hecho, *anticiparse* a él) mediante el dominio (que es *previsión*) del espacio; finalmente, la estrategia *da lugar* a un tipo de *saber* que, sustentado en el poder previo, se definiría como «aquella capacidad de transformar las incertidumbres históricas en espacios legibles».⁶³

La *táctica*, por su parte, es definida como ese otro cálculo de fuerzas que, no teniendo más lugar que aquel ocupado por el otro, carece de la postulación de una frontera que le permita distinguirse de él y controlarlo visualmente. Así pues, la táctica se ve empujada a *jugar* siempre en terreno adverso, esto es, en el espacio acotado por una institución de poder, lo cual le impide tanto mantener las distancias con el enemigo como retirarse de su vigilancia panóptica, tanto acumular beneficios en un lugar propio como preparar desde él sucesivos movimientos.

¿Cuáles son las implicaciones de este «cuerpo a cuerpo sin distancia»⁶⁴? La primera es el privilegio que las tácticas conceden al *tiempo*, escudriñando la insinuación de la «ocasión» propicia y pertrechándose para no dejarla escapar. Ello explica la insistencia del autor francés en el concepto del «golpe», que presupone una perspicacia para detectar *al vuelo* las fisuras transitorias y una astucia para sacarles el mayor partido, apareciendo donde y sobre todo *cuando* nadie lo espera. De aquí surge

62. M. de Certeau, *L'invention du quotidien*, vol. 1, p. XLIV. (La cursiva es mía).

63. *Ibid.*, pp. 59–60 y 63.

64. *Ibid.*, p. 62.

la caracterización de estos procedimientos tácticos nómadas como caza (o pesca) furtiva, esto es, como actividad que se da en un lugar y/o temporada vedados y/o con instrumentos no reglamentarios.⁶⁵

La segunda implicación es que las tácticas están fundamentalmente determinadas por la *ausencia de poder*, la cual acentúa su diseminación a través de los lugares de dominio, su sutileza y sobre todo su tenacidad. A partir de la identificación de estas dos implicaciones cruciales de las tácticas, de Certeau sintetiza de este modo su repercusión:

Las tácticas designan procedimientos cuya validez se basa en la pertinencia que conceden al tiempo –a las circunstancias que el instante preciso de una intervención transforma en situación favorable, a la rapidez de los movimientos que cambian la organización del espacio, a las relaciones entre momentos sucesivos de un «golpe», a los posibles cruces de duraciones y de ritmos heterogéneos, etc. A este respecto [...] mientras que las estrategias apuestan por la resistencia que *el establecimiento de un lugar* ofrece al desgaste del tiempo, las tácticas apuestan por una hábil *utilización de este*, de las ocasiones que él presenta y también de los juegos que él introduce en los cimientos de un poder.⁶⁶

A la hora de describir en detalle dichos procedimientos tácticos, de Certeau se decanta por asimilarlos a las figuras o *tropos* de los que da cuenta la retórica: «Mientras que la gramática vigila la ‘propiedad’ de los términos, las alteraciones retóricas (derivas metafóricas, condensaciones elípticas, miniaturizaciones metonímicas) señalan la utilización de la lengua por locutores en situaciones particulares de combates lingüísticos rituales o efectivos».⁶⁷ En esta misma línea, Umberto Eco define las figuras retóricas como «*cortocircuitos útiles* para sugerir de manera analógica los problemas que no se pueden analizar a fondo»,⁶⁸ señalando cómo en su base reside una contradicción consistente en la «oscilación entre redundancia e información». De este modo, en los tropos convivirán la apelación a algo que el oyente sabe (y quiere) de antemano (*«la retórica como depósito de técnicas argumentales ya comprobadas»*) y el propósito de convencerlo de algo que ignora o de

65. *Ibid.*, pp. 60–62.

66. *Ibid.*, p. 63.

67. *Ibid.*, p. 64.

68. Umberto Eco, *La estructura ausente*, p. 169. (La cursiva es mía).

contemplar algo familiar desde una perspectiva inusitada. Esta función *generativa* «que persuade reestructurando en lo posible lo que ya es conocido» es señalada por Eco como propia de una variante específica de retórica que califica como *«nutritiva»*.⁶⁹

Entre las diferentes *artes* de pensar y actuar susceptibles de proporcionar esquemas de análisis retórico ajustados para las tácticas, Michel de Certeau pone el foco en la *metis* de la Grecia Clásica analizada por Detienne y Vernant.⁷⁰ De entrada, el historiador francés caracteriza dicha *metis* como un principio de *buen hacer*, de astucia práctica que orienta la acción hacia la consecución de efectos máximos con el mínimo de fuerzas, para lo cual son cruciales tres aspectos: La *ocasión*, que implica la concentración sobre la dimensión temporal de la acción al acecho del momento oportuno (*«kairos»*); los *disfraces*, en tanto que esta *metis* entraña «una deserción del lugar propio»⁷¹ que se sirve de muy diversas metaforizaciones o desplazamientos;⁷² y la «invisibilidad paradigmática»⁷³ que determina el hecho de que la *metis* desaparece en el acto mismo en que se realiza, sin conservar ninguna imagen de sí misma.

A reglón seguido, el historiador galo se interesa por el «giro» que este arte griego posibilita para invertir las relaciones de fuerzas pre establecidas, identificando como el elemento clave la *memoria*: «“Memoria” en el sentido antiguo del término, que designa una presencia en la pluralidad de tiempos y no se limita por tanto al pasado». ⁷⁴ A partir de aquí, de Certeau subraya que la memoria no presupone un lugar propio, a modo de base de datos posteriormente *recuperables* mediante una consulta abstracta, sino que se compone mediante la sedimentación (irregular, heterogénea e inseparablemente apegada a los momentos de su adquisición) de una experiencia que permanece

69. *Ibid.*, p. 172.

70. M. Detienne y J.-P. Vernant, *Las artimañas de la inteligencia. La metis en la Grecia antigua*. Taurus, Madrid, 1988. En la página 10 Detienne y Vernant definen la *metis* como «un cierto tipo de inteligencia comprometida con la práctica, enfrentada a obstáculos que debía dominar utilizando la astucia para lograr el éxito en los ámbitos más diversos de la acción». Dicha *metis* abarcaría «una trampa de caza, una red de pesca, el arte del cestero, del tejedor, del carpintero, la maestría del piloto, el olfato del político, el ojo clínico del médico, las artimañas de un personaje retorcido como Ulises, [o] el ilusionismo teórico de los sofistas».

71. M. de Certeau, *L'invention du quotidien*, vol. 1., p. 124.

72. En este sentido, de Certeau nos recuerda que en la Grecia contemporánea los transportes públicos y los servicios de mudanza mantienen la denominación de «*metaphorai*». *Ibid.*, p. 170.

73. *Ibid.*, p. 124.

74. *Ibid.*, p. 320 (nota 7 del capítulo VI).

presente y viva. A partir de dicha sedimentación, la memoria se articula como una práctica metonímica de *singularidad* en medio de un cúmulo de circunstancias preestablecidas. En efecto, la memoria *solo* aporta al conjunto que le viene impuesto detalles, fragmentos, destellos, cada uno de los cuales, no obstante, apunta a un todo cuya ausencia es revelada por la presencia de dichos detalles. En último término, el potencial de este cúmulo de conocimientos sin lugar propio radica en que, al «restaurar en los lugares donde los poderes se distribuyen *la insólita pertinencia del tiempo*»,⁷⁵ posibilita orientar las acciones hacia el porvenir e «inventariar los posibles».⁷⁶

Una duración se introduce así en la correlación de fuerzas y va a cambiarla. La *metis*, en efecto, apuesta por un tiempo acumulado, que le resulta favorable, contra una composición de lugar que le es desfavorable. Pero su memoria permanece escondida (carente de lugar identificable) hasta el instante en que se revela, en el «momento oportuno» [...]. El relámpago de esta memoria brilla en la *ocasión*, [en la que] concentra el *máximo* de saber en el *mínimo* de tiempo.⁷⁷

Concluyo aquí el repaso por la concepción táctica de los modos de hacer que propugna de Certeau, pero juzgo sugerente complementarla con un par de apuntes teóricos más. El primero es la reivindicación del *plagio* que el Critical Art Ensemble (CAE) lanza en 1994, en la cual resuena aquello que más de dos décadas antes avanzara Roland Barthes al afirmar que «en una sociedad sometida a la guerra de los sentidos, [...] hay que practicar un cierto ‘arribismo’ semántico. La crítica ideológica está hoy en día, efectivamente, *condenada a operaciones de hurto*»⁷⁸. De esta forma, tras emprender un desagravio histórico del plagio el CAE señala cómo en el panorama cultural contemporáneo su virtuosismo pasa en buena medida por el *hacking*:

En una sociedad dominada por una explosión de «conocimiento», la exploración de las posibilidades de significado de lo ya existente

75. *Ibid.*, pp. 133-134. (La cursiva es mía).

76. *Ibid.*, p. 126.

77. *Ibid.*, pp. 125-126.

78. Roland Barthes, *Lo obvio y lo obtuso. Imágenes, gestos, voces*, trad. por Carlos Fernández Medrano. Paidós Comunicación, Barcelona, 1986, pp. 329-330.

se hace más acuciante que la adición de información redundante. [...] He aquí donde el plagio progresá más allá del nihilismo. No se limita a injectar escepticismo para ayudar a destruir los sistemas totalitarios que paralizan la invención, sino que *participa él mismo en la invención* y, consecuentemente, es también productivo. [...] El problema actual para los aspirantes a productores culturales es el de acceder a dichas tecnología e información [...] lo cual nos lleva a preguntarnos si, para ser un plagiador exitoso, uno ha de ser también exitoso como *hacker*.⁷⁹

A esta misma cuestión alude Henry Jenkins en su obra *Textual poachers*, donde «poacher» traduce al inglés el término «braconnier» («cazador o pescador furtivo») propuesto por de Certeau. Así, el profesor del MIT analiza la «escandalosa categoría»⁸⁰ que en el campo de la cultura sigue representando el movimiento *fan*, consagrado a un prolífico «garabateo en los márgenes» (como Ada Byron) de los productos mediáticos masivos.⁸¹ No en vano, Jenkins desmonta el estereotipo que atribuye a dichos fans una sumisión casi histérica a los productos de las industrias culturales, y expone una concepción de los mismos que, sin negar el *placer* inherente a su particular modo de consumo, constata que en él existe también insatisfacción, frustración e incluso *antagonismo*:

Impávidos ante las concepciones tradicionales de propiedad literaria e intelectual, los fans hacen incursiones en la cultura masiva, reclamando los materiales de esta para su uso propio, reelaborándolos como base de sus creaciones culturales y de sus interacciones sociales. [...] La cultura fan se sitúa así como *un desafío abierto a la «naturalidad» y deseabilidad de las jerarquías culturales dominantes, un rechazo de la autoridad del autor y una violación de la propiedad intelectual*.⁸²

Llegados a este punto, no obstante, Jenkins identifica en las reflexiones de Michel de Certeau acerca de las tácticas un límite que

79. Critical Art Ensemble, *The Electronic Disturbance*, Autonomedia, Nueva York, 1994, pp. 84 y 90. Disponible en: <http://www.critical-art.net/books/ted> (La traducción y la cursiva son mías).

80. Henry Jenkins, *Textual poachers. Television Fans & Participatory Culture*, Routledge, Nueva York, 1992, p. 15 [ed. cast.: *Piratas de textos: fans, cultura participativa y televisión*, trad. por Alicia Capel Tatjer. Paidós, Barcelona, 2010].

81. *Ibid.*, pp. 152-184.

82. *Ibid.*, p. 18. (La cursiva es mía).

estimo fundamental explorar para emparentar los modos de hacer con las reflexiones en torno al procomún y los públicos recursivos a las que dedicaré el siguiente apartado. La orientación de la crítica del autor estadounidense se identifica ya desde el título del apartado donde desarrolla su argumentación sobre dicho límite: «*What do poachers keep?*» —«¿Qué guardan los cazadores furtivos?». Y es que Jenkins difiere de Michel de Certeau cuando este afirma que «la lectura no posee garantías contra el desgaste del tiempo (uno se olvida y olvida) y no conserva (o lo hace mal) aquello que adquiere». ⁸³ Centrándose en los aficionados televisivos, el teórico de Atlanta repondrá que «los fans logran guardar aquello que producen con los materiales que ‘cazan furtivamente’ de la cultura masiva [...] Los fans poseen [...] una cultura específica construida a partir de la materia prima semiótica que les proporcionan los media, [la cual] desafía las reclamaciones de la industria mediática de ostentar *copyrights* sobre las narrativas populares». ⁸⁴

De este modo, Jenkins contrapone a la colectividad dispersa de los lectores y a su limitada inversión creativa en aquello que leen la conformación por parte de los fans de una *comunidad lectora* con un alto grado de *compenetración e implicación afectiva* con aquello que consumen. Junto a ello, el autor estadounidense subraya la *materialidad y permanencia* de las historias de los fans, que no solo discuten y recrean los productos mediáticos sino que demuestran ya una preocupación *infraestructural* al imprimir, editar y distribuir ellos mismos sus creaciones. Ello se traduce en el hecho de que la comunidad fan haya cabalgado las sucesivas expansiones del *copyright* desarrollando *tradiciones estéticas* y circuitos de producción y distribución cultural considerados «no ya como un campo de entrenamiento sino como una *salida permanente* para su expresión creativa». ⁸⁵ La resonancia de este (auto)sostenimiento de sus prácticas de consumo con las comunidades *hackers* queda aún más clara si constatamos que el devenir digital de las industrias culturales y el acceso de los fans a Internet y a herramientas de remezcla (a menudo programas libres a su vez) han contribuido a avivar el desarrollo de dichos circuitos y tradiciones estéticas.

83. M. de Certeau, *L'invention du quotidien*, vol. I., p. 251.

84. Jenkins, *Textual poachers*, pp. 49 y 279. (La cursiva es mía).

85. *Ibid.*, p. 48.

En definitiva, sin soslayar el papel crucial que para de Certeau tiene la memoria, será el matiz *olvidadizo* que atribuye a la producción de los usuarios el que juzgo menos ajustado a las tácticas que ponen en juego los «públicos recursivos» que define Chris Kelty. Y es que por más que los *golpes* del movimiento del software libre (paradigma de dicho público para Kelty) en el terreno acotado por las industrias informáticas y culturales jueguen con la ocasión y carezcan de garantías de permanencia (de hecho, los contragolpes privativos están a la orden del día), estimo erróneo considerarlos desconectados o, mejor dicho, *desheredados* unos respecto de otros y, en consecuencia, ignorar la marcada *vocación pública* que le atribuye el autor de *Two Bits*:

Y sin embargo, el software libre es público, consiste en hacer públicos sus desarrollos. Este hecho es esencial para captar su trascendencia cultural, su atractivo y su proliferación. Ahora bien, el software libre es público de una forma particular: se trata de un modo autónomo, colectivo y políticamente independiente de crear objetos técnicos muy complejos que se ponen a libre y pública disposición de todo el mundo —un «*procomún*», en lenguaje común.

Modos de hacer y *procomún*: Lugares comunes

¿Qué procedimientos pone en juego esta producción cultural furtiva para preservar y, más allá, transfigurar y recrear esa *herencia* de tradiciones y operaciones estéticas? ¿Cómo podría concebirse en el marco de las infraestructuras comunicativas e informáticas dominantes un *espacio* donde estos golpes tácticos pudieran disponer de una producción justamente basada en la ausencia de (que es desafío a) un lugar *propio*? Tras repasar la doble referencia lingüística (los *usos*) y polemológica (las *tácticas*) con que Michel de Certeau caracteriza los modos de hacer, parte de la cita que cierra el apartado anterior para proponer el *procomún* como una tercera referencia que permite completar dicha caracterización desarrollando las cuestiones de permanencia, legado y materialidad presentes en la crítica de Jenkins.

De entrada, creo importante subrayar la pertinencia de aproximarnos al *procomún* teniendo muy presente su reverso antítetico, los *cercamientos*, histórica amenaza que Antonio Lafuente suele decir que nos redescubre lo común invisibilizado, dado por sentado. No en vano, ya se apuntó arriba la trascendencia que Marx otorga en el

nacimiento del proletariado industrial al violento proceso de *enclosures* que desde el siglo XVI expropió a los campesinos de Inglaterra y Gales de sus *commons*, aquella institución social tradicional que regía una serie de terrenos comunales a los que toda la vecindad tenía derecho a acceder de forma no exclusiva para obtener sustento. En consonancia con ello, la apelación que aquí se hace al procomún implica retomar la «reorientación del saber y el poder» aludida al inicio y, más concretamente, los «cercamientos de la inteligencia colectiva» esbozados por Rodríguez y Sánchez Cedillo.

De esta manera, y sin olvidar que para Kelty dicha reorientación posee una decisiva *trascendencia cultural* (más allá de aspectos económicos y legales), cabría sintetizar la convulsa dinámica productiva en torno a la inteligencia colectiva parafraseando la polisémica divisa *Information wants to be free* de Stewart Brand para afirmar en su lugar:

La inteligencia colectiva quiere ser libre. La inteligencia colectiva también quiere ser cara. La inteligencia colectiva quiere ser libre porque se ha vuelto muy barata de distribuir, copiar y recombinar —demasiado barata para medirla. Y quiere ser cara porque puede tener un valor incalculable [...]. Esta tensión no desaparecerá.⁸⁶

En este sentido, la propuesta de construir un procomún informativo que contrarreste los cercamientos de la inteligencia colectiva aludidos exige antes de nada insistir en los fundamentos económicos de la producción de información, en la medida en que esta presenta peculiaridades no aplicables a los clásicos bienes comunes de ámbito rural.

La primera de estas peculiaridades consiste en que las ideas constituyen bienes *no exclusivos*, en la medida en que solo si alguien las mantiene en secreto puede excluir a otros de su uso y disfrute, pues desde el momento en que las comparte el proverbial genio escapa de la lámpara de modo irreversible. En este punto es casi obligada la referencia a Thomas Jefferson cuando equiparaba las ideas al *aire* que respiramos para descartar su confinamiento privativo:

86. La cita original, que alude a «la información» en vez de a «la inteligencia colectiva», se encuentra en Brand, *The Media Lab: Inventing the Future at MIT*. Viking Penguin Press, Nueva York, 1987, p. 202. Como se verá más adelante, Kelty propone una inversión de este lema singularmente inspiradora.

Si la naturaleza ha creado alguna cosa menos susceptible que todas las demás de ser objeto de propiedad exclusiva, esa es la acción del pensamiento que llamamos idea, la cual solo puede poseer un individuo si la guarda para sí; pero en el momento en que se divulga, se fuerza a sí misma a estar en posesión de todos, y su receptor no puede desposeerse de ella.⁸⁷

La segunda peculiaridad radica en la definición de las ideas como bienes *no rivales*, en la medida en que si alguien las comparte no ve mermado su disfrute (ni total ni parcialmente) por que quienes las reciban las disfruten a su vez. Retomando la cita de Jefferson, en este caso el fundador de EEUU recurre a la analogía del *fuego*, «susceptible de expandirse por todo el espacio sin que disminuya su densidad en ningún punto»,⁸⁸ para evidenciar algo que más de un siglo después remarcaría Antonio Machado en una suerte de canto a la cultura libre *avant la lettre*:

Para nosotros, la cultura ni proviene de energía que se degrada al propagarse, ni es caudal que se aminore al repartirse; su defensa, obra será de actividad generosa que lleva implícitas las dos más hondas paradojas de la ética: solo se pierde lo que se guarda, solo se gana lo que se da.⁸⁹

Una última cita, singularmente ilustrativa en su obviedad/herejía (tácheselo que no proceda), es esta sobre la diferencia entre la com-partición de bienes físicos e intelectuales apócrifamente atribuida a George Bernard Shaw:

Si tú tienes una manzana y yo tengo una manzana y las intercambiamos, entonces tanto tú como yo seguiremos teniendo una manzana cada uno. Pero si tú tienes una idea y yo tengo una idea y las intercambiamos, entonces cada uno tendrá dos ideas.

87. Thomas Jefferson, carta a Isaac McPherson, 13 de agosto de 1813, citado en Lessig, *El Código 2.0*, p. 294.

88. *Idem*.

89. Antonio Machado, «'Sobre la defensa y la difusión de la cultura'. Discurso pronunciado en Valencia en la sesión de clausura del Congreso Internacional de Escritores», *Hora de España*, n.º VIII, Valencia, agosto de 1937, p. 17. Disponible en: www.filosofia.org/hem/193/hde/hde08011.htm

A las dos peculiaridades mencionadas, que Smári McCarthy compendia afirmando que «las ideas son volátiles y pegajosas»,⁹⁰ se añade una tercera «extravagancia crucial» de la producción intelectual que Benkler describe bajo el rótulo de «efecto a hombros de gigantes»:

[...] la información es tanto insumo como producto de su propio proceso productivo. [...] Para poder escribir una novela, una película o una canción actuales, tengo que utilizar y transformar formas culturales existentes, tales como argumentos o giros. Esta característica se conoce entre los economistas como el efecto «a hombros de gigantes», recordando la afirmación atribuida a Newton: «Si he visto más allá es porque me he aupado a hombros de gigantes».⁹¹

Una vez repasados los fundamentos económicos que trazan los contornos de la reorientación del saber y el poder a la que alude Kelty, es momento de abordar la definición de ese «concepto ancho, plural y elusivo»⁹² que con Antonio Lafuente hemos dado en llamar procomún. Una referencia clave para ello proviene de nuevo de Yochai Benkler, quien lo define como «una específica forma institucional de estructurar el derecho de acceso, uso y control de los recursos» que se opone a la propiedad porque, en lugar de asignar a un propietario el control exclusivo de los mismos, establece que:

Cualquier miembro de un grupo (más o menos definido) de personas puede usar o disponer de los recursos regidos por el procomún, de acuerdo con unas normas que pueden ir desde el «todo vale» a reglas formales escrupulosamente articuladas que se aplican de modo efectivo.⁹³

Años más tarde, en un artículo titulado *A Tale of Two Commons?*, Benkler recalcará la especificidad de su concepción respecto de la clásica promovida por Elinor Ostrom, subrayando que, pese a que

90. Véase (escúchese) la charla que Smári McCarthy ofreció en la Casa Invisible el 2 de noviembre de 2010, grabada por Chinowski: <https://archive.org/details/PrimeraParteSmeriMcCarthyLaCasaInvisible> (1^a parte) y <https://archive.org/details/SegundaParteSmeriMcCarthyLaCasaInvisible> (2^a parte).

91. Benkler, *La riqueza de las redes*, p. 73.

92. Véase Antonio Lafuente, «Qué es el procomún», Medialab-Prado: <https://www.medialab-prado.es/noticias/que-es-el-procomun>

93. Benkler, *La riqueza de las redes*, pp. 98–99.

esta también se adscribe al término *commons*, resulta más preciso encuadrarla en los «regímenes de propiedad comunal» (CPR, por sus siglas en inglés):

[...] el marco teórico básico de los estudios contemporáneos sobre el procomún necesita contemplar dos casos paradigmáticos distintivos que marcan nuestra comprensión del procomún. Por un lado, tenemos los pastos y los distritos de regadío que simbolizan el trabajo del que Ostrom es pionera; por otro lado, tenemos las autovías, las calles y las aceras, así como los aspectos tradicionales e incontrovertidos del dominio público. [...] Podría decirse que los CPR son más adecuados para recursos cuya escala de utilización es amplia pero definida. [...] En contraste, el procomún abierto consiste en disposiciones institucionales que abarcan una gama mucho mayor de recursos de las sociedades modernas, los cuales están generalmente abiertos a todo el público o al menos a un conjunto muy grande y ampliamente indefinido de usuarios, tanto individuales como corporativos.⁹⁴

En una línea cercana, Miquel Vidal abría las II Jornadas Copyleft de Barcelona de 2004 explicando cómo el concepto de procomún permitía dotar de sentido a muy diversas prácticas de producción cultural ampliamente accesible (en especial mediante su creciente aprovechamiento de Internet):

El viejo vocablo castellano «procomún» —que alude a los espacios y recursos colectivos cuyo aprovechamiento y gestión se realiza de forma comunal— puede servirnos de forma más precisa y general que la expresión inglesa *copyleft* para encontrar un punto de conexión entre las distintas prácticas (musicales, literarias, de software libre...) que han surgido en los últimos años frente al *copyright* restrictivo. Desde una perspectiva jurídica, todos los ciudadanos tienen acceso libre a los bienes y recursos englobados bajo el procomún, aunque deben respetar ciertas reglas (que varían en cada caso). Es un derecho civil que no se ciñe exclusivamente al ámbito mercantil,

94. Benkler, «Between Spanish Huertas and the Open Road: A Tale of Two Commons?» (artículo preparado para el congreso Convening Cultural Commons celebrado en la NYU el 23 y 23 de septiembre de 2011), pp. 1, 17–18. Disponible en: http://www.benkler.org/Commons_Unmodified_Benkler.pdf

sino que se inserta en una dinámica social mucho más amplia y compleja. De este modo, fomenta no solo el beneficio económico de los autores (como hace el *copyright*), sino también el enriquecimiento creativo y comunitario de todos los agentes implicados en los procesos de transferencia de información y conocimiento.⁹⁵

Antonio Lafuente, por su parte, subraya cómo dicho concepto nos aleja de las nociones de propiedad para acercarnos a las reflexiones antropológicas sobre comunidad(es):

El procomún, los bienes comunes —los *commons*, en inglés— sostienen y son sostenidos por colectivos humanos. Y, así, salimos de la economía y nos metemos en la antropología. [...] De la ética de los valores hemos de transitar a la de las capacidades si queremos entender cómo es la dinámica de producción del procomún, pues *un bien común no es más que una estrategia exitosa de construcción de capacidades para un colectivo humano.*⁹⁶

Con esta delineación del marco conceptual del procomún, incluido su reverso amenazante de los cercamientos, espero desterrar cualesquiera atribuciones de lo que Morozov llama «ciberutopismo» e «internet-centrismo»⁹⁷ a mi defensa de lo inspirador de la investigación de Kelty sobre el software libre. En efecto, si nos atrae el software libre es como paradigma pionero y ampliamente contrastado de la construcción de un procomún informativo que, lejos de asentarse en una Internet mágicamente predispuesta a ello, *se construye con Internet*. O dicho de otro modo, el software libre construye Internet como procomún a la vez que la atraviesa para llevar más lejos su «hacer común» o «comunalización» [*commoning*], por usar la propuesta conceptual de matriz medieval de Peter Linebaugh.⁹⁸ Esta idea nuclear (o acaso sea mejor decir *infraestructural*) en *Two Bits* la reformula magistralmente

95. Véase: <http://www2.unia.es/arteypensamiento04/ezine/ezine04/abr00.html>

96. Antonio Lafuente, «Los cuatro entornos del procomún», en *Archipiélago*, 77-78, Madrid, noviembre de 2007, p. 16. Disponible en: http://digital.csic.es/bitstream/10261/2746/1/cuatro_entornos_procomun.pdf [La cursiva es mía].

97. Evgeny Morozov, *El desengaño de internet. Los mitos de la libertad en la red*, trad. por Eugenio G. Murillo. Destino, Barcelona, 2012, pp. 18-23.

98. Peter Linebaugh, *El Manifiesto de la Carta Magna. Comunes y libertades para el pueblo*, trad. por Yaiza Hernández Velázquez y Astor Díaz Simón. Traficantes de Sueños, Madrid, 2013. Sobre la traducción de «*commoning*» en concreto, véanse pp. 62-63 (nota 46).

Marga Padilla (a la que agradezco su asesoramiento en el marco del Laboratorio del Procomún) cuando afirma que:

La complejidad de Internet no es solo un asunto técnico: es una complejidad política. La Red en sí misma es, recursivamente, a la vez el contexto y la coyuntura, a la vez lo que habla y de lo que se habla, a la vez el campo de batalla y la organización para transformar ese contexto en pro de más libertad (como viejo y nuevo derecho económico) en unas nuevas relaciones de poder.⁹⁹

Es justamente para dilucidar la compleja trabazón entre el movimiento del software libre e Internet en el marco de la «reorientación del saber y el poder» que Kelty subraya la comprensión de dicho movimiento como «creación de un “público”», cuya cualidad distintiva es una «recursividad» ligada a Internet. He aquí el origen de un concepto singularmente fecundo para abordar el interrogante de Kelty sobre la pertinencia de «otra contribución más al debate sobre el público y las esferas públicas» que abre este texto:

A diferencia de otros conceptos de público o de esfera pública, la noción de «público recursivo» refleja el hecho de que el principal modo de asociación y actuación de los *geeks* se da a través de Internet, y es precisamente a través de este medio que puede originarse un público recursivo en primer término. Internet no es en sí misma una esfera pública, un público o un público recursivo, sino una infraestructura compleja y heterogénea que constituye y restringe las aspiraciones prácticas cotidianas de los *geeks*, su capacidad para «hacerse públicos» o para componer un mundo común. [...] Ellos son quienes construyen e imaginan este espacio, y el espacio es lo que les permite construirlo e imaginarlo.

Confieso de partida que las nociones de público recursivo y recursividad me han dado algún que otro quebradero de cabeza. Y ello por más que desde el principio entreviera su potencial esclarecedor, a modo de *sustrato* conceptual para articular una reflexión sobre los diversos *estratos* presentes en dinámicas bien reconocibles en nuestro quehacer

99. Marga Padilla, *El kit de la lucha en Internet*. Traficantes de Sueños, Madrid, 2012, p. 122.

cotidiano. De este modo, han sido abundantes las sesiones de discusión del proyecto Traducciones Procomún y, más allá, las conversaciones y lecturas dedicadas a desentrañar estas nociones. De todo ello extraigo algunas notas con la esperanza de que ayuden (como me ayudaron a mí) a asimilar la brillante y detallada caracterización de dichos públicos recursivos que plantea Kelty.

Dejando de lado su definición más estrictamente técnica de base matemática, lo primero que hay que subrayar es que *la recursividad no equivale a una mera repetición*, sino que alude a la concatenación de un mismo procedimiento que va remitiéndose a un paso sucesivo hasta alcanzar un límite que cierra la cadena mediante una resolución hasta entonces diferida. La complejidad en torno a esta suerte de *operaciones con llevada* queda ampliamente despejada con esta explicación de Marga Padilla:

Hay repetición cuando algo vuelve a ocurrir independientemente de las consecuencias de lo ocurrido previamente. Y hay recursividad cuando algo vuelve a ocurrir a partir de lo ocurrido previamente. Por lo tanto, lo que hace que una repetición sea recursiva es la manera de asociarse con otras repeticiones, es decir, con otros procesos anteriores y posteriores. [...] Podemos decir que cuando hay repetición todo permanece igual, mientras que cuando hay recursividad surge algo nuevo que se da en la relación entre repeticiones consecutivas.¹⁰⁰

Una segunda fuente de orientación a este respecto se la debo a mi colega de la UMA Inmaculada Pérez de Guzmán, quien explica la recursividad como una estrategia de *modularización eficiente* que, al articular los distintos módulos copiando en cada uno de ellos la estructura del todo, apuesta por un aprovechamiento de lo ya realizado que facilita el proceso en cuestión. Marga Padilla ilustra dicha estrategia con el ejemplo del canon, «donde un tema musical es acompañado de una versión de sí mismo retrasada en el tiempo».¹⁰¹ En este sentido, Pérez de Guzmán conecta el aplazamiento inherente a la recursividad con la modalidad de cálculo que en el campo de los lenguajes de programación se denomina «evaluación perezosa», concluyendo que la recursividad

100. *Ibid.*, p. 43.

101. *Ibid.*, pp. 43–44.

no ha de entenderse como una repetición en sí, sino como *un salto*, una forma de *avanzar reutilizando*.

Acaso la ilustración más clara de ello sea la proliferación en el software libre de *acrónimos recursivos*, una peculiar forma de distinción de los nuevos programas y sistemas mediante la cita de aquellos previos en los que se basan (por ejemplo, el reconocimiento por negación a UNIX presente en las siglas GNU, «*GNU (is) Not Unix*»). La resonancia de este avanzar reutilizando con la lógica del «valerse de» [*faire avec*] expuesta previamente me resulta harto significativa, por cuanto la recursividad así entendida se asemeja a la concepción táctica de Michel de Certeau que apela a golpes basados en una astuta reactualización de la memoria y en el imperativo de no poder permitirse la mera repetición de lo que funcionó en otra coyuntura previa.

A partir de estos breves apuntes introductorios, llegamos a la definición de Kelty de público recursivo como:

Un público que está vitalmente implicado en la conservación y modificación material y práctica de los medios técnicos, legales, prácticos y conceptuales de su propia existencia como público; se trata de un colectivo independiente de otras formas de poder constituido y capaz de dirigirse a las formas existentes de poder mediante la producción de alternativas realmente existentes.

Estamos, pues, ante una propuesta conceptual que, aunque bebe de las prácticas concretas de la creación de software y redes, pretende indagar su singular *trascendencia cultural* o, si se prefiere, el motivo por el que el descubrimiento del software libre nos lleva a muchos «conversos» (especialmente a los menos *techies*) a preguntarnos cómo «portarlo» a los campos a los que nos dedicamos. Sin ir más lejos, el proyecto Traducciones Procomún del que surge esta edición en castellano de *Two Bits* nace justamente de esta «revelación» a la que alude Kelty.

Junto a ello, la concepción del público recursivo como un tipo de imaginario social que se desenvuelve en una suerte de «enmedio»¹⁰² entre sistemas operativos y sistemas sociales permite a Kelty cuestionar el logocentrismo de determinadas concepciones de la acción y expresión políticas señalando que «los *geeks* expresan ideas, pero también expresan *infraestructuras* mediante las que las ideas pueden manifestarse (y difundirse) de nuevas maneras».

102. Raunig, *op. cit.*

Tal insistencia de Kelty en «la indómita materialidad técnica de un orden político» para evitar la estéril contraposición entre prácticas e ideas nos devuelve a la caracterización de los modos de hacer como «un estilo de intercambios sociales, un estilo de invenciones técnicas y un estilo de resistencia moral, es decir, una economía del ‘don’ (generosidades *à charge de revanche*), una estética de ‘golpes’ (operaciones de artistas) y una ética de la tenacidad (mil maneras de negarle al orden establecido el estatuto de ley, de sentido o de fatalidad)». ¹⁰³

He aquí el vínculo entre modos de hacer y públicos recursivos por el que sostengo que la teorización de Kelty está llamada a enriquecer los debates y quehaceres actuales vinculados, por ejemplo, a la reclamación de democracia (real ya), a los centros sociales autónomos (desde luego con el edificio de la Casa Invisible hemos aprendido bien que *la rehabilitación es política*), al movimiento de personas con discapacidad¹⁰⁴ o al propio feminismo (cuyo acento en la reproducción y el sostenimiento de la vida abraza sin ambages la materialidad/corporeidad inherente a los cuidados). ¿O acaso cabe desdeñar observaciones como la que sigue sobre el «habitar» de los públicos recursivos como algo cuya interpelación política se ciñe al ámbito *geek*?:

Los públicos recursivos responden a la cuestión gubernamental de forma directa implicándose en, manteniendo y a menudo modificando la infraestructura que pretenden, en cuanto público, habitar y extender —y no solo ofreciendo opiniones o protestando contra las decisiones, como hacen los públicos convencionales (en la mayoría de teorías de la esfera pública). [...] Los públicos recursivos [...] desean inventar modos de dotar al terreno de juego de un cierto tipo de capacidad de acción, efectuada a través de la capacidad de acción de muchos seres humanos diferentes, pero revisada mediante su estructura y apertura técnica y legal. [...] Se trata de una ética de la justicia entreverada con una estética de elegancia técnica y astucia legal.

103. M. de Certeau, *L'invention du quotidien*, vol. 1., p. 46.

104. A este respecto, véase la fundamental contribución de Langdon Winner, «Is there a right to shape technology?», *Argumentos de Razón Técnica*, 10, 2007, pp. 199-213. Disponible en: http://institucional.us.es/revistas/argumentos/10/art_11_rea10.pdf

Modos de hacer y *modulaciones*: Traducciones Procomún

Hablar de la trascendencia cultural del software libre implica necesariamente ocuparse de un último concepto clave de la obra de Kelty: la «modulación». Con esta noción el antropólogo de UCLA alude a la reutilización en otros ámbitos de las prácticas constitutivas del software libre (entendido como un «sistema experimental») bajo la guía de las imaginaciones de orden moral y técnico comunes a los públicos recursivos. En este punto, recalco una vez más cómo las modulaciones de las que habla *Two Bits* pueden interpretarse a la luz de los dos aspectos de la caracterización de los modos de hacer que vimos arriba, a saber, la reutilización y la táctica.

En cuanto a la *reutilización*, este aspecto remite a la conceptualización de Kelty del software libre como «un sistema técnico experimental de carácter colectivo» que, justamente por su imbricación con la reorientación del saber y el poder, lleva incorporadas *de serie* su accesibilidad y sobre todo su *modificabilidad*. En consonancia con ello, el autor estadounidense no ofrece una definición propiamente dicha del software libre sino una caracterización de las cinco prácticas que, en su contingencia y evolución constante, componen dicho sistema en el momento histórico específico (finales de los 90) en que se da su gran expansión. Dichas cinco prácticas son la creación de un movimiento (los debates sobre qué significa software libre), la compartición de código fuente, la concepción de sistemas abiertos, la redacción de licencias de *copyright* y la coordinación de colaboraciones.

En relación a la *táctica*, conviene recordar que dichas modulaciones rara vez se dan en un contexto libre de controversias y disputas, ya sean económicas, legales, políticas, morales, etc. En suma, el software libre y sus modulaciones tampoco están en condiciones de postular un lugar propio desde donde formularse y experimentar sin cortapisas, sino que se mueven en un terreno adverso donde confluyen actores muy asentados y poderosos con los que hay que lidiar. No en vano, al referirse a las dos modulaciones de las que se ocupa la Tercera Parte (*Connexions* y *Creative Commons*), Kelty observa que, por el hecho de adentrarse en el otrora coto reservado de las industrias culturales, ambos proyectos se ven «forzados a ser opositores» y desafiar un *status quo* con el que no comulgan.

A partir de lo expuesto, concluiré esbozando en qué medida el proyecto Traducciones Procomún del que nace esta edición de *Two Bits* supone una modulación del software libre (aunque, por supuesto,

nadie supiera qué era «modular» cuando lo emprendimos). Para ello confrontaré las cinco prácticas descritas por Kelty con los elementos de lo que he llamado la «Receta del proyecto *Traducciones Procomún*»,¹⁰⁵ directamente inspirada (a sugerencia de Fátima Solera) en el célebre símil con que Stallman ilustra la naturaleza del código fuente.¹⁰⁶ En última instancia la propia Fátima plasmaría las claves de nuestra receta en una imagen que incluyo a modo de síntesis de lo que explicaré a continuación.

Empezando por la práctica del *movimiento*, sin duda nuestro proyecto nacido en las aulas de la UMA en 2008 nunca generó (ni lo pretendió) nada parecido a los encendidos debates que alumbraron a finales de los 90 el movimiento del software libre. De este modo, si algún tipo de afinidad intelectual marcó nuestro hacer común, esta se generó sobre la marcha, sin buscarse de antemano (¿cómo podría hacerlo un docente que no escoge a su alumnado?). Ello concuerda con la crucial observación de Kelty de que «los *geeks* no parten de ideologías, sino que más bien llegan a ellas a través de su implicación en las prácticas de creación del software libre y sus derivados».

Dicho esto, creo que el autor de *Two Bits* sigue teniendo razón más de una década después en su observación de que no existe como tal ningún movimiento de «libros de texto libres», por más que haya habido un puñado de meritarias tentativas al respecto (en España cabe destacar las ligadas a la Marea Verde). En todo caso, como también apunta Kelty, hemos encontrado «un primo segundo bastante cercano» en el movimiento de acceso abierto, este sí bastante consolidado y del que hemos aprendido mucho gracias a la incorporación de nuestro proyecto al Laboratorio del Procomún de Medialab-Prado en 2010. En definitiva, lo más cercano a una definición de las motivaciones y la misión que compartimos quienes nos sumamos al proyecto Traducciones Procomún es lo que di en llamar el «*hambre* de aprender e investigar, hambre de compartir esa hambre y hambre de experimentar cómo saciarlas cooperativamente».¹⁰⁷

105. Florencio Cabello, «Receta del proyecto Traducciones Procomún: Cocinas recursivas de conocimiento entre iguales». *Teknokultura*, 10(1), 2013, pp. 265–276. Disponible en: <http://revistas.ucm.es/index.php/TEKN/article/view/48066/44943>

106. Véase, por ejemplo, Richard Stallman, *Software libre para una sociedad libre*, trad. por Jaron Rowan, Diego Sanz Paratcha y Laura Trinidad. Madrid, Traficantes de Sueños, Madrid, 2004, pp. 225-226.

107. Cabello, «Receta del proyecto Traducciones Procomún», p. 269.

Traducciones Procomún



Ten hambre de...
aprender a investigar...
aprender como
componer relativamente
tu hambre comunitaria.
Busca a quienes
estoyes...
Estoyes...
Es muy importante que
dilundas ampliamente
tu convocatoria y que la
abras a multiples perfiles...
propios.



Acepta solo productos
con denominación de
origen procomún.
Puedes encontrar
productos exquisitos de
ilustres chefs cuya
creatividad te permite
realizar obras derivadas.



UTENSILIOS DE COCINA

Empieza utensilios de
cocina libres (Nº1,
ElHerpad, Kune, etc.).
Para alimenter tus
hambres, toma
utensilios libres, ayuda
a que sus
desarrolladores los me-
joren y
comparte con ellos un
pedacito de tu tarta.
(compinches).



TRADUCIÓN ENTRE GUALES

Aprovecha las
despensas informáticas
de los compinches
para alimenter tus
hambres, toma
utensilios libres, ayuda
a que sus
desarrolladores los me-
joren y
comparte con ellos un
pedacito de tu tarta.
(compinches).



INGREDIENTES PROCOMÚN

Ofrece los platos
derivados de dichos
productos al procomún.
La colaboración con
otros proyectos
basadas en el
procomún, enriquecerá
tu selección
de productos.



Colabora



MEDIALAB
PRADO



Colectivo
de Recetas
Comunes



Encuentro Cebolla

Encuentro Cebolla

Kune
Cartel diseñado por Fátima Solera

En cuanto a los componentes de *compartición del código fuente y redacción de licencias de copyright*, su modulación nos venía servida *a plato puesto*: Respecto al código fuente, nos hallamos ante los mismos condicionantes que Kelty describe para Connexions, en la medida en que ambos proyectos tratamos con un contenido libresco (en nuestro caso, obras de mayor envergadura que los manuales al uso), sin que nos hiciera falta elaborar programas informáticos específicos. Tampoco con las licencias tuvimos necesidad de experimentar en la medida en que cuando nuestro proyecto nace Creative Commons estaba ya plenamente consolidado como proveedor de un abanico de licencias abiertas fruto de una delicada modulación de la licencia GPL. Si a todo lo anterior unimos que autores reconocidos (antes Lessig y Benkler y ahora Kelty) apostaron por licencias que consagraban la *modificabilidad* de sus obras, lo único que cabía incluir en el apartado «Ingredientes» de nuestra receta era este aviso: –Acepta solo productos con denominación de origen procomún. –Puedes encontrar productos exquisitos de ilustres *chefs* cuyas licencias Creative Commons te permiten usarlos para realizar obras derivadas». ¹⁰⁸ En suma, nuestro proyecto se ve plenamente reconocido en la inversión que Kelty plantea del citado lema «La información quiere ser libre»: «el acto de compartir produce su propio tipo de orden moral y técnico, es decir, ‘la información hace que las personas quieran libertad’».

En contraste con lo anterior, la modulación relativa a la *concepción de la apertura* sí nos dio más juego para experimentar. Por supuesto, partíamos de un consenso más o menos intuitivo (tanto entre mi alumnado, por razones obvias, como entre el resto de colaboradores) acerca de la inexcusabilidad de que el profesorado pagado con fondos públicos difunda su trabajo abiertamente para provecho común.¹⁰⁹ Más allá de esto, el *quid* de la experimentación radicaba en lo que Kelty denomina «la profundidad recursiva», esto es, el compromiso de llevar a todas las capas posibles el compromiso con la apertura. En este sentido, nuestra receta contenía algunas instrucciones sobre los «utensilios de cocina» más técnicos (los formatos, programas y plataformas donde volcar nuestro trabajo): «-Monta una cocina virtual para coordinar

108. *Ibid.*, p. 271.

109. Véase a este respecto el llamamiento que la organización LERU publicó el 12 de octubre de 2015 con el explícito título «Se acabó la Navidad. ¡La financiación investigadora debería dedicarse a la investigación, no a las editoriales!»: <https://www.leru.org/files/LERU-Statement-Moving-Forwards-on-Open-Access2.pdf>

las aportaciones de todos los (com)pinches. –Descarta ubicar dicha plataforma en despensas en la nube que te privan del control de tu producción. [...] –Emplea utensilios de cocina libres. Recomendamos específicamente coordinar el trabajo abriendo un grupo en la red social libre n-1 y emplear el editor cooperativo libre Etherpad que esta incorpora».¹¹⁰ Lamentablemente esta última recomendación ha quedado desfasada por la desaparición de n-1, pero deseo destacar lo estimulante que fue experimentar con esta pionera red social libre (el propio Kelty se registró para observarnos trabajar) y ello hasta la confirmación última (literalmente) de la virtud de esta recursividad culinaria: gracias al empeño de Alex Haché y Sem Brestel, tras el cierre de la plataforma pudimos rescatar absolutamente todo el trabajo volcado en los *pads* sin que se lo tragara ninguna nube comercial.

En cuanto al último componente, la *coordinación de colaboraciones*, está íntimamente ligada con la reflexión previa sobre los «utensilios de cocina» en la medida en que, como afirma Kelty, «la coordinación es importante porque desarma y resuelve la distinción entre formas técnicas y sociales en [...] una significativa práctica tecnosocial de gestión, toma de decisiones y rendición de cuentas». En el caso de Traducciones Procomún, la colaboración entre múltiples voluntarios venía obviamente marcada por el hecho de que algunos de ellos en algún momento debían recibir de mí su calificación, si bien he de matizar que la participación era opcional y que solía extenderse más allá del cuatrimestre en que yo daba mis clases. En última instancia, a partir de la traducción de Benkler y especialmente en la de Kelty, mi alumnado llegaría a ser minoritario dentro del grupo de colaboradores. Como colofón de estos apuntes, concluyo con una cita extensa acerca de las claves de coordinación de nuestras cocinas recursivas:

- Trocea los ingredientes originales hasta obtener módulos coherentes susceptibles de ser cocinados de modo independiente. En el caso de libros, puedes partir de la estructura de capítulos.
- Afina tus módulos hasta obtener trozos de distinta envergadura adaptados a la disponibilidad de los (com)pinches. [...]
- Es fundamental que todos los (com)pinches registren sus hallazgos en el Glosario común para evitar la duplicidad de esfuerzos.

110. Cabello, «Receta del proyecto Traducciones Procomún», p. 272.

- Adereza el proceso cooperativo con una dosis de jerarquía para las funciones de grano más grueso: selección de materias primas y contacto con autores, convocatoria, diseño de la cocina virtual, coordinación virtual y presencial, revisión y organización de la difusión.
- El intercambio continuo de recetas con otros proyectos basados en el procomún enriquecerá tu repertorio de modos de hacer.¹¹¹

Aquí queda, pues, nuestra receta y tan solo resta decir: ¡Que aproveche!

111. *Ibid.*, pp. 274-275.

PREFACIO

Este es un libro sobre software libre, también conocido como software de código abierto, y va dirigido a cualquiera que desee comprender la trascendencia cultural del software libre. *Two Bits*¹ explica cómo funciona el software libre y cómo ha surgido en tandem con Internet como una forma tanto técnica como social. Comprender en detalle el software libre constituye el mejor modo de comprender muchas transformaciones polémicas y confusas vinculadas a Internet, el «procomún» [«commons»], el software y las redes. Tanto si lo primero que nos viene a la mente es el correo electrónico, Napster, Wikipedia, MySpace o Flickr; como si es la proliferación de bases de datos, robos de identidad e inquietudes en torno a la privacidad; como si se trata del conocimiento tradicional, las patentes genéticas, la muerte de las publicaciones científicas o las licencias obligatorias sobre medicamentos contra el sida; como si pensamos en MoveOn.org, la neutralidad de la Red o Youtube; el caso es que las cuestiones suscitadas por todos estos fenómenos pueden comprenderse mejor observando cuidadosamente el surgimiento del software libre.

1. Una de las primeras decisiones de traducción que discutimos en nuestras sesiones de trabajo en Medialab-Prado atañía al título mismo de la obra. Finalmente la opción que se vio menos infiel al sentido de *Two Bits* fue *dejarlo sin traducir*. En efecto, más allá de la parca literalidad del «dos bits» (como si de dígitos binarios habláramos) o de la alusión a la moneda de cuarto de dólar (equivalente a «dos ochavos», por usar la palabra y la moneda española equiparables en las que se basan los estadounidenses), la expresión «two bits» posee un significado específico en el contexto del póker. Así, alguien pone sus «two bits» como *apuesta inicial* con la que entrar en una partida y recibir cartas, lo cual se ha trasladado al lenguaje coloquial como una suerte de aportación originaria (como un «granito de arena») que alguien ha de realizar para incorporarse a una conversación, proyecto, etc. Actualmente es más común leer en inglés la expresión «*my two cents*» con un sentido similar. Nuestra principal fuente para entender estos matices fue: <https://www.urbandictionary.com/define.php?term=Two-Bit> [N. del E.]

¿Por qué? Porque es en el software libre y en su historia donde dichas cuestiones —desde la propiedad intelectual y la piratería hasta la movilización política en línea y el software «social»— se figuraron y afrontaron por vez primera. Las raíces del software libre se remontan a los años 70 del siglo XX y entrelazan las historias del ordenador personal y de Internet, los altibajos de la tecnología de la información y de las industrias informáticas, la transformación de la legislación de propiedad intelectual, la innovación organizativa y la colaboración «virtual», así como el auge de los movimientos sociales en red. El software libre no explica por qué han ocurrido tan diversos cambios, sino más bien cómo los individuos y grupos responden a ellos: mediante la creación de nuevos objetos, nuevas prácticas y nuevas formas de vida. Son estas prácticas y formas de vida —y no el software en sí mismo— las que resultan más significativas, sirviendo a su vez como patrones que otros pueden usar y transformar: prácticas de compartición de código fuente, de concepción de la apertura, de redacción de licencias de *copyright* (y de *copyleft*) y de proselitismo de todo lo anterior. Existen explicaciones para todos los gustos sobre *por qué* las cosas son como son: por la mundialización, por la sociedad red, por una ideología de transparencia, por la virtualización del trabajo, por la reformulación de la noción de la Tierra plana, por el Imperio. Estamos atiborrados de *porqué*s, tanto populares como académicos, pero ávidos de *cómo*s.

Comprender cómo funciona el software libre no es simplemente una empresa académica sino una experiencia que transforma las vidas y el trabajo de los participantes involucrados. A lo largo de la última década, consagrada al trabajo de campo con programadores informáticos, abogados, empresarios, artistas, activistas y otros *geeks*,² he observado

2. Opto por mantener algunos términos en inglés por la complejidad de matices que poseen. En este caso, «geek» no termina de encajar con equivalentes aproximados en castellano como «aficionado a la informática», «experto informático» o incluso con el coloquial «friki informático». En este sentido, y con el propósito de reflejar (algo de) dicha complejidad, recurro a la definición de «geek» que Eric S. Raymond incluye en la versión 4.4.7 del *Jargon File* («Archivo de Jerga», en inglés), un glosario de argot *hacker* de cuyos mantenimiento y actualización se encarga con la ayuda de Guy Steele desde principios de los 90. Deseo subrayar que el propio Chris Kelty me animó a recurrir al *Jargon File* para estas notas aclaratorias por suponer una forma de *definición recursiva* (*hackers* caracterizando a *hackers*) plenamente pertinente con la máxima «Eres *hacker* cuando otro *hacker* te llama *hacker*»:

Geek: Persona que ha elegido la concentración en vez de la conformidad; alguien que busca la destreza (especialmente la destreza técnica) y la imaginación, y no una amplia aprobación social. [...] La mayoría de *geeks* es experta en ordenadores y emplea *hacker* como un término respetuoso, pero no todos ellos son *hackers* —y en cualquier caso al-

repetidamente que la comprensión del funcionamiento del software libre da lugar a una revelación. Las personas —incluso (o quizás especialmente) aquellas que no se consideran programadoras, *hackers*, *geeks* o tecnófilas— extraen de la experiencia una especie de religión, porque la clave del software libre reside en las prácticas, no en las ideologías y objetivos que se arremolinan en su superficie. El software libre y sus creadores y usuarios no son, en cuanto grupo, antimercantiles o anticomerciales, anti-propiedad intelectual o antiestatales; en definitiva, en cuanto grupo, no están en contra o a favor de nada. De hecho, ni siquiera constituyen un grupo: no son ni una empresa ni una organización; ni una ONG ni una agencia estatal; ni una sociedad profesional ni una horda informal de *hackers*; ni un movimiento ni un proyecto de investigación.

Y sin embargo, el software libre es público, consiste en hacer públicos sus desarrollos. Este hecho es esencial para captar su trascendencia cultural, su atractivo y su proliferación. Ahora bien, el software libre es público de una forma particular: se trata de un modo autónomo, colectivo y políticamente independiente de crear objetos técnicos muy complejos que se ponen a libre y pública disposición de todo el mundo —un «procomún», en lenguaje común. Constituye una práctica de profundización en las promesas de igualdad, justicia, razón y argumentación en un ámbito de redes y programas técnicamente complejos, y en un contexto de leyes de propiedad intelectual poderosas y desequilibradas. El hecho de que algo público en este sentido amplio surja de prácticas tan aparentemente arcanas es la razón de que el primer impulso de muchos conversos sea preguntar cómo puede «portarse» el software libre a otros aspectos de la vida, como las películas, la música, la ciencia y la medicina, la sociedad civil y la educación. Son este impulso proselitista y la facilidad con que se diseminan dichas prácticas los que configuran la trascendencia cultural del software libre. Para bien o para mal, puede que todos nosotros estemos usando software libre antes de darnos cuenta.

gunos que de hecho son *hackers* normalmente se llaman a sí mismos *geeks*, ya que (muy oportunamente) consideran «*hacker*» como una catalogación que debería ser otorgada por otros más que adoptada por uno mismo—. [...] Versiones previas de este léxico definían a un *geek* informático como [...] un monomaníaco asocial, apéstoso y paliducho con toda la personalidad de un rallador de queso. Todavía es frecuente que los no-*geeks* consideren de esta forma a los *geeks*, si bien a medida que la cultura mayoritaria se vuelve más dependiente de la tecnología y de la destreza técnica, las actitudes mayoritarias han tendido a derivar hacia el respeto reticente.

Disponible en: <http://www.catb.org/~esr/jargon/html/G/geek.html> (última consulta: 18 de abril de 2018). [N. del E.]

AGRADECIMIENTOS

La antropología es una ciencia dependiente de extraños que se convierten en amigos y colegas —extraños que aportan la esencia misma del trabajo. En mi caso, estos extraños demuestran también una hiperconciencia en lo relativo al mérito, la reputación, el reconocimiento, la reutilización y la modificación de ideas y objetos. Por consiguiente, esta lista es exhaustiva y detallada.

Sean Doyle y Adrian Gropper abrieron las puertas a este proyecto, proporcionando una perspicacia, hospitalidad, estímulo y curiosidad sin parangón. Axel Roch me presentó a Volker Grassmuck y a mucha más gente. Volker Grassmuck me introdujo en el mundo del software libre berlínés y me invitó a participar en las jornadas *Wizards of OS* (Magos del Código Abierto). Udhay Shankar me presentó a prácticamente toda la gente que conozco, a veces *a posteriori*. Shiv Sastry me ayudó a encontrar alojamiento en Bangalore en casa de su tía Anasuya Satry, que recibe el nombre de «Silicon Valley» y que fue verdaderamente una residencia deliciosa. Bharath Chari y Ram Sundaram me dejaron frecuentar su oficina y sus cables cat-5 durante uno de los periodos más turbulentos de sus carreras. Glenn Otis Brown me visitó, bebió y charló conmigo, me invitó, me estimuló, me entretuvo, me reprendió, me animó, me llevó en su coche, se montó en el mío, me ofreció consejos y recibió los míos. Ross Reedstrom me dio la bienvenida al Grupo de Usuarios de Linux de la Universidad de Rice y a Connexions. Brent Hendricks realizó un trabajo ímpreprobo, soportando mis preguntas e intrusiones. Geneva Henry, Jenn Drummond, Chuck Bearden, Kathy Fletcher, Manpreet Kaur, Mark Husband, Max Starkenberg, Elvena Mayo, Joey King y Joel Thierstein se han mostrado hospitalarios y entusiastas en cada encuentro. Sid Burris ha estimulado y respetado mi trabajo, lo cual ha sido un

honor. Rich Baraniuk escucha todo lo que digo, para bien o para mal; es un magnífico colaborador y amigo.

James Boyle ha sido un apoyo constante, por más que sienta que ha sacado escaso rédito de ello. Muy poca gente es capaz de leer y criticar y ayudar a rehacer la argumentación y la estructura de un libro, y a la vez de aparecer en él. Mario Biagioli me ayudó a apreciar la intrincada estrategia descrita en el Capítulo 6. Stefan Helmreich leyó los primeros borradores y transformó mi concepción de las redes. Manuel DeLanda me explicó el concepto de *ensamblaje [assemblage]*. James Faubion corrigió mi razonamiento del Capítulo 2, me ayudó enormemente con los protestantes y ha sido un colega y jefe de departamento exquisitamente solícito. Mazyar Lotfalian y Melissa Cefkin me cedieron su apartamento y su biblioteca, donde escribí gran parte del Capítulo 1. Matt Price y Michelle Murphy han escuchado pacientemente cómo construía y reconstruía versiones de este libro al menos durante seis años. Tom y Elizabeth Landecker me brindaron hospitalidad y un entorno asombrosamente bello para reescribir varias partes de la obra. Lisa Gitelman leyó cuidadosamente y me ayudó a explicar la cuestión de la documentación y las versiones que trato en el Capítulo 4. Matt Ratto leyó y comentó del Capítulo 4 al 7, me convenció para desechar una distinción inútil y para clarificar la conclusión del Capítulo 7. Shay David me proporcionó apreciaciones estratégicas sobre la apertura a partir de su propio trabajo y me impulsó a explicar la noción de públicos recursivos con mayor claridad. Biella Coleman ha sido una interlocutora constante sobre la temática de este libro (sus contribuciones son demasiado profundas, demasiado variadas y demasiado exhaustivas para detallarlas). Su propio trabajo sobre el software libre y los *hackers* ha sido una constante caja de resonancia y guía, y ha sido un placer trabajar juntos en nuestros respectivos textos. Kim Fortun ayudó a que me figurara todo esto.

George Marcus me contrató en un fantástico departamento de Antropología y ha tenido una inmensa fe en este proyecto durante todo su desarrollo. Paul Rabinow, Stephen Collier y Andrew Lakoff me han brindado un marco extremadamente valioso (el Colaboratorio de Investigación sobre Antropología de lo Contemporáneo) en cuyo seno los planteamientos de este libro se desarrollaron de modos que habrían sido imposibles de tratarse de un proyecto en solitario. Joe Dumit me ha animado, espoleado, cuestionado, aportado ideas, guiado e inspirado. Michael Fischer es el mejor mentor y asesor que jamás haya existido. Lo ha leído todo, ha escrito mucho de lo que

precede y da forma a este trabajo y ha sido un apoyo firme y un amigo todo este tiempo.

Tish Stringer, Michael Powell, Valerie Olson, Ala Alazze, Lina Dib, Angela Rivas, Anthony Potoczniak, Ayla Samli, Ebru Kayaalp, Michael Kriz, Erkan Saka, Elise McCarthy, Elitza Ranova, Amanda Randall, Kris Peterson, Laura Jones, Nahal Naficy, Andrea Frolic y Casey O'Donnell hacen genial mi trabajo. Scott McGill, Sarah Ellen-zweig, Stephen Collier, Carl Pearson, Dan Wallach, Tracy Volz, Rich Doyle, Ussama Makdisi, Elora Shehabudin, Michael Morrow, Taryn Kinney, Gregory Kaplan, Jane Greenberg, Hajime Nakatani, Kirsten Ostherr, Henning Schmidgen, Jason Danziger, Kayte Young, Nicholas King, Jennifer Fishman, Paul Drueke, Roberta Bivins, Sherri Roush, Stefan Timmermans, Laura Lark y Susann Wilkinson hicieron de la ciudad de Houston un lugar maravilloso para vivir o bien me brindaron una oportunidad de huir de ella. Me siento especialmente feliz de que Thom Chivens haya hecho ambas cosas y aún más.

El Centro para el Estudio de las Culturas me concedió una ayuda de investigación en otoño de 2003, lo cual me permitió llevar a término gran parte del trabajo de conceptualización del libro. El Departamento de Historia de la Ciencia de Harvard y el Programa de Historia, Antropología y Estudio Sociales de Ciencia y Tecnología del MIT (*Massachusetts Institute of Technology*, Instituto Tecnológico de Massachusetts) me acogieron en la primavera de 2005, permitiéndome escribir la mayor parte de los capítulos 7, 8 y 9. La Universidad de Rice ha sido extremadamente generosa en todos los aspectos, y un maravilloso lugar para trabajar. Me siento infinitamente agradecido por haber disfrutado de un año sabático júnior que me dio la oportunidad de completar gran parte de este libro. John Hoffman permitió gentil y generosamente el uso del nombre de dominio twobits.net, como apoyo al software libre. Ken Wissoker, Courtney Berger y los revisores anónimos de Duke University Press han hecho de este libro algo mucho, mucho mejor de lo que era cuando lo empecé.

Mis padres, Ted y Anne, y mi hermano, Kevin, me han demostrado siempre su apoyo y su cariño; por más que afirmen que no tienen ni idea de lo que hago, debo mi pequeño éxito a su constante apoyo. Hannah Landecker ha leído y releído y reescrito cada parte de este trabajo; ella nos ha hecho a él y a mí mejores, y la amo con ternura por eso. Por último, si bien no menos importante, mi nuevo proyecto, Ida Jane Kelty Landecker, es mucho más linda y aguda y divertida que *Two Bits*, y la quiero por distraerme de él.

INTRODUCCIÓN

En torno a 1998 el software libre emergió de una existencia felizmente subterránea y oscura que se remontaba aproximadamente a veinte años atrás. En pleno apogeo del *boom* puntocom, de repente el software libre poblaba las páginas de las principales revistas de negocios, se introducía en las discusiones de los ejecutivos sobre estrategia y planificación, confundía los radares de los líderes políticos y reguladores de todo el mundo y permeaba la conciencia de una generación de adolescentes tecnófilos que crecía preguntándose cómo había podido vivir la gente sin correo electrónico. El software libre parecía algo chocante, algo que la historia económica sugería que jamás podría existir: una práctica de creación de software —buen software— cuya propiedad era privada, pero al que se podía acceder libre y públicamente. El software libre, como sugiere su ambiguo apelativo en inglés [«free»], está tanto libre de restricciones como libre de pago. Tales características parecen violar la lógica económica y los principios de propiedad privada y autonomía individual, y sin embargo el software libre cuenta con decenas de millones de desarrolladores y cientos de millones de usuarios. ¿Por qué? ¿Por qué ahora? Y lo que es más importante: ¿Cómo?

El software libre consiste en un conjunto de prácticas para la creación cooperativa y distribuida de código fuente, el cual luego se difunde de forma abierta y libre a través de un uso astuto e insólito de la legislación de *copyright*.¹ Pero es mucho más que eso: el software

1. A lo largo de este volumen, algunas referencias se citarán mediante su «Identificador de mensaje», lo cual debería permitir que cualquier persona interesada acceda a los mensajes originales a través de Google Groups (<http://groups.google.com>).

Una nota sobre terminología: Sigue vivo el debate sobre cómo referirse al software libre, también conocido como software de código abierto. La comunidad académica ha adoptado

libre ejemplifica una considerable reorientación del saber y el poder en la sociedad contemporánea —una reorientación del poder con respecto a la creación, diseminación y autorización del saber en la era de Internet. Esta obra trata de la trascendencia cultural del software libre, y por *cultural* entiendo mucho más que la conducta exótica o las peculiaridades indumentarias de los programadores informáticos, por fascinantes que puedan ser. Entiendo por *cultura* un sistema experimental en curso, un espacio de modificación y modulación, de descodificación y de prueba; la cultura es un experimento difícil de seguir de cerca, sometido a cambios veloces y a veces bruscos. En cuanto sistema experimental, la cultura atraviesa economías y gobiernos, esferas sociales en red, así como la infraestructura de saber y poder en la que actualmente funciona —o deja de funcionar— nuestro mundo. El software libre, como práctica cultural, entreteje un sorprendente repertorio de lugares, objetos y personas, y contiene patrones, umbrales y repeticiones que no son simples o inmediatamente obvios, ni siquiera para los *geeks* que escriben software libre o que desean entenderlo. Mi propósito en este libro es revelar algunos de esos complejos patrones y umbrales, desde una perspectiva tanto histórica como antropológica, y explicar no solo qué es el software libre sino cómo ha surgido en el pasado reciente y cómo continuará cambiando en el futuro próximo.²

La trascendencia del software libre alcanza mucho más allá de las prácticas técnicas arcanas y detalladas de los programadores informáticos y «*geeks*» (tal y como me refiero a ellos aquí). Y es que desde 1998, las prácticas e ideas del software libre se han expandido a nuevas esferas vitales y creativas: desde el software a la música y el cine, de ahí a la ciencia, la ingeniería y la educación; desde la política nacional de propiedad intelectual a los debates mundiales sobre la sociedad civil; desde UNIX a Mac OS X y Windows; desde los registros y bases de datos médicos al seguimiento internacional de enfermedades y la biología

dos fórmulas: la anglosajona FOSS [*Free and Open Source Software*, software libre y de código abierto] y la continental FLOSS (o F/LOSS) [*Free, Libre and Open Source Software*, redundancia que en castellano se traduciría igual]. *Two Bits* se ciñe a la simple denominación *software libre* para referirse a todas estas variantes, excepto cuando es específicamente necesario distinguir entre dos o más nombres, o precisar personas o eventos así aludidos. La razón es primordialmente estética y política, pero también influye que *software libre* es el término más antiguo, además de aquel que comprende cuestiones de orden moral y social. En el Capítulo 3 explicaré por qué existen dos términos para referirse a él.

2. Michael M. J. Fischer, «Culture and Cultural Analysis as Experimental Systems».

sintética; desde el código abierto al acceso abierto. El software libre ya no solo afecta al software, sino que ejemplifica una reorientación más general del poder y el saber.

Los términos *software libre* y *código abierto* no terminan de aprender el alcance de esta reorientación o su propia trascendencia cultural, sino que se refieren, de forma bastante estrecha, a la práctica de crear software —actividad que mucha gente considera bastante alejada de su experiencia. Sin embargo, la creación de software libre es mucho más que eso: incluye una combinación única de prácticas más familiares que abarcan desde la creación y la custodia de la propiedad intelectual hasta la discusión sobre el significado de «apertura» o la organización y coordinación de personas y máquinas a través de lugares y zonas horarias. Observadas en su conjunto, estas prácticas hacen del software libre algo distinto, importante y significativo tanto para quienes lo crean como para quienes se toman el tiempo de comprender cómo llega a nacer.

Con el fin de analizar e ilustrar la trascendencia cultural más general del software libre y sus consecuencias, introduzco el concepto de «público recursivo». Un público recursivo es *un público que está vitalmente implicado en la conservación y modificación material y práctica de los medios técnicos, legales, prácticos y conceptuales de su propia existencia como público; se trata de un colectivo independiente de otras formas de poder constituido y capaz de dirigirse a las formas existentes de poder mediante la producción de alternativas realmente existentes*. El software libre constituye un ejemplo de este concepto, tanto en su aparición en el pasado reciente como en su transformación y diferenciación en el futuro próximo. Pero existen otros, incluyendo los que surgen de las prácticas del software libre, como Creative Commons, el proyecto Connexions y el movimiento de acceso abierto en la ciencia. Estos últimos ejemplos pueden o no vincularse al software libre, o incluso pueden no ser *per se* proyectos de «software», pero están conectados por las mismas prácticas, y lo que los hace relevantes es que pueden constituir también «públicos recursivos» en el sentido que exploro en esta obra. Los públicos recursivos, y los públicos en general, difieren de los grupos de interés, las empresas, los sindicatos, las sociedades profesionales, las iglesias y otras formas de organización en su atención hacia la mutabilidad tecnológica radical de sus propias condiciones de existencia. En cualquier público inevitablemente llega un momento en que surge la preocupación

por cuestiones relativas a cómo se expresan las cosas, quién controla los medios de comunicación o si se atiende adecuadamente a todas y cada una de las voces. Una esfera pública legítima es aquella que da cabida a los extraños: su opinión puede ser atendida o no, pero no tienen que apelar a ninguna autoridad (interna o externa a la organización) para tener voz.³ Tales públicos no son inherentemente modificables, sino que son las prácticas de los participantes las que los vuelven —y mantienen— así. Es posible que el software libre tal y como lo conocemos deje de ser público, o que se convierta en una forma establecida más de poder, pero este libro se centra en el pasado reciente y en el futuro próximo de algo que es (hasta la fecha) público de un modo radical y novedoso.

El concepto de público recursivo no pretende referirse a cualquier ejemplo de público —no se trata de un sustituto del concepto de «esfera pública»—, sino más bien dar a los lectores una idea específica y detallada de los hilos no evidentes pero persistentes que conforman la urdimbre del software libre, y analizar proyectos similares y relaciona-

3. Así, por ejemplo, cuando una sociedad profesional instituida a partir de actas fundacionales e ideales referentes a la afiliación y cualificación habla como un público, representa a sus miembros, como cuando la Asociación Médica Estadounidense se posiciona a favor o en contra del programa de seguridad social Medicare. Sin embargo, si un grupo nuevo —de enfermeros, pongamos por caso— pretende no solo participar en este debate —lo cual puede ser posible, e incluso bienvenido—, sino *cambiar la estructura de representación* con el fin de dotarse de un estatus equivalente al de los médicos, tal cambio es imposible, pues va en contra de los objetivos y principios mismos de la sociedad. Es más, se instará a los enfermeros a formar su propia sociedad, y no a unirse a la de los médicos, pues esta propuesta desmiente las estructuras de poder existentes. Por el contrario, un público es una entidad menos controlada y por ende más agonística, de tal modo que los enfermeros podrían unirse a él, expresarse e insistir en modificar los términos del debate, al igual que los pacientes, los científicos o la gente sin casa. Su éxito, sin embargo, depende enteramente de la fuerza con que sus acciones transformen el enfoque y las condiciones del público. En las últimas décadas las concepciones de la esfera pública han estado sometidas a una crítica categórica por asumir que dicha «igualdad de acceso» basta para alcanzar la representación, cuando de hecho hay otros factores contextuales (raza, clase, sexo) que descompensan inherentemente el poder representativo de los diferentes participantes. Con todo, se trata de dos problemas diferentes que se solapan: es imposible solucionar el problema de las formas de desigualdad perniciosas e invisibles a menos que se solucione antes el problema de garantizar un cierto modo de publicidad estructural. Es precisamente la atención hacia el mantenimiento de la publicidad de un público recursivo, frente a los contundentes y poderosos intentos corporativos y estatales de restringirla, la que sitúo como la lucha central del software libre. El género ciertamente influye en qué voces se escuchan en el seno del software libre, por ejemplo, pero es un error centrarse en esta desigualdad a expensas de la forma más amplia y amenazante de quiebra política de la que se ocupa el software libre. Y estoy convencido de que muchos geeks —hombres, mujeres y animales— comparten esta opinión.

dos que siguen surgiendo de él como formas originales e insólitas de publicidad y acción política.

A primera vista, el hilo que entrelaza estos proyectos parece ser Internet. De hecho, la historia y trascendencia cultural del software libre han estado intrincadamente ligadas a las de Internet a lo largo de las últimas cuatro décadas. Y es que Internet constituye una plataforma única —un entorno o una infraestructura— para el software libre. Ahora bien, Internet debe su forma al software libre. De este modo, el software libre e Internet están vinculados como figura y fondo, o como sistema y entorno; ninguno de los dos elementos es estable o invariable en y por sí mismos, y existen varios puntos prácticos, técnicos e históricos donde resultan esencialmente indistinguibles. Internet no es ella misma un público recursivo, pero es algo de vital importancia para esos públicos, que se preocupan profundamente por ella y actúan para preservarla. A lo largo de las próximas páginas, retornaré a estos tres fenómenos: Internet, una infraestructura de tecnologías y usos heterogénea y diversa, aunque singular; el software libre, un conjunto muy específico de prácticas técnicas, legales y sociales que ahora necesitan Internet; y los públicos recursivos, un concepto analítico que pretende aclarar la relación entre los dos anteriores.

Tanto Internet como el software libre son fenómenos históricamente específicos, esto es, no equiparables a anteriores formas de nuevos medios o tecnologías de la información. Sin embargo, Internet significa cosas muy diversas cosas para muy diversas personas. Tal y como apuntó el revisor de una versión inicial de este libro:

La mayoría de la gente identifica Internet con porno, cotizaciones bursátiles, videos de ejecuciones en Al Jazeera, Skype, ver fotos de los nietos, porno, no tener que volver a comprar una enciclopedia jamás, MySpace, correo electrónico, directorios inmobiliarios *online*, Amazon, consulta en Google de potenciales intereses románticos, etc.

Es imposible ofrecer una explicación conjunta de todo esto; el sentido y la relevancia de la proliferación de pornografía digital representan cuestiones muy diferentes del declive de las enciclopedias en papel y el auge de Wikipedia. Con todo, existen ciertas prácticas subyacentes que vinculan estos fenómenos dispares y ayudan a explicar por qué

se dan en este momento y en este contexto técnico, legal y social. Mi sugerencia es que, a través de la observación cuidadosa del software libre y de sus modulaciones, se puede alcanzar una mejor comprensión de las transformaciones que influyen en la pornografía, Wikipedia, las cotizaciones bursátiles y muchas otras cosas maravillosas y aterradoras.⁴

Two Bits consta de tres partes. La Primera Parte presenta a los lectores el concepto de públicos recursivos mediante la exploración de las vidas, obras y debates de una comunidad internacional de *geeks* reunidos en torno a su interés compartido por Internet. El Capítulo 1 se pregunta, con una entonación etnográfica, «¿por qué los *geeks* se asocian entre sí?». La respuesta —formulada a través de la historia de Napster en 2000 y del proceso de definición de estándares que se halla en el corazón mismo de Internet— es que están creando un público recursivo. El Capítulo 2 explora más de cerca las expresiones y actitudes de los *geeks*, centrándose en los extraños relatos que cuentan (sobre la Reforma Protestante, sobre su polimatía práctica cotidiana, sobre el progreso y la ilustración), relatos que dotan de sentido a la economía política contemporánea de formas a veces sorprendentes. En esta Primera Parte resulta central una explicación de cómo los *geeks* discuten sobre tecnología pero también discuten con y a través de ella, construyendo, modificando y manteniendo los mismos programas, redes e instrumentos legales que les dotan de espacios y herramientas para asociarse. La intención de dicha explicación es ofrecer a los lectores una especie de idea visceral de por qué ciertas conjugaciones de tecnología, organización y legislación —en especial las de Internet y el software libre— resultan tan vitalmente importantes para estos *geeks*.

La Segunda Parte retrocede un paso del vínculo etnográfico para preguntarse: «¿Qué es el software libre y por qué ha surgido en este momento histórico?». De este modo, presenta un retrato históricamente detallado del surgimiento del software libre que comienza en los años 1998-1999 y se remonta hasta finales de los 50 del siglo pasado; y a continuación emprende una recapitulación de la Primera Parte examinando el software libre como paradigma de un público recursivo. Los cinco

4. Wikipedia constituye quizás el ejemplo más generalmente conocido y familiar de aquello de lo que trata este libro. Por más que no se la identifique de este modo, se trata de hecho de un proyecto de software libre y de una «modulación» de este tal y como la describo aquí. Puede que tener este ejemplo en mente ayude a los lectores no aficionados a la técnica a seguir la argumentación de esta obra. En todo caso, volveré explícitamente a ella en la Tercera Parte. Eso sí, para bien o para mal, no incluiré aquí ninguna discusión sobre pornografía.

capítulos de esta parte presentan un relato históricamente coherente, pero cada uno de ellos se centra en un componente diferenciado del software libre. Estos relatos ayudan a distinguir la figura del software libre del fondo de Internet. La diversidad de prácticas técnicas, de intereses económicos, de tecnologías de la información y de prácticas legales y organizativas es inmensa, y estos cinco capítulos distinguen y describen las prácticas específicas en sus contextos y trasfondos históricos: las prácticas de proselitismo y argumentación, de compartición, adaptación y bifurcación de código fuente, de concepción de la apertura y de los sistemas abiertos, de creación de licencias de *copyright* para el software libre y de coordinación de personas y código fuente.

La Tercera Parte retoma el vínculo etnográfico para analizar dos proyectos relacionados que, inspirados por el software libre, modulan uno o más de los cinco componentes examinados en la Segunda Parte, es decir, que adoptan las prácticas desarrolladas por el software libre para experimentar la creación de algo nuevo y diferente. Los dos proyectos son Creative Commons, una organización no lucrativa que crea licencias de *copyright*, y Connexions, un proyecto dedicado a desarrollar un procomún de manuales educativos en línea. A través del rastreo de sus modulaciones de las prácticas, planteo estas cuestiones: ¿Siguen siendo proyectos de software libre? ¿Siguen siendo públicos recursivos? La respuesta a la primera pregunta revela de qué modo las prácticas flexibles del software libre están influyendo en formas prácticas específicas alejadas de la programación informática, mientras que la respuesta a la segunda ayuda a explicar cómo el software libre, Creative Commons, Connexions y otros proyectos similares son todos respuestas estratégicas relacionadas con la reorientación del saber y el poder. La conclusión pone sobre la mesa una serie de interrogantes con la intención de ayudar a los estudiosos a examinar fenómenos similares.

Públicos recursivos y reorientación del saber y el poder

El gobierno y el control de la creación y diseminación del saber han cambiado considerablemente en el contexto de Internet a lo largo de las tres últimas décadas. Casi todos los tipos de productos mediáticos son más fáciles de producir, publicar, difundir, modificar, remezclar o reutilizar. El número de tales creaciones, difusiones y préstamos se ha disparado, y las herramientas para generar y divulgar conocimiento —software y redes— también están cada vez más ampliamente

disponibles. Los resultados también han sido explosivos e incluyen desasosiegos acerca de la validez, calidad, propiedad y control, amén de pánicos morales en abundancia y nuevas inquietudes acerca de la forma y legitimidad de los sistemas mundiales de «propiedad intelectual». Todas estas cuestiones llegan a configurar una *reorientación del saber y el poder* aún incompleta y emergente, cuyas implicaciones alcanzan de lleno el núcleo de la legitimidad, certeza, fiabilidad y, especialmente, de la integridad y temporalidad del saber y las infraestructuras que creamos colectivamente. Se trata de una reorientación a la vez más específica y más general que los grandiosos alegatos diagnósticos sobre la «sociedad de la información» o la «sociedad red», o sobre el auge del trabajo cognitivo o de las economías basadas en el conocimiento: más específica porque concierne a prácticas técnicas y legales precisas y detalladas, y más general porque constituye una reorientación *cultural*, no solo económica o legal.

El software libre ejemplifica esta reorientación; no es una mera empresa técnica sino que supone la creación de un «público», un colectivo que se reivindica como un mecanismo de control sobre otras formas de poder constituidas —como los Estados, la Iglesia y las corporaciones— pero que se mantiene independiente respecto de dichos ámbitos de poder.⁵ El software libre es una respuesta a esta reorientación que ha dado lugar a una nueva forma de acción política democrática, un medio por el que los públicos pueden crearse y mantenerse de modos con los que no estábamos para nada familiarizados en el pasado. El software libre es un público de un tipo particular: un público recursivo. Los públicos recursivos son públicos implicados en la capacidad de construir, controlar, modificar y mantener la infraestructura que les permite nacer en primer término y que, a su vez, constituye sus aspiraciones prácticas cotidianas y las identidades de sus participantes como individuos creativos y autónomos. En los casos aquí explorados, esa infraestructura específica incluye la creación de la propia Internet y de sus herramientas y estructuras asociadas, tales como Usenet, el correo electrónico, la World Wide Web, UNIX y sus sistemas operativos

5. Aunque el término *público* sugiere claramente como su opuesto el de *privado*, el software libre no es anticomercial, pues en su creación interviene una gran cantidad de dinero, tanto real como nocional. Podría usarse también el término *mercado recursivo*, con el fin de enfatizar la importancia (especialmente durante la década de los 90 del siglo pasado) de los atributos económicos de su práctica. La cuestión no es probar si el software libre es «público» o «mercantil», sino construir un concepto adecuado a las prácticas que lo constituyen.

derivados y los protocolos, estándares y procesos de normalización. Durante los últimos treinta años, Internet ha sido el terreno de una contienda en la que el software libre ha sido tanto un combatiente central como un importante arquitecto.

Al definir el software libre como público recursivo, planteo dos cuestiones: en primer lugar, llamo la atención sobre la trascendencia democrática y política del software libre y de Internet; y en segundo lugar, sugiero que nuestra actual comprensión (tanto académica como coloquial) de lo que se considera un público autónomo, o incluso «el público», es radicalmente inadecuada para captar la reorientación contemporánea del saber y el poder. El primer argumento es fácil de defender: resulta obvio que hay algo político en el software libre, pero la mayoría de observadores asume, erróneamente, que se trata simplemente de una postura ideológica tecnoliberal-libertaria [*technolibertarian*]⁶ o contraria a la propiedad intelectual. Espero mostrar de qué modo los *geeks* no parten de ideologías, sino que más bien llegan a ellas a través de su implicación en las prácticas de creación del software libre y sus derivados. Por supuesto, hay multitud de ideólogos, pero son muchas más las personas que empiezan viéndose a sí mismos como liberal-libertarios o liberadores, pero que se vuelven algo bastante diferente a través de su participación en el software libre.

El segundo argumento es más complejo: ¿por qué otra contribución más al debate sobre el público y las esferas públicas? Existen dos razones por las que he considerado necesario inventar, y tratar de precisar, el concepto de público recursivo: la primera es para señalar la necesidad de incluir dentro del espectro de actividad política la creación, modificación y mantenimiento de software, redes y documentos legales. La codificación, el parcheo, la compartición, la compilación

6. Tras bastantes dudas acerca de cómo verter al castellano el término «*libertarian*» tal y como lo emplean diversos autores estadounidenses, optamos por la fórmula «liberal-libertario» y remitimos a David Graeber para la mejor nota aclaratoria que hemos encontrado al respecto (en su obra *La utopía de las normas*, Barcelona, Ariel, 2015):

Debido a un curioso conjunto de circunstancias históricas, la palabra «liberal» ya no significa lo mismo en los Estados Unidos que en el resto del mundo. La palabra se aplicaba originalmente a los partidarios del mercado libre, y en gran parte del mundo sigue siendo así. En los Estados Unidos la adoptaron los socialdemócratas, y en consecuencia se convirtió en anatema para la derecha, por lo que los partidarios del mercado libre se vieron obligados a adoptar el término *libertarian*, que en su origen era intercambiable con «anarquista», y se empleaba en definiciones como «socialista libertario» o «comunista libertario» [p. 224]. [N. del E.]

y la modificación de programas informáticos son formas de acción política que hoy acompañan rutinariamente a formas de expresión política familiares como la libertad de expresión, las asambleas, las peticiones y la libertad de prensa. Tales actividades son expresivas de modos que la teoría política y las ciencias sociales convencionales no reconocen: pueden tanto expresar como «implementar» ideas acerca del orden social y moral de la sociedad. El software y las redes pueden expresar ideas en el sentido escrito convencional así como crear (expresar) infraestructuras que permitan que las ideas circulen de formas novedosas e inesperadas. En el nivel analítico, el concepto de público recursivo es una manera de insistir en la importancia para el debate público de la indómita materialidad técnica de un orden político, y no solo del discurso plasmado sobre él (por más material que sea). A lo largo de este libro, planteo la cuestión de cómo el software libre e Internet constituyen en sí mismos un público, así como aquello que ese público crea, construye y mantiene.

La segunda razón por la que uso el concepto de público recursivo es que los públicos convencionales han sido descritos como «autofundamentadores», esto es, como constituidos exclusivamente a través del discurso en el sentido convencional del habla, la escritura y la asamblea.⁷ Los públicos recursivos no solo son «recursivos» por la «autofundamentación» de aspiraciones e identidades sino también porque están comprometidos con la profundidad o los estratos de dicha autofundamentación: las capas de infraestructura técnica y legal que son necesarias para que, por ejemplo, Internet exista como la infraestructura de un público. Cada acto de autofundamentación que constituye a un público se apoya a su vez en la existencia de un medio o fundamento a través del cual es posible la comunicación —ya sea una conversación cara a cara, un intercambio epistolar o una asamblea a través de la Red— y los públicos recursivos cuestionan implacablemente el estatuto de estos medios, sugiriendo que también ellos han de ser independientes para que un público sea auténtico. En cada una de estas capas, las decisiones técnicas, legales y organizativas pueden influir en si la infraestructura permitirá, o incluso garantizará, la existencia prolongada de los públicos recursivos comprometidos con ella. La independencia de los públicos recursivos respecto del poder no es

7. Véase, por ejemplo, Warner, *Publics and Counterpublics*, pp. 67–74 [ed. cast: *Públicos y contrapúblicos*].

absoluta, sino provisional y estructurada en respuesta a la estratificación históricamente constituida de poder y control en el seno de las infraestructuras informáticas y comunicativas.

Por ejemplo, un aspecto muy importante de la Internet contemporánea que se ha visto sometido a una encarnizada disputa (recientemente bajo la bandera de la «neutralidad de la red») es su *singularidad*: solo existe una Internet. Ello no constituía un desenlace inevitable o técnicamente determinado, sino el resultado de una batalla en la que se tomaron una serie de decisiones sobre capas que abarcan desde la misma configuración física básica de Internet (redes conmutadas de paquetes y sistemas de enrutamiento indiferentes a los tipos de datos) a los estándares y protocolos que la hacen funcionar (por ejemplo, el TCP/IP o el DNS), pasando por las aplicaciones que se ejecutan en ella (correo electrónico, www, ssh). Como consecuencia de estas decisiones, se ha privilegiado la singularidad de Internet y se ha propugnado su estandarización, en lugar de haberse promovido su fragmentación en múltiples redes incompatibles. Esta misma clase de decisiones se discute, sopesa y programa rutinariamente en la actividad de los diversos proyectos de software libre, así como de sus derivados. Mi planteamiento es que se trata de decisiones inscritas en imaginaciones de orden que son simultáneamente morales y técnicas.

En contraste, los gobiernos, empresas, organizaciones no gubernamentales (ONGs) y otras instituciones tienen multitud de motivos —lucro, seguridad, control— para buscar la fragmentación de Internet. Pero es el control sobre este poder por parte de los públicos recursivos, y especialmente mediante las prácticas que actualmente constituyen el software libre, el que ha mantenido hasta la fecha la integridad de Internet. Sin ser en modo alguno absoluto, este mecanismo de control está sin embargo rigurosa y técnicamente comprometido con su legitimidad e independencia, no solo respecto de las formas estatales de poder y control, sino también respecto del poder corporativo, comercial y no gubernamental. En la medida en que Internet es pública y extensible (incluyendo la capacidad de crear subredes privadas), lo es gracias a las prácticas discutidas aquí y a su culminación en un público recursivo.

Los públicos recursivos responden a la cuestión gubernamental de forma directa implicándose en, manteniendo y a menudo modificando la infraestructura que pretenden, en cuanto público, habitar y extender —y no solo ofreciendo opiniones o protestando contra las decisiones, como hacen los públicos convencionales (en la mayoría de teorías de

la esfera pública). Los públicos recursivos tratan de crear lo que podría entenderse, enigmáticamente, como un terreno de juego nivelado en constante «autonivelación». Y es en su intento de hacer que el terreno de juego se autonivele donde plantan cara y se resisten a las formas de poder que tratan de nivelarlo en beneficio de una u otra parte: Estados, gobiernos, empresas o grupos profesionales. Es importante comprender que los *geeks* no quieren simplemente nivelar el terreno de juego en su beneficio, pues carecen de afinidad o identidad como tales. En lugar de ello, desean inventar modos de dotar al terreno de juego de un cierto tipo de capacidad de acción, efectuada a través de la capacidad de acción de muchos seres humanos diferentes, pero revisada mediante su estructura y apertura técnica y legal. Los *geeks* no desean competir con capitalistas o empresarios a no ser que puedan asegurarse de que (en cuanto actores públicos) pueden hacerlo de forma justa. Se trata de una ética de la justicia entreverada con una estética de elegancia técnica y astucia legal.

El hecho de que los públicos recursivos respondan de este modo —mediante la implicación y modificación directas— supone un aspecto clave de la reorientación del poder y el saber que ejemplifica el software libre. Dichos públicos están reconstituyendo así la relación entre libertad y saber en un contexto técnica e históricamente específico. Los *geeks* crean y modifican y discuten sobre licencias y código fuente y protocolos y estándares y control de versiones e ideologías de libertad y pragmatismo no simplemente porque esas cosas sean inherente o universalmente importantes, sino porque conciernen a la relación de las formas de gobierno con la libertad de expresión y la naturaleza del consenso. El código fuente y las licencias de *copyright*, el control de versiones y las listas de distribución son los panfletos, cafés y salones del siglo XXI: las *Tischgesellschaften* [«sociedades de mesa» —primeras tertulias de Alemania—] se convierten en *Schreibtischgesellschaften* [«sociedades de escritorio»].⁸

La «reorientación del poder y el saber» posee dos aspectos clave que forman parte del concepto de públicos recursivos: disponibilidad y modificabilidad (o adaptabilidad). La disponibilidad es una cuestión amplia, difusa y familiar. Incluye elementos como la transparencia, el gobierno abierto o la organización transparente, la confidencialidad y

8. Habermas, *The Structural Transformation of the Public Sphere*, esp. pp. 27–43 [ed. cast.: *Historia y crítica de la opinión pública*, pp. 69-88].

la libertad de información y el acceso abierto en ciencia. Igualmente incluye las teorías de las escuelas de negocios acerca de la «desintermediación» y la «transparencia y responsabilidad», así como la difusión de la «cultura de la auditoría» y los denominados regímenes de gobierno neoliberales. Este aspecto es tan a menudo un objeto de sospecha como un imperativo moral, como en el caso del acceso abierto a los resultados y publicaciones científicos.⁹ Ciertamente todos estos asuntos se tocan de forma detallada y práctica en la creación de software libre. Así, los debates sobre el modo de disponibilidad de la información van desde los sistemas de gestión de derechos digitales y de protección contra la copia hasta la seguridad nacional y el espionaje corporativo, pasando por el progreso científico y las sociedades abiertas.

No obstante, la modificabilidad representa el aspecto más fascinante e inquietante de la reorientación del poder y el saber. Esta incluye la capacidad no solo de acceder a una obra —es decir, de reutilizarla en el sentido trivial de emplear algo sin restricciones— sino de transformarla para su uso en nuevos contextos, con fines diferentes o con el propósito de participar directamente en su mejora y redistribuir dichas mejoras dentro de las mismas infraestructuras al tiempo que se garantizan los mismos derechos para cualquier otra persona. De hecho, la práctica esencial del software libre es la reutilización y modificación del código fuente. Otros proyectos que toman como modelo el software libre (como Connexions y Creative Commons) también se marcan como meta las ideas clave de reutilización y modificación. Así, Creative Commons tiene como lema «La cultura siempre se basa en el pasado», y pretende que ello signifique «a través de una apropiación y modificación legales». Connexions, que permite que distintos autores creen en línea fragmentos de libros de texto, promueve explícitamente la reutilización, modificación y apropiación del trabajo de los demás. Por consiguiente, la modificabilidad suscita una cuestión muy específica e importante acerca del *acabado*. ¿Cuándo se considera algo (software, cine, música, cultura) acabado? ¿Cuánto tiempo permanece acabado? ¿Quién decide todo esto? O, más generalmente, ¿qué forma adopta su temporalidad, y cómo reestructura estas las relaciones políticas? Estos asuntos solo resultan generalmente familiares a los historiadores y expertos literarios

9. Entre las críticas a la demanda de disponibilidad y a la superioridad presuntamente inherente de la transparencia figuran Coombe y Herman, «Rhetorical Virtues» y «Your Second Life?»; Christen, «Gone Digital»; y Anderson y Bowery, «The Imaginary Politics of Access to Knowledge».

que comprenden la transformación de los cánones, la interacción de la imitación y la originalidad y las cuestiones teóricas que plantea, por ejemplo, la erudición textual. Pero el significado contemporáneo de la modificación incluye tanto un enorme incremento de la velocidad y alcance de la modificabilidad como una cierta automatización de tal práctica que era inusual antes del advenimiento de formas sofisticadas y distribuidas de software.

La modificabilidad constituye una ventaja habitualmente citada del software libre. En efecto, este puede actualizarse, modificarse, extenderse o adaptarse a otros entornos cambiantes: nuevos dispositivos, nuevos sistemas operativos, tecnologías imprevistas o nuevas leyes y prácticas. En el plano de la infraestructura, tal modificabilidad tiene pleno sentido como respuesta y alternativa a las formas tecnocráticas de planificación. Se trata de un modo de incorporar en la planificación la capacidad de superarla; una iniciativa para garantizar constantemente la capacidad de manejar la sorpresa y los resultados inesperados; una forma de hacer que infraestructuras flexibles y modificables como Internet sean tan seguras como aquellas otras permanentes e inflexibles como las carreteras o los puentes.

Ahora bien, ¿cuál es la trascendencia cultural de la modificabilidad? ¿Qué supone la incorporación de la modificabilidad a la cultura, la música, la educación o la ciencia? En un ámbito administrativo, tal cuestión se hace patente cuandoquiera que los académicos no pueden recuperar documentos escritos en un procesador de texto WordPerfect 2.0 o en un disco para el que ya no existen unidades lectoras, o cuando una biblioteca se plantea conservar tanto sus archivos como las máquinas que los leen. Por tanto, la modificabilidad es un imperativo para construir infraestructuras más perdurables. Sin embargo, no solo se trata de una solución a un problema administrativo, sino de la creación de nuevas posibilidades y nuevos problemas para prácticas arraigadas como la publicación, o para los fines y la estructura de los sistemas de propiedad intelectual, o bien para la definición del acabado, la permanencia, la monumentalidad y, especialmente, la identidad de una obra. Así, prácticas arraigadas y aparentemente invulnerables —como la autoridad de los libros publicados o el poder gubernamental para controlar la información— quedan de repente commocionadas y desnaturalizadas por las técnicas de modificabilidad.

Durante las últimas décadas, a medida que Internet se expandía exponencialmente y se insinuaba en las prácticas más íntimas de todo

tipo de personas, la cuestión de la disponibilidad y modificabilidad, así como la reorientación del saber y el poder que ellas denotan, se han convertido en lugares comunes. A medida que esto ocurría, la trascendencia y las prácticas asociadas con el software libre también se han expandido —y se han ido modulando. Estas prácticas proporcionan un material y un punto de partida significativo para un despliegue de públicos recursivos que juegan con ellas, las modulan y las transforman al tiempo que debaten y construyen nuevos modos de compartir, crear, asignar licencias y controlar sus respectivas producciones. No todos ellos comparten los mismos objetivos, inmediatos o a largo plazo, pero mediante su implicación en las prácticas técnicas, legales y sociales iniciadas por el software libre, de hecho comparten un «imaginario social» que define una relación específica entre tecnología, órganos de gobierno (ya sean estatales, corporativos o no gubernamentales) e Internet. Puede tratarse de científicos de un laboratorio o de músicos de una banda; de eruditos que crean un libro de texto o de movimientos sociales que estudian formas de organización y protesta; de burócratas gubernamentales que publican datos o de periodistas que investigan la corrupción; de compañías que gestionan datos personales o de cooperativas que supervisan el desarrollo comunitario —todos estos grupos y otros pueden encontrarse adoptando, modulando, rechazando o refinando las prácticas que han generado el software libre en el pasado reciente y que seguirán haciéndolo en el futuro próximo.

Experimento y modulación

¿Qué es exactamente el software libre? Acaso resulte asombroso comprobar que se trata de una pregunta increíblemente común en la vida *geek*. Los debates sobre la definición y las discusiones y acusaciones se dan por doquier. En cuanto antropólogo, he participado rutinariamente en tales discusiones y debates, y es a través de mi participación inmediata como se abre *Two Bits*. En la Primera Parte relataré historias sobre *geeks*, con las que pretendo dar a los lectores la clásica sensación antropológica de ser arrojados a un mundo extraño. Dichas historias revelan aspectos generales de los temas y modos de conversación de los *geeks*, sin entrar en detalles sobre qué es el software libre. Empiezo de esta forma porque fue así como empezó mi proyecto. Inicialmente no tenía intención de estudiar el software libre, pero resultaba imposible ignorar su surgimiento y manifiesta centralidad para los *geeks*.

Los debates sobre la definición del software libre en los que participé tanto en línea como en el trabajo de campo acabaron desviándome del estudio de los *geeks per se* para orientar mi atención investigadora hacia el interés principal de esta obra: ¿Cuál es la trascendencia cultural del software libre?

En la Segunda Parte lo que ofrezco no es una definición del software libre, sino una historia de cómo llegó a nacer. La historia comienza en 1998, con el importante anuncio de Netscape de que liberaría el código fuente de su principal producto, Netscape Navigator, y a partir de aquí se remonta a los relatos en torno al sistema operativo UNIX, los «sistemas abiertos», la legislación de *copyright*, Internet y las herramientas para coordinar a personas y código. En conjunto, estas cinco historias constituyen una descripción de cómo funciona el software libre en la práctica. Desde la perspectiva del análisis cultural, estas historias destacan hasta qué punto son experimentales estas prácticas, y de qué modo los individuos realizan su seguimiento y las modulan sobre la marcha.

La decisión de Netscape llegó en un momento importante de la vida del software libre. Por entonces el software libre comenzaba a adquirir conciencia de sí mismo como un *movimiento* coherente y no solo como una amalgama diversa de proyectos, herramientas o prácticas. Curiosamente, este reconocimiento también supuso una escisión: ciertos participantes empezaron a insistir en denominar el movimiento software «de código abierto», para así remarcar sus aspiraciones prácticas por encima de las ideológicas. La propia propuesta desató una monumental discusión pública acerca de lo que definía el software libre (o de código abierto). Este enigmático suceso, en el que un movimiento adquiere conciencia de sí mismo al tiempo que comienza a cuestionarse su misión, es objeto de análisis en el Capítulo 3. Empleo el término *movimiento* para designar uno de los cinco componentes esenciales del software libre: las prácticas de argumentación y discrepancia acerca del significado del software libre. A través de estas prácticas de discusión y crítica empiezan a ponerse de relieve las otras cuatro prácticas y los participantes tanto en el software libre como en el de código abierto se percatan de algo sorprendente: pese a todas las distinciones ideológicas en el ámbito discursivo, están *haciendo exactamente lo mismo* en el ámbito práctico. El sentido histrionismo con que los *geeks* discuten sobre la definición de lo que hace libre el software libre o abierto el código abierto solo es equiparable a la sobria especificidad de las detalladas prácticas que comparten.

El segundo componente del software libre no es sino una actividad mundana: la compartición de código fuente (Capítulo 4). Se trata de una práctica esencial y fundamentalmente rutinaria, pero que posee una historia que revela los objetivos de portabilidad del software, las interacciones del desarrollo de software comercial y académico y la centralidad del código fuente (y no solo de conceptos abstractos) en los contextos pedagógicos. Los detalles de la «compartición» del código fuente también conforman el relato del auge y proliferación del sistema operativo UNIX y de su miríada de derivados.

El tercer componente, la concepción de la apertura (Capítulo 5), alude a los específicos sentidos técnico y «moral» de la apertura, especialmente tal y como surgieron en los debates sobre «sistemas abiertos» que se dieron en la industria informática en la pasada década de los 80. Dichos debates atañían a la creación de una infraestructura particular, que incluía tanto estándares y protocolos técnicos (un UNIX estándar y los protocolos de redes) como una infraestructura mercantil ideal que permitiría el florecimiento de tales sistemas abiertos. El Capítulo 5 narra la historia del fracaso en la consecución de una infraestructura mercantil para los sistemas abiertos, debido en parte a un importante punto ciego: el papel de la propiedad intelectual.

El cuarto componente, la aplicación de licencias de *copyright* (y *copyleft*) (Capítulo 6), implica el problema de la propiedad intelectual con el que se toparon los programadores y *geeks* a finales de los 70 y principios de los 80. En este capítulo detallaré la historia de la primera licencia de software libre —la GPL (*GNU General Public License*, la Licencia Pública General de GNU)— que surgió a partir de la controversia en torno a una aplicación muy famosa llamada EMACS. Dicha controversia coincidió con modificaciones legislativas (en 1976 y 1980) y con prácticas cambiantes en la industria informática —una deriva general desde el secreto comercial a la protección mediante *copyright*— y representa también una historia sobre la alabada «ética *hacker*» que la revela en su contexto práctico de nacimiento, en vez de presentarla como una selecta lista de reglas.

El quinto componente, la práctica de coordinación y colaboración (Capítulo 7), es del que más se habla: la idea de decenas o cientos de miles de voluntarios dedicando su tiempo a contribuir a la creación de programas complejos. En este capítulo muestro cómo se desarrollaron novedosas formas de coordinación en la década de los 90 y cómo funcionaron en los casos canónicos de Apache y Linux; asimismo

resalto cómo la coordinación facilita la aspiración de adaptabilidad (o modificabilidad) frente a la planificación y la jerarquía, y cómo dicha aspiración resuelve la tensión entre el virtuosismo individual y la necesidad de control efectivo.

Tomados en su conjunto, estos cinco componentes conforman el software libre, pero no suponen una definición. Dentro de cada una de estas cinco prácticas, podrían incluirse razonablemente muchas actividades similares y desemejantes. El propósito de tal redescipción de las prácticas de software libre es su conceptualización como un tipo de *sistema técnico experimental de carácter colectivo*. Dentro de cada componente existe un abanico de diferencias prácticas que va de lo convencional a lo experimental. En el centro, por así decirlo, se hallan las versiones más comunes y aceptadas de una práctica; en los márgenes están las versiones más inusuales o controvertidas. En conjunto, todos los componentes conforman un sistema experimental cuya infraestructura es Internet y cuyas «hipótesis» conciernen a la reorientación del saber y el poder.

Por ejemplo, apenas se puede disponer de software libre sin el código fuente, pero no necesariamente ha de estar escrito en lenguaje C (aunque la inmensa mayoría lo esté), sino que puede escribirse en Java o perl o TeX. No obstante, si expandimos el significado de código fuente hasta incluir la música (donde el código fuente sería la partitura y el código binario sería la interpretación), ¿qué sucede? ¿Sigue siendo software libre? ¿Qué sucede cuando tanto la partitura como la ejecución «nacen digitales»? O, por tomar un ejemplo distinto, el software libre requiere licencias de software libre, pero las condiciones de estas a menudo son objeto de cambios, discusiones acaloradas y celosa vigilancia por parte de los *geeks*. ¿Qué nivel de modificación excluye una licencia de la esfera del software libre y por qué? ¿Cuánta flexibilidad está permitida?

Así concebido, el software libre es un sistema de umbrales, no de clasificación; el entusiamo que sienten los participantes y observadores procede de la modulación (experimentación) de cada una de estas prácticas y del subsiguiente descubrimiento de la localización de los umbrales. Son numerosísimas las personas que han escrito sus propias licencias «de software libre», pero solo algunas de ellas se mantienen dentro del umbral de la práctica tal y como viene definida por el sistema. Las modulaciones se dan cada vez que alguien aprende cómo funciona algún componente del software libre y se pregunta: «¿Puedo experimentar estas prácticas en algún otro ámbito?».

La realidad de la modulación constante supone que estas cinco prácticas no definen el software libre de una vez por todas, sino que lo hacen con respecto a su constitución en el contexto contemporáneo. Se trata de un conjunto de prácticas definidas «en torno al punto» de los años 1998-1999, un intensivo espacio coordinado que permite explorar los componentes del software libre de modo retrospectivo y prospectivo: en su futuro próximo y en su pasado reciente. El software libre es una máquina para mapear el (re)surgimiento de una problemática de poder y saber a medida que se va filtrando a través de las realidades técnicas de Internet y de la configuración política y económica contemporánea. Cada una de estas prácticas posee su propia temporalidad de desarrollo y surgimiento, pero recientemente han confluido en este conglomerado denominado software libre o código abierto.¹⁰

Esta visión del software libre como un sistema experimental tiene un propósito estratégico en *Two Bits*: crear el marco para la Tercera Parte, en la que me pregunto qué tipos de modulación podrían seguir caracterizándose como públicos recursivos pese a no poder entrar ya *per se* en la categoría de software libre. Fue en torno a 2000 cuando la discusión en torno al «procomún» comenzó a propagarse más allá de los debates sobre software libre al ámbito de los materiales educativos y la biodiversidad, de la música, el texto y el video, así como de la información médica y de los resultados y datos científicos.¹¹ Por una parte, ello iba en la línea del interés por crear «archivos digitales», «colecciones en línea» o «bibliotecas digitales»; por otra parte, representaba una conjugación de la colección digital con los problemas y prácticas de la propiedad intelectual. El mismo término *procomún* —al mismo tiempo un nombre nuevo y un objeto de investigación teórica— se proponía sugerir algo más que una simple colección, sea de objetos digitales o de cualquier otra cosa; se proponía señalar el interés público, la gestión colectiva y el estatuto legal de dicha colección.¹²

10. Esta descripción del software libre podría denominarse también un «ensamblaje». La fuente más reciente para ello es Rabinow, *Anthropos Today*. El lenguaje de umbrales e intensidades está desarrollado del modo más claro por Manuel DeLanda en *A Thousand Years of Non-linear History* [ed. cast.: *Mil años de historia no lineal*] y en *Intensive Science and Virtual Philosophy*. El término *problematisación*, tomado de Rabinow (que lo canaliza a partir de Foucault), es un sinónimo de la expresión «reorientación del saber y el poder» tal y como la uso aquí.

11. Véase Kelty, «Culture's Open Sources».

12. La genealogía del término *commons* (*procomún*) cuenta con un buen número de fuentes. Una obvia es el célebre artículo «*The Tragedy of the Commons*» («La tragedia del procomún»),

En la Tercera Parte, examino en detalle dos «procomunes» entendidos como modulaciones de estas prácticas constitutivas del software libre. Más que tratar estos proyectos del procomún como simples usos metafóricos o inspirados del software libre, los trato como modulaciones, lo que me permitirá mantener la conexión directa con las cambiantes prácticas implicadas. El propósito de la Tercera Parte es comprender cómo proyectos del procomún como Connexions y Creative Commons traspasan los umbrales de estas prácticas y sin embargo mantienen parte de la misma orientación. Así, por ejemplo, ¿qué cambios han posibilitado imaginar nuevas formas de contenido libre, cultura libre, música de código abierto o procomún científico? ¿Qué sucede cuando nuevas comunidades de personas adoptan y modulan los cinco componentes? ¿Se convierten ellas también en públicos recursivos, implicados en el mantenimiento y la expansión de las infraestructuras que les permiten nacer en primer término? ¿Están interesadas en las implicaciones de disponibilidad y modificabilidad que continúan desplegándose y explorándose en las esferas educativa, musical, cinematográfica, científica y literaria?

Las respuestas de la Tercera Parte dejan claro que, hasta el momento, estas preocupaciones siguen bien vivas en las modulaciones del software libre: tanto Creative Commons como Connexions pugnan por adaptarse a nuevos modos de crear, compartir y reutilizar contenidos en el entorno legal contemporáneo, empleando Internet como infraestructura. Los Capítulos 8 y 9 ofrecen un detallado análisis de un experimento técnico y legal: una modulación que parte del código abierto,

publicado por Garrett Hardin en 1968. James Boyle ha contribuido más que cualquier otro a especificar el término, especialmente durante un congreso de 2001 sobre el dominio público, que incluyó la inspirada yuxtaposición en la lista de invitados del colectivo de apropiación musical gozosa Negativland y la dama de los estudios sobre «procomún», Elinor Ostrom, cuyo libro *Governing the Commons* [ed. cast. *El Gobierno de los bienes comunes*] ha supuesto una inspiración indiscutible para la reflexión sobre procomún versus dominio público. Boyle, por su parte, ha impulsado incesantemente la metáfora «ecológica» para posicionarse en defensa del dominio público del mismo modo que los ecologistas de los 60 y 70 del siglo XX se posicionaban en defensa del medioambiente (véase Boyle, «The Second Enclosure Movement and the Construction of the Public Domain» [ed. cast.: «El segundo movimiento de cercamiento y la construcción del dominio público»] y «A Politics of Intellectual Property»). El término *procomún* resulta útil en este contexto precisamente porque distingue el «dominio público», entendido como un objeto imaginado de pura transacción y coordinación públicas, frente al «procomún», que puede constar de objetos/espacios de titularidad privada que se gestionan de tal modo que efectivamente funcionan como se imagina que lo hace el «dominio público» (véase Boyle, «The Public Domain»; Hess y Ostrom, *Understanding Knowledge as a Commons* [ed. cast.: *Los bienes comunes del conocimiento*]).

pero que rápidamente requiere modulaciones en la configuración de las licencias y en las formas de coordinación. Es ahí donde *Two Bits* proporciona el relato más detallado de una figuración que tiene como trasfondo la reorientación del saber y el poder. En concreto, este relato versa sobre la *reutilización*, sobre la modificabilidad y los problemas que surgen en el intento de incorporarla a las prácticas cotidianas de escritura pedagógica y producción cultural de una mirada de formas. Dicho intento enfrenta a los actores directamente implicados con la cuestión de la existencia y ontología de las normas: normas académicas de producción, préstamo, reutilización, cita, reputación y propiedad. Estos últimos capítulos dejan abiertas preguntas sobre la estabilidad del conocimiento moderno, y ello no como un problema archivístico o legal, sino como uno social y normativo; suscitan interrogantes acerca de la invención y el control de las normas, y acerca de las formas de vida que pueden surgir de estas prácticas. Los públicos recursivos llegan a existir allá donde queda claro que tales invención y control han de ser ampliamente compartidos, abiertamente examinados y celosamente supervisados.

Tres formas de contemplar *Two Bits*

Two Bits realiza tres tipos de contribuciones académicas: empíricas, metodológicas y teóricas. Al basarse en gran medida en el trabajo de campo (que incluye labores históricas y archivísticas), estas tres contribuciones a menudo aparecen mezcladas unas con otras. El trabajo de campo, especialmente en la antropología cultural y social de las tres últimas décadas, ha pasado a concebirse cada vez menos como un instrumento específico dentro de una caja de herramientas metodológica, y cada vez más como un modo distintivo de encuentro epistemológico.¹³ Las cuestiones de las que partí surgieron de los estudios de ciencia y tecnología, pero podrían acabar teniendo sentido en multitud de campos, abarcando desde los estudios legales a la informática.

Desde una perspectiva empírica, los actores de mis relatos están figurándose algo, algo que es insólito, desconcertante, impreciso y en ocasiones impactante para todo el mundo que se ocupa de ello en

13. Marcus y Fischer, *Anthropology as Cultural Critique* [ed. cast.: *La antropología como crítica cultural*]; Marcus y Clifford, *Writing Culture* [ed. cast.: *Retóricas de la antropología*]; Fischer, *Emergent Forms of Life and the Anthropological Voice*; Marcus, *Ethnography through Thick and Thin*; Rabinow, *Essays on the Anthropology of Reason and Anthropos Today*.

diferentes momentos y medidas.¹⁴ Existen dos clases de relatos de figuración: los contemporáneos en los que he sido participante activo (los de Connexions y Creative Commons), y los históricos que elaboro mediante la investigación «archivística» y la relectura de ciertos tipos de textos, discusiones y análisis de coyuntura (los de UNIX, EMACS, Linux, Apache y Open Systems). Algunos relatos versan sobre figuraciones técnicas, pero la mayoría se ocupa de figuraciones acerca de un problema que parece haber surgido. Algunos relatos implican a actores inexpertos y fervorosos, otros contienen maquinaciones y estrategias, pero en todos ellos la figuración se presenta «en proceso» y no como algo que puede narrarse convenientemente como obvio e incontestable mirado en retrospectiva. A lo largo de este libro expongo relatos que ilustran cómo son los *geeks* en algunos aspectos, pero que, más importante aún, les muestran en pleno proceso de figuración —una práctica que puede darse tanto en una discusión como durante la fase de diseño, planificación, ejecución, escritura, depuración, *hackeo* y reparación.

Hay también miradas de formas con las que los *geeks* narran sus acciones a los demás y a ellos mismos, a medida que van figurándose cosas. De hecho, aquí no existe crisis de representación del otro: los *geeks* son elocuentes, estridentes, persistentes y locuaces. Los superalternos pueden expresarse por sí mismos. Con todo, tales representaciones no deberían tomarse necesariamente como prueba de que los *geeks* ofrecen explicaciones analíticas o críticas adecuadas de sus propias acciones. Algunos de los textos disponibles combinan excelentes descripciones con análisis confusos, como por ejemplo la obra de Eric Raymond.¹⁵ Durante el transcurso de mi trabajo de campo, la obra de Raymond ha estado siempre presente como una excelente guía para las prácticas y preguntas que asolan a los *geeks* —en buena medida como el clásico «informante principal» de la antropología. Y sin embargo sus análisis,

14. El lenguaje de la «figuración» [*figuring out*] tiene como fuente inmediata la obra de Kim Fortun «Figuring Out Ethnography». Dicha obra refina otras dos fuentes, el trabajo de Bruno Latour en *Science in Action* [ed. cast: *Ciencia en acción*] y el de Hans-Jörg Rheinberger en *Towards History of Epistemic Things*. Latour describe la diferencia entre «ciencia acabada» y «ciencia en proceso de elaboración» y el modo en que el análisis cuidadoso de nuevos objetos puede revelar cómo llegan a surgir. Rheinberger extiende este enfoque mediante el análisis de las prácticas detalladas que están involucradas en la figuración de un nuevo objeto o un nuevo proceso —prácticas que solo pueden ser nombradas o explicadas por sus participantes *a posteriori*.

15. Raymond, *The Cathedral and the Bazaar* [ed. cast.: «La catedral y el bazar»].

a los que muchos *geeks* se adhieren, resultan confusos. Es cierto que son imaginativos y en ocasiones deliciosos e iluminadores, pero no abordan la trascendencia cultural del software libre. En este sentido, estoy menos interesado en tratar a los *geeks* como nativos susceptibles de ser explicados y más en discutir con ellos: las personas que aparecen en *Two Bits* son una condición *sine qua non* de la etnografía, pero no constituyen los objetos de su análisis.¹⁶

Dado que las historias que narro aquí son recientes según los estándares de la historia académica, no abundan las referencias bibliográficas en comparación con el material empírico. Me apoyo en diversos libros y artículos sobre la historia de la Internet inicial, especialmente en la obra de Janet Abbate y en el único trabajo histórico sobre UNIX, el libro de Peter Salus *A Quarter Century of Unix*.¹⁷ También hay un par de excelentes obras periodísticas, destacando el libro de Glyn Moody *Rebel Code: Inside Linux and the Open Source Revolution* (que, como *Two Bits*, se apoya fuertemente en la novedosa accesibilidad a las discusiones detalladas que brindan las listas de distribución públicas). De modo similar, el estudio académico sobre el software libre y su

16. La literatura sobre «comunidades virtuales», «comunidades *online*», la cultura de *hackers* y *geeks* o el estudio social de la tecnología de la información ofrece una importante información de contexto, aunque no constituye el tema de este libro. Una revisión exhaustiva del trabajo al respecto en antropología y otras disciplinas se encuentra en Wilson y Peterson, «The Anthropology of Online Communities». Otras obras indispensables son Miller y Slater, *The Internet*; Carla Freeman, *High Tech and High Heels in the Global Economy*; Hine, *Virtual Ethnography*; Kling, *Computerization and Controversy*; Star, *The Cultures of Computing*; Castells, *The Rise of the Network Society* [ed. cast., *La era de la información: economía, sociedad y cultura. Vol. 1 La sociedad real*]; Boczkowski, *Digitizing the News*. La mayoría de trabajos de ciencias sociales sobre la tecnología de la información ha abordado las cuestiones de la desigualdad y la denominada brecha digital, encontrándose una excelente visión general en DiMaggio *et al.*, «From Unequal Access to Differentiated Use». Más allá de las obras de antropología y estudios sociales, un buen número de trabajos procedentes de otras disciplinas han asumido recientemente temáticas similares, especialmente Adrian MacKenzie, *Cutting Code*; Galloway, *Protocol*; Hui Kyong Chun, *Control and Freedom*; y Liu, *Laws of Cool*. En contraste, si consideramos los estudios de ciencias sociales sobre la tecnología de la información que se ubican en un contexto de estudios históricos y etnográficos acerca de la «figuración» de problemas de tecnologías de la información, software o redes específicos, entonces la literatura es exigua. Entre los ejemplos de antropología y estudios científicos sobre el proceso de figuración están Barry, *Political Machines*; Hayden, *When Nature Goes Public*; y Fortun, *Advocating Bhopal*. Matt Ratto también ha retratado esta actividad en el software libre en su tesis doctoral, «The Pressure of Openness».

17. Además de Abbate y Salus, véase Norberg y O'Neill, *Transforming Computer Technology*; Naughton, *A Brief History of the Future*; Hafner, *Where Wizards Stay Up Late*; Waldrop, *The Dream Machine*; Segaller, *Nerds 2.0.1*. Para una clásica autodocumentación de un aspecto de Internet, véase Hauben y Hauben, *Netizens*.

historia está empezando a asentarse en torno a un conjunto coherente de cuestiones.¹⁸

Metodológicamente, *Two Bits* ofrece un ejemplo de cómo estudiar etnográficamente los *fenómenos distribuidos*. El software libre e Internet son objetos que carecen de una única ubicación geográfica donde pueda acudirse a estudiarlos. Por consiguiente, esta obra es multisede en el sencillo sentido de que investiga dichos objetos en múltiples sedes: Boston, Bangalore, Berlín y Houston. Dicha investigación se llevó a cabo entre personas, proyectos y empresas específicos, así como en congresos y encuentros en línea demasiado abundantes para ser enumerados, pero no ha supuesto el estudio de un único proyecto de software libre distribuido por todo el mundo. En todos estos lugares y proyectos los *geeks* con los que he trabajado eran personas de filiaciones aleatorias y difusas con vidas e historias muy diversas. Algunos se identificaban como *hackers* del software libre, pero la mayoría no; algunos no habían coincidido jamás en la vida real, y otros sí. Estas personas representaban a múltiples empresas e instituciones, y procedían de diversas naciones, pero pese a todo compartían un cierto conjunto de ideas y expresiones que me permitían viajar de Boston a Berlín y de allí a Bangalore y retomar una conversación en curso con personas diferentes y en lugares muy distintos sin jamás perder el hilo.

El estudio de fenómenos distribuidos no implica necesariamente el estudio detallado y local de cada manifestación de un fenómeno, ni exige visitar cada lugar geográfico relevante —de hecho, tal proyecto no solo es extremadamente difícil, sino que confunde mapa y territorio. En palabras de Max Weber: «No es la interconexión ‘real’ de las ‘cosas’, sino la interconexión *conceptual* de los *problemas* lo que define el ámbito de las diferentes ciencias».¹⁹ Las decisiones sobre dónde acudir, a quién estudiar y cómo reflexionar sobre el software libre son arbitrarias en el sentido

18. Kelty, «Culture’s Open Sources»; Coleman, «The Social Construction of Freedom»; Ratto, «The Pressure of Openness»; Joseph Feller *et al.*, *Perspectives on Free and Open Source Software*; véase también <http://freesoftware.mit.edu/>, organizado por Karim Lakhani, una enorme recopilación de obras sobre proyectos de software libre. Los primeros trabajos en este ámbito provienen de los escritos de participantes como Raymond y de expertos de negocios y gestión que advirtieron en el software libre un extraordinario y sorprendente conjunto de contradicciones aparentes. Hasta la fecha el mejor de ellos es *The Success of Open Source*, de Steven Weber. Sus conclusiones son similares a las que presento aquí, y además el autor posee una familiaridad criptoetnográfica (que no confiesa explícitamente) con los actores y prácticas. *The Wealth of Networks* [ed. cast.: *La Riqueza de las Redes*], de Yochai Benkler, extiende y generaliza algunos de los argumentos de Weber.

19. Max Weber, «Objectivity in the Social Sciences and Social Policy», p. 68.

preciso de que, al tratarse de fenómenos tan extensamente distribuidos, es posible convertir cualquier nodo dado en una fuente de conocimiento rico y detallado sobre los propios fenómenos distribuidos, y no solo sobre ese nodo local. Así, por ejemplo, si no hubiera aceptado un trabajo en Houston, probablemente habría seguido ignorando ampliamente el proyecto Connexions, pero pese a todo este posee conexiones precisas e identificables con otros lugares y grupos que he estudiado, y por tanto resulta reconocible como parte de *este* fenómeno distribuido, y no de algún otro. El caso es que en ese momento buscaba activamente algo como Connexions para formular preguntas sobre en qué se estaba convirtiendo el software libre y cómo se estaba transformando. De no haberse cruzado Connexions en mi camino, me habría servido algún otro campo de estudio similar.

Es en este sentido en el que afirmo que el objeto etnográfico de este estudio no son los *geeks* ni ningún proyecto, lugar o grupo de personas específico, sino el software libre e Internet. Siendo más específico, el objeto etnográfico de este estudio son los «públicos recursivos» —con la salvedad de que este concepto es también la *obra* de la etnografía, y no su objeto preliminar. Al inicio me habría sido imposible identificar a los «públicos recursivos» como el objeto de la etnografía, y ello supone una prueba nítida de que el quehacer etnográfico supone un tipo particular de encuentro epistemológico, el cual requiere un considerable trabajo conceptual durante y después de la labor material de trabajo de campo, y a lo largo de la labor material de escritura y reescritura, con el fin de dotarlo de sentido y reorientarlo hacia un interrogante que en retrospectiva parecerá deliberado y susceptible de respuesta. Esta clase de etnografía requiere un compromiso a largo plazo y una capacidad para soslayar la superficie obvia de la transformación rápida y percibir una temporalidad más oscura y pausada de trascendencia cultural, sin dejar por ello de plantear preguntas y refinar debates sobre el futuro próximo.²⁰ Históricamente hablando, los capítulos de la Segunda Parte

20. Pese a lo que pudiera sonar como un enfoque de «primero dispara y después pregunta», lo cierto es que el diseño de este proyecto se llevó a cabo de acuerdo con metodologías específicas. La más destacada es la teoría del actor-red: Latour, *Science in Action* [ed. cast: *Ciencia en acción*]; Law, «Technology and Heterogeneous Engineering»; Callon, «Some Elements of a Sociology of Translation»; Latour, *Pandora's Hope* [ed. cast: *La esperanza de Pandora*]; Latour, *Re-assembling the Social* [ed. cast: *Reensamblar lo social*]; Callon, *Laws of the Markets*; Law y Hassard, *Actor Network Theory and After*. Curiosamente, no han existido estudios de actor-red sobre las redes, es decir, sobre tecnologías de información y comunicación específicas como Internet. La confusión de la palabra *red* (como término analítico y metodológico) con

pueden interpretarse como una contribución a una historia de la infraestructura científica —o quizá a una comprensión de la experimentación colectiva a gran escala.²¹ Tanto Internet como el software libre suponen una importante transformación práctica que tendrá efectos en la práctica científica y un tipo de práctica técnica compleja para la que existen pocos modelos de estudio.

Una precisión metodológica sobre la peculiaridad de mi materia de estudio resulta también pertinente. Los lectores perspicaces advertirán que apenas incluyo transcripciones de fragmentos de material etnográfico convencional (por ejemplo, entrevistas o notas). Allí donde sí aparecen, tienden a ser «públicamente disponibles» —esto es, accesibles a través de Internet— y se las cita como tales, con tanto detalle como sea necesario para permitir a los lectores recuperarlas. La sabiduría convencional tanto antropológica como histórica sostiene que lo que hace interesante un estudio es, en parte, el trabajo que los investigadores dedican a la recopilación de aquello que no está de antemano disponible, esto es, las fuentes primarias frente a las secundarias. En algunos casos ofrezco acceso a fuentes primarias (específicamente en los capítulos 2, 8 y 9), pero en muchos otros ello resulta ya literalmente imposible: *prácticamente todo está archivado*. Debates, disputas, colaboraciones, conversaciones, artículos, software, noticias, relatos históricos, aplicaciones antiguas con sus correspondientes manuales, reminiscencias, notas y dibujos —todo ello queda archivado por alguien en alguna parte y, lo que es más importan-

la palabra *red* (como configuración particular de cables, ondas, software y chips, o de gente, carreteras y autobuses, o de bases de datos, nombres y enfermedades) supone que siempre es necesario distinguir *esta-red-específica* de *una-red-cualquiera*. Mi enfoque comparte mucho con los interrogantes ontológicos planteados en obras como Law, *Aircraft Stories*; Mol, *The Body Multiple*; Cussins, «Ontological Choreography»; Charis Thompson, *Making Parents*; y Dumit, *Picturing Personhood*.

21. A mi entender, la preocupación por la infraestructura científica parte de *Leviathan and the Air Pump*, de Steve Shapin y Simon Schaffer, si bien su genealogía es sin duda más compleja, incluyendo Shapin, *The Social History of Truth*; Biagioli, *Galileo, Courtier*; Galison, *How Experiments End e Image and Logic*; Daston, *Biographies of Scientific Objects*; Johns, *The Nature of the Book*. Una amplia variedad de obras exploran la cuestión de la infraestructura y las herramientas científicas: Kohler, *Lords of the Fly*; Rheinberger, *Towards a History of Epistemic Things*; Landecker, *Culturing Life*; Keating y Cambrosio, *Biomedical Platforms*. En «What Rules of Method for the New Socio-scientific Experiments», Bruno Latour proporciona un ejemplo de hacia dónde podrían encaminarse los estudios sociales con estos interrogantes. Entre los textos importantes sobre la cuestión de las infraestructuras técnicas se encuentran Walsh y Bayma, «Computer Networks and Scientific Work»; Bowker y Star, *Sorting Things Out*; Edwards, *The Closed World*; Misa, Brey y Feenberg, *Modernity and Technology*; Star y Ruhleder, «Steps Towards an Ecology of Infrastructure».

te, quienes lo recopilan suelen ponerlo instantáneamente a disposición general. El repertorio de conversaciones e interacciones consideradas privadas (ya sea en el sentido de que desaparecen de la memoria escrita o de que solo son accesibles para las partes implicadas) ha disminuido de modo manifiesto desde aproximadamente 1981.

Tamaña obsesión archivística implica que la investigación etnográfica se encuentra estratificada en el tiempo. Cuestiones que de otra manera habrían requerido «encontrarse allí» resultan mucho más fáciles de investigar *a posteriori*, y esto se hace de lo más evidente en la reconstrucción a partir de fuentes de USENET y listas de distribución que realizó en los capítulos 1, 6 y 7. La abrumadora disponibilidad de materiales cuasiarchivísticos es algo a lo que me refiero, en un guiño al editor de texto EMACS, como «historia autodocumentada». Es decir, una de las actividades en la que a los *geeks* les encanta participar, y que alientan, es la creación, análisis y archivo de su propio papel en el desarrollo de Internet. Sin importar cuán oscuro o arcano sea, parece que la mayoría de los *geeks* posee un sentido de la posibilidad bien desarrollado —un sentido de que su contribución podría haber resultado transformativa, importante, seminal. Aquello que a los *geeks* pueda faltarles en habilidades sociales, lo compensan con su desmesurada arrogancia archivística.

Finalmente, la contribución teórica de *Two Bits* consiste en un refinamiento de los debates sobre los públicos, las esferas públicas y los imaginarios sociales que se presentan agitados en el contexto de Internet y el software libre. Términos como *comunidad virtual*, *comunidad en línea*, *ciberespacio*, *sociedad red* o *sociedad de la información* no son por lo general constructos teóricos, sino formas de designar un subgénero de investigación disciplinaria que tiene que ver con las redes electrónicas. La necesidad de un análisis más preciso de los tipos de asociación que tienen lugar en y a través de la tecnología de la información está clara; el primer paso para ello es precisar qué tecnologías de la información y qué prácticas específicas marcan la diferencia.

Existe una literatura relativamente amplia y creciente sobre Internet como esfera pública, pero en general tal literatura se preocupa menos por refinar el concepto por medio de la investigación y más por dictaminar si Internet se ajusta o no a la definición de Habermas de la esfera pública burguesa, una definición concebida primordialmente para la Gran Bretaña del siglo XVIII, no para la Internet del siglo XXI.²²

22. Dreyfus, *On the Internet*; Dean, «Why the Net Is Not a Public Sphere».

Los hechos de la vida técnica y humana, tal y como se despliegan a través de Internet y en torno a las prácticas del software libre, no son fáciles de encajar en la definición de Habermas. El propósito de *Two Bits* no es este sino ofrecer claridad conceptual basada en trabajo de campo etnográfico.

Los textos clave para comprender el concepto de público recursivo son los trabajos de Habermas, la obra de Charles Taylor *Modern Social Imaginaries* y los libros de Michael Warner *The Letters of the Republic* y *Publics and Counterpublics*. Entre los textos secundarios que refinan estas nociones destacan *The Public and its Problems*, de John Dewey, y *The Human Condition*, de Hannah Arendt. Aquí el centro del análisis no es la esfera pública *per se*, sino las «ideas de orden moral y social moderno» y la terminología de los «imaginarios sociales modernos»²³. Encuentro que estos conceptos son útiles como puntos de partida por una razón muy específica: porque permiten distinguir el sentido de orden moral del sentido de *orden moral y técnico* que exploro con respecto a los *geeks*. Mi intención no es someter a examen el concepto de imaginario social, sino construir algo a partir de él.

Si el concepto de público recursivo resulta útil es porque contribuye a elaborar la cuestión general de la «reorientación del saber y el poder». En particular este concepto pretende poner de relieve los modos en que Internet y el software libre están vinculados con la economía política de la sociedad moderna a través de la creación no solo de conocimiento nuevo, sino de nuevas infraestructuras para divulgarlo, mantenerlo y modificarlo. Del mismo modo que el libro de Warner *The Letters of the Republic* se ocupaba del surgimiento del discurso del republicanismo y del desarrollo simultáneo de una república estadounidense de las letras, o que el análisis de Habermas se ocupaba de la relación de la esfera pública burguesa con las revoluciones democráticas del siglo XVIII, este libro plantea una serie de preguntas similares: ¿cómo se relacionan las prácticas emergentes de los públicos recursivos con las relaciones emergentes de la vida política y técnica en un mundo que se somete a

23. Véase además Lippmann, *The Phantom Public* [ed. cast.: *Público fantasma*]; Calhoun, *Habermas and the Public Sphere*; Latour y Weibel, *Making Things Public*. El debate sobre los imaginarios sociales comienza de forma alterna con *Imagined Communities*, de Benedict Anderson [ed. cast.: *Comunidades imaginadas*], o con *The Imaginary Institution of Society*, de Cornelius Castoriadis [ed. cast.: *La institución imaginaria de la sociedad*].; véase también Chatterjee, «A Response to Taylor's 'Modes of Civil Society'»; Gaonkar, «Toward New Imaginaries»; Charles Taylor, «Modes of Civil Society» y *Sources of the Self*.

Internet y a sus formas de circulación? ¿Sigue habiendo un papel para una república de las letras, o al menos para una especie de público que pueda reivindicar seriamente independencia y autonomía respecto de otras formas de poder constituido? ¿Son las críticas pesimistas de Habermas a la quiebra de la esfera pública en el siglo XX igualmente aplicables a las estructuras del siglo XXI? ¿O acaso es posible que los públicos recursivos representen un resurgimiento de públicos fuertes y auténticos en un mundo atravesado por el cinismo y la sospecha acerca de los medios masivos, el conocimiento verificable y la racionalidad de la Ilustración?

PARTE I INTERNET

El concepto de «El Estado», como muchos conceptos que van precedidos del artículo determinado, es a la vez demasiado rígido y demasiado susceptible de discusión como para que pueda utilizarse sin más. Es un concepto al que es más fácil aproximarse con un movimiento desde los flancos que con un ataque frontal. En el momento en que pronunciamos las palabras «El Estado», surge toda una serie de fantasmas intelectuales que nos nublan la visión. Sin quererlo y sin darnos cuenta, la idea de «El Estado» nos lleva imperceptiblemente a considerar la relación lógica mutua de diversas ideas, y nos aleja de los hechos de la actividad humana. Es mejor, de ser posible, partir de esta última y ver si ello nos conduce a una idea de algo que resulte que implica el signo distintivo de lo que caracteriza a la conducta política.

JOHN DEWEY, *Liberalismo y Acción Social*: 62.

I. GEEKS Y PÚBLICOS RECURSIVOS

Desde aproximadamente 1997 he estado conviviendo con *geeks* dentro y fuera de Internet. Ello me ha llevado de Boston a Bangalore, de Berlín a Houston y Palo Alto, de congresos a fiestas, pubs y canales de IRC (*Internet Relay Chat*, Protocolo de Charla Basada en Internet). A lo largo de mi trayectoria investigadora han surgido interrogantes sobre compromiso y práctica, sobre ideología e imaginación, incluso cuando la naturaleza exacta de las conexiones entre estas personas e ideas me seguían resultando inescrutables: ¿qué une a los *geeks*? A medida que mi trabajo de campo me arrastraba desde una compañía emergente de Boston que trabajaba con radiografías a un laboratorio multimedia y tecnológico en Berlín, y de ahí a las jóvenes élites emprendedoras de Bangalore, mi duda logística finalmente desembocó en un concepto analítico: los *geeks* se unen como un público recursivo.

¿Cómo llegué a concebir a los *geeks* como un público constituido en torno a las ideas técnicas y morales de orden que les permiten asociarse entre sí? A través de esta pregunta, se puede empezar a comprender la narrativa más amplia de *Two Bits*: la del software libre como un ejemplo modélico de público recursivo y como un conjunto de prácticas que permite que tal público se expanda y se despliegue. En este capítulo describo etnográficamente las diversas, dispersas y novedosas formas de relación que unen a los *geeks* y establezco el concepto de público recursivo para explicar esos vínculos.

Un público recursivo es un público que se constituye por medio de una implicación compartida en el mantenimiento de los medios de asociación a través de los que se congrega como público. Los *geeks* encuentran afinidad entre ellos porque comparten una imaginación moral perdurable de la infraestructura técnica (Internet) que les ha

permitido desarrollar y mantener dicha afinidad en primer término. Mi elaboración del concepto de público recursivo (que no es un término usado por los *geeks*) está relacionada con teorías sobre ideología, públicos, esferas públicas e imaginarios sociales. Para ilustrar el concepto recurro a historias y ejemplos etnográficos que resaltan las imaginaciones de los *geeks* respecto al orden técnico y moral de Internet. Entre estas historias se incluyen la del destino de Amicas, una empresa sanitaria emergente de Boston, entre 1997 y 2003, la de mi participación con académicos y activistas de los nuevos medios en Berlín entre 1999 y 2001 y la de las actividades dentro y fuera de Internet de un grupo de profesionales de la tecnología de la información provenientes en su mayoría de Bangalore, especialmente la de los acontecimientos que rodearon a la aplicación de compartición de archivos P2P (*peer-to-peer*, entre iguales) Napster entre 2000 y 2001.

La expresión «orden moral y técnico» indica tanto la tecnología—principalmente software, hardware, redes y protocolos— como una concepción del orden adecuado de las acciones colectivas políticas y comerciales, es decir, de cómo deberían ordenarse colectivamente la economía y la sociedad. Los públicos recursivos están tan preocupados por el orden moral de los mercados como por el del procomún: no son ni anticomerciales ni antiestatales. Dichos públicos existen independientemente de, y como formas de control sobre, las formas de poder establecidas, que incluyen los mercados y las corporaciones. A diferencia de otros conceptos de público o de esfera pública, la noción de «público recursivo» refleja el hecho de que el principal modo de asociación y actuación de los *geeks* se da a través de Internet, y es precisamente a través de este medio que puede originarse un público recursivo en primer término. Internet no es en sí misma una esfera pública, un público o un público recursivo, sino una infraestructura compleja y heterogénea que constituye y restringe las aspiraciones prácticas cotidianas de los *geeks*, su capacidad para «hacerse públicos» o para componer un mundo común. Como tal, su participación en cuanto públicos recursivos estructura su identidad como individuos creativos y autónomos. El hecho de que los *geeks* aquí descritos se hayan unido a través de listas de distribución y correos electrónicos, boletines de noticias y páginas web, libros y módems, viajes en avión y espacios académicos, y conversaciones y publicaciones cruzadas de modos que no eran posibles antes de Internet está en la raíz de su propio razonamiento acerca del motivo de su asociación. Ellos son quienes

construyen e imaginan este espacio, y el espacio es lo que les permite construirlo e imaginarlo.

¿Por qué recursivos? Denomino *recursivos* a estos públicos por dos razones: en primer lugar, para señalar que este tipo de público incluye las actividades de creación, mantenimiento y modificación de software y redes, además del discurso más convencional que dicha infraestructura posibilita; y en segundo lugar, para sugerir la «profundidad» recursiva de este público, la serie de capas técnicas y jurídicas —desde las aplicaciones a los protocolos y a las infraestructuras físicas de ondas y cables— que son objeto de dicha creación, mantenimiento y modificación. La primera de estas características es evidente al constatar que los *geeks* usan la tecnología como un tipo de argumento para un tipo específico de orden: discuten *sobre* tecnología pero también *a través de* ella; expresan ideas, pero también expresan *infraestructuras* mediante las que las ideas pueden manifestarse (y difundirse) de nuevas maneras. La segunda característica —relativa a las capas— se refleja en la capacidad de los *geeks* para percibirse inmediatamente de las conexiones entre, por ejemplo, Napster (una aplicación de usuario) y TCP/IP (un protocolo de red) y extraer las consecuencias para ambos. Mediante la conexión de estas capas, Napster viene a representar una Internet en miniatura. La cuestión de hasta dónde llegan dichas capas (¿hasta el hardware? ¿hasta las leyes y normativas? ¿hasta las constantes físicas? etc.) circunscribe los límites de la imaginación de orden técnico y moral compartida por los *geeks*.

Ante todo, «público recursivo» es un concepto, no un objeto. Con él pretendo realizar distinciones, permitir la comparación, resaltar características importantes y relacionar dos clases de objetos diversos (Internet y el software libre) en un contexto histórico particular de cambio en las relaciones de saber y poder. Las historias recogidas en este capítulo (y a lo largo de todo el libro) ofrecen una idea aproximada de *cómo* interactúan los *geeks* y de qué hacen técnica y legalmente, pero el concepto de público recursivo ofrece una explicación de *por qué* los *geeks* (o la gente involucrada en el software libre y sus derivados) se relacionan entre ellos, así como un modo de comprobar si otros casos parecidos de afinidad contemporánea y tecnológicamente mediada se estructuran de forma similar.

Recursión

La recursión (o lo «recursivo») es un concepto matemático que constituye un componente estándar de cualquier formación en programación informática. La definición ofrecida por el *Oxford English Dictionary* reza así:

2. a. Relativo a aquel procedimiento repetido donde el resultado requerido en cada paso excepto el último viene dado por el resultado o resultados del siguiente paso, hasta que tras un número finito de pasos se alcanza un resultado final con una evaluación global del resultado.

Este concepto debería distinguirse de la mera iteración o repetición. La recursión siempre está sujeta a un límite y se asemeja más a un proceso de aplazamiento repetido hasta alcanzar el último paso del proceso, momento en el que se calculan todos los pasos aplazados y se obtiene el resultado.

La potencia de la recursión en el campo de la programación se debe a que permite definir los procedimientos a partir de ellos mismos —algo que en principio parece contraintuitivo. Así, por ejemplo:

(defun (factorial n):	Este es el nombre de la función siendo <i>n</i> su valor de entrada
(si (=n 1):	Este es el límite final, o profundidad recursiva
1	si <i>n</i> =1, entonces devuelve 1
(* n (factorial (- n 1))):	si no devuelve <i>n</i> veces el factorial de <i>n</i> -1
	llama al procedimiento desde sí mismo,
	calcula el siguiente paso del resultado antes de
	dar una respuesta ¹

En *Two Bits* un público recursivo es aquel cuya existencia (que consiste únicamente en la interpellación a través del discurso) solo es posible mediante la referencia discursiva y técnica a los medios de creación de dicho público. La recursión está siempre supeditada a un límite que determina la profundidad de un procedimiento recursivo. De esta forma, por ejemplo, un proyecto de software libre puede depender de otro tipo de software o sistema operativo, que a su vez puede depender de un proceso o de unos protocolos abiertos específicos, los cuales a su vez dependen de ciertos tipos de hardware que los

1. Abelson y Sussman, *The Structure and Interpretation of Computer Programs*, p. 30.

implementen. La «profundidad» de la recursión viene determinada por la apertura que necesite el proyecto mismo.

James Boyle ha señalado también la naturaleza recursiva del software libre en particular:

Además, y este es un giro verdaderamente fascinante, cuando el proceso de producción necesita una coordinación más centralizada, un gobierno que guíe el modo de unir las pegadizas porciones modulares, es posible, al menos en teoría, que el sistema de control surja *exactamente de la misma forma*. En este sentido, la producción distribuida es potencialmente recursiva.²

Desde los hechos de la actividad humana

Boston, mayo de 2003. Cafetería Starbucks. Sean y Adrian vienen a recogerme para ir a cenar. Ya he tomado demasiado café, así que me siento junto a la ventana a leer el periódico. Adrian llama por fin para averiguar dónde estoy, se lo digo y promete llegar en quince minutos. Me aburro y salgo a esperarle fuera. Observo el tráfico. Más o menos a la hora acordada (solo tras la caída de las puntocom Adrian logra llegar a tiempo), el nuevo Beetle azul de Sean aparece ante mi vista. Adrian salta del asiento de copiloto para sentarse detrás y yo me subo. Sean lleva conduciendo poco más de un año. Parece confiado, precavido, pero serpentea por las calles de Cambridge. Nuestro destino es Winchester, un municipio cercano al río Charles, donde cenaremos en un restaurante indio que nos ha recomendado un amigo de Sean. Cuando les pregunto qué tal van las cosas, responden: «Bien, bien». Adrian añade: «Bueno, Sean está mejor de lo que ha estado en dos años». «¿En serio?», pregunto impresionado.

Sean dice:

Bueno, al menos más feliz que el año pasado. Yo, bueno, permíteme decirlo así: perdóname, padre, porque he pecado. Aún albergo pensamientos impuros sobre algunos de los altos cargos de la compañía; en ocasiones me da por pensar que no hacen las cosas buscando lo mejor para la empresa sino por su propia conveniencia, y a veces les deseo lo peor.

2. Boyle, «The Second Enclosure Movement and the Construction of the Public Domain», p. 46 [ed. cast.: «El segundo movimiento de cercamiento y la construcción del dominio público», pp. 23-24].

A bordo de este confesionario rodante de color azul, Sean describe a alguna de la gente con la que estoy familiarizado y sobre la que él se esfuerza en no pensar. Le miro y le digo: «Diez ave marías y diez padrenuestros y quedarás absuelto, hijo mío». Girándome hacia Adrian, le pregunto: «¿Qué hay de ti?». Adrian sigue con la broma: «Yo también he pecado. He llegado al punto en que veo que de esta empresa no va a salir nada bueno pero que puedo mantener mi inversión en ella el tiempo suficiente para poder pagarles la universidad a mis hijos». Repongo: «A ti, hijo mío, no te puedo ayudar». Sean dice: «Bueno, es curioso esto del dinero sucio... Uno nunca tiene bastante suciedad».³

Me quedo atónito. Cuando conocí a Sean y Adrian, allá por 1997, Amicas era una prometedora empresa emergente con cinco empleados que trabajaban desde el salón de Adrian y con grandes planes que revolucionarían el mundo de la imagen médica. Juntos se habían confabulado para lograr que el Hospital General de Massachusetts instalase su rudimentario sistema y así poder competir con los grandes perezosos corporativos que normalmente acechaban en los departamentos administrativos: General Electric, Agfa, Siemens. Según Sean y Adrian, estos gigantes estaban defraudando a los hospitales y a los proveedores sanitarios con promesas de tecnologías curalotodo y «silos» de diseños horribles, «sistemas heredados» [*legacy systems*]⁴ y otros monstruos de sistema cerrado procedentes de corporaciones tecnológicas que se remontan a los días de las unidades centrales de IBM. Evidentemente, estas bestias no pertenecían al resplandeciente futuro de la escalabilidad propiciada por Internet. En junio de 2000, Amicas había contratado a nuevos administradores «profesionales», se había trasladado a Watertown y había crecido hasta emplear a cien trabajadores. Habían alcanzado su objetivo de crear una alternativa de Sistema de Archivo y Transmisión de Imágenes (PACS, por sus siglas en inglés) basada en los estándares de Internet y disponible para su uso en los departamentos hospitalarios de radiología.

3. Juego de palabras intraducible fruto de la homofonía entre el verbo «taint» («ensuciar», «manchar») y la contracción «t'ain't», procedente de «there is not» («no hay»). El origen de esta expresión suele atribuirse a William Booth, fundador del Ejército de Salvación, quien al ser interrogado sobre el origen dudoso de ciertas donaciones («tainted money») respondía: «*T'ain't enough!*» («¡No hay suficiente!»). [N. del E.]

4. Según nos explicó Marga Padilla en una sesión de discusión del grupo Traducciones Procomún en Medialab-Prado, este término designa sistemas informáticos antiguos que se han quedado obsoletos y para los que ya no se desarrollan actualizaciones. [N. del E.]

En aquel momento, en la primavera de 2000, Sean aún podía presentarme efusivamente a su nuevo jefe, el mismo hombre al que llegaría a odiar con toda su alma. Pero hacia 2002 Sean estaba frustrado por la extraordinaria variedad de chapuzas y, más concretamente, por la complacencia con que los administradores ignoraban sus recomendaciones y lanzaban programas que casi con toda seguridad fallarían más pronto que tarde. La única explicación que acertó a encontrar Sean, en cierto modo un eterno inexperto en asuntos corporativos, era que la nueva administración era malvada.

Sin embargo, hacia 2003 la compañía había triunfado, llegando a los doscientos empleados y alcanzando unos ingresos regulares y una posición estable en el campo sanitario. Tanto Sean como Adrian se hicieron ricos (no asquerosamente ricos, pero sí lo suficiente) gracias a este éxito. No obstante, en el proceso la compañía se había transmutado exactamente en aquello contra lo que Sean y Adrian pretendían luchar al crearla: una monstruosa corporación suministradora de promesas y software defectuoso. Promesas que Adrian había formulado y software que Sean había creado. El fracaso de Amicas en la transformación de la sanidad era demasiado complejo y técnico como para que la mayoría de estadounidenses lo entendiera, pero se cimentaba en el éxito de Amicas en términos más fácilmente comprensibles: una compañía que crecía y daba beneficios. El propósito de Adrian y Sean al fundar la compañía no era ganar dinero, sino enmendar un sistema sanitario deficiente; sin embargo, el sistema mantuvo sus deficiencias mientras ellos ganaban dinero.

Pese a las bromas, en el confesionario rodante Sean y Adrian sí que me veían como una especie de redentor, un sacerdote (aunque perteneciente a una orden carente de rebaño), cuyo juicio de los asuntos pasados era esencial para la narración de su empresa como un éxito, un fracaso o como una mezcla insatisfactoria y compleja de ambos. Reflexioné sobre aquel extraño momento de confesión, sobre la combinación de reconocimiento y rechazo, sobre la nueva cosificación que Adrian hacía de la empresa como una oportunidad de inversión y sobre la permanente lucha de Sean por armonizar vida y trabajo para hacer el bien en el mundo. Solo la promesa del próximo proyecto, la próxima misión (y la ostensible razón de nuestra cena) podía llegar a mitigar el desastre emocional que de otro modo representaría su empresa. El eterno y arcano fervor de Sean y de Adrian por la promesa de las nuevas tecnologías no remitió, pese a las calamidades cotidianas que estas tecnologías dejaban a su paso. Su fe era sólida y estaba a prueba constantemente.

La pasión de Adrian y Sean no era el dinero —aunque este fuera una droga potente— sino Internet: las formas en que Internet podía sustituir la infraestructura existente de los hospitales y servicios sanitarios, cumplir las viejas promesas de la telemedicina y la telerradiología y, por encima de todo, nivelar un terreno de juego sistemáticamente distorsionado y manipulado por instituciones empresariales y gubernamentales que buscaban el secretismo y el control privado y obstaculizaban el progreso. Como Adrian me explicaba repetidamente, en el ámbito sanitario tal terreno de juego descompensado no solo era injusto sino también malicioso e irresponsable: estaba costando vidas, estaba frenando la creación y despliegue de tecnologías y soluciones que podrían reducir los costes y con ello ofrecer más asistencia sanitaria a más personas. Internet no era parte del problema, sino parte de la solución a los problemas que aquejaban a la asistencia sanitaria en los 90.

Al final de nuestro viaje en coche, en el restaurante indio en Winchester, me enteré de su siguiente plan, un proyecto llamado MedCommons, que se basaría en los ideales del software libre y daría a las personas una manera de controlar y gestionar de forma segura sus datos sanitarios personales. La retórica del procomún y la promesa de Internet como infraestructura dominaron nuestra conversación, pero la realidad de la financiación y la cuestión de si MedCommons podría realizarse sin fundar otra empresa quedaron en el aire. Traté de imaginar la forma que adoptaría una futura confesión.

Los *geeks* y sus Internets

Sean y Adrian son *geeks*. Cada uno a su manera, son empresarios e idealistas, una combinación a veces paradójica. Ciertamente están obsesionados con la tecnología, pero especialmente con Internet, y se distinguen claramente de aquellos otros que se obsesionan con cualquier clase de tecnología. No son demasiado representativos —no se corresponden con todos los *geeks*—, pero su manera de pensar sobre Internet y sus posibilidades sí puede serlo. Dentro de la rica historia de sus éxitos y fracasos, se pueden vislumbrar los contornos de una pregunta: ¿dónde recaen sus simpatías? ¿Con quiénes están? ¿A quiénes reconocen como semejantes a ellos? ¿Qué podría vincularles con otros *geeks* si no es una empresa, una nación, un lenguaje o una causa? ¿Qué liga a estos dos *geeks* con cualesquiera otros?

En los años 80 Sean trabajó para la Reserva Federal, y allí entró en contacto con UNIX, el lenguaje de programación C, EMACS, Usenet, el software libre y la Fundación para el Software Libre. Con todo, Sean no era un *hacker* del software libre; de hecho, se resistía a todos mis tentativas de llamarle *hacker*. No obstante, lanzó una serie de proyectos y empresas con Adrian que se valían del repertorio de prácticas e ideas ligadas al software libre, incluyendo su proyecto MedCommons, que se basaba de forma más o menos explícita en los ideales del software libre. Adrian tiene el título de Medicina y de Ingeniería, y es un emprendedor en serie, siendo Amicas su mayor éxito hasta ahora. A lo largo de la última década ha asistido a toda clase de congresos y encuentros dedicados al software libre, el código abierto, los estándares abiertos y demás, casi siempre como el único representante del sector sanitario. Ambos se graduaron en el MIT (Sean en Economía, Adrian en Ingeniería), uno de los más destacados núcleos de Internet y legendario hogar de *hackers*, pero ninguno de ellos fueron *hackers* del MIT, ni siquiera fueron grandes figuras de la informática.

Sus metas al crear una empresa emergente radicaban en su comprensión de Internet como una infraestructura: una *infraestructura estandarizada* con ciertas propiedades extremadamente potentes, entre las cuales la menor no era su flexibilidad. Sean y Adrian hablaban sin parar de sistemas abiertos, estándares abiertos y de la necesidad de que Internet permanezca abierta y estandarizada. Adrian hablaba genéricamente de cómo así se revolucionaría la sanidad; Sean hablaba específicamente de cómo dicha consideraciones estructuraron la forma de diseñar y escribir el software de Amicas. Ambos participaron en comités de estandarización y en las discusiones dentro y fuera de la Red que constituyen el equivalente a la elaboración de políticas en el mundo de Internet. La empresa que crearon era «virtual», es decir, basada en herramientas que dependían de Internet y que permitían a sus empleados administrar y ejecutar tareas desde muy diversas localizaciones, sin por ello quedar exentos de frustraciones, claro: Sean esperó años para acceder a banda ancha en su casa y los hospitales donde trabajaban estaban atenazados en redes virtuales privadas, *intranets* y cortafuegos de seguridad que traicionaban las promesas de apertura que Sean y Adrian pregonaban.

Internet no era el objeto de sus trabajos y sus vidas, pero representaba al detalle un tipo de orden moral o social incorporado en un sistema técnico y disponible para que cualquiera lo usara como una

plataforma a través de la que poder competir para mejorar e innovar en cualquier ámbito. Aunque no todos los empresarios de Internet de los 90 veían la Red de la misma forma, no cabe duda de que Sean y Adrian no eran ni mucho menos los únicos con esa visión. Algo de su particular forma de entender Internet como representación de un orden moral —simultáneamente una red, un mercado, un público y una tecnología— era compartido por un gran número de gente, aquellos a los que ahora me refiero simplemente como *geeks*.

El término *geek* pretende ser inclusivo e indicar la problemática de los públicos recursivos. Puede haber otros igualmente útiles, pero quizás estén sobredeterminados semánticamente, especialmente el término *hacker*, que independientemente de su alcance definitorio, tiende a connotar alguien subversivo y/o criminal y a excluir a empresarios afines a los *geeks*, así como a abogados y activistas.⁵ Al igual que la noción de *público* en «público recursivo», el término *geek* pretende señalar que los *geeks* se sitúan al margen del poder, al menos en algunos aspectos, y que no son capitalistas ni tecnócratas, por más que abran empresas o trabajen en el gobierno o la industria.⁶ El término *geek* pretende señalar un modo de pensar y de trabajar, no una identidad; es un modo o una

5. Para la historia canónica, véase Levy, *Hackers*. El término *hack* se refería (y aún lo hace) a un uso ingenioso de la tecnología, habitualmente no intencionado, para llevar a cabo cierta tarea de un modo elegante. Los medios masivos han redefinido exitosamente el término para referirse a usuarios de ordenadores que se cuelan y cometan actos criminales en computadoras corporativas, gubernamentales o personales conectadas a una red. Muchas personas que se identifican como *hackers* insisten en que se aluda a los elementos criminales como *crackers* (véase, en particular, las entradas sobre «*Hackers*» «*Geeks*» y «*Crackers*» en el Jargon File, <http://www.catb.org/~esr/jargon/>, también publicado por Raymond en el *The New Hackers' Dictionary*). Sobre la cuestión de las definiciones y las características culturales y éticas de los hackers, véase Coleman, «The Social Construction of Freedom», cap. 2.

6. Un ejemplo del uso de *geek* se halla en Star, *The Cultures of Computing*. Diversas denuncias (por ejemplo, Barbrook y Cameron, «The California Ideology»; Borsook, *Techno-libertarianism*) tienden a centrarse en los relatos periodísticos de una ideología que tiene poco que ver con lo que realmente hacen los *hackers*, *geeks* y empresarios. Una distinción categórica más relevante que la establecida entre *hackers* y *geeks* es la que se realiza entre *geeks* y tecnócratas; en el caso de los tecnócratas, la «antropología de la tecnocracia» se propone como el estudio de los límites de la racionalidad técnica, en concreto de las formas mediante las que la «planificación» crea «grietas en la forma que funcionan como ‘objetivos de intervención’» (Riles, «Real Time», p. 393). Ciertamente los «tecnócratas» de Riles no son los «*geeks*» que retrato aquí (o al menos solo lo son en sus frustrantes jornadas laborales). Los *geeks* tienen inclinaciones liberal-libertarias, específicamente vinculadas a Hayek o Feyerabend, pero es más probable que contemplen los fallos técnicos no como fallos de planificación sino como errores, ineficiencias o en ocasiones como productos de la arrogancia o la estupidez humanas que nacen de la fe en la planificación.

cualidad que permite a la gente encontrarse, por razones distintas al hecho de compartir oficina, titulación, idioma o nación.

Hasta mediados de los 90, *hacker*, *geek* y *nerd informático*⁷ designaban a un tipo de gente muy específico: programadores y mirones (*lurkers*) de redes relativamente ocultas, usualmente estudiantes universitarios, informáticos y «*amateurs*» o «*hobbyists*» [«aficionados»]. Un clásico autodiagnóstico en clave burlesca llamado el *The Geek Code* [«El Código Geek»] de Robert Hayden, detallaba de forma precisa y cómica las diversas formas en las que alguien podía ser un *geek* en 1996 —habilidades con UNIX/Linux, amor/odio hacia *Star Trek*, hábitos particulares de alimentación y vestimenta—, si bien el propio Hayden señala que los *geeks* de principios de los 90 han desaparecido. La élite subcultural relativamente homogénea que una vez existió ha sido invadida:

La Internet de 1996 era todavía un paraíso salvaje y virgen de *geeks* y lumbres sin rastro de *script kiddies* o de habitantes de AOL. Cuando las cosas cambiaron, me quedé totalmente desorientado. Es decir, todo lo *geek* de Internet se esfumó y fue reemplazado por clichés de *Expediente X* y políticos que aprobaron leyes sobre una tecnología que se negaban a comprender.⁸

Para los puristas como Hayden, los *geeks* llegaron primero y entendieron algo, vivieron de una forma que no puede ser comprendida por los «*script kiddies*» (adolescentes que ejecutan el equivalente *hacker* de las pintadas con spray o del derribo de vacas),⁹ *crackers* o usuarios de

7. Opto de nuevo por prescindir de una traducción literal de «*nerd*» («empollón») y dejar el término en inglés. Igualmente vuelvo a recurrir a la definición recursiva basada en la versión 4.4.7 del *Jargon File* («Archivo de Jerga», en inglés) con el fin de dotar de mayor densidad connotativa al término en su contexto *geek*:

Nerd: 1. [lenguaje común] Término peyorativo aplicado a cualquiera con un coeficiente de inteligencia superior a la media y con escasas dotes para la chábola y los rituales sociales ordinarios. 2. [jerga] Término elogioso aplicado (en consciente referencia irónica al sentido 1) a alguien que sabe lo que es realmente importante e interesante y no presta atención a las distracciones derivadas de la charla trivial y los tontos juegos de estatus. Compárese con *geek*.

Disponible en: <http://www.catb.org/~esr/jargon/html/N/nerd.html> [N. del E.]

8. Véase *The Geek Code*: <http://www.geekcode.com/>. A menudo también se identifica a los *geeks* por la jocosidad y agilidad con que manipulan estas etiquetas y caracterizaciones. Para un buen ejemplo, véase Michael M. J. Fischer, «Worlding Cyberspace».

9. Intento completar esta definición de Kelty de *script kiddie* (y también justificar nuestra opción por mantener este otro término en inglés) remitiéndome una vez más a la versión 4.4.7 del *Jargon File* («Archivo de Jerga», en inglés):

AOL, todos ellos despreciados por los *geeks* del estilo de Hayden por considerarlos usuarios no cualificados que desfilan por Internet como si les perteneciera. Aun siendo ciertamente elitista, Hayden capta la distinción entre quienes pueden legítimamente llamarse *geeks* (o *hackers*) y quienes no, distinción que a menudo se formula de forma recursiva, por supuesto: «Eres *hacker* cuando otro *hacker* te llama *hacker*».

Sin embargo, a partir del explosivo crecimiento de Internet, el término *geek* se ha convertido en una designación más común y, en consecuencia, mi uso del mismo sugiere un papel más amplio que el de programador/*hacker*, pero no tan amplio como el de «todos los usuarios de Internet». Pese a la frustración de Hayden, los *geeks* todavía permanecen unidos en una élite y pueden distinguirse fácilmente de los «usuarios de AOL». Algunas de las personas con las que discuto no se llamarían a sí mismos *geeks*, y otras sí. No todos son ingenieros o programadores: he conocido a empresarios, abogados, activistas, blogueros, gastroenterólogos, antropólogos, lesbianas, esquizofrénicos, científicos, poetas, enfermos de malaria, capitanes de barco, traficantes de droga y gente que tiene lémures de mascota, muchos de los cuales se refieren a sí mismos como *geeks* en algunos momentos. También hay abogados, políticos, sociólogos y economistas que pueden no llamarse a sí mismos *geeks* pero cuidan de Internet tal y como lo harían los *geeks*. En contraste, los «usuarios» de Internet, incluso quienes la usan dieciocho horas al día para jugar o despachar mercancías, no son necesariamente *geeks* según esta caracterización.

Sistemas Operativos y Sistemas Sociales

Berlín, noviembre de 1999. Estoy en un club muy sofisticado de Mitte llamado WMF. Son casi las ocho —cinco horas pronto para considerarme un *hipster*, pero el contexto es extremadamente enrollado. WMF está en un edificio abandonado y difícil de encontrar de la antigua Alemania del Este; está parcialmente reformado, decorado con

Script kiddie: 1. [muy común] La forma más baja de *cracker*; [...] Usada para gente con pericia técnica limitada que emplea herramientas de uso sencillo, preconfiguradas y/o automatizadas para llevar a cabo actividades disruptivas contra sistemas en red. 2. De forma más general, un *script kiddie* escribe (o más probablemente copia y pega) código sin tener ni desechar tener un modelo mental de lo que hace dicho código; alguien que concibe el código como conjuros mágicos y solo pregunta: «¿Qué necesito teclear para que suceda esto?».

Disponible en: <http://www.catb.org/jargon/html/S/script-kiddies.html> [N. del E.]

una mezcla de mobiliario nuevo y viejo, videoproyectores, altavoces, barras improvisadas e iluminación de pista de baile. Una multitud de unas cincuenta personas se esparce en medio del humo y las botellas de cerveza Beck's, sentadas en taburetes, sillas, sofás y por el suelo. Estamos escuchando a un académico que lee un artículo sobre Claude Shannon, el ingeniero del MIT a quien se atribuye la creación de la teoría de la información. El autor está fumando y leyendo en alemán mientras el público escucha educadamente. Su charla dura unos setenta minutos. Hay preguntas y una discusión superficial. En cuanto la multitud se disuelve, me encuentro a mí mismo, en un vacilante alemán que rápidamente cambia al inglés, manteniendo una serie de animadas conversaciones sobre la licencia GPL, la distribución Debian de Linux, los estándares abiertos de la radio por Internet y una variedad de temas de los que Claude Shannon es el perfecto tecnopaterfamilias espectral, aun cuando su invocación de setenta minutos haya desentonado fuertemente con el entorno.

Pese a mi deficiente alemán,igo arreglándome las para entrar en profundidad en cuestiones que me parecen extremadamente familiares: estándares de Internet y sistemas abiertos y problemas de licencias y espacios de nombres de dominio y legislación de patentes, etc. No estoy entre gente de negocios, esto no es una *start-up*. Como acabaría descubriendo, había incluso cierto desdén hacia *die Krauttenfaktor*, el factor corbata-y-traje, en estos esporádicos eventos híbridos acogidos por Mikro e.V., un colectivo sin ánimo de lucro de periodistas, académicos, activistas, artistas y otros interesados en nuevos medios, Internet y temas relacionados. Mikro cuenta con el respaldo de gente de Alemania, Holanda, Austria, y apunta más al Este. Describen con cierto orgullo Berlín como «lo más oriental a lo que llega Occidente» y organizan una foto grupal en la que, mirando hacia el Oeste, posan tras la estatua de Marx y Lenin, que de cara al Este miran eternamente la icónica torre de radio de Alemania del Este (*Funkturm*) de Alexanderplatz. Los miembros de Mikro son resueltamente activos y consideran el tema de Internet-como-infraestructura no en función de su potencial para oportunidades de negocio, sino en términos urgentemente políticos e impenitentemente estéticos —términos que, sin embargo, son similares a los de Sean y Adrian, de quienes aprendí el lenguaje que me permite mezclarme con la multitud de Mikro en el WMF. Ahora soy un *geek*.

En poco tiempo, me encuentro hablando con Volker Grassmuck, miembro fundador de Mikro y organizador del exitoso congreso

«*Wizards of OS*»,¹⁰ celebrado meses antes bajo el intrigante subtítulo «Sistemas Operativos y Sistemas Sociales». Grassmuck me invita a participar en una sesión de planificación de la próxima edición del congreso, enmarcada en el Chaos Computer Congress, un encuentro de *hackers* que se celebra cada año en diciembre en Berlín.¹¹ En los siguientes meses, conoceré a innumerables personas que, de forma inusitada para ser artistas y activistas, están extrañamente obsesionados con la configuración de sus distribuciones de Linux o con el *hackeo* del protocolo http o con asistir a las audiencias del Parlamento alemán sobre la reforma del *copyright*. No en vano las vidas políticas de estas personas han mezclado sistemas operativos y sistemas sociales de formas que son más que metafóricas.

La idea de orden frente al teclado

Si la intuición puede llevarle a uno de *geek* en *geek*, de una compañía emergente a un club nocturno, y a través de diversos países, idiomas y orientaciones profesionales, ello solo puede deberse a un conjunto compartido de ideas sobre cómo encajan las cosas en el mundo. Estas ideas pueden ser «culturales» en el sentido tradicional de encontrar expresión en una comunidad de personas que comparten experiencias, hogar, nación, lengua, dialecto, etnia, normas u otros marcadores de pertenencia y co-presencia. Pero dado que Internet —como el colonialismo, la emisión por satélite y el transporte aéreo, entre otras cosas— franquea todas estas líneas sin ningún miramiento, la idea compartida de orden se entiende mejor como parte de un público, o esfera pública, una enorme república de letras, medios e ideas que penetran y atraviesan nuestros pensamientos, documentos, cartas y conversaciones, a una escala y con un alcance planetarios.

Sin embargo, la «esfera pública» es algo extraño. Es a la vez un concepto —que pretende dar sentido a un espacio que trasciende el aquí y ahora y se compone de escritos, ideas y discusiones— y un

10. La traducción literal sería «Magos de los Sistemas Operativos», si bien estamos ante un juego de palabras intraducible que alude a la novela de L. Frank Baum *El Mago de Oz* (originalmente titulada *The Wonderful Wizard of Oz*). [N. del E.]

11. Debido a la cada vez más desbordante demanda de asistentes, en 2012 el congreso trasladó su sede de Berlín a Hamburgo (donde, por otro lado, se había celebrado desde su primera edición en 1984 y hasta 1997, pasando a Berlín en 1998). En 2017 el CCC se trasladó a Leipzig debido a las obras en la sede del Centro de Congresos de Hamburgo. [N. del E.]

conjunto de ideas que la gente tiene sobre sí misma y sobre su propia participación en dicho espacio. Debo ser capaz de imaginarme a mí mismo hablando y escuchando en tal espacio y a muchas otras personas haciendo lo propio de acuerdo con reglas no escritas que todos compartimos. No necesito una teoría completa ni tampoco llamarlo «esfera pública», pero de alguna forma debo compartir una idea de orden con toda esa otra gente que también se imagina a sí misma participando en y sometiéndose a dicho orden. De hecho, si la esfera pública existe como algo más que una teoría, su base no es otra que ese mismo imaginario compartido de orden, un imaginario que proporciona una guía desde la que formular juicios y un mapa para modificar o alcanzar dicho orden. Sin este imaginario compartido, una esfera pública no es más que una cacofonía de voces e información, una corriente de datos, estructurados y formateados por y para máquinas, ya estén en papel o en formato electrónico.

Charles Taylor se basa en la obra de Jürgen Habermas y Michael Warner para sugerir que la esfera pública (tanto la idea como la cosa) que surgió en el siglo XVIII se creó mediante prácticas de comunicación y asociación que reflejaban un orden moral en el que el público se sitúa al margen del poder y guía o controla su operación a través del discurso compartido y la discusión ilustrada. Contrariamente a la experiencia de los cuerpos que se encuentran en un espacio común (lo que Taylor llama «espacios tópicos», como conversaciones, rituales, asambleas...), el componente crucial es que la esfera pública

trasciende cualquier espacio tópico. Podríamos decir que teje una pluralidad de espacios de este tipo en un espacio mayor de concurrencia no presencial. Se considera que el debate que hemos mantenido hoy entre nosotros forma parte de la misma discusión pública que proseguirá mañana en la seria conversación que tendrá otra persona, y en la entrevista del periódico del jueves, etc. [...] La esfera pública que surge en el siglo XVIII es un espacio común metatópico.¹²

Por todo esto, Taylor define su versión de público como un «imaginario social», una forma de capturar un fenómeno que oscila entre una

12. Taylor, *Modern Social Imaginaries*, p. 86 [ed. cast.: *Imaginarios sociales modernos*, p. 108].

existencia concreta «ahí fuera» y una existencia racional imaginada «aquí dentro». Hay otros diversos espacios imaginados semejantes —la economía, el pueblo soberano, la sociedad civil— y en la historia filosófica de Taylor todos ellos se relacionan entre sí mediante «ideas de orden moral y social» que se han desarrollado en Occidente y alrededor del mundo.¹³

El imaginario social de Taylor tiene un objetivo específico: resistir al «espectro del idealismo», la distinción entre ideas y prácticas, entre las «ideologías» y el llamado mundo material como «agentes causales rivales». Taylor sugiere:

Las prácticas humanas son la clase de cosa que se define por tener un sentido, y eso significa que son inseparables de ciertas ideas; es imposible separar un aspecto de otro para poder plantear la pregunta «¿qué es causa de qué?».¹⁴

Incluso si la explicación causal materialista es satisfactoria, como a menudo ocurre, Taylor sugiere que lo es «al precio de resultar poco plausible como principio universal», y ofrece en su lugar un análisis del crecimiento de los imaginarios modernos de orden moral.¹⁵

El concepto de público recursivo, como el de la esfera pública de Taylor, se entiende aquí como un tipo de imaginario social. La razón primordial es la de evitar la dicotomía entre ideas y práctica material. Dado que la creación de software, redes y documentos legales es precisamente el tipo de actividad que trastoca esta distinción —pues lo que se crea son a la vez ideas y objetos que tienen efectos materiales en el mundo, tanto expresivos como performativos—, es extremadamente difícil identificar la materialidad propiamente material (¿el código fuente? ¿los chips informáticos? ¿los centros de fabricación de semiconductores?). Esta es la primera de las razones por las que un público recursivo ha de distinguirse de las clásicas fórmulas de la esfera pública.

13. Sobre la cuestión de las comunidades imaginadas y el papel de las tecnologías de la información en redes imaginadas, véase Green, Harvey y Knox, «Scales of Place and Networks»; y Flichy, *The Internet Imaginaire*.

14. Taylor, *Modern Social Imaginaries*, p. 32 [ed. cast.: *Imaginarios sociales modernos*, pp. 47-48].

15. *Ibid.*, pp. 33-48. La historia que narra Taylor sobre la transición de la nobleza feudal a la sociedad civil y al auge de las democracias republicanas (por más que incompleta) es comparable a la historia de Foucault acerca del nacimiento de la biopolítica, en *La naissance de la biopolitique* [ed. cast.: *Nacimiento de la biopolítica*], como un intento de contextualizar históricamente las formas de gobierno en relación a sus teorías y sistemas, así como dentro de las formas materiales que adopta.

En otras palabras, un público recursivo requiere un tipo de imaginación que incluye la escritura, la publicación, el habla y la argumentación con las que estamos familiarizados, pero también la elaboración de nuevos tipos de infraestructuras informáticas para la difusión, archivo, movimiento y modificabilidad de nuestros enunciados.

El concepto de imaginario social evita también acertijos que nacen del concepto de «ideología» y su distinción de la práctica material. La ideología en su uso técnico ha quedado arrinconada de forma lenta pero segura por su significado peyorativo: «Lo ideológico nunca es la posición de uno mismo; es siempre la postura de otros, es siempre *su* ideología».¹⁶ Si uno intentara explicar cualquier ideología particular en términos no peyorativos, no hay aparentemente nada que pudiera salvar dicha explicación de convertirse ella misma en ideológica.

El problema no es nuevo. Clifford Geertz lo apuntó en «La ideología como sistema cultural», como antes lo hizo Karl Mannheim en *Ideología y Utopía*: se trata de la dificultad de emplear un concepto no evaluativo de ideología.¹⁷ De todas las versiones de la lucha por el concepto de una sociología científica u objetiva, la pretensión de explorar la ideología objetivamente es la que más exaspera. Como lo expresó Geertz:

A los hombres no les importa tener creencias a las cuales puedan asignar una gran significación moral examinadas desapasionadamente, por pura que sea su finalidad; y si los hombres están ideologizados en alto grado, puede resultarles imposible creer que un tratamiento desinteresado de las cuestiones fundamentales de convicción social y política pueda ser otra cosa que una impostura escolástica.¹⁸

Mannheim ofreció una respuesta: una versión del relativismo epistemológico donde el análisis de ideología incluyera la posición ideológica del analista. Geertz ofreció otra: una ciencia de «acción sim-

16. Ricoeur, *Lectures on Ideology and Utopia*, p. 2. [ed. cast.: *Ideologías y utopía*].

17. Geertz, «Ideology as a Cultural System» [ed. cast.: «La ideología como sistema cultural»]; Mannheim, *Ideology and Utopia* [ed. cast.: *Ideología y utopía*]. Ambos, por supuesto, también sitúan el origen del uso científico del término aproximadamente en la obra «La ideología alemana» de Karl Marx y de forma algo más distante en los escritos ilustrados de Destutt de Tracy.

18. Geertz, «Ideology as a Cultural System», p. 195 [ed. cast.: «La ideología como sistema cultural», p. 172].

bólica» basada en la obra de Kenneth Burke con la contribución de un ejército de filósofos y críticos literarios.¹⁹ Ni el concepto de ideología, ni los métodos de la antropología cultural han sido los mismos desde entonces. La «ideología» se ha convertido en una de las armas de crítica más utilizadas (más difundidas, dirían algunos), entendiendo la crítica como el análisis de los patrones culturales que se dan en el lenguaje y las estructuras simbólicas, con el fin de sacar a la luz sistemas de hegemonía, dominación, autoridad, resistencia y/o falta de reconocimiento.²⁰ Sin embargo, hay tantas (o más) probabilidades de que las prácticas de crítica se dirijan hacia los propios investigadores críticos, con el fin de mostrar cómo los procesos de análisis, los supuestos ocultos, las funciones latentes de la universidad u otros atributos no reconocidos del mundo real material y no ideológico llevan al analista a caer en una trampa ideológica.

El concepto de ideología da un giro hacia el de «imaginario social» en *Lectures on Ideology and Utopia*, donde Paul Ricoeur propone el pensamiento ideológico y el utópico como dos componentes de «la imaginación social y cultural». La visión de Ricoeur divide los enfoques del concepto de ideología en tres tipos básicos —distorsionador, integrador y legitimador— en función del modo en que los actores interactúan con la realidad a través de la imaginación (simbólica). ¿La imaginación distorsiona la realidad, la integra o la legitima de cara al Estado? Ricoeur defiende la segunda opción, con resabios geertzianos: las ideologías integran la estructura simbólica del mundo en un todo coherente y «solo porque la estructura de la vida social ya es simbólica puede distorsionarse».²¹

19. *Ibid.*, pp. 208–213 [ed. cast.: pp. 182-187].

20. La profundidad y el alcance de esta cuestión es obviamente enorme. *Lectures on Ideology and Utopia* [ed. cast.: *Ideologías y utopía*] de Ricoeur es un análisis excelente del problema de la ideología previo a 1975. Los libros de Terry Eagleton *The Ideology of the Aesthetic* [ed. cast.: *La estética como ideología*] y *Ideology: An Introduction* [ed. cast.: *Ideología: Una introducción*] constituyen exploraciones marxistas que incluyen debates sobre hegemonía y resistencia en el contexto de la teoría artística y literaria de los 80. Slavoj Žižek crea un sistema de análisis algebraico de inspiración lacaniana que combina marxismo y psicoanálisis de modos novedosos (véase Žižek, *Mapping Ideology* [ed. cast.: *Ideología: Un mapa de la cuestión*]). Existe incluso un intento de reemplazar el concepto de ideología por una metáfora de «software» y «memes» (véase Balkin, *Cultural Software*). El núcleo de la cuestión de la ideología como una práctica (y las vicisitudes de índole materialista que la perturban) está también en el corazón de obras de Pierre Bourdieu y sus seguidores (sobre la relación entre ideología y hegemonía, véase Laclau y Mouffe, *Hegemony and Socialist Strategy*). En antropología, véase Comaroff y Comaroff, *Ethnography and the Historical Imagination*.

21. Ricoeur, *Lectures on Ideology and Utopia*, p. 10. [ed. cast.: *Ideologías y utopía*].

Para Ricoeur, la propia esencia de la vida empieza en la interpretación de la realidad y, por consiguiente, las ideologías (así como las utopías, y quizás las conspiraciones) pueden ser tratadas como sistemas que integran esas interpretaciones en los conjuntos significativos de la vida política. Con todo, el análisis de Ricoeur de la integración de la realidad a través de la imaginación social no aborda el modo en que funciona la imaginación: ¿cuál es exactamente la naturaleza de esta acción o interpretación simbólica, de esta imaginación? ¿Puede uno conocerla desde fuera, y resiste la distinción entre ideología y práctica material? Tanto Ricoeur como Geertz albergan la esperanza de que la ideología pueda hacerse científica, de que la integración de la realidad a través de la acción simbólica requiera solo el desarrollo de conceptos adecuados al trabajo.

Volvamos a Charles Taylor. En *Modern Social Imaginaries* el concepto de imaginario social es distintivo en el sentido de que trata de captar específicamente las imaginaciones integradoras de orden moral y social moderno. Taylor recalca que se trata de *imaginaciones* —no necesariamente teorías— del orden moral y social moderno:

Por imaginario social entiendo algo mucho más amplio y profundo que las construcciones intelectuales que puedan elaborar las personas cuando reflexionan sobre la realidad social de un modo distanciado. Pienso más bien en el modo en que imaginan su existencia social, el tipo de relaciones que mantienen unas con otras, el tipo de cosas que ocurren entre ellas, las expectativas que se cumplen habitualmente y las imágenes e ideas normativas más profundas que subyacen a estas expectativas.²²

Los imaginarios sociales se han desarrollado históricamente y dan lugar a nuevas instituciones y nuevas subjetividades; los conceptos de público, mercado y sociedad civil (entre otros) se sitúan en las facultades imaginativas de los actores que reconocen la existencia común y compartida de esas ideas, aunque difieran en ciertos detalles, y las prácticas de esos actores reflejan un compromiso en el desarrollo de estos conceptos compartidos.

Los imaginarios sociales son una extensión del «trasfondo» en sentido filosófico: «Una comprensión en gran medida inarticulada

22. Taylor, *Modern Social Imaginaries*, p. 23 [ed. cast.: *Imaginarios sociales modernos*, p. 37].

de nuestra situación».²³ El ejemplo que utiliza Taylor es el de nuestra marcha en una manifestación: la acción está en nuestro repertorio imaginativo y tiene un significado que no puede reducirse al contexto local.

Sabemos que debemos reunirnos, tomar pancartas y caminar [...]. Comprendemos el ritual [...]. La idea inmediata que tenemos de lo que estamos haciendo, por ejemplo lanzar un mensaje al gobierno y a nuestros conciudadanos para que terminen los recortes, cobra sentido en un contexto más amplio, donde nos hallamos inmersos en una relación continuada con otras personas, y donde es aceptable que nos dirijamos a ellas de esta manera.²⁴

Pero también nos situamos «internacionalmente» e «históricamente» en un contexto de historias, imágenes, leyendas, símbolos y teorías.

El trasfondo que da sentido a cualquier acto particular es, pues, amplio y profundo [...] Nuestros actos cobran sentido en el marco del conjunto de nuestro mundo, es decir, de nuestra concepción del lugar que ocupamos en el tiempo y en el espacio, en la historia y entre las demás personas.²⁵

El imaginario social no lo conforman simplemente las normas que estructuran nuestras acciones, sino que también abarca una concepción de lo que hace esas normas alcanzables o «realizables», en palabras de Taylor. Esta es la idea de un «orden moral», uno que esperamos que exista o, en caso contrario, que proporcione un plan para alcanzarlo. Para Taylor, existe algo así como una «idea moderna de orden» que incluye, entre otras cosas, ideas sobre qué significa ser un individuo, sobre cómo las pasiones y deseos individuales entran en relación con la asociación colectiva y, principalmente, ideas sobre la convivencia en el tiempo (destaca una concepción radicalmente secular del tiempo, en un sentido más amplio que el simple de «fuera de la religión»). En ningún caso insiste en que esta sea la única definición de la modernidad (la puerta está completamente abierta a comprender modernidades alternativas), sino en que la idea moderna de orden moral es una que domina y estructura un amplio abanico de instituciones e individuos por todo el mundo.

23. *Ibid.* p. 25 [ed. cast.: *Imaginarios sociales modernos*, p. 39].

24. *Ibid.* pp. 26-27 [ed. cast.: *Imaginarios sociales modernos*, p. 39].

25. *Ibid.* p. 28 [ed. cast.: *Imaginarios sociales modernos*, p. 42].

La «idea moderna del orden moral» es un buen punto para regresar a la cuestión de los *geeks* y sus públicos recursivos. ¿Son distintas las ideas de orden que comparten los *geeks* y las que esboza Taylor? ¿Poseen los *geeks* como Sean o Adrian, o como los activistas de Berlín, un imaginario social distintivo? ¿O participan (pese a su dispersión planetaria) en esta idea moderna común de orden moral? ¿Tienen algo distintivo las historias y narrativas, las herramientas y tecnologías, las teorías e imaginaciones que siguen y sobre las que se basan? Por ejemplo, la aspiración de Sean y Adrian de transformar el sistema sanitario parece estar motivada por una noción de orden moral en que los medios de dispensación de asistencia sanitaria podrían ser más justos, pero también está atravesada por ideas técnicas sobre el papel de los estándares, Internet o los problemas con las actuales soluciones técnicas. Así, aunque pueda parecer que se limitan a propugnar una mejor asistencia sanitaria, lo hacen utilizando una jerga técnica y unas prácticas que probablemente sean bastante ajenas a los legisladores, altos ejecutivos y grupos de promoción de la salud que de otro modo podrían estar en plena sintonía.

La afinidad entre *geeks* se procesa a través de ideas de orden que son tanto morales *como* técnicas —ideas de orden que de hecho mezclan «sistemas operativos y sistemas sociales». Estos sistemas incluyen los medios técnicos (la infraestructura) mediante los que los *geeks* se conocen, se reúnen, colaboran y planifican, así como la manera en la que hablan y piensan sobre estas actividades. La infraestructura —Internet— permite que una variedad verdaderamente amplia y heterogénea de personas se encuentre e interaccione entre sí. Ello equivale a afirmar que la idea de orden compartida por los *geeks* es compartida porque son *geeks*, porque «lo pillan», porque la estructura y el software de Internet han adoptado una forma específica a través de la cual los *geeks* llegan a entender el orden moral que dota de consistencia al tejido de sus vidas políticas.

La Ruta de la Seda de Internet

Bangalore, marzo de 2000. Estoy en otro bar, esta vez en una de las calles más de moda de Bangalore. El bar se llama *Purple Haze* y me ha llevado allí al día siguiente de mi llegada Udhay Shankar N. El interior es oscuro, lleno de humo y hombres entre los dieciocho y los treinta, y decorado con pósters de Jimi Hendrix, Black Sabbath, Jim Morrison

(Udhay: «Odio ese grupo»), Led Zeppelin y, algo fuera de lugar, Frank Zappa (Udhay: «Uno de mis héroes políticos y musicales»). Todos los hombres parecen estar cantando la música al unísono, *heavy metal* sin excepción.

Entablé una conversación algo forzada con Udhay y su primo Kirti sobre la diferencia entre la música carnática y el *rock and roll*, que podría resumirse en lo siguiente: la música carnática disminuye el metabolismo y el pulso, y favorece así un estado mental de relajación; el *rock* hace lo contrario. Dado que mi objetivo era centrarme en Internet y en las cuestiones de apertura, ya estaba decidido a no prestar atención a esta conversación musical. Al echar la vista atrás me doy cuenta de que cometí un grave error metodológico: infravaloré hasta qué punto el tema de la música era precisamente una de mis primeras vías hacia cuestiones como la «reorientación del saber y el poder» en las que estaba interesado. A lo largo de esa velada y de los días siguientes, Udhay me presentó, como me había prometido, a un gran número de personas que, o bien conocía, o bien habían sido compañeros de trabajo. Casi todos ellos parecían amar sinceramente el *heavy metal*.

Conocí a Udhay Shankar N. en 1999 a través de un boletín de noticias por correo electrónico llamado *Tasty Bits from the Technology Front* [«Sabrosos Bits/Bocados desde el Frente Tecnológico»]. Era una de las fuentes que seguía de cerca mientras estaba en Berlín tratando de encontrar conexiones con la cultura *geek*. El boletín hablaba de una nueva compañía en Bangalore encargada de crear un puente entre Internet y la telefonía móvil, una operación totalmente gestionada desde la India, según el boletín, aunque presumiblemente con financiación estadounidense. Yo quería encontrar una empresa comparable a Amicas: recién creada, dirigida por *geeks*, con un acercamiento similar a Internet, separados por medio mundo y en una «cultura» que presumiblemente ocuparía un orden moral muy diferente. Udhay me invitó a visitarle y prometió presentarme a toda la gente que conocía. Se autodenominaba «trabajador en red fortuito» [«random networker»]: realmente no era ni programador ni diseñador ni un *geek* del software libre. Tampoco un empresario, sino que más bien se describía como el tipo que «traduce entre los trajeados y los *techies*».

Udhay «reúne a gente interesante» y fue precisamente por el placer que hallaba en esta tarea que conocí a tanta gente en India: activistas cosmopolitas, abogados de élite, capitalistas de riesgo, ingenieros y primos, hermanos y hermanas de ingenieros; ejecutivos de publicidad,

azafatas de aviones y consultores de Bombay; periodistas, gastroenterólogos, profesores de informática, músicos y la madre de un robot científico de Bangalore. Entre ellos había musulmanes, hindúes, jainistas, judíos, parsis y cristianos, pero la mayoría se consideraba más seculares y científicos que religiosos. Muchos eran autodidactas o, como sus homólogos estadounidenses, habían dejado la universidad pero seguían aprendiendo sobre ordenadores y redes. Algunos eran estudiantes o trabajaban para el Instituto Indio de Ciencia de Bangalore, una de las instituciones más famosas entre los *geeks* de India (muchos dirían que tan famosa como la Universidad de Stanford para Silicon Valley). Los *geeks* que Udhay me presentó solo tenían dos cosas en común: eran mayoritariamente hombres y les encantaba el *heavy metal*.²⁶

Durante mi estancia en Bangalore Udhay me invitó a unirme a una lista de correo, llamada *Silk-list*, irregular, sin moderación y dedicada a la «conversación inteligente». La lista no estaba centrada en nada particular; con frecuencia estallaban en ella largas divagaciones sobre la política de la India, religión, economía e historia. Los temas oscilaban entre la gastronomía, la ciencia-ficción, el análisis de películas, las discusiones sobre Cachemira, Harry Potter, la singularidad o la nanotecnología. Udhay abrió *Silk-list* en 1997 con Bharath Chari y Ram Sundaram, y sus destinatarios han incluido a centenares de personas de todo el mundo, algunas muy conocidas, programadores, abogados, un ejecutivo publicitario de Bombay, autores de ciencia-ficción, empresarios, un miembro de la compañía emergente Amicas, al menos dos transhumanistas, un esquizofrénico (diagnosticado) y yo mismo. Los participantes activos normalmente rondaban las diez o quince personas, mientras muchos más fisgaban en segundo plano.

26. La cuestión del género plaga el tema de la cultura informática. La configuración de género de *hackers* y *geeks* y la exclusión más general de las mujeres en informática han sido observadas ampliamente por los académicos. Aquí no puedo hacer más que dirigir a los lectores a la literatura cada vez más extensa y sofisticada sobre el asunto. Véase especialmente Light, «When Computers Were Women»; Turkle, *The Second Self y Life on the Screen* [ed. cast.: *La vida en la pantalla*]. Con respecto al software libre, véase Nafus, Krieger, Leach, «Patches Don't Have Gender». Más generalmente, véase Kirkup *et al.*, *The Gendered Cyborg*; Downey, *The Machine in Me*; Faulkner, «Dualisms, Hierarchies and Gender in Engineering»; Grint y Gill, *The Gender-Technology Relation*; Helmreich, *Silicon Second Nature*; Herring, «Gender and Democracy in Computer-Mediated Communication»; Kendall, «Oh No! I'm a NERD!»; Margolis y Fisher, *Unlocking the Clubhouse*; Green y Adam, *Virtual Gender*; P. Hopkins, *Sex/Machine*; Wajcman, *Feminism Confronts Technology* y «Reflections on Gender and Technology Studies»; y Fiona Wilson, «Can't Compute, Won't Compute». Véanse asimismo las novelas y relatos de Ellen Ullman, incluyendo *Close to the Machine* y *The Bug: A Novel*.

Silk list es un índice excelente de la relación entre la red de gente de Bangalore y su conexión con una comunidad mundial en Internet —una increíble historia del poder de redes y medios conectados de forma heterogénea. Udhay me explicó que a principios de los 90 participó por primera vez, para luego aprender él mismo, en la configuración y funcionamiento de un módem basado en el sistema de red conocido como BBS (*Bulletin Board Service*, Servicio de Tablón de Anuncios) en Bangalore. En 1994 oyó hablar de un libro de Howard Rheingold llamado *The Virtual Community*, que se convirtió en su primera introducción a Internet. Un par de años después, cuando por fin tenía acceso a Internet, mandó un correo electrónico a John Perry Barlow, de cuyo trabajo había oído hablar a través de la revista *Wired*, para preguntarle la dirección de correo de Rheingold y poder contactar con él. En cierta manera, Rheingold y Barlow ocupan el centro de una especie de mundo geek; los libros de Rheingold son un amplio registro escrito de los aspectos sociales y comunitarios de las nuevas tecnologías que a menudo han tenido un impacto internacional considerable; Barlow ayudó a fundar la EFF (*Electronic Frontier Foundation*, Fundación de la Frontera Electrónica) y es el responsable de popularizar la frase «La información quiere ser libre».²⁷ Ambos fueron una gran influencia para Udhay y en última instancia le proporcionaron las ideas principales para poner en marcha una comunidad en línea. Una serie de conexiones paralelas de características similares —algunas personales, algunas surgidas de otros medios y canales, otras totalmente aleatorias— completaron la composición de *Silk-List*.²⁸

27. Originalmente acuñada por Steward Brand, la expresión fue ampliamente citada tras su mención en el artículo que Barlow publicó en 1994 con el título «The Economy of Ideas» [ed. cast. «Vender vino sin botellas»].

Estimo pertinente completar esta referencia de Kelty apuntando que la formulación original de Brand posee una coletilla tan habitualmente omitida como significativa en su juego con el doble sentido del adjetivo inglés «free» como «libre» y «gratis»:

La información quiere ser libre. La información también quiere ser cara. La información quiere ser libre porque se ha vuelto muy barata de distribuir, copiar y recombinar —demasiado barata para medirla. Y quiere ser cara porque puede tener un valor incalculable para su receptor [...]. Esta tensión no desaparecerá.

Véase *The Media Lab: Inventing the Future at MIT*. Nueva York, Viking Penguin Press, 1987, p. 202 [ed. cast.: *El Laboratorio de Medios: Inventando el futuro en el M.I.T.*, trad. por Lidia Espínosa de Mattheu y rev. por Jorge A. Andrade Padilla. Madrid, Fundesco, 1989] [N. del E.]

28. Sobre la génesis de las «comunidades virtuales» y el papel de Steward Brand, véase Turner, «Where the Counterculture Met the New Economy».

Como muchas otras comunidades de «*digerati*»²⁹ contemporáneas y posteriores al *boom* de las punto.com, *Silk-list* se constituyó de forma más o menos orgánica alrededor de gente que «lo pillaba», es decir, gente que decía entender Internet y su potencial transformador, y que tenía la capacidad de participar en su expansión. Esta no era la única lista de este tipo. Otras como el boletín de noticias *Tasty Bits*, la lista de correo FoRK (*Friends of Rohit Khare*, Amigos de Rohit Khare), ambas radicadas en Boston, y las listas Nettime y Syndicate (ambas de Holanda), tenían razones de existencia ostensiblemente diferentes, pero muchas tenían los mismos miembros suscriptores y comunidades solapadas de *geeks*. La suscripción estaba abierta a cualquiera y ocasionalmente alguien se topaba con la lista y se unía a ella, pero a la mayoría les invitaban otros miembros, o amigos de amigos, o bien acababan conectados en virtud de mensajes cruzados de otras listas a las que estaban suscritos.

/pub

Silk-list es una lista pública en muchos sentidos de la palabra. En el sentido más práctico, no hace falta invitación para entrar y el material que aparece en ella se archiva públicamente y puede encontrarse fácilmente en Internet. Udhay se esfuerza por animar a todo el mundo a hablar y participar, a la vez que evita las formas de discurso que según él convertirían a los participantes en mirones. *Silk-list* no está asociada a ningún gobierno, corporación u organización no gubernamental, sino que está constituida por la actividad de los *geeks* al buscar y hablar con otros *geeks* de la lista (lo cual puede suceder de muchas maneras: a través del trabajo, la escuela, los congresos, la fama, la asociación aleatoria, etc.). Si recordamos la distinción de Charles Taylor entre espacio tópico y espacio metatópico, veremos que *Silk-list* no es un espacio tópico convencional: en ningún momento se da un encuentro cara a cara de todos sus miembros (aunque hay encuentros regulares en ciudades de todo el mundo), y tampoco están todos en línea al mismo tiempo (aunque el volumen y el ritmo de los mensajes a menudo reflejan quiénes están «hablando» en línea en un momento determinado). Con todo, se trata de un espacio tópico si lo consideramos desde la perspectiva

29. Término surgido de la combinación de las palabras «*digital*» y «*literati*» («letrado», «versado») que originalmente se refería a cierta «élite» de las primeras comunidades virtuales y actualmente se usa como equivalente de expertos o gurús de Internet. [N. del E.]

de la máquina: todos los nombres de los suscriptores se reúnen en una base de datos, en un archivo o en el servidor que administra la lista de correo. Sería una exageración llamarlo «asamblea», sin embargo, porque reúne solo los avatares de los lectores de la lista de correo, muchos de los cuales probablemente ignoren o borren la mayoría de mensajes.

Por otro lado, *Silk-list* es sin duda un público «metatópico»: la lista «entreteje» una variedad de espacios tópicos: mi debate con unos amigos de Houston y los de otros miembros con gente de todo el mundo, así como las fuentes de múltiples debates como artículos de prensa y revistas, películas, eventos y demás materiales de los que se informa y discute en Internet. Pero *Silk-list* no es «El» público —está lejos de ser el único foro donde se entreteje la esfera pública. Existen muchas, muchas listas como esta.

En *Publics and Counterpublics* Michael Warner ofrece una distinción más. «El» público es un imaginario social operativo en los términos formulados por Taylor: como una especie de visión de orden evidenciada a través de historias, imágenes, narrativas y demás, las cuales constituyen la imaginación de lo que significa formar parte del público, así como los planes necesarios para crear el público, si es necesario. Sin embargo, Warner distingue entre una audiencia concreta, encarnada, como en una obra de teatro, una manifestación o un disturbio (un público tópico, según Taylor) y una audiencia creada mediante el discurso y su difusión, una audiencia que no es metatópica en tanto que constituye un público que es concreto de una forma distinta: no en la temporalidad cara a cara del acto de habla, sino en el sentido de dar vida a un público a través de una interpellación que posee una temporalidad diferente. Se trata de un público que es concreto en su relación específica con unos medios: depende de las estructuras de creación, difusión, uso, ejecución y reutilización de formas particulares de discurso, de objetos o de instancias particulares de discurso.

La distinción de Warner tiene varias implicaciones. En primer lugar, tal y como Warner se cuida de señalar, la existencia de un medio en particular no es suficiente para que un público exista. Que se imprima un libro no quiere decir que exista un público, pues también hace falta que el público actúe de forma correspondiente, es decir, que lo lean. Ser parte de un público en particular es decidir prestar atención a aquellos que deciden dirigirse a aquellos que deciden prestar atención... y así sucesivamente. O como explica Warner:

La circularidad es esencial en este fenómeno. Puede que un público sea real y eficaz, pero su realidad radica precisamente en esta reflexividad por la cual se hace aparecer un objeto de interpelación con el fin de posibilitar el mismo discurso que le dota de existencia.³⁰

Esta característica «autotélica» del público es crucial para entender su *función* como una entidad situada al margen del poder. Si un público busca la legitimidad derivada de funcionar de forma independiente, de ningún modo puede ser organizado por el Estado, por una empresa o por cualquier otra totalidad social. Como lo expresa Warner:

Un público se organiza independientemente de las instituciones estatales, la ley, los marcos formales de ciudadanía o las instituciones preexistentes como la Iglesia. Si no fuera posible pensar en el público como algo organizado independientemente del Estado u otras estructuras, el público no podría ser soberano con respecto al Estado. [...] Hablar, escribir y pensar nos agrupa —activa e inmediatamente— en un público, y de ahí la experiencia de la soberanía.³¹

La descripción de Warner no afirma que cualquier público o incluso El Público realmente adopte esta forma en el presente: es una descripción de un imaginario social o una «fe» que permite a los individuos dotar de sentido a sus acciones de acuerdo con una idea moderna de orden social. Tal y como Warner (y Habermas antes que él) sugiere, la existencia de tales públicos autónomos —y, desde luego, de la idea de «opinión pública»— no siempre se ajusta a esta idea de orden. A menudo resulta que tales públicos han estado controlados desde el principio por los Estados, las corporaciones, el capitalismo y otras formas de totalidad social que determinan la naturaleza del discurso de formas insidiosas. Un público cuyos participantes no tienen fe en su carácter autotélico y autónomo es poco más que una charada destinada a aplacar la oposición a la autoridad, a transformar el poder político y la igualdad en una negociación entre partes desiguales.

¿Es *Silk-list* un público? O lo que es más importante, ¿es un público soberano? La distinción que hace Warner entre las distintas

30. Warner, «Publics and Counterpublics», p. 51 [ed. cast: *Públicos y contrapúblicos*].

31. Ibid., pp. 51-52. Véase también Warner, *Publics and Counterpublics*, p. 69 [ed. cast: *Públicos y contrapúblicos*].

formas de asamblea vinculadas a medios específicos es crucial para responder a esta pregunta. Si uno desea saber si una lista de correo electrónico es más o menos susceptible de ser un público soberano que un público de lectores de libros o de oyentes del noticiero vespertino, es necesario aproximarse a ella desde la especificidad de la forma de discurso. Esta especificidad no solo incluye si se trata de texto, video o audio, o si el texto es ASCII o Unicode, o si el video es PAL o NTSC, sino que también incluye los medios de creación, difusión y reutilización de ese discurso.

Consideremos, por ejemplo, las diferencias entre un libro publicado de forma convencional, por una editorial comercial convencional y distribuido en librerías y a través de Amazon.com, y un libro publicado por una compañía emergente de Internet que pone a libre disposición una copia electrónica con licencia *copyleft*, pero cobra (un precio más bajo) por las copias en papel por encargo. Ambos libros podrían entrar fácilmente en el espacio metatópico del Público: podrían ser debatidos en casas, escuelas o listas de distribución, podrían cosechar críticas favorables o demoledoras, y quizá incluso provocar efectos en las conductas empresariales, el Estado o las políticas públicas. Ahora bien, el primero está sometido a elevadas restricciones con respecto a su autoría, su forma de distribución, comercialización, edición, revisión, etc. La legislación de *copyright* restringirá lo que los lectores pueden hacer con él, incluyendo el modo en que podrían leerlo, difundirlo o hacer obras derivadas del mismo. Sin embargo, un libro publicado tradicionalmente también se ve enriquecido por su asociación con una empresa reputada: de inmediato se lo trata como una obra acreditada, que quizá alcanza ciertos estándares de exactitud, precisión o incluso verdad, y su calidad se mide primordialmente en función de las ventas.

En contraste, el libro por encargo y mediado por Internet tendrá una temporalidad de difusión muy diferente: puede languidecer en la oscuridad por falta de *marketing* o de una autoridad reputada, o puede ser mencionado en algún sitio como el *New York Times* y convertirse repentinamente en un éxito. Para tal libro, la legislación de *copyright* (en forma de licencia *copyleft*) puede permitir un rango mucho más amplio de usos y reutilizaciones, pero restringirá ciertas formas de comercialización del texto. Por tanto, los dos públicos pueden terminar pareciendo muy diferentes, sin duda con ciertos solapamientos pero variando en sus formas de control y condiciones de acceso. Lo que

está en juego es el poder de uno u otro de estos públicos para aparecer como una entidad independiente y soberana —libre de control y restricciones sospechosas— cuya función es discutir con otras formas constituidas de poder.

El libro publicado de manera convencional puede satisfacer todos los criterios para constituir un público, al menos en el sentido coloquial de hacer accesible un conjunto de ideas y un discurso, esperando influir o recibir una respuesta por parte de formas constituidas de poder soberano. Sin embargo, solo el último esquema de publicación por encargo satisface el criterio de ser un público *recursivo*. Las diferencias en este ejemplo ofrecen una cruda indicación de por qué Internet es tan importante para los *geeks*, tan importante como para unirse en su defensa, como una infraestructura que permite la creación de públicos concebidos como autónomos, independientes y autotélicos. Los *geeks* comparten la idea de un orden moral y técnico en lo que se refiere a Internet; y no solo eso, comparten la responsabilidad de mantener ese orden por ser aquello que de entrada les permite asociarse como un público recursivo. A través de su asociación descubren, o redescubren, el poder y la posibilidad de ocupar la posición de público independiente —uno no controlado por los Estados, las corporaciones u otras organizaciones, sino abierto de cabo a rabo (según reivindican ellos)— y desarrollar un deseo de defenderlo de la intrusión, la destrucción o la refeudalización (por usar el término de Habermas para definir la fragmentación de la esfera pública).

El público recursivo por tanto no es solo el libro y el discurso alrededor del libro. No es ni siquiera el «contenido» expandido para incluir todo tipo de medios de comunicación. También es la estructura técnica de Internet: su software, sus protocolos y estándares, sus aplicaciones y software, su estatus legal y las licencias y regulaciones que lo gobiernan. Esto permite captar las dos razones por las que los públicos recursivos son algo distintivo: 1) No solo incluyen los discursos de un público, sino también la habilidad de crear, mantener y manipular las infraestructuras de esos discursos; y 2) están estratificados por «capas» e incluyen tanto discursos como infraestructuras, hasta un cierto punto técnico (no de forma ilimitada). La idea sobre qué capas son importantes se extrae más o menos inmediatamente de la implicación directa con el medio. En el siguiente caso, por ejemplo, Napster representa el potencial de Internet en miniatura —en cuanto aplicación—, pero también

conecta inmediatamente con preocupaciones acerca de los protocolos básicos que rigen Internet y el proceso de estandarización que gobierna el desarrollo de estos protocolos: de ahí la recursión a través de las capas de una infraestructura.

Estos dos aspectos del público recursivo también se relacionan con la preocupación sobre la fragmentación o refeudalización de la esfera pública: *existe una única Internet*. Su singularidad no está técnicamente determinada ni es de ninguna manera necesaria, pero es lo que convierte Internet en algo tan valioso para los *geeks*. Es una lucha, cuyo objetivo consiste en mantener Internet como una infraestructura para el surgimiento de públicos autónomos y autotélicos como parte de El Público, entendido como parte de un imaginario de orden moral y tecnológico: sistemas operativos y sistemas sociales.

De Napster a Internet

El 27 de Julio de 2000 Eugen Leitl envió un mensaje a *Silk-list* bajo el asunto «Preludio a la Singularidad». El autor original del mensaje, Jeff Bone (que aún no era miembro de *Silk-list*), había enviado inicialmente la «columna de opinión» a la lista de correo de FoRK en respuesta a las acciones de la RIAA (*Recording Industry Association of America*, Asociación de la Industria Discográfica de EEUU) contra Napster. La RIAA acababa de conseguir que la jueza de distrito Marylin Hall Patel, del Noveno Tribunal del Circuito de Apelaciones, emitiera un mandato judicial para que Napster detuviera las descargas de música sujeta a *copyright*. El artículo de Bone sostenía que:

El folclor popular mantiene que Internet fue diseñada con protocolos de enrutamiento descentralizados para resistir un ataque nuclear. Es decir, Internet «percibe el daño» y «lo sortea». Se ha dicho que, en la Red, la censura se percibe como un daño y es consecuentemente sorteada. La RIAA, en cierto sentido, ha adoptado el papel de censor. Por consiguiente, la industria de la música será percibida como un daño, y será sorteada. No hay duda de que esto pasará y la tecnología evolucionará más rápido de lo que son capaces las empresas y las instituciones sociales; hay numerosos proyectos ya en desarrollo que intentan crear tecnología invulnerable a desafíos legales de distinto tipo. Julian Morrison, el creador de un proyecto (llamado Fling) para construir un conjunto de protocolos de red

completamente anónimos/irrastreables, expresa esto de forma particularmente elocuente.³²

El mensaje de Bone está repleto de detalles que ilustran el significado y el valor de Internet para los *geeks*, y que ayudan a clarificar el concepto de público recursivo. Aunque se trata tan solo de un mensaje, condensa y expresa una gran variedad de historias, imágenes, folclore y detalles técnicos que elaboraré a continuación.

El cierre de Napster en 2000 tristeció a aficionados de la música y a *geeks* por igual, y tampoco ayudó realmente a las compañías discográficas que lo perpetraron. Para muchos *geeks*, Napster representaba Internet en miniatura, una innovación que manifestaba algo a un alcance y escala nunca vistos antes, y que también conectaba a gente alrededor de algo que les importaba profundamente —su interés compartido por la música. Napster suscitó interesantes preguntas acerca de su propio éxito: ¿Se debía a que permitía a las personas *desarrollar nuevos intereses musicales* a un alcance y escala nunca experimentados con anterioridad? ¿O quizás a que proporcionaba a la gente que ya tenía intereses musicales una forma de *compartir música* a un alcance y escala que ellos nunca antes habían experimentado? Es decir, ¿fue una innovación de *marketing* o de distribución? La industria musical lo vivió como esto último y, por tanto, como una competencia directa con sus propios medios de distribución. Muchos aficionados a la música lo vivieron como lo primero, lo que Cory Doctorow bien calificó como «picoteo seguro», es decir, la habilidad de probar una casi inimaginable diversidad de música antes de elegir en qué invertir intereses (y dinero). En gran medida, Napster era, por tanto, una recapitulación de lo que Internet ya significaba para los *geeks*.

El mensaje de Bone, el cierre de Napster y las variadas reacciones al mismo ilustran muy bien los dos aspectos claves del público recursivo: primero, la forma en la que los *geeks* discuten no solo sobre derechos e ideas (por ejemplo, ¿es legal compartir música?) sino también sobre las infraestructuras que permiten tal discusión y compartición; segun-

32. El resto del mensaje puede encontrarse en los archivos de *Silk-list* en <http://groups.yahoo.com/group/silk-list/message/2869> (última consulta: 18 de abril de 2018). La referencia a «Fling» alude a un proyecto ahora disponible en <http://fling.sourceforge.net/> (enlace no accesible en la última consulta de la misma fecha). Los archivos completos de *Silk-list* pueden encontrarse en <http://groups.yahoo.com/group/silk-list/> y los archivos completos de la lista FoRK pueden encontrarse en <http://www.xent.com/mailman/listinfo/fork/>.

do, las «capas» de un público recursivo quedan en evidencia ante la inmediata conexión de Napster (una aplicación conocida por millones de personas) con los «protocolos de enrutamiento descentralizados» (TCP/IP, DNS, y otros) que posibilitaban que Napster funcionase como lo hacía.

El mensaje de Bone contiene cuatro puntos interrelacionados. El primero está relacionado con el concepto de progreso técnico autónomo. El título «Preludio a la Singularidad» se refiere al artículo de 1993 de Vernor Vinge sobre la noción de «singularidad», un punto en el tiempo en que la velocidad del desarrollo tecnológico autónomo sobrepasa la capacidad humana de controlarlo.³³ La noción de singularidad tiene el estatus de un tipo de «ley» coloquial similar a la Ley de Moore o a la Ley de Metcalfe, además de enlazar con un tipo de literatura más general cuyas raíces provienen de ideas de orden social liberal-libertarias o clásicamente liberales que van desde John Locke y John Stuart Mill a Ayn Rand y David Brin.³⁴

La afinidad de Bone con las historias transhumanistas de la teoría evolucionista, la teoría económica y la rápida innovación sienta las bases para el resto de su mensaje. Aquí la táctica retórica crucial es la aparición de lo inevitable (como en el enfático «no hay duda de que esto ocurrirá»): Bone parte de la base de que se dirige a una audiencia acostumbrada a oír hablar de lo inevitable del progreso técnico y la imposibilidad de maniobrar legalmente para cambiarlo, pero su audiencia no necesariamente está de acuerdo con esas suposiciones. Los *geeks* ocupan un espectro que va desde los «polímatas» a los «transhumanistas», espectro que incluye su entendimiento del progreso tecnológico y su relación con la intervención humana. El mensaje de Bone aterriza claramente en el lado transhumanista.

Un segundo punto tiene que ver con la censura y la situación del poder: según Bone, el poder no reside principalmente en el Gobierno o la Iglesia, sino que viene del sector privado, en este caso de la coalición de corporaciones representada por la RIAA. La importancia de esto tiene

33. Vinge, «The Coming Technological Singularity».

34. La Ley de Moore —nombrada así por Gordon Moore, antiguo presidente de Intel— establece que la velocidad y capacidad de las CPU (*central processing units*, unidades centrales de procesamiento) de los ordenadores se duplica cada dieciocho meses, lo que se ha verificado desde aproximadamente 1970. La ley de Metcalfe —nombrada así por Robert Metcalfe, inventor de Ethernet— establece que la utilidad de una red equivale al cuadrado del número de sus usuarios, lo cual sugiere que el número de cosas que podemos hacer con una red se incrementa exponencialmente a medida que se agregan miembros de forma lineal.

que ver con el hecho de que se espera que un «público» sea su propia entidad soberana, ajena a la Iglesia, el Estado o las corporaciones. Y si bien la censura de la Iglesia o el Estado es una forma familiar de agresión contra el público, la censura por parte de las corporaciones (o de los consorcios que las representan) supone un desarrollo novedoso, y como tal llama la atención de Bone y de otras personas. Que el bloqueo a la compartición de archivos pueda llamarse legítimamente censura resulta también controvertido y muchos de quienes respondieron al mensaje en la *Silk-list* encontraron insostenible tal acusación de censura.

Demostrando la afirmación de Bone, a lo largo de los siguientes años y procesos judiciales, la RIAA y la MPAA (*Motion Picture Association of America*, Asociación Cinematográfica de EEUU) han recibido una autoridad de control considerablemente superior a la de muchas agencias federales —especialmente con respecto al control de las redes por su propia cuenta (un problema que, dado su carácter abstruso, raramente ha sido mencionado en los principales medios masivos). Ambas organizaciones no solo han intentado procesar a quienes comparten archivos, sino que han obtenido el derecho de requerir a los proveedores de servicios de Internet información sobre las actividades de sus clientes y han reclamado de forma constante el derecho a desactivar secretamente (manipular, inutilizar o destruir) ordenadores privados sospechosos de actividades ilegales. Aunque tales prácticas no puedan definirse *per se* como censura, suponen excelentes ejemplos de los problemas que más inquietan a los *geeks*: el uso de los medios legales por unos pocos (en este caso, corporaciones privadas) para eliminar o transformar tecnologías de uso generalizado por la mayoría. También señalan los problemas de monopolio, defensa de la competencia y control técnico que no son evidentes y a menudo se expresan, por ejemplo, en alegorías de reforma y control de los legos que comparten música por parte de las autoridades papales.

En tercer lugar, el mensaje de Bone puede entenderse en sí mismo en el marco de la reorientación del saber y el poder. Pese a que el sentido de llamar a su texto «columna de opinión» pueda parecer obvio, el mensaje de Bone no se publicó en ningún lugar en ningún sentido convencional. Tampoco parece haber sido ampliamente citado y enlazado. Sin embargo, al menos por un día, fue objeto de una acalorada discusión en tres listas de correo, incluyendo la *Silk-list*. La «publicación» en este ejemplo supone un tipo de evento diferente de conseguir una tribuna de opinión en el *New York Times*.

El material en la *Silk-list* se sitúa en un punto intermedio entre la conversación privada (en un lugar público, quizás) y la opinión publicada; ningún editor tomó una decisión para «publicar» el mensaje —Bone se limitó a darle a «Enviar». Y sin embargo, como con cualquier publicación impresa, en teoría su texto era accesible a cualquiera, y más aún, se podía archivar un número potencialmente enorme de copias en muchos lugares distintos (los ordenadores de todos los participantes, el servidor que alberga la lista, los grupos de servidores de Yahoo! que la archivan, las bases de datos de Google, etc.). El mensaje de Bone ejemplifica la naturaleza recursiva del público recursivo: es una declaración pública sobre la apertura de Internet y es un *ejemplo* de las nuevas formas de público que esta posibilita mediante su apertura.

Las restricciones a quienes hablan en la esfera pública (como el poder de los impresores y editores, los requisitos de la legislación o las cuestiones de coste y accesibilidad) son mucho más laxos en la era de Internet que en cualquier otra anterior. Internet otorga a un anteriormente desconocido Jeff Bone el poder para hilvanar un manifiesto sin siquiera pensarlo dos veces. Por otro lado, la facilidad de distribución contrasta con la dificultad de ser escuchado: la multitud de otros Jeff Bones hace mucho más difícil conseguir una audiencia. Hablando de públicos, el mensaje de Bone puede constituir un público en el mismo sentido que un artículo de opinión del *New York Times*, pero su impacto y significado serán diferentes. Su mensaje es abierto y libremente disponible durante tanto tiempo como haya *geeks* y leyes y máquinas que lo mantengan, pero la columna del *New York Times* tendrá más autoridad, será menos accesible, y, lo que es más importante, no estará disponible para cualquiera. Los *geeks* imaginan un espacio donde cualquiera pueda hablar con similar alcance y poder de permanencia —aunque ello no implique automáticamente autoridad—, e imaginan que debería mantenerse abierto a toda costa. Por eso Bone está interesado precisamente en una infraestructura técnica que asegure su derecho a hablar sobre dicha infraestructura y a ofrecer críticas y orientaciones al respecto.

No obstante, la capacidad para crear y mantener tal público recursivo plantea la cuarta y más sustancial cuestión que el mensaje de Bone deja clara. El salto a hablar sobre los «protocolos de enrutamiento descentralizado» representa claramente el orden moral y técnico compartido por los *geeks*, derivado en este caso de los detalles específicos de Internet. El mensaje de Bone comienza con una serie de declaraciones

que son parte del repertorio de historias técnicas e imágenes entre los *geeks*. Bone comienza haciendo referencia al «folclore» de Internet, en el que comúnmente se piensa que los protocolos de enrutamiento han sido creados para soportar un ataque nuclear. Llamándolo folclore sugiere que no es una descripción precisa de Internet, sino una imagen que captura sus objetivos de diseño. Bone compendia esta idea en un fragmento más reciente del folclore: «Internet trata la censura como un daño y la sortea».³⁵ Ambas partes del folclore son ampliamente citadas y difundidas por condensar una de las ideas intelectuales centrales sobre la arquitectura de Internet, a saber, su interconectividad abierta y distribuida. Ciertamente hay un trasfondo técnico específico para esta cuestión: los protocolos de interconexión de redes TCP/IP fueron diseñados para conectar múltiples redes sin hacerlas sacrificar su autonomía y control. Sin embargo, Bone usa este argumento técnico más a la manera de un imaginario social que de una teoría, es decir, como un modo de reflexionar sobre el orden técnico (y moral) de Internet, de cómo se supone que es Internet.

A comienzos de los 90, esta versión del orden técnico de Internet era parte de un vibrante dogma liberal-libertario que aseveraba que Internet simplemente no podía ser gobernada por ningún soberano territorial y que era fundamentalmente un lugar de libertad. Este era el mensaje central de gente como John Perry Barlow, John Gilmore, Howard Rheingold, Esther Dyson y otros muchos que poblaron tanto la Internet anterior a 1993 (es decir, antes de que la World Wide Web estuviera ampliamente disponible) como las páginas de revistas como *Wired* y *Mondo 2000* —por cierto, el mismo grupo de personas cuyas ideas fueron visibles y significativas para Udhay Shankar y sus amigos de India incluso antes del acceso a Internet allí, sin mencionar a Sean y Adrian en Boston, así como a artistas y activistas en Europa, todos

35. Esta cita de la década de los 90 se atribuye al fundador de la EFF y «ciberliberal-libertario» John Gilmore. No está claro si la creencia generalizada expresada en esta frase contiene o no algo de verdad. Por un lado, el protocolo al que se refiere el folclore —el sistema general de «comunicación de mensajes» y más tarde de «comunicación de paquetes» inventado por Paul Baran, de la RAND Corporation— sí que parece prestarse a la robustez (sobre esta historia, véase Abbate, *Inventing the Internet*). Sin embargo, no está claro que las amenazas nucleares fueran la única razón de que tal robustez fuera un objetivo del diseño, sino simplemente que la garantía de la comunicación en una red distribuida era necesaria en sí misma. Con todo, la historia circula profusamente como un mito de la naturaleza y estructura de Internet. Paul Edwards sugiere que ambas historias son verdaderas (*«Infrastructure and Modernity»*, pp. 216-220, 225n13).

los cuales solían reaccionar con más virulencia contra esta estética liberal-libertaria.

Para Jeff Bone (y un gran número de *geeks*), la noción folclórica de que «la red trata la censura como un daño» es muy poderosa: sugiere que la censura no es posible porque no hay un punto central de control. Un sentimiento relacionado y frecuentemente citado es que «intentar quitar algo de Internet es como intentar sacar pis de una piscina». Los *geeks* perciben esto como una virtud de Internet, no como un inconveniente.

Al otro lado del espectro, sin embargo, esta visión de la naturaleza irregulable de Internet ha sido sometida a rotundas críticas, destacando la de Lawrence Lessig, quien por lo demás suele concordar con la cultura *geek*. Lessig sugiere que solo porque Internet tiene una estructura particular no significa que esta deba ser siempre así.³⁶ Su argumento tiene dos vertientes: primera, que Internet está estructurada del modo en que lo está porque se compone de código que la gente escribe y que, por tanto, podría haber sido y será distinto, dado que constantemente se dan cambios e innovaciones; segunda, que en consecuencia la estructura particular de Internet gobierna o regula la conducta de formas específicas: el Código es Ley. Así que aunque pueda ser cierto que nadie puede volver «cerrada» Internet aprobando una ley, también lo es que Internet podría volverse cerrada si se quisiera alterar la tecnología con tal fin, un proceso que bien puede ser impulsado y guiado por leyes, reglamentos y normas.

La crítica de Lessig está en realidad en el núcleo de la inquietud de Bone, y más generalmente de los públicos recursivos: Internet entraña una *lucha* y una que necesita ser retomada repetida y constantemente con el fin de mantenerla como la infraestructura legítima a través de la cual los *geeks* se asocian entre sí. Los *geeks* discuten en detalle sobre qué distingue los factores técnicos de los legales o sociales. La apertura de Internet está complejamente entrelazada con cuestiones de disponibilidad, precio, restricción legal, usabilidad, elegancia de diseño, censura, secreto comercial, etc.

Sin embargo, incluso donde la apertura está presente como una tendencia natural para la tecnología (en las frecuentes analogías con la aptitud reproductiva y la biodiversidad, por ejemplo), se trata solo

36. Lessig, *Code and Other Laws of Cyberspace* [ed. cast.: *El código y otras leyes del ciberespacio*]. Véase Gillespie, «Engineering a Principle», sobre la historia relacionada del principio de diseño «extremo a extremo».

de una afirmación parcial, pues representa solo una de las «capas» de un público recursivo. Por ejemplo, cuando Bone sugiere que la red es «invulnerable a desafíos legales» porque «la tecnología evolucionará más rápido de lo que son capaces las empresas y las instituciones sociales», no solo se está refiriendo al hecho de que la novedosa configuración técnica de Internet tiene pocos puntos centrales de control, lo que dificulta que una sola institución la controle, sino que también habla de las redes distribuidas y difusamente conectadas de personas que tienen derecho a escribir y reescribir software y lidian regularmente con los protocolos subyacentes de Internet —en otras palabras, de los propios *geeks*.

Sistemas operativos y sistemas sociales: el imaginario de orden que comparten los *geeks* es tanto moral como técnico. No se trata solo de la estructura técnica de Internet, por más innovadora que esta sea, sino también de la estructura legal y social que ha surgido con ella, el tipo de orden que ha posibilitado que los *geeks* se asocien en un público planetario y se vuelvan conscientes del valor del espacio que han creado.

Muchos *geeks*, quizás incluyendo a Bone, descubren la naturaleza de este orden al llegar a entender cómo funciona Internet —cómo funciona técnicamente, pero también quiénes la crearon y cómo. Algunos han llegado a tal entendimiento mediante su participación en el software libre (un «público recursivo» ejemplar), otros a través de relatos, tecnologías, proyectos e historias que iluminan el proceso de creación, crecimiento y evolución de Internet. La historia del proceso de estandarización de Internet es quizás la más célebre: se trata de la historia del IETF (*Internet Engineering Task Force*, Grupo Especial de Ingeniería de Internet) y su sistema RFC (*Requests for Comments*, Petición de Comentarios).

Requests for Comments

Para muchos *geeks*, el IETF y su sistema RFC ejemplifican las características clave del orden moral y técnico que comparten, los «relatos y prácticas» que crean un imaginario social, según Charles Taylor. El IETF es una veterana asociación de ingenieros de Internet que trata de ayudar a diseminar algunos de los estándares centrales de la Red a través del proceso de RFC. La asociación está abierta a la incorporación de cualquier individuo y tiene muy poco control real sobre la estructura o el crecimiento de Internet —solo sobre el proceso clave de la

estandarización de Internet. Sus estándares no suelen gozar del tipo de legitimidad política que uno asocia con los tratados internacionales y con los organismos de normalización de Ginebra, y no obstante están legitimados *de facto*. El proceso de RFC es una estandarización inusual que permite hacer modificaciones en las tecnologías existentes antes de que el estándar esté finalizado. Los estándares de Internet y el proceso de RFC forman el trasfondo del debate sobre Napster y las afirmaciones de Jeff Bone sobre los «protocolos de enrutamiento de Internet».

Un famoso fragmento del folclore sobre el gobierno de Internet (atribuido a David Clark, miembro del IETF) expresa de forma sucinta la combinación de orden moral y técnico que comparten los *geeks*: «Rechazamos reyes, presidentes y votaciones. Creemos en el consenso general y el código que funciona»³⁷. Esta cita enfatiza la necesidad de discutir con y por medio de la tecnología, el primer aspecto de un público recursivo: el único argumento convincente es el código que funciona. Si funciona, se puede implementar; si se implementa, sorteará el daño legal hecho por la RIAA. La noción de «código que funciona» es clave para entender la relación que los *geeks* establecen entre las discusiones mediante tecnología y las discusiones de palabra. Muy a menudo, la respuesta de los *geeks* a la gente que discutía sobre Napster ese verano —y a las sentencias judiciales al respecto— era desechar sus quejas como si fueran mera palabrería. Muchos sugerían que si se cerrase Napster, miles de programas similares surgirían tras su estela. Como dice Ashish «Hash» Gulhati, un participante de una lista de correo electrónico:

Son precisamente todas estas decisiones judiciales no ejecutables y absurdas las que comenzarán a parecer mero onanismo satisfecho de sí mismo cuando exista código disponible que haga que las leyes valgan menos que el papel en que están escritas. Cuando se trata de luchar contra esta mierda de una forma efectiva, cualquier cosa que no sea código no es más que mera palabrería.³⁸

37. Esta frase se repite constantemente en Internet y es atribuida a David Clark, pero nadie sabe realmente dónde o cuándo la pronunció. Aparece en una entrevista de 1997 que Jonathan Zittrain realizó a David Clark y cuya transcripción está disponible en <http://cyber.law.harvard.edu/jzfallsem//trans/clark/> (última consulta: 18 de abril de 2018).

38. Ashish «Hash» Gulhati, correo electrónico a la lista de distribución *Silk-list*, 9 de septiembre de 2000, <http://groups.yahoo.com/group/silk-list/message/3125>.

Tal retórica rotunda a menudo quiebra el propio proceso, ya que alguien tiene que escribir el código. Ello puede dar lugar a algo paradigmático: hay una necesidad de hablar enérgicamente sobre la necesidad de menos palabrería y más código, como demostró Eugen Leitl cuando yo objeté que los usuarios de *Silk-list* «tan solo estaban hablando».

Por supuesto que deberíamos hablar. ¿Acaso mi última entrada consistió en un código Python cojonudo que añadía una funcionalidad anhelada a Mojonation? No. Tan solo más divagaciones de nivel superior sobre la importancia de divagar menos y codificar más. Mi cabeza carece del equipo mental adecuado para ser un buen programador, de ahí que intente corromper a jóvenes inocentes impresionables para que contribuyan a la causa. Lo digo sin ninguna vergüenza. Demándame siquieres.³⁹

La frivolidad de Eugen revela un reconocimiento de que existe un componente político en la codificación, por más que, a fin de cuentas, las palabras desaparezcan y solo quede el código. Puede que a Eugen y otros *geeks* les guste adoptar una retórica que sugiera que «simplemente ocurrirá», pero en la práctica ninguno de ellos actúa realmente así. Mejor dicho, las actividades de codificación, escritura de software o mejora y diversificación del que ya existe no son inevitables o automáticas sino que tienen características específicas. Requieren tiempo y «el equipo mental adecuado». La inevitabilidad a la que se refieren no consiste en alguna fantasía sobre la inteligencia de las máquinas, sino en un imaginario social compartido en redes difusamente conectadas por mucha gente que emplea todo su tiempo libre en construir, descargar, hackear, probar, instalar, parchear, codificar, bloguear y hacer proselitismo; en resumen, en crear un público recursivo habilitado por Internet.

La columna de opinión de Jeff Bone, típicamente entusiasta acerca de la inevitabilidad de las nuevas tecnologías, se toma su tiempo para referirse a uno de los miles (quizá decenas de miles) de proyectos dignos de atención y apoyo, un proyecto llamado Fling, que intenta reescribir los protocolos básicos de Internet.⁴⁰ El objetivo del proyecto es escribir

39. Eugen Leitl, correo electrónico a la lista de distribución *Silk-list*, 9 de septiembre de 2000, <http://groups.yahoo.com/group/silk-list/message/3127>. Python es un lenguaje de programación. Mojonation era una aplicación P2P muy prometedora en 2000 que desde entonces ha dejado de existir.

40. En particular, este proyecto se centra en el TCP (*Transmission Control Protocol*, Protocolo de Control de Transmisión), el UDP (*User Datagram Protocol*, Protocolo de Datagramas

un software de implementación de estos protocolos con la finalidad explícita de hacerlos «anónimos, irrastreables e inexplotables». Fling no es una empresa, una compañía emergente o un proyecto de investigación universitario (aunque algunos proyectos similares sí lo son); es solo una página web. Los protocolos básicos de Internet, incluidos en los RFC, son poco más que documentos que describen cómo los ordenadores deberían interactuar entre sí. Son estándares, pero de un tipo inusual⁴¹. Lo que no es inusual es el salto de Bone de una discusión sobre Napster a una sobre los protocolos básicos de Internet. Y es que dicho salto representa el segundo aspecto de un público recursivo: la importancia de entender Internet como un juego de «capas», cada una de las cuales posibilita la siguiente y requiere una apertura que impide el control central al tiempo que propicia la máxima creatividad.

A medida que el IETF y la ISOC (*Internet Society*) se han convertido en entidades más burocráticas, los RFC han pasado de ser un sistema informal de memorandos a un proceso formal de estandarización sobre la vida de Internet. El proceso de redacción y mantenimiento de estos documentos es específico de Internet, precisamente porque Internet es el tipo de experimento en red que facilita la compartición de recursos a través de redes administrativamente delimitadas. Es un proceso que ha permitido a todos los experimentadores compartir la red y proponer cambios a la misma, todo en un espacio común. Los RFC son principalmente sugerencias, no demandas. Son documentos de «dominio público» y por tanto disponibles para cualquier persona con acceso a Internet. Tal y como muestra la referencia de David Clark sobre «el consenso general y el código que funciona», el componente esencial de la estandarización de Internet es una *implementación* de los protocolos correcta y operativa. Alguien ha de escribir los programas que funcionen según las especificaciones del RFC, pues, después de todo, este es solo un documento, no un programa informático. Las diferentes implementaciones de, por ejemplo, el protocolo TCP/IP o el ftp (*File Transfer Protocol*, Protocolo de Transferencia de Archivos) dependen inicialmente de los individuos, grupos y/o empresas que los incorporan al núcleo de un sistema operativo o a una aplicación de

de Usuario) y el DNS (*Domain Name System*, Sistema de Nombres de Dominio). Los dos primeros han permanecido en gran medida estables a lo largo de los cuarenta últimos años, pero el sistema DNS ha estado sujeto a una amplia politización (véase Mueller, *Ruling the Root*).

41. Sobre los estándares de Internet, véase Schmidt y Werle, *Coordinating Technology; Abbate y Kahin, Standards Policy for Information Infrastructure*.

usuario, y ulteriormente de la existencia de un gran número de usuarios de ese sistema operativo o aplicación.

En muchos casos, tras la diseminación y adopción de una implementación, se han modificado los RFC para reflejar esas implementaciones funcionales y establecerlas como estándares. Así pues, los estándares actuales son realmente de arranque autosostenido («*bootstrapped*»),⁴² a través de un proceso de redacción de RFC seguido por un proceso de creación de implementaciones que se adhieran difusamente a las reglas del RFC, implementaciones cuyo progreso es luego observado para acabar reescribiendo el RFC y retomar de nuevo todo el proceso. El hecho de que los *geeks* puedan discutir por correo electrónico depende de la existencia de un RFC que define el protocolo de correo electrónico y de implementaciones de software para enviar dichos correos.

42. Tras no pocas dudas y más consultas, opto por traducir a lo largo de estas páginas el neologismo «*bootstrap*» o «*bootstrapping*» como «arranque autosostenido». Pese al riesgo de no satisfacer ni a «*geeks*» ni a «*legos*», creo que la introducción del adjetivo «autosostenido» (en lugar de otras variantes como «autoarranque», «arranque secuenciado» o «secuencia de arranque», por no hablar de la tentación de dejarlo en inglés) puede contribuir a *ensamblar* la acepción primigenia del término con sus sucesivas traslaciones al ámbito técnico, y más específicamente informático (que recogo sucintamente a continuación), e incluso con la idea de «autofundamentación» (concretamente del compromiso de los públicos recursivos «con la profundidad o los estratos de dicha autofundamentación») que Chris Kelty plantea más arriba.

Así, de entrada, el término «*bootstrap*» alude literalmente a la trabilla o tira de piel situada en la parte posterior de una bota para ayudar a hacer presión mientras se introduce el pie en la bota, todo ello sin necesidad de calzador externo. A partir de aquí, la expresión «*pull oneself up from the bootstraps*» (literalmente, «impulsarse hacia arriba tirando de la trabilla de las botas») designa generalmente la acción de salir adelante por sus propios medios, de modo autosuficiente, sin ayuda externa. Sobre esta base, el término «*bootstrapping*» se emplea en distintos campos (física, lingüística, biología, electrónica, finanzas...) para aludir a procesos de inicio autónomo que, sin necesidad de entradas, recursos o estímulos externos, desencadenan una serie de pasos conducentes a procesos o sistemas más amplios y complejos.

Según Wikipedia, en el ámbito específico de la informática, la metáfora «*bootstrapping*» comenzó a aplicarse en la década de los 50 a dos procesos fundamentales. Por un lado, «el proceso de arranque de un ordenador [que] implica una cadena de fases, en cada una de las cuales un programa más pequeño y simple carga y luego ejecuta el programa mayor y más complicado de la siguiente fase». Por otro, el proceso de desarrollo de entornos de programación cada vez más complejos y rápidos a partir de otros más simples, a lo cual volverá Kelty en el Capítulo 6: «El entorno más simple sería, quizás, un editor de textos muy sencillo [...] y un programa ensamblador. Utilizando estas herramientas, se puede escribir un editor de texto más complejo y un compilador simple para un lenguaje de más alto nivel y así sucesivamente, hasta obtener un entorno de desarrollo integrado y un lenguaje de programación de muy alto nivel». Véanse las entradas al respecto de Wikipedia (en inglés: <https://en.wikipedia.org/wiki/Bootstrapping#Computing>; en castellano: https://es.wikipedia.org/wiki/Bootstrapping_inform%C3%A1tica), así como el interesante hilo de discusión del foro de lengua española de Stackexchange: <https://spanish.stackexchange.com/questions/8492/c%C3%B3mo-se-dice-bootstrap-en-castellano> [N. del E.]

Este proceso de estandarización invierte de manera esencial el proceso de planificación. En lugar de planificar un sistema y a continuación estandarizarlo, refinarlo y finalmente construirlo según las especificaciones, el proceso de los RFC permite que los planes se propongan, implementen, refinen, repropongan, reconstruyan, y así hasta que los usuarios los adoptan y se convierten en protocolos aprobados por el IETF. La implicación para la mayoría de los *geeks* es que este proceso está permanente y fundamentalmente abierto: se pueden proponer, implementar y adoptar cambios sin fin, y cuanto mejor sea la tecnología, más difícil será mejorarla y por tanto habrá menos razones para subvertirla o reinventarla. También abundan contraejemplos en los que surge el estándar pero nadie lo adopta, lo cual sugiere que el proceso de estandarización se extiende más allá del círculo propuesta-implementación-propuesta-estándar para incluir el problema de lograr convencer a los usuarios para que dejen una tecnología que funciona bien por otra mejor. Con todos, tales fallos de adopción también son percibidos como una especie de confirmación de la calidad o facilidad de uso de la solución *actual*, y es más que probable que provoquen resistencia cuando alguna organización o entidad política intente obligar a los usuarios a pasarse al nuevo estándar —algo que el IETF se ha abstenido de hacer en general.

Conclusión: PÚBLICO RECURSIVO

Napster era un ejemplo familiar y ampliamente discutido de la «reorientación del poder y el saber» (o en este caso, del poder y la música) originada por Internet y las prácticas de los *geeks*. Sin embargo, Napster no era un público recursivo o un proyecto de software libre, sino un plan de negocio inspirado en las punto.com por el que se regalaba un programa privativo con la esperanza de que lograr un flujo de ingresos procedentes de la Bolsa, la publicidad o de versiones expandidas del software. Por tanto, más que defender a Napster, los *geeks* experimentaron sus restricciones legales como un toque de atención: Internet posibilita Napster y posibilitará otras muchas cosas, pero las leyes, las corporaciones, los grupos de presión, el dinero y los gobiernos pueden destruirlo todo.

Empecé este capítulo preguntándome qué une a los *geeks*: ¿Qué constituye la cadena que vincula a *geeks* como Sean y Adrian con *hipsters* de Berlín y empresarios y programadores de Bangalore? ¿Qué

constituye su afinidad si no es ninguno de los candidatos convencionales como la cultura, la nación, la empresa o la lengua? Una respuesta coloquial podría ser que es simplemente Internet lo que los mantiene unidos: ciberespacio, comunidades virtuales, cultura en línea. Pero esto no responde a la cuestión de por qué. ¿Porque pueden? ¿Porque la Comunidad es Dios? Si el objetivo es la mera asociación, ¿por qué no usar AOL o una gran red privada proporcionada por Microsoft?

Mi respuesta, por contra, es que la afinidad entre los *geeks* se estructura por medio de concepciones compartidas de orden técnico y moral. Constituyen un público, un público independiente que posee la capacidad de construirse, mantenerse y modificarse a sí mismo, y que no está restringido a las actividades de hablar, escribir, discutir o protestar. Los públicos recursivos se forman a través de su experiencia con Internet precisamente porque Internet es el tipo de elemento que pueden habitar y transformar. Dos características hacen de los públicos recursivos algo distintivo: la capacidad para incluir la práctica de crear esta infraestructura como parte de la actividad de ser público o de impugnar el control; y la habilidad de «hacerse recursivo» a través de las capas de dicha infraestructura, manteniendo en cada nivel su condición de público sin convertirla en algo inmutable, estático, inmodificable.

La afinidad constituida por un público recursivo, a través del medio de Internet, crea *geeks* que entienden claramente lo que significa la asociación mediante la Red. Esta afinidad estructura su idea de qué es Internet y qué permite: crear, distribuir y modificar conocimiento, música, ciencia, software. La infraestructura —*esta-infraestructura-concreta*, la Red— debe ser entendida como parte de este imaginario (además de ser una palpante maraña de ordenadores, cables, ondas y electrones).

Internet no es el único medio para tal asociación. Una empresa, por ejemplo, también se basa en un imaginario compartido de la economía, de cómo se supone que funcionan los mercados, los intercambios y los ciclos de negocio; se trata de la creación de un conjunto de relaciones y prácticas, uno generalmente inflexible —incluso en esta era del llamado capitalismo flexible— porque requiere la dedicación de tiempo, personas y capital. Incluso en el capitalismo veloz uno necesita alquilar oficinas, comprar papel higiénico, instalar programas para las nóminas, etc.

El software y las redes pueden ser igualmente concretos —conectando a personas, capital y otros recursos a lo largo del tiempo y creando así una infraestructura—, pero podría afirmarse que son más flexibles,

más modificables y más reprogramables —que una empresa, que un sistema de aguas residuales, que un mercado de valores. Internet en particular, especialmente en las historias de la IETF y del proceso de los RFC, representa una radicalización de esta flexibilidad: no solo se puede crear una aplicación como Napster que aprovecha inteligentemente las capas (protocolos, enrutadores y rutas) de la Red, sino que podemos reescribir esas mismas capas, posibilitando una nueva clase de Napsters. La dificultad de tal tarea se acrecienta con la profundidad cada vez mayor de las capas, pero la posibilidad no está (aún) restringida por ninguna organización, persona, ley o gobierno. La afinidad —la pertenencia a un público recursivo— depende de la adopción de las ideas morales y técnicas de este tipo de orden.

La urgencia evidenciada en el caso Napster (y repetida en muchos otros ejemplos, como el debate sobre la neutralidad de la red) está vinculada a una idea moral de orden en la que hay un imaginario compartido de El Público, y no solo una vasta multiplicidad de públicos competidores. Es una urgencia ligada directamente al hecho de que Internet provee a los *geeks* de una plataforma, un entorno, una infraestructura a través de la cual no solo se asocian sino que crean, y ello de una forma ampliamente percibida como autónoma, autotélica e independiente como mínimo de las formas de poder más convencionales: Estados y corporaciones —de hecho, lo bastante independiente como para que tanto los Estados como las corporaciones puedan hacer un uso amplio de esta infraestructura (ellos mismos pueden convertirse en *geeks*) sin poner necesariamente en peligro su independencia.

II. REFORMADORES PROTESTANTES, POLÍMATAS Y TRANSHUMANISTAS

Los *geeks* hablan mucho. No suelen hablar sobre públicos recursivos, ni tampoco sobre imaginarios, infraestructuras, órdenes morales o técnicos. Pero hablan mucho. La creación de software y redes requiere una gran cantidad de tiempo y de escritura: aprender y hablar, enseñar y discutir, contar historias y leer polémicas, reflexionar acerca del mundo desde y en torno a la infraestructura que uno habita. En este capítulo, me recreo en los relatos que cuentan los *geeks*, y especialmente en relatos y reflexiones que delimitan problemas contemporáneos en torno al saber y el poder —relatos de grandes asuntos como el progreso, la ilustración y la libertad.

De manera evidente, cuestiones como la ilustración, el progreso y la libertad siguen formando parte de un «imaginario social», especialmente las imaginaciones acerca de la relación del saber y la ilustración con la libertad y la autonomía, tan claramente en juego en la noción de público o de esfera pública. Y por más que el ejemplo del software libre ilumine el modo en que se proponen, refutan e implementan las cuestiones de la ilustración, el progreso y la libertad en y a través del software y las redes, este capítulo contiene relatos que se comprenden mejor en cuanto «pasados usables» —narrativas menos técnicas y más accesibles que dotan de sentido al mundo contemporáneo mediante la reflexión sobre el pasado y sus diferencias con la actualidad.

Pasados usables es un término más caritativo para lo que podría denominarse mitos modernos de los *geeks*: relatos que son reconocidos por sus narradores como una combinación de realidad y ficción. Estas historias no se cuentan con la intención de recordar el pasado, sino de dar sentido al presente y al futuro. Confieren sentido a unas prácticas que no son cuestionadas al realizarse, pero que tampoco se

entienden fácilmente en los términos coloquiales o intelectuales al uso. El primer conjunto de relatos se refiere a la Reforma Protestante: alegorías que recurren a las Iglesias Católica y Protestante, los legos, el clero, los sumos sacerdotes y las imágenes de control y liberación propias de la época de la Reforma. Podría resultar sorprendente que los *geeks* volvieran al pasado (y especialmente a la alegoría religiosa) para dar sentido a su presente, pero la razón es muy simple: no hay relatos preestablecidos que doten de sentido a las prácticas contemporáneas de los *geeks*. Precisamente porque los *geeks* están «figurándose» cosas que no son claras ni obvias, se hallan necesariamente desprovistos de formas efectivas de hablar sobre ello. La Reforma Protestante funciona bien como alegoría porque separa el poder del control; se basa en relatos acerca del catecismo y el ritual, los alfabetos, los panfletos y las liturgias, las indulgencias y la autoayuda, todo ello con el propósito de dar a los *geeks* una forma de dar sentido a la distinción entre poder y control y a cómo esta se relaciona con la economía técnica y política que ocupan. La relación contemporánea entre Estados, corporaciones, pequeñas empresas y *geeks* no queda reflejada en oposiciones familiares como comercial/no comercial, a favor/contra la propiedad privada, o capitalista/socialista —se trata de una relación de reforma y conversión, no de revolución o derrocamiento.

Los pasados usables son relatos, pero del tipo que refleja actitudes y formas específicas de pensar acerca de la relación entre pasado, presente y futuro. Los *geeks* piensan y hablan mucho sobre el tiempo, el progreso y el cambio, pero sus conclusiones y actitudes no son uniformes en modo alguno. Algunos *geeks* están mucho más al tanto que otros de las circunstancias históricas y los contextos específicos en que se desenvuelven. En este capítulo planteo una pregunta a través del famoso ensayo breve de Michel Foucault *¿Qué es la Ilustración?* Concretamente mi pregunta es ¿son modernos los *geeks*? Para Foucault, en su relectura del texto epónimo de Kant de 1784, el problema de ser moderno (o de que una época sea «ilustrada») no atañe al periodo histórico o la época en que vive la gente; más bien, implica una relación subjetiva, una *actitud*. La explicación de Kant sobre la Ilustración no sugiere que esta constituya en sí misma un universal, sino que se da a través de una manera de reflexionar sobre la diferencia que marcan los cambios de nuestro pasado histórico inmediato en nuestra concepción de los supuestos universales que se dan en una historia mucho más dilatada —es decir, que debemos preguntarnos por qué es necesario pensar como pensamos hoy acerca de problemas que han

sido afrontados en épocas pasadas. Para Foucault, tales reflexiones deben estar enraizadas en las «formas históricamente singulares bajo las cuales han sido problematizadas las generalidades de nuestra relación con las cosas»¹. De este modo, quiero plantear respecto de los *geeks* esta pregunta: ¿cómo conectan los problemas singulares que afrontan —desde Internet a Napster, pasando por la propiedad intelectual, la compartición y la reutilización de código fuente— con las generalidades de las relaciones dentro de las que los narran como problemas de libertad, saber, poder e ilustración? O como lo expresa Foucault: ¿Son *modernos* en este sentido? ¿«Desprecian el presente» o no?

Las actitudes que adoptan los *geeks* al responder a estas preguntas cubren un espectro que he identificado que abarca desde los «polímatas» a los «transhumanistas». Estos apodos están extraídos de discusiones reales con *geeks*, pero no designan a un tipo de persona. Son acaso «subrutinas», invocaciones desde un programa más amplio de imaginación de orden técnico y moral. Es posible que una misma persona sea un polímatha en el trabajo y un transhumanista en casa, pero en términos generales se trata de roles encontrados y opuestos. En las rutinas polímatas, la tecnología es una intervención dentro de un complicado e históricamente singular campo de personas, costumbres, organizaciones, leyes y otras tecnologías; en las rutinas transhumanistas, la tecnología es vista como una fuerza inevitable —producto de la acción humana, pero no del diseño humano— que es imposible controlar o resistir a través de medios legales o convencionales.

Reforma Protestante

A los *geeks* les encantan las alegorías acerca de la Reforma Protestante; se regodean con historias sobre Lutero y Calvin, sobre el papado y la iconoclastia, sobre la reforma frente a la revolución. Las alegorías de la revuelta protestante permiten a los *geeks* conferir sentido a la relación entre el Estado (la monarquía), las grandes corporaciones (la Iglesia Católica), las pequeñas empresas emergentes, los programadores individuales y los eruditos entre quienes pasan la mayor parte de su tiempo (los reformadores protestantes), y los legos (conocidos como «*lusers*»²

1. Foucault, «What Is Enlightenment», p. 319 [ed. cast.: «¿Qué es la Ilustración?», trad. por Antonio Campillo. *Daimon. Revista de Filosofía*, nº 7, 1993, p. 18].

2. Juego de palabras intraducible fruto de la combinación de los términos ingleses «*losers*» («perdedores») y «*users*» («usuarios»). [N. del E.]

y «*sheeple*»).³ Esto les brinda un modo de aseverar que (para salvar al capitalismo de los capitalistas) prefieren la reforma a la revolución. Obviamente, no todos los *geeks* cuentan relatos acerca de «las guerras religiosas» y la Reforma Protestante, pero estas imágenes reaparecen en las conversaciones lo bastante a menudo como para que la mayoría de *geeks* las reconozca más o menos instantáneamente como un modo de dotar de sentido al moderno poder político, corporativo y estatal en el terreno de las tecnologías de la información: las figuras del Papa, la Iglesia Católica, el Vaticano, los monarcas de diferentes naciones, los legos, los rebeldes eruditos como Lutero y Calvino, así como las formas de sectarismo, iconoclastia («En el principio fue... la línea de comandos»), poder político-religioso y discusiones teológicas arcanas⁴. Las alegorías que despliegan proveen a los *geeks* de una forma de dar sentido a una situación moderna de similar complejidad en la que la lucha ya no es entre la Iglesia y el Estado, sino entre la Corporación y el Estado; una situación en la que la lucha de los *geeks* no atañe a los asuntos de doctrina u organización eclesiástica, sino a la tecnología de la información y su organización como propiedad intelectual y motor económico. Subrayo aquí que esta analogía no la realizo yo mismo (pese a que me sirva de ella con gozo), sino que circula ampliamente entre los *geeks* que estudio. Para los historiadores o críticos religiosos, puede parecer incompleta, absurda o extravagante, pero sigue desempeñando una función específica, y es por ello que la destaco como un componente de las ideas prácticas y técnicas de orden que comparten los *geeks*.

En un primer nivel están las alegorías de «guerra religiosa» o «guerra santa» (y, cada vez más, de «yihad»). Estos relatos revelan un cierto cinismo: describen una guerra técnica de detalles entre dos aplicaciones que ejecutan la misma función por medios diferentes, de modo que la devoción por una u otra es vista como una forma de fidelidad teológica arbitraria, la cual depende de una racionalidad pura a la vez que requiere un juicio estético o político. Tales relatos implican que dos tecnologías son igualmente buenas o malas, y que la elección de secta que alguien hace es enteramente irracional y se basa en las vicisitudes de su trayectoria y sus creencias. Algunas personas

3. Juego de palabras intraducible fruto de la combinación de los términos ingleses «*sheep*» («oveja, borrego») y «*people*» («gente»). [N. del E.]

4. Stephenson, *In the Beginning Was the Command Line* [ed. cast.: *En el principio fue... la línea de comandos*].

son celosas proselitistas de una tecnología, otras no. Como explica un mensaje de Usenet:

Las «guerras» religiosas han tendido a producirse por tecnicismos teológicos y doctrinales de un tipo u otro. El paralelismo entre esto y los tecnicismos *informáticos* que ocasionan las «guerras informáticas» es muy fuerte.⁵

Acaso la más conocida y famosa de estas guerras sea la mantenida entre Apple y Microsoft (previamente entre Apple e IBM), un conflicto que a menudo se dirime mediante golpes bruscos y dramáticos que sugieren la existencia de diferencias fundamentales, cuando en realidad las diferencias son extremadamente pequeñas.⁶ Los *geeks* también están familiarizados con una pléthora de «guerras santas» no tan conocidas: EMACS contra vi; KDE contra Gnome; Linux contra BSD; Oracle contra todas las demás bases de datos.⁷

A menudo, el lenguaje de la Reforma se desliza jocosamente en los por lo demás serios intentos de formular juicios estéticos sobre tecnología, como en este análisis del lenguaje de programación tcl/tk:

Tampoco está claro que el principal criterio de diseño en tcl, perl o Visual BASIC fuera la belleza visual —ni con toda probabilidad debería haberlo sido. Ousterhout dijo que la gente votaría con los pies.⁸ Esto es importante. Mientras los Sumos Sacerdotes en sus

5. Identificador del mensaje: [thr55.221960\\$701.2930569@news4.giganews.com](mailto:thr55.221960$701.2930569@news4.giganews.com)

6. Umberto Eco dotó al conflicto entre Apple y Microsoft de una expresión memorable en un artículo ampliamente leído donde comparaba la interfaz de usuario de Apple con el catolicismo y la interfaz de usuario del PC con el protestantismo («La bustina di Minerva», *Espresso*, 30 de septiembre de 1994, contraportada).

7. Una entrada de Wikipedia diferencia las guerras religiosas de las «guerras de *flames*» (*«flame wars»*) normales y corrientes como sigue: «Mientras que una guerra de *flames* suele consistir en un aluvión particular de mensajes hostiles en un contexto no hostil, una guerra santa es un desacuerdo pertinaz que puede durar años o incluso abarcar toda una carrera» [*Flaming [Internets]*] http://en.wikipedia.org/wiki/Flame_war [última consulta: 18 de abril de 2018]).

Para una aclaración añadida sobre las «guerras de *flames*», véase nota 39 [N. del E.]

8. Traducimos literalmente la expresión inglesa *«to vote with one's feet»*, introducida por Charles Tiebout en 1956 para sintetizar su modelo de provisión de bienes públicos a escala local. Con este modelo el economista estadounidense aborda el problema del polizón derivado del hecho de que el Gobierno federal (o, por extensión, cualquier otro Gobierno central) carece de un modo efectivo de obligar a la ciudadanía a expresar sus preferencias sobre la oferta de bienes públicos y, en consecuencia, a sufragar dicha oferta. A partir de siete premisas estrictas, Tiebout desplaza el eje del debate de lo nacional a lo local para proponer que los «consumidores-votantes» manifiesten sus preferencias mudándose a aquella localidad que mejor las satisfaga, esto es, que

Torres de Marfil diseñan para su deleite propio lenguajes prístinos de belleza austera y equilibrada perfección, el resto de los mortales, en su ignorancia ciega y feliz, se las apañará usando pequeños lenguajes desagradables como los enumerados arriba. Estos pobres borrachines sacarán adelante una gran cantidad de trabajo, pondrán pan en la mesa para sus hijos y llegarán de noche a casa para compartirlo con ellos. La diferencia es que los sacerdotes señalarán a los legos agitando el dedo, y a estos les dará igual porque estarán dormidos en su cama.⁹

En este caso, la «guerra religiosa» atañe a la diferencia entre los lenguajes de programación académicos y los de los programadores profesionales, la cual se hace equivaler a la distinción entre la insularidad de la Iglesia Católica y la autoayuda de los legos protestantes: los héroes (como tcl/tk, perl y python —todos software libre—) son los «pequeños lenguajes desagradables» de los legos; los Sumos Sacerdotes diseñan (presumiblemente) Algol, LISP y otros lenguajes «académicos».

En un segundo nivel, no obstante, la alegoría hace un uso preciso de detalles de la Reforma Protestante. Por ejemplo, en un hilo de debate acerca de las múltiples luchas surgidas en torno al GCC (*Gnu C Compiler*, Compilador Gnu de C), un componente central de los diversos sistemas operativos UNIX, Christopher Browne publicó esta alegoría de la Contrarreforma en un grupo de Usenet:

El proyecto EGCS comenzó hace dos años cuando el desarrollo de C++ (y GCC) se quedó bastante «atascado». EGCS buscaba integrar varios grupos de parches que la gente estaba construyendo para la «familia» GCC. En efecto, se había producido una «Reforma Protestante», con escisiones entre:

- a) La confesión GNU FORTRAN;
- b) La Secta de la Puesta a Punto de Pentium;
- c) La confesión de los Planificadores de Instrucciones de IBM en Haifa;
- d) Los Acólitos del C++ Estándar;

ofrezca el nivel de bienes públicos que demandan al precio (impuestos) que están dispuestos a pagar: «En este modelo los gobiernos locales no intentan 'adaptarse a' las preferencias de los consumidores-votantes. En lugar de ello, puede considerarse que aquellos gobiernos locales que atraigan al número óptimo de residentes son 'adoptados' por el sistema económico» (C. Tiebout, «*A Pure Theory of Local Expenditures*», *The Journal of Political Economy*, vol. 64, nº 5, octubre de 1956, p. 420). [N. del E.].

9. Identificador del mensaje: 369tva\$8l0@csnews.cs.colorado.edu.

Estos grupos habían sido incapaces (por razones varias) de integrar sus iniciativas con la Versión Católica, GCC 2.8. La Sociedad Ecuménica del Compilador GNU intentó atraer a estos grupos al seno del rebaño católico. El proyecto tuvo bastante éxito: GCC 2.8 fue sucedido por GCC 2.9, que no era una actualización directa desde la versión 2.8 sino más bien el resultado del proyecto EGCS. EGCS es ahora GCC.¹⁰

Además del placer obvio con que los *geeks* despliegan la dimensión sectaria de la Reforma Protestante, también se permiten ver sus luchas como las de eruditos al estilo de Lutero, enfrentados a poderosas instituciones mundanas que son distintas pero están entrelazadas entre sí: la Iglesia Católica y los monarcas absolutistas. Algunas veces estas comparaciones pretenden mofarse de la discusiones teológicas; otras son más directamente hagiográficas. Por ejemplo, un artículo de 1998 en *Salon* compara a Martín Lutero y a Linus Torvalds (creador del núcleo de Linux).

En los días de Lutero, la Iglesia Católica Romana poseía un cuasimonopolio sobre la vida cultural, intelectual y espiritual de Europa. Pero el principal texto fuente que informaba esa vida —la Biblia— estaba fuera del alcance de la gente ordinaria. [...] Linus Torvalds es un reformador de la era de la información cortado por el mismo patrón. Como Lutero, su periplo comenzó mientras estudiaba para ordenarse en el sacerdocio moderno de los ingenieros informáticos en la Universidad de Helsinki —lejos de las sedes de poder en Redmond y Silicon Valley. También como Lutero, tuvo la idea divina y un poco chiflada de eliminar a las burocracias entrometidas y poner a la gente corriente en relación directa con un poder superior —en este caso, sus ordenadores. Disolviendo la distinción entre programador y usuario, alentó a la gente ordinaria a participar en el desarrollo de su entorno informático. Y al igual que Lutero buscó hacer accesible al *hoi polloi*¹¹ todo el *shebang*¹²

10. Identificador del mensaje: c1dz4.145472\$mb.2669517@news6.giganews.com. Debería señalarse, en caso de que los lectores no estén seguros de la seriedad de esto, que las siglas EGCS significaban *Extended GNU Compiler System* (Sistema de Compilación GNU Extendido), no *Ecumenical GNU Compiler Society* (Sociedad Ecuménica del Compilador GNU).

11. Término procedente de la expresión del griego clásico *oi πολλοί* que significa literalmente «los muchos», «la mayoría» y que se usa en inglés de modo peyorativo como sinónimo de «masa» o «plebe». [N. del E.]

12. Según Wikipedia, «shebang» es, en la jerga de Unix, el nombre que recibe el par de caracteres #! que se encuentran al inicio de los programas ejecutables interpretados. [...] A

sacramental —el vino, el pan y la Palabra traducida—, Linus busca revocar el acceso privativo del desarrollador al sistema operativo, insistiendo en que el código fuente completo del sistema operativo se le entregue —sin coste— a cualquier hijo de vecino con un ordenador.¹³

Los eruditos con fuertes convicciones —monjes y curas cuya iniciación y maestría son evidentes— hacen que la alegoría funcione. Otros usos de la iconografía cristiana son menos fieles a las fuentes, por así decirlo. Otra personalidad prominente, Richard Stallman, de la Fundación para el Software Libre, es dado a vestirse de su álder ego: San IGNUCIO, santo patrón de la iglesia de EMACS —una iglesia sin dios, pero con una intensa devoción hacia un barroco procesador de textos de potencia innegable, cuasimilagrosa.¹⁴

A menudo el atractivo de la retórica de la Reforma proviene de una especie de impugnación del presente: pese a todo este superfabuloso portento computróntico de tecnología punta, no somos para nada menos feudales, menos violentos, menos arbitrarios y antide-mocráticos; o dicho de otra forma, los *geeks* han progresado, han visto la luz y el camino, pero el resto de la sociedad —especialmente los sectores de la gestión y la mercadotecnia— no lo ha hecho. En este sentido, las alegorías de la Reforma constituyen relatos acerca de cómo «las cosas nunca cambian».

Pero el uso más convincente de la Reforma Protestante como pasado usable se da en las interpretaciones más detalladas de los *geeks* acerca de la economía política de las tecnologías de la información. Por ejemplo, la alegorización de la Iglesia Católica con Microsoft constituye un componente frecuente, como se ve en este breve mensaje relativo a las combinaciones de teclas de arranque del sistema operativo Be: «Estos apretones de manos secretos pretenden reforzar a un alto clero cabalístico y no deberían haberse revelado a los legos.

continuación de estos caracteres se indica la ruta completa al intérprete de las órdenes contenidas en el mismo. Véase: <https://es.wikipedia.org/wiki/Shebang> [N. del E.]

13. «Martin Luther, Meet Linus Torvalds,» *Salon*, 12 de noviembre de 1998, http://www.salon.com/1998/11/12/feature_258/ (*última consulta: 18 de abril de 2018*).

14. Véase: <http://www.stallman.org/saint.html> (*última consulta: 18 de abril de 2018*) y <http://www.dina.kvl.dk/~abraham/religion/> (enlace no accesible en la última consulta de la misma fecha). Sobre EMACS, véase el Capítulo 6. [N. del E.: Pese a que el enlace previo no sea accesible, aún es posible seguir a la Iglesia de EMACS fundada por Per Abrahamsen en este grupo de Google+: [https://plus.google.com/communities/101818454981911211159\]](https://plus.google.com/communities/101818454981911211159)

Olvida que alguna vez viste esta entrada y ve a comprarte algo de Microsoft».¹⁵

Más generalmente, las grandes corporaciones como IBM, Oracle o Microsoft son identificadas con el Catolicismo, mientras que los burocráticos congresos y parlamentos con sus grupos de presión asumen el papel de los monarcas absolutistas y sus compinches. Los *geeks* pueden entonces verse a sí mismos como luchadores que defienden la Cristiandad (el verdadero capitalismo) contra la Iglesia (las corporaciones) y como reformadores de un modo de vida corrompido por la Iglesia y los monarcas, en lugar de como revolucionarios derrocando un sistema que consideran defectuoso. Hay un componente histórica y técnicamente específico de esta economía política en la que corporaciones como IBM y Microsoft están interesadas en mantener a los usuarios «tan confinado[s] al Gigante Azul»¹⁶ como los desdichados encadenados en una mazmorra medieval.¹⁷

Tales relatos atraen porque eluden el lenguaje de la política estadounidense contemporánea (liberales y conservadores, Demócratas y Republicanos) en el que cualquier asunto solo comprende dos bandos. También eluden la discusión entre capitalismo y socialismo, por la cual si no eres pro-capitalista debes ser comunista. Se trata de relatos que permiten a los *geeks* más pragmáticos involucrarse en la intervención y la reforma, en lugar de en la revolución. Pese a que rara vez la he visto articulada tan abiertamente, la alegoría a menudo implica que uno debe «salvar al capitalismo de los capitalistas», un sentimiento que implica al menos algún tipo de control humano sobre el capitalismo.

De hecho, el uso alegórico de la Reforma y la Iglesia genera todo tipo de comparaciones ingeniosas. Una descripción típica de tales comparaciones puede ser algo así: la Iglesia Católica representa a las grandes corporaciones que cotizan en bolsa, especialmente a aquellas que controlan grandes cantidades de propiedad intelectual (cuya concesión puede equipararse a grandes rasgos con las ceremonias de confesión y comunión) para las que dependen de la asistencia y el

15. Identificador del mensaje: 6ms27l\$6e1@bgtnsc01.worldnet.att.net. En un caso muy cómico la comparación se hace literal: «Microsoft adquiere la Iglesia Católica» (Identificador del mensaje: gaijin-870804300-dragonwing@sec.lia.net).

16. El apelativo Gigante Azul (o *Big Blue*) deriva de la tradicional (hoy ya extinta) norma de vestuario por la cual los empleados de IBM debían vestir traje azul. Fuente: <https://es.wikipedia.org/wiki/IBM> [N. del E.]

17. Paul Fusco, «The Gospel According to Joy», *New York Times*, 27 de marzo de 1988, Sunday Magazine, p. 28.

apoyo de los gobiernos nacionales. Naturalmente, son los renombrados excesos de la Iglesia (indulgencias, complejidad litúrgica, ceremoniales rituales y corrupción) los que ponen fácil la alegoría. Las corporaciones modernas pueden ser retratadas como un reducido cuerpo papal de élite compuesto por teólogos (los ejecutivos y sus abogados, las juntas directivas y sus abogados), los cuales dirigen a un clero mucho más amplio (los empleados), que a su vez sirve a una comunidad de legos (los consumidores) en gran medida concebida como pecaminosa (por gastar muy poco en música y películas —de hecho, por llegar a «robarlas»—) y como tal necesitada de una elaborada purificación ritual (campañas publicitarias y demandas judiciales) por parte de la Iglesia. Solo por mediación de la Iglesia se alcanza la Gracia (el Sueño Americano), y esta toma forma a través de los sagrados actos de ir de compras y de reformar el hogar. Los ejecutivos predicen mensajes de condenación al Gobierno, mensajes que la mayoría de cargos gubernamentales está más que deseosa de escuchar: no interfieran en nuestra cuota de mercado, no influyan en nuestra política de precios, no limiten nuestra capacidad de expansión mercantil. Los ejecutivos también ofrecen inexplicables promesas de salvación en forma de desregulación y de la versión estadounidense de la «reforma» —la demolición del Estado y los servicios sociales nacionales. Por su parte, los cargos gubernamentales han desarrollado su propio «derecho divino de reyes», el cual justifica ciertas formas de manipulación (una vez llamadas «elecciones») de la sucesión. Los grupos de presión o las asociaciones industriales venden indulgencias a diestro y siniestro, y los decretos del papado no evidencian más que su total desconexión respecto de la miserable existencia cotidiana de su grey.

De hecho, resulta extraordinario lo sencillas que se vuelven estas comparaciones a medida que uno se entera de más detalles de la economía política de la información. Pero las alegorías de la Reforma y el poder clerical pueden llevar fácilmente al cinismo, que en este caso quizá debería interpretarse como despojamiento de derechos políticos y no como falta de fe. Y sin embargo los pasados usables de estos modernos monjes y sacerdotes de mentalidad reformista no solo afloran regularmente porque ofrezcan alivio frente a la chábbara técnica sino porque explican una situación política, técnica y legal que carece de relatos preestablecidos. Los *geeks* viven en un mundo controlado minuciosamente por organizaciones corporativas, medios masivos, departamentos de *marketing* y grupos de presión, pero al

mismo tiempo comparten una profunda desconfianza hacia la regulación gubernamental —necesitan otro repertorio de *cuentos de así fue*¹⁸ para dotarla de sentido. Los consabidos pasados *no usables* acerca de la liberalización de los mercados, la inevitabilidad del capitalismo y la democracia o, más recientemente, la necesidad de seguridad no hacen justicia a su experiencia.

Las alegorías de la Reforma son relatos que dan sentido a la economía política de la información. Pero también tienen un uso más preciso: dotar de sentido a la distinción entre poder y control. Dado que los *geeks* se encuentran «más cerca de la máquina» que el resto de los legos, uno podría esperar razonablemente que sean ellos quienes ostenten el poder. Sin embargo, es evidente que no es así, y son las frustraciones y los misterios por los que los Estados, las corporaciones y los individuos manipulan detalles técnicos con vistas a un desplazamiento del poder los factores que a menudo provocan la ira más profunda de los *geeks*. Así pues, el control incluye los métodos detallados y las prácticas efectivas por los que las corporaciones, las agencias gubernamentales o los individuos intentan manipular a la gente (o reclutarla para manipularse a sí misma y a otros) de modo que tome decisiones técnicas que sirvan al poder, en vez de a la racionalidad, la libertad, la elegancia o a cualquier otra preocupación *geek*.

Consideremos el tema de lo malvado. Durante mis conversaciones con Sean Doyle a finales de los 90, así como con otros muchos *geeks*, el término *malvado* [«evil»] se empleaba habitualmente para referirse a algún tipo de problema técnico o de diseño. Le pregunté a Sean qué quería decir con ello:

SD: [*Malvado* es] solo un término que uso para decir que algo está mal, pero normalmente implica que está mal a propósito, que ha habido intervención detrás. No logro recordar [el ejemplo que pusiste] pero creo que pudo haber sido algún equipo de General Electric, donde por defecto al equipo le gusta enviar cosas en su propio formato privado en lugar de en DICOM [el estándar de la

18. El autor emplea aquí la expresión inglesa «*just-so stories*», procedente de la obra homónima en la que Rudyard Kipling fabula acerca del origen de distintos fenómenos, especialmente de cómo diferentes animales adquirieron sus rasgos distintivos. En castellano se la puede encontrar con el título *Cuentos de así fue*, de ahí nuestra traducción. Por extensión, la expresión «*just-so stories*» se aplica a cualquier explicación sobre el origen de un proceso o característica natural que, o bien no está plenamente probada (acepción que la asemeja al concepto de «hipótesis»), o bien es inverificable (acepción próxima al concepto de «mito»). [N. del E.]

industria radiológica para imágenes digitales] si le dejas opción. No sé por qué habrán hecho algo así, no resuelve ningún problema de retrocompatibilidad, en realidad es solo algo exclusivista. Así que supongo que hay algo malvado en ello...

CK: Otro de los ejemplos que tenías... ¿no tenía que ver con Internet Explorer 3.0?

SD: Sí, ah, sí, hay tantas cosas en IE3 que son completamente malvadas. Como por ejemplo, una de ellas: en el protocolo http existe una cosa llamada «campo de agente de usuario», donde un navegador declara al servidor quién es. Si observas IE, este declara que es Mozilla, que es [el nombre en clave de] Netscape. ¿Por qué hace esto? Bueno, pues porque un montón de servidores web estaban enviando cierto código según el cual si el navegador fuera Mozilla, servirían las cosas bien, [y si no] mandarían algo muy simple o estúpido que se vería fatal. Pero resultó que [IE3 o quizás IE2] no admitía algunas cosas cuando salió. Por ejemplo, no creo que admitieran tablas, y luego sus versiones de Javascript eran tan diferentes que no había forma de que el navegador fuera compatible —simplemente añadió una tremenda complejidad. No era más que un modo de pitorrearse de Internet y decir que no hay ninguna ley que diga que tengamos que seguir estos estándares de Internet. Podemos hacerlo como nos dé la puñetera gana, y somos tan grandes que no nos lo podéis impedir. Por eso lo veo como malvado. Quiero decir, obviamente tienen el talento para hacerlo. Obviamente tienen los recursos para hacerlo. Y obviamente han hecho su trabajo, es solo que les da este pronto de no ofrecer asistencia técnica a un cierto tipo MIME o de darla a algunas cosas de forma diferente que los demás.

CK: Pero estos problemas de incompatibilidad pueden ser fruto de una falta de comunicación o de coordinación, la cual podría implicar cierto grado de intencionalidad, ¿no?

SD: Bueno, yo lo veo más como Estupidez que como Maldad (risas). No, lo malvado se da cuando existe la oportunidad de hacer algo, y la comprensión de la oportunidad, y los recursos y todo eso —y entonces haces algo solo por fastidiar a los demás. Sabes, estoy seguro de que sucede como en los divorcios conflictivos, en los que uno prefiere vender la casa por la mitad de su valor antes que dejársela a la pareja.

Sean relaciona el control con el poder proyectando un enfoque moral sobre las decisiones de una gran corporación. Aunque la alegoría específica de la Reforma Protestante no opere aquí, los detalles sí lo hacen. La decisión de Microsoft de manipular el funcionamiento de Internet Explorer no surge de una carencia de sofisticación técnica y, de acuerdo con Sean, tampoco es un «accidente» de la complejidad, sino que constituye una manifestación deliberada de poder económico y político para corromper aquellos detalles por los que se crea y estandariza software y se cumplen sus expectativas de funcionamiento. El claro objetivo de esta actividad es la *conversión*, la expansión del rebaño de Microsoft mediante un control minucioso de las creencias y prácticas (navegadores y funcionalidades) de los usuarios informáticos. Aplicar a Microsoft el calificativo de «malvada» de esta forma tiene un sentido muy similar al cuestionamiento del uso que la Iglesia Católica hace del ritual, la ceremonia, la alfabetización y la historia —los detalles de la «implementación» de la religión, por decirlo de algún modo.

En los propios términos de la Reforma Protestante, las prácticas de conversión, así como las de liberación, aprendizaje y autoayuda son cruciales para la historia. No es casualidad que muchos historiadores de la Reforma Protestante llamen ellos mismos la atención sobre las promesas de liberación a través de las «tecnologías de la información» reformistas¹⁹. Afirmaciones coloquiales (y a menudo académicas) acerca de que la imprenta fue tecnológicamente necesaria o suficiente para provocar la Reforma surgen constantemente como una parábola de esta nueva era de la información. Con frecuencia la imprenta es la única causa «tecnológica» tenida en cuenta, pero los académicos de la verdadera, histórica Reforma Protestante también prestan especial atención a la extensión de la alfabetización, a la circulación de panfletos devotos, catecismos y folletos teológicos, así como al abanico de transformaciones en las relaciones políticas y legales que se desarrollaron simultáneamente a la introducción de la imprenta.

19. Véase, por ejemplo, Matheson, *The Imaginative World of the Reformation*. Existe un debate riguroso acerca de la relación entre la imprenta, la religión y el capitalismo: un ejemplo paradigmático es la obra de Eisenstein *The Printing Press as an Agent of Change* [ed. cast. (abreviada): *La revolución de la imprenta en la Europa moderna*], inspirada en *The Gutenberg Galaxy* de McLuhan [ed. cast. *La Galaxia Gutenberg*]. Véase también Ian Green, *Print and Protestantism in Early Modern England* y *The Christian's ABCs*; Chadwick, *The Early Reformation on the Continent*, caps. 1-3.



Una última forma de demostrar la efectividad de estas alegorías —su capacidad para operar en la mente de los *geeks*— es demostrar cómo han empezado a operar en *mí*, hasta qué punto me he convertido en *geek* —una forma de alegorización participante, por decirlo así. Cuanto más considera uno los problemas que conforman la economía política contemporánea de las tecnologías de la información que habitan los *geeks*, más probabilidades hay de que estas alegorías comiencen a aparecer casi automáticamente —como, por ejemplo, cuando leí *The Story of A*, un libro delicioso sin relación alguna con los *geeks*, una obra sobre la alfabetización en los albores de EEUU. Su autora, Patricia Crain, expone que la Cruz de Cristo (véase arriba) se usaba a menudo en la creación de cartillas o tablillas de alfabetización, pequeñas palas con el lomo de cuero inscritas con el Padrenuestro y el alfabeto, las cuales se usaban para enseñar el abecedario a los niños ya desde el siglo XV y hasta entrado el XIX:

En sus primeras manifestaciones impresas, el alfabeto pedagógico no viene encabezado por la letra A sino por la «Cruz de Cristo»:
¶ [...] Dado que el alfabeto está asociado a la iconografía católica, como si los dos conjuntos de signos fueran realmente parte de un sistema semiológico, una de las luchas de la Reforma será la de arrebatar el alfabeto a la Iglesia Católica.²⁰

Aquí, alegóricamente, el control del alfabeto por parte de la Iglesia Católica (como la programación de Internet Explorer por parte de Microsoft para ofuscar los estándares de Internet) no es simplemente ideológico; no es solo una fantasía de origen o posesión sembrada en el barbecho mental de los creyentes, sino verdaderamente una herramienta de control normativo muy concreta, muy objetiva y muy específica en sus medios. Como explica Crain:

Hoy ¶ representa el imprimátur de la Iglesia Católica en las páginas con *copyright*. En su conexión con el primer alfabeto moderno, esta cruz comporta también un imprimátur o efecto de licencia. Sin

20. Crain, *The Story of A*, pp. 16-17.

embargo, este «imprímase» no se dirige a los impresores artesanos sino a la mente y la memoria de los jóvenes estudiantes. [...] Como el *copyright* moderno, la cruz autoriza la existencia del alfabeto y asocia sus letras con la autoría sagrada, especialmente porque otra perdurable función de ✕ en los misales litúrgicos es la de señalar los pasajes evangélicos. El símbolo transmite información a la vez que genera comportamiento ritual.²¹

Hoy la © comporta el mismo poder ideológico y legal, si no más, que la cruz de la Iglesia Católica. Es el símbolo por excelencia de la autoría, por más que en origen y función solo regule la propiedad y los derechos. El pensamiento mágico sobre el *copyright* abunda, pero una función importante del símbolo ©, además de sus implicaciones legales, es alcanzar lo mismo que la cruz de Cristo: asociar en la mente del lector la propiedad de un texto particular (o, en este caso, un programa informático) con una organización o persona concreta. Es más, pese a que el símbolo es un artificio de las leyes nacionales e internacionales, no crea una asociación entre el texto y el Estado o el Gobierno, sino entre el texto y corporaciones, editoriales, imprentas o autores particulares.

Como la cruz de Cristo, el símbolo de *copyright* comporta tanto un efecto de licencia (exclusiva, limitada o no exclusiva) como un imprimatur en las mentes de los lectores: «imprímase en la memoria» que este trabajo es de tal y tal autor y que es *propiedad* de tal y tal corporación.

Sin la alegoría de la Reforma Protestante, la única narrativa disponible para tal maldad —ya sea el comportamiento de Microsoft o de cualquier otra empresa— es que las corporaciones están «compitiendo en el mercado según las reglas del capitalismo» y, por ende, cuando los geeks condenan dicho comportamiento es por pura frustración. Si las corporaciones no violan ninguna ley, ¿por qué debería impedírseles tomar el control de esta manera? En esta narrativa no hay sitio para una *evaluación moral de la competición* —al parecer, todo vale. Afirmar que Microsoft simplemente juega según las reglas del capitalismo coloca a los demás en el cajón de los competidores o en el de los no competidores. En cambio, usar la alegoría de la Reforma Protestante dota a los geeks de un modo de conferir sentido a una distribución desigual entre poderes en competencia —entre corporaciones grandes y pequeñas y

21. *Ibid.*, pp. 20–21.

entre el poder de mercado y los detalles del control. Dicha alegoría ofrece una imagen alternativa con la que juzgar las acciones técnica y legalmente específicas que realizan corporaciones e individuos, e imaginar formas de acción justificada que les den respuesta.

Sin una alegoría así, los *geeks* que se oponen a Microsoft se ven generalmente obligados a asumir una posición anticapitalista o a adoptar la postura de que todos los estándares deberían generarse y controlarse públicamente, algo que pocos desean defender. De hecho, muchos *geeks* preferirían un tipo completamente diferente de imaginario —tal vez un público recursivo. En lugar de una infraestructura sujeta a distribuciones desiguales del poder y atravesada por «malvadas» distorsiones de control técnico, existe la posibilidad, según los *geeks*, de un terreno de juego «auto-nivelado», un sistema autotélico de reglas, tanto técnicas como legales, por el que se espera que todos los participantes compitan en igualdad. Aunque siga siendo un imaginario, la alegoría de la Reforma Protestante dota de sentido (confiere orden) a la economía política del mundo contemporáneo de las tecnologías de la información y permite a los *geeks* concebir sus intereses y acciones de acuerdo con una narrativa de reforma, más que de revolución o de sumisión. En la Reforma, lo primordial no era cuestionar la interpretación o veracidad de las enseñanzas cristianas: no se trataba de una revolución doctrinal, sino burocrática. Del mismo modo, los *geeks* no cuestionan la virtud de las redes, el software o los protocolos y estándares, ni tampoco se oponen al capitalismo o la propiedad intelectual, sino que desean mantener un espacio para la crítica y la evaluación moral del capitalismo y la competencia contemporáneos.

Polímatas y transhumanistas

Los pasados usables articulan la conjunción entre «sistemas operativos y sistemas sociales», dando forma narrativa a las imaginaciones de orden moral y técnico. Afirmar que no existen relatos preestablecidos sobre la economía política contemporánea significa simplemente que las explicaciones coloquiales convencionales acerca del estado del mundo moderno no hacen justicia a los tipos de imaginaciones de orden moral y técnico que los *geeks* poseen en virtud de sus prácticas. Los *geeks* habitan y construyen un tipo de mundo —un mundo de programas, redes e infraestructuras—, pero a menudo se ven confrontados con historias y explicaciones que simplemente no concuerdan con su experiencia, ya

sea en prensa y televisión, o entre su círculo de amigos no *geeks*. Para muchos *geeks*, el proselitismo aparece como una senda obvia: ¿por qué no ayudar a amigos y vecinos a entender el mundo oculto de las redes y el software si, tal es su certeza, dicho mundo llegará a estructurar también sus vidas?

Los *geeks* se reúnen a través de Internet y, como un pueblo autónomo, poseen ideas incipientes de independencia, contrato y constitución por las que desean gobernarse a sí mismos y resistir la autoridad ajena.²² Filosofías políticas convencionales como el liberal-libertarismo, el anarquismo o el (neo)liberalismo captan solo parcialmente estos imaginarios sociales precisamente porque no hacen referencia a los sistemas operativos, el software y las redes en las que los *geeks* viven y trabajan, y que a su vez tratan de construir y extender.

Los *geeks* viven de modos específicos en el tiempo y el espacio. No son meros usuarios de tecnología, ni una «sociedad red», ni una «comunidad virtual», sino actores encarnados e imaginativos cuya afinidad mutua viene posibilitada por las herramientas y tecnologías con las que mantienen conexiones afectivas tan profundas. Viven inmersos en *esta-red-concreta*, una forma históricamente única arraigada en determinadas especificidades sociales, morales, nacionales e históricas que sin embargo remiten a generalidades como el progreso, la tecnología, la infraestructura y la libertad. Con todo, la postura de los *geeks* acerca de dichas generalidades no es para nada unánime, y a menudo tienen formas de pensar al respecto altamente sofisticadas.

El artículo de Foucault «¿Qué es la Ilustración?» capta parte de esta problemática. Para Foucault, la visión kantiana de la modernidad era un intento de repensar la correspondencia entre el paso del tiempo histórico y la relación subjetiva que los individuos tienen con él.

Teniendo como referencia el texto de Kant, me pregunto si no se puede considerar la modernidad como una actitud más que como un periodo de la historia. Y por «actitud» quiero decir un modo de relación con respecto a la actualidad; una elección voluntaria que hacen algunos; en fin, una manera de pensar y de sentir, una manera también de actuar y de conducirse que, simultáneamente, marca

22. A un nivel populista, ello fue captado por John Perry Barlow en su «Declaración de Independencia del Ciberespacio»: <http://homes.eff.org/~barlow/Declaration-Final.html> [disponible en castellano en http://biblioweb.sindominio.net/telematica/manif_barlow.html]. (Última consulta: 18 de abril de 2018).

una pertenencia y se presenta como una tarea. Un poco, sin duda, como eso que los Griegos llamaban un «*ethos*». Y consecuentemente, más que querer distinguir el «periodo moderno» de las épocas «pre» o «post-moderna», creo que sería mejor averiguar cómo la actitud de la modernidad, desde que se formó, se ha encontrado en lucha con actitudes de «contra-modernidad».²³

Al reflexionar sobre cómo los *geeks* conciben el presente, el pasado y el futuro, planteo la pregunta de si son «modernos» en este sentido. Foucault se sirve de Baudelaire como contrapunto para explicar en qué consiste la actitud de la modernidad: «Para [Baudelaire,] ser moderno [...] consiste en apoderarse de algo eterno que no está más allá del instante presente, ni detrás de él, sino en él».²⁴ Sugiere así que la interpretación de la modernidad por parte de Baudelaire se refiere a «la actitud que permite captar lo que hay de ‘heroico’ en el momento presente. La modernidad no es un fenómeno de sensibilidad hacia el presente fugitivo, es una voluntad de ‘heroizar’ el presente».²⁵ «Heroico» significa aquí una suerte de redescipción de los eventos supuestamente fugaces del presente en términos que evoquen el carácter universal o eterno que los anima. En la lectura foucaultiana de Baudelaire, dicha actitud es incommensurable con la que ve en el paso del tiempo presente al futuro una versión del progreso autónomo (ya sea en forma de espíritu absoluto o de degeneración decadente), y la enseña que usa para ello es «No tenéis derecho a despreciar el presente». Ser moderno supone afrontar el presente como un problema que puede transformarse por la acción humana, y no como el resultado inevitable de procesos más allá del control humano individual o colectivo, esto es, con las «actitudes de la contra-modernidad». Cuando los *geeks* cuentan historias del pasado para darle sentido al futuro, a menudo lo hacen precisamente para «heroizar» el presente en este sentido —aunque no en todos los casos. Dentro del espectro que va de los polímatas a los transhumanistas caben actitudes tanto de modernidad como de contra-modernidad.

Las cuestiones que suscito aquí son también las de la *política* en sentido clásico: ¿Están los *geeks* vinculados por una actitud hacia el presente a la que conciernen asuntos como la relación entre lo público,

23. Foucault, «What Is Enlightenment», pp. 309–310 [ed. cast.: «¿Qué es la Ilustración?», trad. por Antonio Campillo. *Daimon. Revista de Filosofía*, nº 7, 1993, p. 11].

24. *Ibid.*, p. 310 [«¿Qué es la Ilustración?», p. 11].

25. *Idem*.

lo privado y lo social (en la estela de Hannah Arendt), la relación entre la economía y la libertad (en la línea de John Stuart Mill y John Dewey) o las posibilidades de organización racional de la sociedad mediante la aplicación del conocimiento científico (siguiendo a Friedrich Hayek o a Michel Foucault)? ¿Son los *geeks* «ilustrados»? ¿Son racionalistas de la Ilustración? ¿Qué podría significar esto tanto tiempo después de la Ilustración y sus vigorosas críticas de amplio alcance? ¿Cómo se relaciona su ilustración con los compromisos técnicos e infraestructurales que han contraído? En otras palabras, ¿qué hace a la Ilustración nuevamente necesaria *ahora*, en el medio [«*milieu*»] de Internet, el software libre y los públicos recursivos? ¿Qué tipo de relaciones aparecen cuando nos preguntamos cómo los *geeks* relacionan su propia apreciación consciente de la historia y la política de su tiempo con sus prácticas y aspiraciones cotidianas? ¿Desprecian los *geeks* el presente?

Los polímatas y los transhumanistas difieren al hablar de conceptos como tecnología, infraestructura, redes y software, y tienen ideas distintas acerca de su temporalidad y relación con el progreso y la libertad. Algunos *geeks* ven la tecnología como una forma de intervención en un campo constituido de organizaciones, dinero, política y personas; otros la ven como una fuerza autónoma compuesta de humanos y fuerzas impersonales de evolución y complejidad. Diferentes *geeks* hablan del papel de la tecnología y de su relación con el presente y el futuro de modos diferentes, y su visión de esta relación está ligada a sus propias y ricas interpretaciones del complejo entorno técnico y político en que viven y trabajan.

Polímatas

La polímatía es «dilettantismo declarado», no inteligencia extrema. Deriva de una curiosidad que parece enganchar a un considerable número de personas que pasan su tiempo en Internet, así como de la necesidad básica de ser capaz de evaluar e incorporar campos de conocimiento a menudo dispares con el fin de construir software que funcione. La polímatía surge inevitablemente en el contexto de grandes proyectos de software y redes: es una criatura nacida de restricciones, un proceso de arranque autosostenido por el complejo sedimento de tecnologías, negocios, personas, dinero y planes. Esta idea también podría formularse de modo negativo: un mal diseño de software suele ser resultado de un insuficiente dilettantismo declarado. Los polímatas *deben* poseer

conocimientos muy amplios y profundos con vistas a intervenir en una distribución dada preexistente de máquinas, personas, prácticas y lugares. *Deben* poseer una idea muy detallada del presente, y del proyecto del presente, con vistas a imaginar cómo podría el futuro ser diferente.

Mi polímata favorito es Sean Doyle, quien construyó las primeras versiones de una aplicación que conforma la piedra angular de la compañía de gestión de imágenes radiológicas Amicas. Con tal fin, Sean aprendió lo siguiente: lenguaje Java, para programarla; matemáticas de ondículas, para codificar las imágenes; el flujo de trabajo de los radiólogos en el hospital y el modo en que realizan diagnósticos a través de imágenes, para hacer usable la interfaz; varias bases de datos incompatibles y el lenguaje de base de datos SQL, para construir el archivo y el repositorio; y manual tras manual de estándares técnicos, de los que el más voluminoso y aterrador fue el estándar DICOM (*Digital Imaging and Communication*, Escaneo Digital y Comunicación) para imágenes radiológicas. Sean también leyó las revistas *Science* y *Nature* con regularidad, buscando inspiración para el diseño de la interfaz; leyó libros y artículos sobre escaneo digital de cosas muy pequeñas (rodillas de mosquito), cosas muy grandes (galaxias y polvo interestelar), cosas muy antiguas (fósiles) y cosas muy bellas (patrones de alas de mariposa como una función de las vías de desarrollo). Sean también me dio a conocer la comida tibetana, las películas de Jan Svankmeyer, el software de código abierto, la Cladística y a la Paleoherpetología, las políticas de tierra quemada de Disney respecto a la cultura, y otras muchas cosas fascinantes.

Sean es claramente un personaje inusual, pero no tanto. Con los años he conocido a mucha gente con conocimientos igualmente amplios y profundos (aunque raras veces con la humildad de Sean, lo cual le distingue). La polimatía es un riesgo laboral para los *geeks*. Para un buen programador, arquitecto de software o arquitecto de información simplemente no tiene sentido especializarse en el código. La especialización no es vista como un fin en sí mismo, sino más bien como una suerte de prerequisito técnico para luego poder llevar a cabo otro trabajo —el trabajo real. El trabajo real es el *diseño*, el proceso de insertar software usable en una amalgama completamente ignota de personas, organizaciones, máquinas y prácticas. El diseño es un trabajo duro, mientras que la parte técnica —como elegir el lenguaje correcto, adherirse a un estándar o encontrar un fragmento de código ya existente para insertarlo en algún lugar— no lo es.

Para los *geeks* de Internet y los arquitectos de software es posible pensar así debido en parte a que muchos de los problemas técnicos a los que se enfrentan están extremadamente bien definidos y resultan muy fáciles de abordar mediante una rápida búsqueda y descarga. Es fácil ser un dilettante declarado en la era de las listas de distribución, los grupos de noticias y las publicaciones científicas en línea. Yo mismo he aprendido una gran variedad de prácticas técnicas de esta forma, pero no he *diseñado* ninguna tecnología digna de mención.

El socio de Sean en Amicas, Adrian Gropper, también da el perfil de polímata, aunque no es programador. Adrian, físico y graduado en el programa de ingeniería del MIT, podría ser denominado como un «polímata de alto funcionamiento». Escanea el horizonte de hallazgos técnicos y científicos buscando formas de incorporarlos a su visión de la tecnología médica como intervención. Sean habla burlonamente de ello como «*delirios*», pero ambos coinciden en que Amicas no iría a ninguna parte sin ellos. Adrian y Sean ejemplifican cómo los significados de tecnología, intervención, diseño e infraestructura son interpretados por los polímatas como una forma particular de intervención pragmática, un progreso logrado a través de la reforma deliberada y gradual de sistemas ya existentes. Como Adrian comenta:

Creo firmemente que a la larga la única forma en la que puedes ahorrar dinero y mejorar la asistencia sanitaria es añadir tecnología. Creo en ello más firmemente de lo que creo, por ejemplo, en que si la gente inventa mejores pesticidas podrá cultivar más arroz, y en que constituye un bien universal apoyar que más gente se dedique a ello. Tengo algunas dudas sobre si considerar «de los buenos» a la gente que lleva a cabo ingeniería genética de cultivos y pesticidas. Sí creo, sin embargo, que la asistencia sanitaria es diferente en tanto que a la larga puedes impactar tanto en sus costes como en su calidad mediante la adición de tecnología. Y puedes llamarlo una creencia religiosa siquieres, no es racional. Pero supongo que lo que pretendo decir es que la asistencia sanitaria tradicional no basada en tecnología se ha quedado bastante desfasada.²⁶

En esta conversación, lo «tecnológico» está restringido a las cosas novedosas que pueden hacer la asistencia sanitaria menos costosa

26. Adrian Gropper, entrevista con el autor, 28 de noviembre de 1998.

(o sea, que pueden reducir costes, no recortarlos), aliviar el sufrimiento o prolongar la vida. Ciertos tipos de intervención tecnológica son superfluos o incluso inútiles, y Adrian no puede identificar completamente esta «clase» —no es «tecnología» en general, pero incluye algunos tipos de cosas que son tecnológicas. Lo que es más importante es que la tecnología no resuelve nada por sí misma, ni evita los problemas políticos del racionamiento de la asistencia sanitaria:

Ahora, sin embargo, encuentras este otro problema, que es que la forma de racionar la asistencia sanitaria es a través del miedo al dolor, dolor financiero hasta cierto punto, pero sobre todo dolor físico; así que si tienes una tecnología que, por ejemplo, hace relativamente indoloro un tratamiento... Supongo, por decirlo sin rodeos, que en la mayoría de los casos es más barato dejar a la gente morir, y eso es simplemente innegable. Así que lo que encuentro interesante de todo esto es que la mayoría de personas responsables de las cuestiones políticas de la gestión de recursos sanitarios no quiere debatir esto, nadie quiere hablar de ello, los doctores no quieren hablar de ello porque es tan deprimente hablar sobre el valor de... Y ellos realmente no tienen un mandato para hablar sobre tecnología.²⁷

El rol autodefinido de Adrian en este ámbito es el de un físico no practicante que también es ingeniero y empresario —por tanto, su polimatía ha surgido de intentar ejercer de traductor entre médicos, ingenieros y empresarios. Su objetivo es doble: primero, crear tecnologías que ahorren dinero y mejoren la dispensación de la asistencia sanitaria (y el gran sueño de la telemedicina concierne precisamente a este objetivo: la reasignación de los activos más valiosos, esto es, los individuos y sus experiencias); segundo, elevar el nivel de debate en el mundo médico-empresarial sobre el papel de la tecnología en la gestión de los recursos sanitarios. Aquí la polimatía es esencial, pues la doble misión de Adrian requiere comprender el lenguaje y las vidas de al menos tres grupos distintos que trabajan codo con codo en la asistencia sanitaria: ingenieros y arquitectos de software, médicos y enfermeras, y empresarios.

La tecnología tiene dos significados diferentes según los dos objetivos de Adrian: en el primer caso, la *tecnología* se refiere a la intervención por medio de nuevas tecnologías (desde software a materiales, desde electrónica a productos farmacéuticos) en situaciones sanitarias

27. Adrian Gropper, entrevista con el autor, 28 de noviembre de 1998.

específicas que pueden entrañar altos costes o acceso limitado a la atención médica. En algunas ocasiones se da una asignación de tecnología, en otras es la tecnología la que realiza la asignación. El objetivo de Adrian es combinar su conocimiento sobre tecnología de última generación —en particular, la tecnología de Internet— con una situación específica de la asistencia sanitaria para de ese modo efectuar una reorganización de prácticas, personas, herramientas e información. La herramienta creada por Amicas se distinguía por su uso inteligente de la compresión de datos, los estándares de Internet y los medios baratos de almacenamiento para competir con sistemas «heredados»²⁸ y «pre-configurados» [«turnkey»] mucho más grandes, caros y afianzados. El hecho de si Amicas inventó algo «nuevo» es menos interesante que la naturaleza de esta intervención en un medio existente. Esta intervención es lo que Adrian llama «tecnología». Para su empresa, la tecnología relevante —la intervención importante— era Internet, concebida como una herramienta para cambiar la naturaleza de la organización de la asistencia sanitaria. Su objetivo era reemplazar la infraestructura del departamento de radiología del hospital (y potencialmente de los demás departamentos también) con Internet. Amicas era capaz de desafiar y reformar las prácticas de poderosas entidades establecidas, desde la administración de grandes hospitales a sus compañeros de cama corporativos, como HBOC, Agfa, Siemens y GE.

En cuanto a la elevación del nivel de debate, sin embargo, *tecnología* se refiere a un tipo de argumento político-retórico: la tecnología no salva el mundo (ni tampoco lo destruye), solo salva vidas —y ello solo cuando uno toma decisiones específicas sobre su asignación. O, dicho de otro modo, el medio es la tecnología, pero es en los fines donde se encuentra la acción. De ahí que la grandilocuencia en torno a la tecnología de la información aplicada a la salud horrorice a Adrian: las promesas preceden a las tecnologías, y dichas promesas sugieren que los medios pueden reemplazar los fines. Las grandes corporaciones que prometen «tecnología» pero no ofrecen intervenciones acabadas (según el primer significado de *tecnología* para Adrian) cuyo efecto en la reducción de costes o en la mejora de la gestión esté demostrado concretamente son simplemente un desperdicio de recursos. Dichas compañías son doblemente frustrantes porque usan la «tecnología» como anteojeras que permiten a la gente *no* pensar sobre los arduos

28. Véase nota 28.

problemas (los fines) de la distribución, la equidad, la gestión y la organización; es decir, tratan la «tecnología» (los medios) como si fuera una solución *en sí misma*.

Adrian analiza rutinariamente los usos retóricos y prácticos de la tecnología sanitaria con este tipo de sutileza; claramente, tal sutileza de pensamiento es rara, y esto distingue a Adrian como alguien que entiende que esa intervención en, y transformación de, las organizaciones y los estilos de pensamiento modernos ha de darse mediante una reforma —mediante el uso inteligente de la tecnología por quienes la comprenden íntimamente—, y no mediante una revolución. La reforma a través de la innovación técnica se opone aquí al control a través de la consolidación del dinero y el poder.

En mis observaciones, Adrian siempre resaltaba la importancia de hacer la tecnología —las herramientas de software y el sistema de archivo de imágenes— fácilmente accesible, fácilmente demostrable a los clientes. Cuando hablaba con compradores de los hospitales, a menudo decía algo así: «Puedo mostrarle el software, darle el precio y demostrarle el problema que resolverá». Por contra, un despliegue de comerciales (habitualmente llamados consultores) de corporaciones enormes probablemente diría algo más parecido a esto: «Su hospital necesita más tecnología, nuestra corporación es grande y estable —de-nos tanto dinero y resolveremos su problema». Para Adrian, la decisión de «ir de la mano», como él dice, de una corporación confortablemente grande era irracional si en su lugar el hospital pudiera adquirir una tecnología específica que hiciera una cosa específica por un precio ajustado.

Las reflexiones sobre la tecnología de Adrian lo son también sobre la naturaleza del progreso. El progreso es una intervención limitada que se estructura por objetivos que no fija la propia tecnología, por más que la actividad empresarial se centre específicamente en encontrar nuevos usos y nuevas ideas para nuevas tecnologías. Pero los debates sobre los costes de la dispensación de asistencia sanitaria —problema que Adrian considera susceptible de ciertos tipos de soluciones técnicas— se estructuran, en cambio, como si la tecnología fuera irrelevante a la naturaleza de los fines. Ante ello Adrian se resiste: «Creo firmemente que a la larga la única forma en la que puedes ahorrar dinero y mejorar la asistencia sanitaria es añadir tecnología».

Sean expresa una frustración similar ante la homogeneización del concepto de «tecnología», especialmente cuando se usa para sugerir, por ejemplo, que los hospitales «van a la zaga» de otras industrias en

cuanto a la informatización, una queja habitualmente planteada con objeto de instigar inversiones o explicar fallos. Sean se opone de entrada a dicha noción tan homogénea de «tecnológico»:

La verdad es que no tengo ni idea de qué significa eso de «ir a la zaga». Porque ciertamente en muchos aspectos del procesamiento de imagen o en algunos aparatos de tecnología punta, los hospitalares probablemente vayan muy por delante. Y si se refieren a lo que tiene el personal en sus escritorios, desde que en torno a 1984 llegó al Hospital General de Massachusetts siempre ha habido un ordenador en el escritorio de casi todo el mundo. [...] Al parecer, la mayoría de hospitales donde he estado tienen un fuerte compromiso con las redes informáticas y la automatización, etc. [...] No sé cuál es la situación en gran parte de las fábricas —puede que tengan consolas informáticas, pero estas son de otra especie. Es probable que las granjas sí que vayan *realmente* a la zaga, y de los parques de atracciones mejor *ni hablar*. En cierto modo, los hospitales son pequeñas comunidades muy complicadas, por lo que decir que esta entidad en su conjunto va a la zaga no tiene mucho sentido.²⁹

Asimismo Sean se opone a la noción de que tal rezago ocasione fallos causados por la tecnología, en lugar de por factores como la incompetencia o la mala gestión. De hecho, podría ser justo afirmar que, para los polímatas, a veces la tecnología realmente se disuelve. Sus fronteras no resultan fáciles de trazar, como tampoco sus usos, ni sus supuestas «consecuencias imprevistas». Por un lado hay reglas, regulaciones, protocolos, normas y formas de conducta; por el otro hay estructuras organizativas, lógicas y planes de negocio, habilidades humanas y otras máquinas. Este medio complejo requiere una reforma desde dentro: no puede reemplazarse en su conjunto ni adelantar a otras industrias en cuanto a informatización, pues toda intervención es siempre local y estratégica y entraña una relación más compleja con el proyecto del presente que la de simplemente «ir a la zaga» o «ir por delante».

La polimatía —en tanto es una polimatía derivada de la experiencia vivida de la necesidad de múltiples saberes expertos para resolver una situación— vuelve a la gente pragmática. La tecnología nunca es sim-

29. Sean Doyle, entrevista con el autor, 30 de marzo de 1999.

plemente la solución a un problema, sino parte de una serie de factores. Los polímatas, a diferencia de los tecnófobos, saben distinguir cuándo la tecnología importa y cuándo no. Los polímatas poseen una perspectiva de la tecnología muy mundana: no hay en ella ni misterio ni promesa, tan solo ingenio y error humanos. De esta manera, podría describirse mejor a los polímatas como feyerabendianos que como pragmáticos (y, de hecho, Sean resultó ser un ávido lector de Feyerabend). Los polímatas estiman que no hay un único método por el que la tecnología efectúe su magia: esta depende en gran medida de reglas, de acciones pautadas y de la observación de factores contingentes y contextuales. La intervención en este campo ya instituido de personas, máquinas, herramientas, deseos y creencias requiere un tipo de genialidad científico-técnica, pero difícilmente es singular o incluso autónoma. Esta versión del pragmatismo es, como Feyerabend la designa en ocasiones, simplemente un tipo de *sensibilidad*: de estándares, de reglas, de historia, de posibilidad.³⁰ De ahí que los polímatas no se permitan a sí mismos despreciar el presente, sino que insisten en reflexionar e intervenir sobre él.

Sean y Adrian son personas declaradamente científicas y técnicas y, como Feyerabend, asumen que sus interlocutores creen en la buena ciencia y en los beneficios del progreso. Tienen escasa paciencia con los ludditas, la *New Age*, la intolerancia religiosa o con cualquier otra actitud no derivada de la Ilustración. Y no desprecian el presente porque tienen un sentido bien desarrollado de cuán provisionales son las convenciones de la tecnología y los negocios modernos. Muy pocas cosas son sagradas para ellos y las reglas, cuando existen, son frágiles. Frente a la inmodestia de romperlas sin sentido, la innovación suele verse en sí misma como una manera de transformar una serie de reglas o prácticas aceptadas para otros fines. El progreso es una intervención limitada.³¹

30. Feyerabend, *Against Method*, pp. 215-225 [ed. cast.: *Tratado contra el método*].

31. Uno de los modos en que Adrian discute sobre innovación es mediante el argumento de *The Innovator's dilemma* [ed. cast.: *El dilema de los innovadores*] del profesor de la Escuela de Negocios de Harvard Clayton Christensen. Este libro describe el enfrentamiento entre tecnologías «sostenibles y disruptivas» como una cuestión que tiene menos que ver con cómo funcionan o de qué están hechas las tecnologías, y más con cómo se mide su éxito y rendimiento. Véase Adrian Gropper, «The Internet as a Disruptive Technology,» *Imaging Economics*, diciembre de 2001, <http://www.imagineconomics.com/library/200112-10.asp> (enlace no accesible en la última consulta de 18 de abril de 2018).

Estimo pertinente añadir que la obra de Christensen ha sido objeto de un polémico debate que puede remontarse a 2014, cuando la historiadora Jill Lepore publicó en *The New Yorker* un exhaustivo artículo crítico («*The disruption machine. What the gospel of innovation*

Qué sarcástico y perturbador, pues, constatar que la compañía de Sean y Adrian finalmente se convertiría en aquello que querían reformar cuando fundaron Amicas. Al margen de la intervención limitada, ciertos tipos de impulso parecen irresistibles: la demanda de inversión y las rondas de financiación, la necesidad de «gestión profesional» y la inercia de las prácticas de adquisición de equipos ya racionalizadas y altamente conservadoras del sector sanitario. Para Sean y Adrian, Amicas se convirtió en un fracaso a causa de su éxito. No obstante, mantienen su determinación de polímatas modernos: no desprecian el presente. Como Kant describe en «¿Qué es la ilustración?», el deber de los ciudadanos se divide entre lo público y lo privado: por un lado, el deber de cumplir las responsabilidades de despacho; por el otro, el deber de ofrecer crítica donde la crítica es debida, como un «académico» ante una audiencia de lectores. La iniciativa de Sean y Adrian, en forma de compañía tecnológica emergente, bien podría entenderse como la expresión del deber académico de ofrecer crítica, y ello a través de la creación de un tipo particular de crítica técnica a un sistema sanitario existente (y a su juicio) éticamente sospechoso. La mezcla de capital privado, instituciones públicas, ciudadanía y tecnología es, sin embargo, algo que Kant no podía haber previsto —y las pesquisas técnicas de Sean y Adrian deben entenderse como algo más: un tipo de moderno deber cívico al servicio de la libertad y en respuesta a las particularidades de la vida técnica contemporánea.³²

Transhumanistas

La polimatía nace del compromiso práctico y pragmático con situaciones específicas, y en cierta manera viene demandado por dichas exigencias. En el lado opuesto a la polimatía, y más inclinadas a un interés por el todo, por la totalidad y lo universal, se encuentran las actitudes a las que me refiero con la etiqueta de transhumanismo, que alude al modo de creer en la Línea del Tiempo del Progreso Técnico.³³

gets wrong», véase: <https://www.newyorker.com/magazine/2014/06/23/the-disruption-machine>), respondido días después por el propio Christensen vía Bloomberg (véase: <https://www.bloomberg.com/news/articles/2014-06-20/clayton-christensen-responds-to-new-yorker-take-down-of-disruptive-innovation>). [N. del E.].

32. Sobre tipos de deber cívico, véase Fortun y Fortun, «Scientific Imaginaries and Ethical Plateaus in Contemporary U.S. Toxicology».

33. Existe, de hecho, un grupo muy específico de personas llamadas transhumanistas, sobre las que apenas hablaré. Si invoco esta etiqueta es porque creo que ciertos aspectos del transhumanismo atraviesan el espectro de ingenieros, científicos y geeks.

El transhumanismo, tanto el movimiento como la filosofía, se centra en el poder de la tecnología para trascender las limitaciones del cuerpo humano tal y como ha evolucionado en la actualidad. Sus partidarios creen —pero este es ya un término erróneo— en la posibilidad de descargar conciencia en placas de silicio, de la suspensión criobiológica, del inminente surgimiento de inteligencia artificial muy poderosa y de otras diversas formas de aumento técnico del cuerpo humano con el propósito de alcanzar la inmortalidad —o al menos, mucho más tiempo de vida.³⁴

Bajo esta etiqueta podrían incluirse razonablemente diversos grupos. Así encontramos a los más ardientes difusores de la visión, los Extropianos; a un amplio conjunto de personas que se autodenominan transhumanistas; a una subclase franco-canadiense, los Raelianos, que son más un culto de adoración alienígena que un grupo estrictamente científico y son vehementemente denunciados por los dos anteriores; asimismo a la diversidad de cosmólogos e ingenieros que no se consideran formalmente transhumanistas, pero de cuyas creencias participan de una u otra forma: Stephen Hawking, Frank Tipler y John Barrow (famosos por su principio cosmológico antrópico), Hans Moravic, Ray Kurzweil, Danny Hillis, y todos aquellos que abrazan las ciencias cognitivas, la filosofía de la inteligencia artificial, la filosofía de la mente, la filosofía de la ciencia y demás disciplinas análogas.

Históricamente hablando, la línea de descendencia es difusa. Teilhard de Chardin es ampliamente influyente, a veces de forma admitida, a veces no (dependiendo de la cantidad de misticismo permitido). Un punto de partida más generalmente reconocido es el artículo «Transhumanismo» de Julian Huxley en *New Bottles for New Wine*.³⁵ El «transhumanismo» de Huxley, como el de Teilhard, emana un extraño olor a Nietzsche, aunque tiende mucho más hacia el surgimiento evolutivo del superhombre que hacia el sentido más propiamente moral que Nietzsche le dio. A partir de Huxley, la noción de transhumanismo se identifica demasiado fácilmente con la eugenesia y se ha convertido en una de la serie de corrientes subculturales de mediados de siglo que

34. Véase la World Transhumanist Association, <http://transhumanism.org/> (*última consulta: 18 de abril de 2018*) o el Extropy Institute, <http://www.extropy.org/> (*última consulta: 18 de abril de 2018*). Véase también Doyle, Wetwares, y Battaglia, «For Those Who Are Not Afraid of the Future», para una mirada oblicua al respecto.

35. Huxley, *New Bottles for New Wine*, pp. 13-18.

halla su máxima expresión en lugares pequeños y no convencionales, desde los liberal-libertarios a Esalen.³⁶

Para muchos observadores, los transhumanistas son unos fanáticos flanqueados de un lado por los abducidos por extraterrestres y del otro por los objetivistas que citan sin parar a Ayn Rand. Sin embargo, como tantos fanáticos, ellos meramente representan de forma cristalina actitudes que parecen permear los debates de una manera más amplia, ya sea como creencias profesadas o como creencias atribuidas. Así, aunque probablemente sea un anatema para la mayoría de la gente, el transhumanismo en realidad revela una actitud muy concreta hacia la innovación técnica, la intervención técnica y la vida política que está muy extendida entre los individuos técnicamente versados. Es una creencia que tiene todo que ver asimismo con la línea del tiempo del progreso y el papel de la tecnología en ella.

La mejor forma de entender la interpretación transhumanista del progreso tecnológico es a través del concepto, unas veces serio y otras lúdico, de «singularidad», popularizado por el escritor de ciencia-ficción y matemático Vernor Vinge.³⁷ La «singularidad» es el punto en el que la velocidad del progreso técnico es más rápida que la comprensión humana de tal progreso (y, por implicación, que el control humano sobre el mismo). Es una especie de parábola del hombre de las cavernas, quizás retratada del modo más bello por la película de Stanley Kubrick *2001: Una Odisea del espacio* (particularmente, en el salto de montaje del comienzo de la película que transforma un hueso arrojado en una estación espacial giratoria, recapitulando en dos breves segundos la notable aventura de la tecnología que de otro modo parecería una película sin fin).

En la figura 1 (pág. siguiente), en la parte izquierda de la línea del tiempo, aparece la historia o, mejor dicho, una sucesión de invenciones tecnológicas (en la que está implícito que cada invención allana el camino para las siguientes) tan espaciadas que producen una curva logarítmica que puede parecerse mucho a las curvas de población apocalípticas que comenzaron a aparecer en los 60. Cada invención está asociada con un nombre o a veces una nación. Del borde del gráfico a la

36. El informático Bill Joy escribió un largo artículo en *Wired* advirtiendo de los resultados de la investigación realizada sin salvaguardas éticas y de los peligros de la eugenésia en el pasado, «Why the Future Doesn't Need Us», *Wired*, 8.4, abril de 2000: <http://www.wired.com/wired/archive/8.04/joy.html> (*última consulta: 18 de abril de 2018*).

37. Vinge, «The Coming Technological Singularity».

derecha está el futuro: la historia pasa aquí de una serie de invenciones a una tecnología autónoma autoinventora no asociada con inventores individuales sino con una complejo sistema de adaptación evolutiva que incluye tanto formas tecnológicas como biológicas. Es un futuro en el que los «humanos» ya no son necesarios para el progreso de la ciencia y la tecnología: así pues, tecnología-como-extensión-humana a la izquierda, una inteligencia técnica autónoma parecida a un Borg a la derecha. La operación fundamental para construir la «singularidad» es la «extrapolación razonada» familiar a los escritores de «ciencia-ficción dura» o a los futuristas. Se toma la tecnología presente como la condición inicial para futuras posibilidades y se efectúa una extrapolación basada en las pruebas (caprichosamente manejadas) de la aceleración y el cambio tecnológico del pasado.

Figura 1

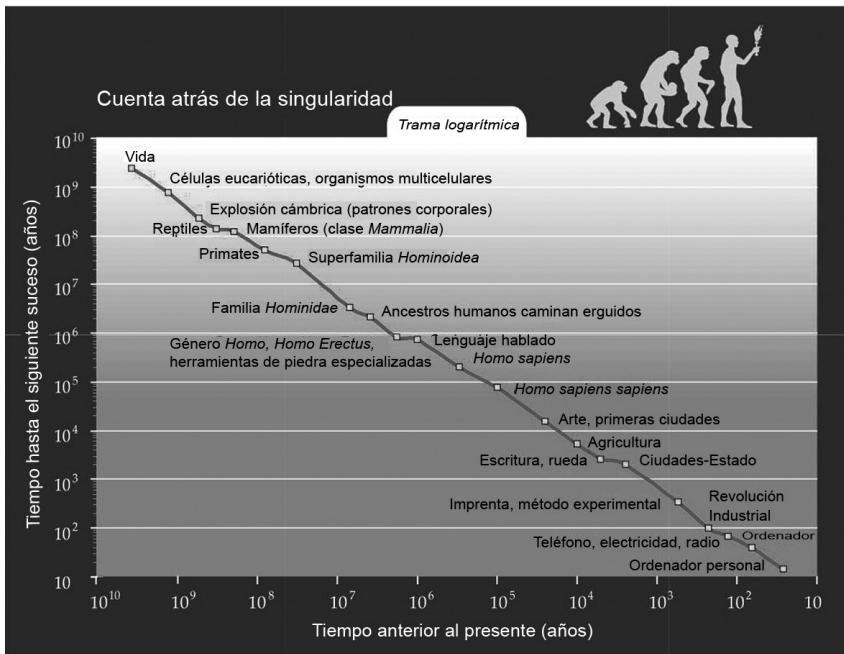


Ilustración © 2005 Ray Kurzweil. Modificaciones © 2007 C. Kelty.

Obra original sujet a licencia Creative Commons Atribución:
<http://en.wikipedia.org/wiki/Image:PPTCountdowntoSingularityLog.jpg>

La posición de observadores es siempre un poco incierta, puesto que de manera natural nos proyectamos al punto más alto de esta curva (o al más bajo, dependiendo de nuestra orientación), pero una implicación resulta clara: que la función o la necesidad de la reflexión humana sobre el presente desaparecerá al mismo tiempo que los humanos, convirtiendo la ilustración en un paso pintoresco, pero necesario, en la ruta hacia la inmortalidad superracional y transhumana.

De forma extraña, la noción de que el progreso técnico posee *aceleración* parece preceder a cualquier cuestionamiento acerca de qué podría significar de entrada la *velocidad* del progreso; se presume que la tecnología existe en un tiempo absoluto —desde el Big Bang hasta la muerte térmica del universo— y sin relación alguna con la vida o la conciencia humanas. La singularidad se describe siempre desde el punto de vista de un dios que no es Dios. El hecho de la aceleración tecnológica se trata generalmente como el asunto más obvio del mundo, reforzado por la constante cantinela mediática acerca del increíble ritmo de cambio de la sociedad contemporánea.

¿Por qué es importante la singularidad? Porque implica siempre que el hecho absoluto de la aceleración técnica —esa mirada intencionada al futuro— debería ordenar los tipos de intervención que se dan en el presente. No es una espera muda o una certidumbre escatológica lo que rige esta actitud, sino más bien un modo de conciencia histórica que privilegia la inevitabilidad del progreso tecnológico sobre la inevitabilidad del poder humano. Solo asomándose al futuro es posible manipular el presente de un modo que sea ampliamente significativo, una actitud que podría expresarse así: «Los que no aprendan del futuro están condenados a sufrir en él». Puesto que se trata de una filosofía basada en el éxito de la racionalidad y el ingenio humanos, ambos siguen siendo claramente esenciales en el futuro, si bien llevan a un tipo de estado poshumano en constante devenir tecnológico que resulta inconcebible para la mente humana individual —y que solo puede ser aprehendido por una inteligencia trascendental que no es Dios.

He aquí una descripción adecuada de algunas vertientes del transhumanismo, y si hago hincapié en ellas es para caracterizar las clases de actitudes hacia la tecnología-como-intervención y las ideas de orden moral y técnico que los *geeks* pueden poner de manifiesto. Situados en el extremo de la polimatía, los *geeks* están demasiado cerca de la máquina para ver un panorama general o para reflexionar sobre cuestiones filosóficas imponderables; en el extremo transhumano, por contraste, se da

una constante reestimación de los detalles arcanos del cambio técnico cotidiano con respecto a una visión de la totalidad —una visión de la evolución de la tecnología y su relación con los humanos que (por el momento) deben crearla y tratar de canalizarla.

Mi transhumanista favorito es Eugen Leitl (que de hecho es un auténtico transhumanista y ha sido vicepresidente de la Asociación Mundial Transhumanista). Eugen es de origen ruso, vive en Munich y trabajó en un laboratorio de investigación criobiológica. Está muy versado en química, nanotecnología, investigación en Inteligencia Artificial, investigación en complejidad computacional y de redes, órganos artificiales, criobiología, ingeniería de materiales y ciencia-ficción. Eugen escribe, por ejemplo:

Si se considera IA codificada a mano por humanos, sí. Sin embargo, dada una cantidad considerable de recursos computacionales (~un metro cúbico de computronio), y usando una población de partida adecuada, se puede hacer coevolucionar la inteligencia mecánica en una escala temporal muy inferior a un año. Tras alcanzar un nivel humano aproximado, es potencialmente capaz de entrar en un bucle de auto-retroalimentación. Dado que incluso el computronio en grado de autoensamblaje es capaz de desarrollar un intelecto a escala humana en un volumen que va desde un terrón de azúcar hasta una naranja y a una velocidad que va de 10^4 [a] 10^6 , es fácil ver que el bucle de auto-retroalimentación posee una dinámica explosiva.

(Espero que lo anterior resulte inteligible, he estado expuesto a memes sobrecededores durante demasiado tiempo).³⁸

Eugen es también un polímata (y encima autodidacta), pero en el sentido convencional. La polimatía de Eugen es una necesidad *no vocacional*: los transhumanistas necesitan estar al día de todos los avances en tecnología y ciencia con objeto de estimar mejor qué clases de tecnologías de aumento u obsolescencia humana van apareciendo. No es para este mundo que los transhumanistas expanden su conocimiento, ni siquiera para el próximo, sino para un «este mundo» aún por llegar.

38. Eugen Leitl, correo a la lista de distribución Silk-list, 16 de mayo de 2000, <http://groups.yahoo.com/group/silk-list/message/2410>.

A Eugen y a mí nos presentaron durante los debates sobre Napster de 2001, que por entonces parecían una conflagración sin cuartel pero que él, tras haber participado en tantas guerras de *flame*³⁹ en línea, probablemente vivió como un mero parpadeo en lo que por lo demás era una pugna constante con inteligencias menos evolucionadas como la mía. Con todo, fue uno de los ejemplos más clarificadores de cómo piensan los *geeks*, y cómo difieren en sus pensamientos, acerca de la tecnología, la infraestructura, las redes y el software. El transhumanismo no tiene relación con el humanismo trasnochado.

>> De: Ramu Narayan...

>>No me gusta la

>>noción de tecnología como una fuerza imparable con voluntad propia que

>>no tiene nada que ver con las necesidades de las personas reales. [Eugen Leitl:] El comportamiento emergente a gran escala no es nada nuevo. ¿Cómo pretendes controlar el comportamiento individual de una gran población de agentes solo parcialmente racionales? Estos agentes no vienen con múltiples y convenientes ganchos modificadores de conducta (feromonas como en los insectos sociales, aunque fíjate en la sincronización de la menarquia en mujeres que comparten cuarto), y ello por una buena razón. Los pocos ganchos que tenemos (multitud, guerra, política, religión) ya han sido infamemente manipulados. De forma análoga a la apoptosis, los metaindividuos pueden funcionar utilizando procesos deletéreos para sus componentes (nosotros).⁴⁰

39. Opto por dejar como «guerras de *flame*» el original «*flame wars*» por la dificultad de verter al castellano los matices de enconamiento dialéctico del término «*flame*» aplicado a los debates en listas y foros de Internet. En todo caso, incluyo la definición del término que Eric Raymond recoge en la versión 4.4.7 del *Jargon File*:

Flame:

1. vi. Publicar un mensaje destinado a insultar o provocar.
2. vi. Hablar incesante y/o virulentamente sobre algún asunto relativamente falto de interés o con una actitud patentemente ridícula.
3. vt. Cualquiera de las acciones de los sentidos 1 o 2 dirigida con hostilidad a una persona o personas en particular.
[...]

Disponible en: <http://www.catb.org/~esr/jargon/html/F/flame.html> [N. del E.]

40. Eugen Leitl, correo a la lista de distribución Silk-list, 7 de agosto de 2000, <http://groups.yahoo.com/group/silk-list/message/2932>.

La interpretación de Eugen de lo que significa el «progreso tecnológico» es lo bastante compleja como para confundir a la mayoría de sus interlocutores. Por un motivo sorprendente, su razonamiento no es exactamente inevitable. La manera en que Leitl discute con la gente es a menudo una especie de ráfaga parlanchina de sorites coevolutivos, criptográficos y de teoría de Juegos. A medida que Eugen va apilando el razonamiento científico y transhumanista, sus interlocutores van lentamente retirándose del debate. Pero no se trata de locura, grandilocuencia o ciencia popular a medio digerir —Eugen generalmente sabe de lo que habla—, simplemente es un discurso que encaja de un modo que casi nadie más llega a captar. Eugen ve la adopción y proliferación a gran escala de tecnologías (particularmente los dispositivos moleculares autorreplicantes y los algoritmos de software evolutivos) como un peligro que trasciende toda posibilidad de control de ámbito individual o estatal. Miles de millones de decisiones individuales no «promedian» una sola voluntad, sino que en su lugar producen dinámicas complejas que están peligrosamente supeditadas a las condiciones iniciales. Al tratar la posibilidad de la singularidad, Eugen sugiere: «Podría ser literalmente un proyecto de feria de ciencias (lo que causa la singularidad)». Si la interpretación de Francis Bacon de la relación entre Hombre y Naturaleza era la de dueño y poseedor, la de Eugen supone su radicalización: el Hombre es una fuerza poderosa pero en última instancia arbitraria en el progreso de la Inteligencia Vital [«*Life-Intelligence*»].⁴¹ En este relato el Hombre está plenamente incorporado en la Naturaleza, hasta el punto de que se disuelve en ella. Eugen escribe que cuando «la vida pase a esta placa de Petri que se está desarrollando, las cosas se pondrán mucho más estimulantes [...]. Espero que lo consigamos».

Para Eugen, las discusiones sobre tecnología en que se involucran los polímatas no podrían ser más provincianas. Tan solo son importantes en tanto que establecerán las «condiciones iniciales» para la gran aventura coevolutiva de la tecnología por venir. Para el transhumanista, la tecnología *no* se disuelve. En vez de eso, es la solución en la que los humanos quedan disueltos. El sufrimiento, la asignación de recursos, la toma de decisiones —todo eso es intrascen-

41. Agradezco a Albano Cruz (<http://albanocruz.com/>) sus explicaciones sobre transhumanismo y, en el caso concreto del concepto de «*Life-Intelligence*», su propuesta de traducción como «Inteligencia Vital». Con ella esperamos conservar (que no despejar) lo que de etéreo e inasible tiene tal concepto. [N. del E.]

dente para el resultado definitivo del progreso tecnológico; se trata de asuntos mundanos, aunque conciernen a la vida y la muerte, y como tales se los puede denunciar o apoyar, pero solo con vistas a afinar la aceleración hacia la singularidad. Para los transhumanistas, no se puede luchar contra la inevitabilidad de la evolución técnica, pero sin duda se puede *contribuir* a ella. El progreso técnico es por ende tanto asimilable a la ley como susceptible de manipulación inteligente; el progreso técnico es inevitable, pero solo por la potencia de una curiosidad humana paralela masiva.⁴²

Considerado como uno de los modos de pensamiento presentes en el debate político mundial, los transhumanistas (como los polímatas) convierten la tecnología en un argumento retórico. La tecnología es el argumento político más poderoso porque «funciona». No tiene sentido discutir «sobre» tecnología pero sí lo tiene discutir a través de y con ella. No tiene sentido hablar de si parar la tecnología es bueno o malo, porque alguien sencillamente construirá una tecnología que invalidará tal argumento.

La invención técnica sigue teniendo un papel reservado, pero este se encuentra claramente diferenciado de las intervenciones políticas, legales, culturales o sociales. Para la mayoría de transhumanistas, no existe retórica aquí, tampoco sofistería, solo la pura verdad del «funciona»: la pura, innegable, imparable e indeconstruible realidad de la tecnología. Para la actitud transhumanista, la realidad del «código que funciona» posee una realidad de la que carecen otras afirmaciones sobre el mundo. El transhumanismo extremo reemplaza el mundo de la vida con el mundo del ordenador, donde las malas (*éticamente* malas) ideas *no serán compiladas*. Versiones menos fervorosas del transhumanismo simplemente permiten la confusión para operar de manera oportunista: el progreso de la tecnología es incuestionable (omnisciente) y solo sus efectos en los humanos merecen ser investigados.

Los transhumanistas puros son, pues, antimodernos. Los transhumanistas desprecian el presente por su descenso intolerablemente lento hacia el futuro de la inmortalidad y la autosuperación humanas, y temen la destrucción por causa de una resistencia humana demasiado turbulenta (e ignorante). No hemos de tener una concepción

42. El procesamiento paralelo masivo es una arquitectura que se sirve de varios procesadores conectados en paralelo para ejecutar un programa o una tarea. El autor emplea la analogía implícita en el juego de palabras para describir el modo en que los transhumanistas entienden el funcionamiento del progreso técnico. [N. del E.]

individual del presente, tampoco una reflexión o una comprensión sintética sobre él. Tan solo hemos de *contribuir* a él correctamente. Podría incluso irse tan lejos como para sugerir que las formas de reflexión sobre el presente que no contribuyen al progreso técnico ponen en peligro el mismo futuro de la inteligencia vital. La curiosidad y la innovación técnica no son rasgos históricos de la ciencia occidental, sino rasgos naturales de un animal humano que ha creado sus propias condiciones de desarrollo. Por ende, la conciencia histórica de los transhumanistas consiste en gran medida en una línea de tiempo que da un sentido ordenado a nuestro lugar en el progreso hacia la Singularidad.

Con todo, la moraleja del relato no es simplemente que la tecnología determina la historia. El transhumanismo es una posición radicalmente *antihumanista* en la que la agencia humana —si existe tal cosa— no es ontológicamente distinta de la de las máquinas, los animales y la vida misma. Por más que sea necesario organizar, hacer cosas, tomar decisiones, participar, construir, hackear, *innovar*, todo ello equivale a una creencia en la capacidad de los humanos para controlar su destino, individual o colectivamente. En último término, los transhumanistas son incapaces de detallar exactamente qué parte de esta historia es inevitable —excepto quizás la propia historia. La tecnología no se desarrolla sin millones de humanos distribuidos contribuyendo a ella; los humanos no pueden evolucionar sin la explícita adopción humana de tecnologías que alteran la vida y la identidad; la evolución no puede hacerse inevitable sin la manipulación de los entornos y las luchas por la adecuación. Como en el dilema del calvinismo (donde no podemos saber si la salvación procede de nuestras buenas obras), el transhumanismo aún debe crear tecnologías acordes a las demandas particulares y ordinarias del momento, pero ello en modo alguno determina el resultado último del progreso tecnológico. Se trata de una percepción bien articulada por Adam Ferguson y subrayada repetidamente por Friedrich Hayek con respecto a la sociedad humana: «resultado de la actividad humana pero no del designio humano».⁴³

43. Friedrich A. Hayek, *Law, Legislation and Liberty*, vol. 1, p. 20 [ed. cast.: *Derecho, legislación y libertad*, p. 51].

Conclusión

Para muchos observadores, los *geeks* exhiben una mezcla acaso desconcertante de liberalismo, liberal-libertarismo, anarquismo, idealismo y pragmatismo, y aun así tienden a encajar perfectamente en cualquier categoría política constituida (liberal, conservadora, socialista, capitalista, neoliberal, etc.). Con la exposición de cómo los *geeks* se sirven de la Reforma Protestante como un pasado usable y cómo ocupan un espectro de creencias relativas al progreso, la libertad y la intervención, espero resistir a esta urgencia clasificatoria. Los *geeks* son un caso interesante precisamente porque están involucrados en la creación de *nuevas cosas* que cambian el significado de nuestras categorías políticas constituidas. Sus ideas políticas se mezclan y combinan con los detalles técnicos de Internet, el software libre y las diversas y variadas organizaciones, leyes, personas y prácticas con las que lidian de manera regular: sistemas operativos y sistemas sociales. Pero tal mezcla no hace a los *geeks* meramente tecnoliberal-libertarios o tecnoconservadores. Más bien, revela su manera de reflexionar desde la situación específica e históricamente única de Internet acerca de los problemas generales del saber y el poder, la libertad y la ilustración, el progreso y la intervención.

Los *geeks* no son un tipo de *persona*: los *geeks* son *geeks* solo en la medida que se reúnen mediante nuevas formas técnicamente mediadas de creación propia y de maneras nada fáciles de identificar (ni lenguaje, ni cultura, ni mercados, ni naciones, ni guías telefónicas o bases de datos). Aunque su afinidad se constituye claramente a través de Internet, Internet no es la única razón de tal afinidad. Lo es la afinidad colectiva a la que me refiero como público recursivo. Debido a que es imposible entender esta afinidad intentando identificar tipos de personas en particular, es necesario recurrir a conjuntos de prácticas históricamente específicos que conforman la esencia de su afinidad. El software libre es un caso ejemplar —quizás el parangón— de público recursivo. Entender el software libre a través de sus cambiantes prácticas no solo ofrece un mejor acceso al mundo de la vida de los *geeks*, sino que también revela cómo la estructura de un público recursivo se origina y logra resistir y transformarse, y cómo puede convertirse en una poderosa forma de vida que extienda sus afinidades más allá de los tecnófilos *geeks* al ámbito de la vida ordinaria.

PARTE II

SOFTWARE LIBRE

III. EL MOVIMIENTO

La Segunda Parte de *Two Bits* describe qué es el software libre y de dónde procede, y cada uno de sus cinco capítulos detalla la narrativa histórica de una clase particular de práctica: la creación de un movimiento, la compartición de código fuente, la concepción de sistemas abiertos, la redacción de licencias de *copyright* (y de *copyleft*) y la coordinación de colaboraciones. Tomadas en su conjunto, estas historias describen el software libre. Su punto final (o su punto de partida, genealógicamente hablando) se sitúa en los años 1998-1999, cuando el software libre irrumpió en escena: en la portada de la revista *Forbes*, como parte del *boom* de las empresas puntocom y en los consejos de administración de empresas de capital de riesgo y corporaciones como IBM y Netscape. Aunque los capítulos que componen esta segunda parte pueden leerse separadamente para comprender las prácticas que son condición *sine qua non* del software libre, también pueden leerse de continuo, como un relato serpenteante de la historia del software y las redes que se extiende desde finales de los 50 hasta el presente.

Más que definir lo que hace libre al software libre o abierto al código abierto, *Two Bits* trata cada una de las cinco prácticas mencionadas como partes de un *sistema técnico experimental de carácter colectivo*: cada componente tiene su propia historia, desarrollo y temporalidad, pero se ensamblan como un todo y se vuelven reconocibles hacia 1998-1999. Como sucede con cualquier sistema experimental, la modificación de los factores modifica el funcionamiento y los resultados del conjunto. El software libre así concebido es un tipo de sistema experimental: sus prácticas se pueden adoptar, adaptar y modular en nuevos contextos y lugares, pero es un sistema cuyas normas se determinan de manera colectiva y se modifican frecuentemente. En cada una de las cinco

prácticas es posible ver el momento en que las opciones de *cómo* hacer software libre alcanzaron, o superaron, ciertos límites, manteniéndose sin embargo como parte de un sistema cuya identidad finalmente se concretó en el periodo 1998-1999 y en los siguientes años.

La primera de estas prácticas —la constitución del software libre como un movimiento— es tanto la más inmediatamente obvia como la más difícil de entender. Por *movimiento* entiendo la práctica, entre los *geeks*, de debatir y discutir la estructura y el significado del software libre: en qué consiste, para qué sirve y si es o no un movimiento. Algunos *geeks* se refieren al software libre como un movimiento, otros no; algunos hablan de la ideología y los objetivos del software libre, otros no; algunos lo llaman software libre, mientras que otros lo llaman código abierto. En medio de toda esta controversia, sin embargo, los *geeks* del software libre reconocen que todos ellos *están haciendo lo mismo*: la práctica de crear un movimiento es la práctica de hablar sobre el significado y la necesidad de las otras cuatro prácticas. Fue en 1998-1999 cuando los *geeks* llegaron a reconocer que todos ellos estaban haciendo lo mismo y, casi inmediatamente, comenzaron a discutir por qué.¹

Una forma de entender el movimiento es a través de la historia de

1. Por ejemplo, Richard Stallman escribe: «El movimiento del software libre y el movimiento *open source* son como dos campos políticos dentro de la comunidad del software libre. Grupos radicales de la década de 1960 desarrollaron una reputación de sectarismo: las organizaciones se escindían por desacuerdos en detalles estratégicos, y luego se trataban entre sí como enemigas. O por lo menos, esa es la imagen que la gente tiene de ellos, tanto si era verdad como si no. La relación entre el movimiento del software libre y el movimiento *open source* es justo la contraria a esa imagen. Estamos en desacuerdo en los principios básicos, pero estamos más o menos de acuerdo en las recomendaciones prácticas. Así que trabajamos juntos en muchos proyectos específicos. No pensamos el movimiento *open source* como enemigo. El enemigo es el software privativo». («Why ‘Free Software’ Is Better than ‘Open Source,’» GNU’s Not Unix! <http://www.gnu.org/philosophy/free-software-for-freedom.html> [última consulta: 18 de abril de 2018] [ed. cast.: «Por qué ‘software libre’ es mejor que ‘código abierto’», en *Software libre para una sociedad libre*, pp. 57–58]. En contraste, la Open Source Initiative caracteriza esta relación como sigue: «¿Cómo se relaciona el ‘código abierto’ con el ‘software libre’? La Open Source Initiative es un programa de *marketing* para el software libre. Es un argumento de venta del ‘software libre’ porque funciona, no porque es la única opción justa. Estamos vendiendo libertad por sus propios méritos» (<http://www.opensource.org/advocacy/faq.php> [enlace no accesible en la última consulta de 18 de abril de 2018]. Hay un gran número de definiciones de Software Libre: las definiciones canónicas recogen los escritos de Richard Stallman en el sitio web de la FSF, www.fsf.org, incluyendo «La definición de software libre» y «Algunas palabras que se deben evitar». Del lado del Código Abierto existe la «Open Source Definition» (<http://www.opensource.org/licenses/>) [traducida al castellano en: https://es.wikipedia.org/wiki/Open_Source_Definition]. Para definiciones sin afiliación, véase: www.freedomdefined.org.

los navegadores web Netscape y Mozilla (ahora conocido como Firefox). Esta historia no solo dota de contexto a los relatos de los *geeks* presentados en la Parte I —y aquí paso de la observación participante directa a la investigación histórica y archivística sobre un fenómeno que se estaba dando prácticamente al mismo tiempo— sino que también contiene todos los elementos necesarios para entender el software libre. Dicha historia está repleta de discusiones y debates acerca de las prácticas que componen el software libre: la compartición de código fuente, la concepción de sistemas abiertos, la redacción de licencias y la coordinación de colaboraciones.

La bifurcación del software libre, 1997-2000

El software libre se bifurcó en 1998 con la aparición repentina del término *código abierto* (anteriormente usado solo por la CIA para referirse a las fuentes de información no confidenciales). Los dos términos se tradujeron en dos tipos diferentes de narrativa: el primero, relativo al software libre, se remontaba a los 80, promoviendo la libertad del software y la resistencia a la «avaricia» del software privativo, en palabras de Richard Stallman, director de la FSF (*Free Software Foundation*, Fundación para el Software Libre); el segundo, relativo al código abierto, estuvo asociado al *boom* de las puntocom y al evangelismo del *hacker* liberal-libertario pro-negocios Eric Raymond, quien se centró en el valor económico y el ahorro de costes que representaba el software de código abierto, incluido el enfoque pragmático (y polimático) que regía el uso cotidiano de software libre en algunas de las mayores compañías emergentes *online* (Amazon, Yahoo!, HotWired y otras, todas las cuales «promocionaron» el software libre usándolo para sus actividades).

Un momento crítico en la aparición del software libre ocurrió en 1998-99: nuevos nombres y nuevas narrativas, pero también nuevas riquezas y nuevos retos. El «código abierto» se basaba en las promesas puntocom de reducción de costes, «desintermediación» y otras diversas estrategias para ganar dinero con él (Cygnus Solutions, una de las primeras empresas de software libre, se autoasignó el jocoso eslogan *Making Free Software More Affordable*, «Haciendo más asequible el software gratis/libre»). Así, por ejemplo, VA Linux, que vendía ordenadores personales con sistemas operativos de código abierto preinstalados, tuvo la salida a Bolsa más lucrativa de la burbuja puntocom, viendo cómo sus acciones se revalorizaban un 700% en un solo día.

El «software libre», en contraste, atizó las llamas de la inquietud por el expansionismo de la propiedad intelectual y se volcó en una incipiente resistencia legal contra la DMCA (*Digital Millennium Copyright Act*, Ley de *Copyright* del Milenio Digital) y la Sonny Bono *Copyright Term Extension Act* (Ley Sonny Bono de Extensión del Plazo del *Copyright*) de 1998. Antes de 1998, el *software libre* se refería tanto a la FSF (y a la atenta mirada de microgestión de Richard Stallman) como a uno de los miles de diferentes proyectos, procesos, licencias e ideologías de matriz comercial, *amateur* o académica que recibían denominaciones muy diversas: *sourceware*, *freeware*, *shareware*, software abierto, software de dominio público, etc. El término *código abierto*, por el contrario, procuró abarcar todas esas denominaciones en un solo movimiento.

El acontecimiento que precipitó esta tentativa de golpe de estado semántico fue la liberación del código fuente del navegador web Netscape Communicator, cuya importancia para la suerte del software libre es difícil de sobreestimar. Netscape es justamente célebre por su salida a Bolsa de 1995 y por su decisión de ofrecer su producto principal, Netscape Navigator, de forma gratuita (es decir, se podía descargar e instalar una versión binaria compilada por «cero dólares»). Pero Netscape es mucho más célebre entre los *geeks* por regalar algo más en 1998: el código fuente de Netscape Communicator (nacido Navigator). Este regalo de la *aplicación* Navigator atrajo a Netscape la simpatía de los *geeks* y confundió a los inversores, pero pasó inadvertido para los consumidores.

Netscape es importante desde varias perspectivas. Los empresarios e inversores conocían Netscape como el proyecto favorito de Jim Clarke, exitoso hombre de negocios que había fundado la compañía de fabricación de equipos especializados Silicon Graphics Incorporated (SGI). Para los informáticos e ingenieros, especialmente en la pequeña ciudad universitaria de Champaign-Urbana (Illinois), Netscape era conocido como el mejor postor por el equipo de WWW del NCSA (*National Center for Supercomputing Applications*, Centro Nacional para Aplicaciones de Supercomputación) de la Universidad de Illinois. Aquel equipo —Marc Andreessen, Rob McCool, Eric Bina, Jon Mittelhauser, Aleks Totic y Chris Houck— había creado Mosaic, el primer y más cariñosamente recordado «navegador gráfico» para surfear la World Wide Web. De ahí que a Netscape se la conociera al inicio como Mosaic Communications Corporation y solo cambiara su nombre tras las amenazas legales por parte del NCSA y de una empresa rival, Spyglass. Entre los *geeks*,

Netscape era conocida por albergar a varios *hackers* y defensores del software libre, muy especialmente a Jamie Zawinski, quien de modo bastante ostentoso había roto filas con la FSF mediante la bifurcación del código de GNU EMACS para crear lo que se conoció primero como Lucid Emacs y más tarde como XEmacs. Zawinski pasaría a liderar el proyecto del recién liberado navegador de Netscape, ahora llamado Mozilla.

Mientras tanto, los usuarios informáticos más asiduos recuerdan Netscape tanto como el emblema de la demencia inversora del *boom* puntocom como otra de las víctimas de Microsoft. Aunque Netscape irrumpió en escena en 1995, ofreciendo un navegador rico en funcionalidades que suponía una alternativa al muy básico Mosaic, pronto comenzó a perder terreno frente a Microsoft, que adoptó relativamente rápido la estrategia de regalar su navegador, Internet Explorer, como si fuera parte del sistema operativo Windows. Esta práctica acabó siendo declarada una violación de las leyes antimonopolio por el Departamento de Justicia de EEUU, y por ella Microsoft recibió una condena que sin embargo nunca conllevó sanción alguna.

La naturaleza de la decisión de Netscape de liberar el código fuente difiere según la perspectiva desde la que se observe. Podría parecer un plan de negocio inspirado en su éxito original: regalar el producto y ganar dinero en Bolsa. Podría parecer una jugada estratégica de última hora para desbancar a Microsoft. También podría parecer, y muchos *geeks* lo creyeron así, un intento de recuperar parte de la «credibilidad *hacker*» que había adquirido al llevarse al equipo del NCSA; o incluso un intento de «hacer lo correcto» mediante la liberación de una de las aplicaciones más útiles del mundo. Pero, ¿por qué llegaría Netscape a tal conclusión? ¿Mediante qué razonamiento parecería correcta semejante opción? Los motivos de la decisión de Netscape de «liberar el código fuente» recapitulan las cinco prácticas nucleares del software libre —y proveyeron el impulso clave para el nuevo movimiento.

Compartición de código fuente

La decisión de Netscape de compartir su código fuente solo podría parecer sorprendente en el contexto de la práctica generalizada de mantener en secreto el código fuente. Este secretismo constituía una práctica que en gran medida se seguía con el fin de evitar que los competidores copiasen un programa y compitiesen con él, pero también como un medio para controlar el mercado mismo. La misma World Wide Web

en la que se había baqueteado el equipo de Andreessen del NCSA se diseñó para ser «independiente de plataforma» y accesible por cualquier dispositivo en la red. En la práctica, no obstante, ello implicaba que alguien tenía que crear «navegadores» para cada equipo o dispositivo. Mosaic se creó inicialmente para UNIX, usando la biblioteca Motif del Sistema de Ventanas X11 —en resumen, un tipo de acceso muy específico. Netscape, en contraste, se enorgullecía de la «portabilidad» de Netscape Navigator para casi todas las arquitecturas de computadora disponibles. De hecho, en 1997 había planes en marcha para crear una versión del navegador —escrito en Java, el lenguaje de programación creado por Sun Microsystems para «escribir una vez, ejecutar en cualquier lugar»— que sería completamente independiente de plataforma.

Sin embargo, el navegador basado en Java (llamado Javagator, por supuesto) creó un problema con respecto a la práctica de mantener en secreto el código fuente. Cada vez que se ejecutaba un programa en Java, este creaba un conjunto de «códigos de *byte*» a los que era fácil aplicar ingeniería inversa, ya que tenían que transmitirse desde el servidor al equipo que ejecutaba el programa, siendo así visibles para cualquiera que supiese cómo y dónde mirar. Los ingenieros de Netscape barajaron la idea de ofuscar deliberadamente estos códigos *byte* para disuadir a los competidores de copiarlos. ¿Cómo se puede competir, dictaba su lógica, si cualquiera puede copiar tu programa y hacer su propia versión alternativa?

Zawinski, entre otros, sugirió que esto era una mala idea: ¿por qué no limitarse a *compartir* el código fuente y hacer que la gente contribuya a mejorarlo? Como participante veterano del software libre, Zawinski comprendió los beneficios potenciales de recibir ayuda de un conjunto enorme de potenciales colaboradores y urgió a sus colegas de Netscape a ver la luz. No obstante, por más que les contó y les mostró éxitos, nunca logró convencerles de que se trataba de un inteligente plan de *negocios*, solo de que era un plan de ingeniería de software eficiente. Desde el punto de vista de la dirección y de los inversores, semejante movimiento parecía equivalente a desprenderse de la propiedad intelectual de la empresa.

Frank Hecker, un director comercial, estableció el vínculo entre desarrolladores y directivos.

[Para los desarrolladores] era obvio por qué esto era importante. Desde la óptica de la alta dirección no estaba nada claro por qué la liberación del código fuente podría ser de utilidad, ya que nadie había planteado nunca un argumento comercial al respecto.

Hecker escribió un documento llamado «El código fuente de Netscape como producto de Netscape» y lo difundió entre diversas personas, incluyendo a Andreessen y al Consejero Delegado de Netscape Jim Barksdale. Como sugiere el título, el argumento comercial era que el código fuente también podría ser un producto, y en el contexto de Netscape, cuyo modelo de negocios era «regalar la aplicación y hacer caja en Bolsa», tal propuesta parecía menos disparatada de lo que habría sido de otro modo:

Cuando Netscape comenzó a habilitar la descarga sin restricción de Navigator por Internet, muchos vieron esto como un atentado a la sabiduría convencional del negocio informático y se preguntaban cómo nos sería posible ganar dinero «regalando nuestro software». Ahora, por supuesto, se analiza retrospectivamente esta estrategia como una innovación exitosa que fue un factor clave en el rápido crecimiento de Netscape, y rara es hoy la compañía de software que no imita nuestra estrategia de un modo u otro. Entre otras cosas, esto provoca la siguiente pregunta: ¿Y si tuviéramos que repetir este guión, solo que esta vez con el código fuente?²

Bajo la influencia de Hecker, Zawinsky y del director tecnológico Eric Hahn (que también había escrito diversos «documentos heréticos» internos sugiriendo enfoques similares), Netscape acabó tomando la decisión de compartir su código fuente con el mundo exterior, decisión que se tradujo en el famoso comunicado de prensa de enero de 1998 que describía los objetivos y beneficios de tal movimiento. En aquel momento particular de la vida de Netscape, y en medio del *boom* de las puntocom, esta decisión fue sin duda trascendental, pero no condujo ni a una afluencia de inversiones ni a un producto repentinamente superior.³

2. Frank Hecker, citado en Hamerly y Paquin, «Freeing the Source», p. 198.

3. Véase Moody, *Rebel Code*, cap. 11, para una versión más detallada de la historia.

Concepción de sistemas abiertos

La liberación del código fuente fue, en cierto modo, un intento de recuperar la confianza de la gente que había imaginado por primera vez la web. Tim Berners-Lee, el arquitecto inicial de la www, siempre había insistido en que el protocolo y todas sus implementaciones deberían ser libremente accesibles (en el sentido de «estar en el dominio público» o «publicados como software libre»). De hecho, Berners-Lee había hecho justo eso con sus primeras implementaciones básicas de la www, declarándolas orgullosamente parte del dominio público.

A lo largo de los 90, la «guerra de navegadores» provocó que tanto Netscape como Microsoft se alejaran de esta visión: cada compañía había implementado sus propias extensiones y «funcionalidades» para navegadores y servidores, las cuales no estaban presentes en el protocolo que Berners-Lee había creado o en los posteriores estándares creados por el consorcio W3C (*World Wide Web Consortium*). Incluidas en dichas implementaciones había diversos tipos de «maldades» que podían llevar a que los navegadores no funcionaran en ciertos sistemas operativos o con ciertos tipos de servidor. La «guerra de navegadores» repetía así una batalla de sistemas abiertos que venía de los 80, en la cual el intento de estandarizar un sistema operativo de red (UNIX) se vio obstaculizado por la competencia y el secretismo, al tiempo que se formaban consorcios consagrados a la «apertura» para intentar impedir la propagación del mal. A pesar de que tanto Microsoft como Netscape eran miembros del W3C, la incompatibilidad de sus navegadores claramente representaba la manipulación de los procesos de estandarización en nombre de la ventaja competitiva.

Así pues, la liberación del código fuente de Communicator se percibió ampliamente como el único modo de sortear el pozo envenenado de las implementaciones de navegadores competitivamente enmarañadas y no estandarizadas. Debido a los derechos de redistribución asociados a la licencia de código abierto, sería posible diseñar un navegador libre que respetara los estándares —si no a través de sus creadores inmediatos, sí mediante una bifurcación del programa que los cumpliera. El código abierto sería así la solución a un problema de los sistemas abiertos jamás resuelto, ya que nunca se había afrontado la cuestión de la propiedad intelectual directamente. El software libre, por el contrario, tenía una sofisticada solución en su licencia GNU GPL, también conocida como licencia *copyleft*, que permitía preservar la libertad del software y reavivar la esperanza de mantener los estándares abiertos.

Redacción de licencias

Pero he aquí el problema: Netscape se vio inmediatamente envuelta en una controversia con los *hackers* del software libre al optar por redactar sus propias licencias a medida para distribuir el código fuente. En lugar de confiar en una de las licencias existentes, como la GNU GPL o la BSD de Berkeley o las licencias del MIT, crearon las suyas propias: la NPL (*Netscape Public License*, Licencia Pública de Netscape) y la MPL (*Mozilla Public License*, Licencia Pública de Mozilla). La preocupación inmediata de Netscape tenía que ver con su red de contratos y convenios vigentes con terceros desarrolladores —tanto aquellos que en el pasado hubieran aportado partes del código fuente existente sobre las cuales Netscape podría carecer de derechos de redistribución como software libre, como aquellos que esperaban comprar y redistribuir una versión comercial en el futuro. Las licencias de software libre existentes eran, o bien demasiado permisivas, otorgando a tercera partes derechos que la misma Netscape podría no tener, o bien demasiado restrictivas, obligando a Netscape a poner a libre disposición el código fuente (la GPL) cuando ya había firmado contratos con compradores del código no libre.

Fue esta situación comercial compleja y específica —una red de contratos y licencias de código ya existentes— la que creó en Netscape la necesidad de redactar su propia licencia. De este modo, la NPL contenía una cláusula que concedía a Netscape un permiso especial para relicenciar cualquier contribución particular al código fuente como un producto privativo con vistas a la conciliación con sus contratos con terceros; esencialmente, ello otorgaba a Netscape derechos especiales que ningún otro titular de licencia tendría. Aunque esta opción no socavaba necesariamente las licencias de software libre —y aunque ello sin duda era prerrogativa de Netscape—, iba en contra del espíritu del software libre, dividiendo al «público recursivo» en dos mitades. Con vistas a apaciguar a los *geeks* del software libre, Netscape redactó una licencia para el código existente (la NPL) y otra diferente para las *nuevas* contribuciones: la MPL.

Ni Stallman ni ningún otro *hacker* del software libre quedaron completamente contentos con esta situación. Stallman señaló tres defectos: «Uno envía un mensaje filosófico negativo, otro pone a la comunidad de software libre en posición de debilidad y el tercero crea un grave problema práctico en el seno de la comunidad de software libre. Dos de estos defectos también incumben a la Licencia Pública Mozilla». Ante ello, Stallman exhortó a la gente a no usar la NPL. De modo similar, Bruce Perens sugirió:

Muchas compañías han adoptado una variación de la MPL [*sic*] para sus propios programas. Esto resulta desafortunado, ya que la NPL se diseñó para la situación comercial específica en que se hallaba Netscape en el momento de su redacción, y no es necesariamente adecuada para otros usos. Debería mantenerse como la licencia de Netscape y Mozilla, y los demás deberían usar la GPL, la BSD u otras.⁴

Las discusiones sobre los pequeños detalles de las licencias pueden parecer escolásticas, pero la decisión tuvo un enorme impacto en la estructura del nuevo producto. Como señaló Steven Weber, la elección de la licencia determina la vía de organización de un producto y puede determinar los tipos y fuentes de contribuciones a un proyecto.⁵ No se trata de una decisión baladí: cada nueva licencia se inspecciona con la misma intensidad o se denuncia con la misma urgencia.

Coordinación de colaboraciones

Uno de los atractivos comerciales del software libre, especialmente de su mercadotecnia *como* código abierto, es que moviliza el trabajo de miles o centenas de miles de colaboradores voluntarios a través de Internet. Semejante argumento casi inevitablemente conduce a un alegato espurio sobre sistemas «autoorganizados» y propiedades emergentes de la colaboración distribuida. El comunicado de prensa de Netscape prometía «aprovechar la energía creativa de miles de programadores en Internet mediante la incorporación de sus mejores avances», citando estas palabras del Consejero Delegado Jim Barksdale: «Mediante la liberación del código fuente para futuras versiones, podremos espollear las energías creativas de toda la comunidad de la Red y animar unos niveles de innovación sin precedentes en el mercado de navegadores»⁶. Pero como cualquiera que alguna vez haya intentado poner en marcha un proyecto de software libre sabe, la cosa nunca funciona así.

4. Bruce Perens, «The Open Source Definition», p. 184.

5. Steven Weber, *The Success of Open Source*.

6. «Netscape Announces Plans to Make Next-Generation Communicator Source Code Available Free on the Net», comunicado de prensa de Netscape, 22 de enero de 1998, <http://wp.netscape.com/newsref/pr/newsrelease558.html> [enlace no accesible en la última consulta de 18 de abril de 2018].

La ingeniería informática constituye un problema notoriamente complicado.⁷ Los pasillos de la industria de *software* están flanqueados por los cadáveres de advertencia de las metodologías de desarrollo muertas. Y la creación de software en pleno *boom* de las empresas puntocom no era en nada diferente, excepto por que la rapidez de los ciclos de lanzamiento y la velocidad del consumo de fondos (el «flujo de caja negativo») era mayor que nunca. Las metodologías de desarrollo internas de Netscape fueron diseñadas para afrontar estas presiones y, como muchos trabajadores del sector atestiguarán, tal método consiste en una carrera semiestructurada, marcada por los plazos de entrega y alimentada a base de cafeína y nootrópicos.⁸

En consecuencia, la liberación del código de Mozilla requería un sistema de coordinación que diferiría de la práctica corriente de desarrollo de software realizada por los programadores en nómina de la empresa. Se requería incorporar las contribuciones de desarrolladores externos, esto es, que no trabajaran para Netscape. También requería *incitar* a la gente a contribuir, ya que este era el trato en que se basaba la decisión de liberar el código fuente, y permitirle el seguimiento de sus aportaciones, de modo que pudiera verificar que las mismas se incluían o descartaban por razones legítimas. En suma, si hubiera de manifestarse algún tipo de autoorganización mágica de código abierto, esta requeriría un sistema de coordinación meticulosamente transparente con base en Internet.

Para empezar, esto implicaba cuestiones prácticas: obtener el nombre de dominio mozilla.org, configurar (liberando a su vez su código fuente) el sistema de control de versiones (el estándar de software libre cvs), la interfaz de control de versiones (Bonsai), el «sistema de integración» que gestionase y mostrase los diversos árboles y ramas (rotas) de un proyecto complejo de software (Tinderbox) y el sistema de seguimiento de errores para monitorizar los fallos enviados por usuarios y desarrolladores (Bugzilla). Todo ello requería un sistema organizativo en el seno del proyecto Mozilla, en el que a los desarrolladores contratados se les encargaba revisar los envíos tanto internos y externos, y a los mantenedores o editores inspeccionar estas contribuciones y verificar si deberían incorporarse.

7. Sobre la historia de las metodologías de desarrollo de software, véase Mahoney, «The Histories of Computing(s)» y «The Roots of Software Engineering».

8. Pueden encontrarse descripciones especialmente buenas de este ciclo en Ullman, *Close to the Machine* y *The Bug*.

Al final, la liberación del código fuente de Mozilla fue tanto un éxito como un fracaso. Su éxito se hizo esperar hasta el año 2004, cuando el navegador Firefox, basado en Mozilla, se encaramó silenciosamente a los primeros puestos de la clasificación de navegadores más populares, convirtiéndos en la aplicación de software libre más visible y ampliamente utilizada. El fracaso, en cambio, fue más inmediato: Mozilla no logró reportar a Netscape los masivos beneficios que en 1995 le dio ofrecer gratuitamente el Netscape Navigator. En su carta abierta de dimisión de abril de 1999 (un año después de la liberación), Zawinsky expresó esta noción de fracaso. En ella atribuía el declive de Netscape posterior a 1996 a que había «dejado de innovar» y se había vuelto demasiado grande para ser creativa, y describía la decisión de liberar el código fuente de Mozilla como un regreso a la innovación:

[El anuncio] supuso un rayo de esperanza para mí [...] Era algo tan loco que podría funcionar. Siguiendo la inspiración, me puse manos a la obra, registrando el dominio esa misma noche, diseñando la estructura de la organización, escribiendo la primera versión de la página web y, junto a mis co-conspiradores, explicando sala a sala a todos los empleados y directivos de Netscape cómo funcionaba el software libre, y que teníamos que hacer para que funcionara.⁹

Para Zawinsky, la decisión suponía tanto una posibilidad de que Netscape recuperara su gloria como una oportunidad para demostrar la potencia del software libre:

Lo vi como una ocasión para que el código realmente prosperase. Al hacer que ya no fuera un proyecto de Netscape sino un proyecto público donde Netscape meramente colaboraba, el hecho de que Netscape no fuese ya capaz de construir nuevos productos no importaría: los participantes externos le mostrarían a Netscape cómo hacerlo. Poniendo el control del navegador web en manos de cualquiera que se preocupase de encarar esta tarea, aseguraríamos que esas personas lo mantendrían en marcha por su propio interés.¹⁰

9. Jamie Zawinski, «resignation and postmortem» 31 de marzo de 1999, <http://www.jwz.org/gruntle/nomo.html>.

10. *Ibid.*

Pero esta promesa no se hizo realidad —o, al menos, no lo hizo a la velocidad a la que estaban acostumbrados Zawinsky y otros participantes del sector. Zawinsky ofreció varias razones para ello: el proyecto se componía primordialmente de empleados de Netscape y por ende parecía seguir siendo cosa de la empresa; el proyecto era demasiado extenso para que los colaboradores externos pudiesen sumergirse en él e introducir pequeñas modificaciones; el código era demasiado «farragoso», es decir, contenía demasiadas complicaciones, sobreescrituras e impurezas. Aunque acaso lo más importante era que el código fuente en realidad *no funcionaba*: «Nunca distribuimos el código fuente a un navegador web en uso y, lo que es más importante, nunca lo distribuimos *al navegador web que la gente realmente usaba*».¹¹

A resultas de ello, Netscape no logró incitar a la colaboración. Como lo expresó Zawinsky:

Si alguien ejecutara un navegador web, dejara de hacerlo, añadiera al código fuente un simple comando nuevo, lo recompilara y obtuviera el mismo navegador *más su añadido*, se sentiría motivado para volver a hacerlo y posiblemente para emprender proyectos aún mayores.¹²

Para Zawinsky el fracaso en «despachar» un navegador que funcionara fue el mayor fallo, y se dejó la piel en sugerir que dicho fracaso no era imputable al software libre como tal:

Permitidme aseguraros que, sean cuales sean los problemas del proyecto Mozilla, no se deben a que el código abierto no funcione. El código abierto sí funciona, pero está claro que no es la panacea. Si hay alguna moraleja que extraer de aquí, es que no se puede coger un proyecto moribundo, rociarlo con los polvillo mágicos del código abierto y esperar que todo se resuelva mágicamente. El desarrollo de software es muy exigente. Las cuestiones no son tan simples.¹³

Fomento de movimientos

El periodo entre el 1 de abril de 1998, cuando se liberó por vez primera el código fuente de Mozilla, y el 1 de abril de 1999, cuando Zawinski

11. *Ibid.*

12. *Ibid.*

13. *Ibid.*

anunció su fracaso, no pudo haber sido más embriagador y emocionante para los participantes en el software libre. La decisión de Netscape de liberar el código fuente supuso una oportunidad magnífica para los geeks involucrados en el software libre. Dicha oportunidad llegaba en pleno apogeo de la burbuja puntocom, pero también de la adopción generalizada de herramientas de software libre cruciales: los sistemas operativos Linux para servidores, el servidor web Apache para páginas web, los lenguajes de guión¹⁴ perl y python para la creación de aplicaciones rápidas de Internet y otras varias herramientas de menor nivel como Bind (una implementación del protocolo DNS) o sendmail para la transferencia de correos electrónicos.

Quizás lo más importante fue que la decisión de Netscape llegó en un periodo de febril e intensa autorreflexión por parte de quienes de una forma u otra llevaban involucrados en el software libre ya desde mediados de los 80. El artículo de Eric Raymond «*The Cathedral and the Bazaar*», presentado en 1997 en el Linux Kongress y en el Congreso Perl de O'Reilly, había causado gran agitación entre los *hackers* del software libre. Frank Hecker y Eric Hahn lo citaron como una de las fuentes que les llevó a la decisión de liberar Mozilla; de hecho, tanto Eric Raymond como Bruce Perens actuaron como consultores para Netscape sobre la estrategia del software libre. Y en abril de ese mismo año, Tim O'Reilly, editor de manuales sobre software libre, organizó un congreso llamado Freeware Summit.

El propio nombre del congreso era indicativo de la preocupación por establecer una definición y una dirección. Pese a su obvia centralidad, pero también debido a ella, Richard Stallman no fue invitado a la Freeware Summit, y la FSF no fue presentada como la guía filosófica nuclear del evento. En lugar de ello, y como rezaba el comunicado de prensa emitido tras el congreso:

El propósito del encuentro ha sido facilitar un debate de alto nivel sobre los éxitos y desafíos que se les presentan a los programadores. Aunque en el pasado este tipo de software ha sólidamente denominarse «*freeware*» o «software libre», los programadores coincidieron en que su desarrollo comercial es un aspecto importante y que los

14. Traduzco «scripting languages» como «lenguajes de guión» siguiendo la tendencia reciente de castellanizar el término «script» como «guión» también en el ámbito de la programación informática [N. del E.].

términos «código abierto» y «software de código abierto» describen mejor el método de desarrollo que ellos apoyan.¹⁵

Fue en este congreso donde la sugerencia de Raymond de «Código Abierto» como denominación alternativa se debatió por primera vez en público.¹⁶ Poco después, Eric Raymond y Bruce Perens crearon la Open Source Initiative y redactaron la «Definición de Código Abierto». La pretensión de toda esta autorreflexión era capitalizar la ola de atención dirigida al software libre a raíz del anuncio de Netscape.

Las motivaciones de estos cambios tenían muy diversas procedencias —que iban desde el deseo de inclusión en el *boom* de las puntocom hasta la fuerte resistencia (ideológica) a ser ideológico. Linus Torvalds proclamaba abiertamente que la razón por la que desarrollaba software libre era porque era «divertido»; otros insistían en que era mejor para los negocios o que la estabilidad de infraestructuras como Internet dependía de una robusta capacidad de mejorarlas desde cualquier dirección. Con todo, nadie cuestionó la manera de hacer software libre ni propuso cambiarla.

El artículo «*The Cathedral and the Bazaar*» rápidamente se convirtió en el relato más divulgado acerca del funcionamiento y la importancia del código abierto. En él se recalca la centralidad de las nuevas formas de coordinación por encima del papel de las nuevas licencias de *copyright* o de las prácticas de compartición de código fuente. En *The Cathedral and the Bazaar*, Raymond expone sus experimentos con software libre (el modelo bazar) y reflexiona sobre las diferencias entre este y las metodologías adoptadas por la industria (el modelo catedral). El artículo se desmarca del discurso sobre la libertad y no contiene ninguna denuncia de la avaricia de las empresas de software al estilo de Stallman. Significativamente, tampoco comenta la cuestión de las licencias. Como buen hacker, sin embargo, Raymond ofreció una «historia de revisiones» de su artículo donde se muestra orgullosamente la revisión 1.29 del 9 de febrero de 1998: «‘Software libre’ cambiado por ‘código abierto’».¹⁷

15. «Open Source Pioneers Meet in Historic Summit», comunicado de prensa, 14 de abril de 1998, O'Reilly Press, <http://press.oreilly.com/pub/pr/796>.

16. Véase Hamerly y Paquin, «Freeing the Source». La historia está elegantemente relatada en Moody, *Rebel Code*, pp. 182-204. Raymond atribuye a Christine Petersen, del Foresight Institute, la acuñación del término *código abierto*.

17. Extraído de Raymond, *The Cathedral and the Bazaar* [ed. cast.: «La catedral y el bazar»]. El registro de cambios solo está disponible en línea: <http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/>.

Eric Raymond estaba decidido a rechazar la filosofía de libertad que representaban Richard Stallman y la FSF, aunque no con el fin de crear su propio movimiento político. Más bien, Raymond (y el resto de participantes de la Freeware Summit) buscaban sacar tajada del auge de la economía de Internet haciendo del desarrollo de software libre algo más comprensible para inversores, capitalistas de riesgo y pequeños accionistas. Para Raymond, Richard Stallman y la FSF no representaban la libertad, sino una especie de dogmático comunismo imposible. Al ser Raymond un liberal-libertario convencido, era de esperar que sus creencias fundamentales en la necesidad de sólidos derechos de propiedad entrara en conflicto con el extraño communalismo del software libre —y de hecho su retórica se centró en los usos del software libre pragmáticos, comerciales y orientados a los beneficios y al mercado. Para Raymond, el interés esencial del software libre no era su expansión de la libertad humana, sino la innovación en la producción de software que este representaba (el «modelo de desarrollo»). Estaba claro que el software libre logró algo asombroso a través de una astuta inversión de los sólidos derechos de propiedad, una inversión de la que podía esperarse extraer beneficios masivos de alguna otra forma, fuera mediante la reducción de los costes o, como Netscape, a través del mercado bursátil.

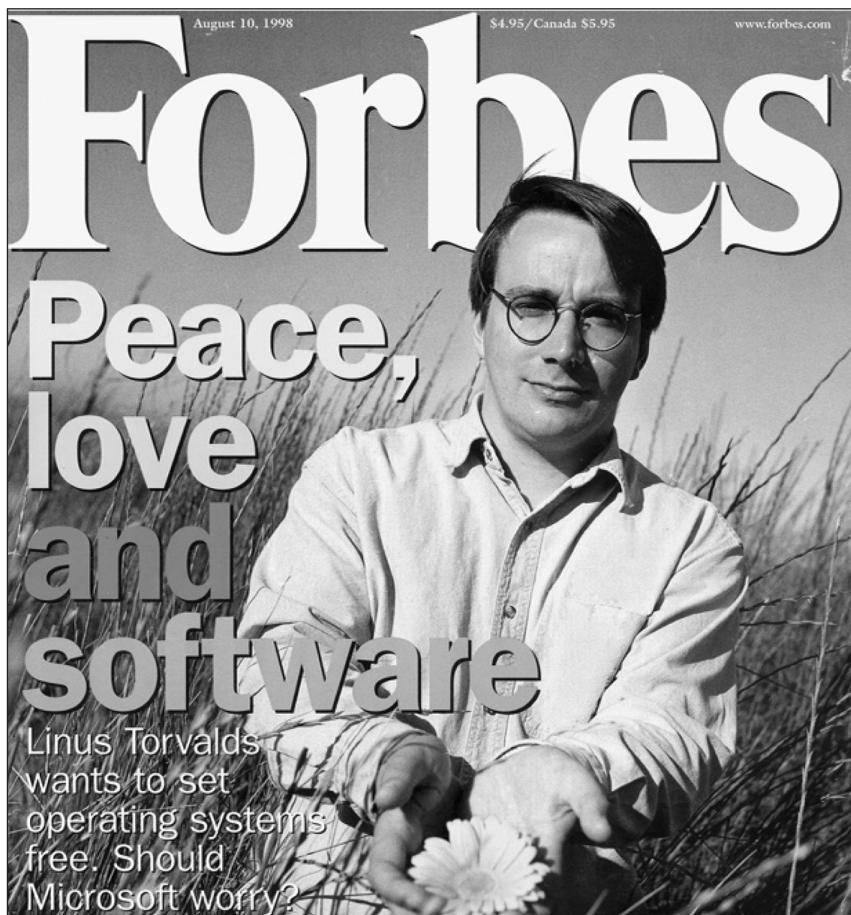
Raymond quería que el mundo empresarial y la gran industria reconocieran el potencial del software libre, pero tenía la impresión de que la retórica de Stallman se interpondría en su camino. Así, por ejemplo, la insistencia de Stallman en llamar «avaricia» a la protección del software mediante la propiedad intelectual estaba haciendo más mal que bien a la aceptación del software libre entre las empresas como una práctica, si no exactamente como un producto.

Los artículos de Eric Raymond canalizaron la frustración de una generación entera de *hackers* del software libre que podían o no haber compartido la posición filosófica dogmática de Stallman, pero que no obstante querían participar en la creación de software libre. Tanto el artículo de Raymond como el comunicado de Netscape y la Freeware Summit apelaban a una ansiedad palpable: que en medio de la mayor creación de riqueza contable de la historia de EEUU quienes se estaban enriqueciendo a través del software libre e Internet no eran quienes los creaban, los mantenían y los *entendían*.

El saldo de Internet entrañaba un conflicto de propiedad: los *hackers* y *geeks* que habían construido el software que la hacía funcionar, bajo

la enseña de hacerla libre para todo el mundo, veían que sus programas generaban una riqueza inaudita para gente que no había participado en dicha construcción (y que, para colmo, no tenía ninguna intención de mantenerla libre para todo el mundo). Subyacente a la creación de riqueza había la aspiración a una suerte de libertad técnica permanente —un orden moral— no compartida por quienes estaban extrayendo los mayores beneficios. Esta ansiedad relativa a la expropiación del trabajo (aunque hubiera sido un acto de amor) se ramificó con el anuncio de Netscape.

Figura 2



«Peace, Love and Software» [«Paz, amor y software»], portada de la revista *Forbes*, 10 de agosto de 1998. Usada con el permiso de *Forbes* y Nathaniel Welch.

Durante todo 1998 y 1999 el revuelo en torno al código abierto fue creciendo. Compañías poco conocidas como Red Hat, VA Linux, Cygnus, Slackware y SuSe, que venían proporcionando mantenimiento técnico y servicios a los consumidores, se situaron súbitamente bajo los focos mediáticos y empresariales. Durante la primavera y el verano de 1998 los principales periódicos les dedicaron diversos artículos, que a menudo intentaban dotar de sentido al cambio terminológico y ver si se correspondía con un cambio en las prácticas. Un artículo de portada de la revista *Forbes*, que incluía fotografías de Richard Stallman, Larry Wall, Brian Behlendorf y Linus Torvalds, no llegaba a decantarse por ningún término en concreto, y recurría tanto a software libre, como a código abierto y *freeware*.¹⁸

A principios de 1999, O'Reilly Press publicó *Open Sources: Voices from the Open Source Revolution* [«Códigos abiertos: Voces desde la revolución del código abierto»], un libro apresuradamente escrito pero muy leído. La obra incluía una serie de artículos —esta vez incluyendo uno de Stallman— que creaban la primera historia pública ampliamente disponible del software libre, abarcando tanto las prácticas como las tecnologías implicadas. El artículo de Kirk McKusick detallaba la historia de importantes tecnologías como la versión BSD de UNIX, mientras que Brian Behlendorf, de Apache, recorría los desafíos prácticos de desarrollar proyectos de software libre. Raymond aportó una historia de *hackers* y un artículo de autoengrandecimiento acerca de su importancia en la creación del movimiento, mientras que la contribución de Stallman refería su propia versión del ascenso del software libre.

En diciembre de 1999 el revuelo en torno al software libre había alcanzado su paroxismo. Cuando VA Linux, una compañía legítima que realmente se dedicaba a algo real —vender ordenadores con Linux preinstalado—, salió al mercado bursátil, sus acciones se revalorizaron un 700% en un día, en lo que constituyó la salida a Bolsa más rentable de la época. VA Linux tomó la insólita decisión de permitir que quienes habían contribuido al núcleo de Linux adquirieran acciones antes de la salida a Bolsa, introduciendo al menos a un grupo de ellos en el preponderante esquema Ponzi de la economía puntocom de Internet. Aquellos que se las apañaron para vender sus acciones acabaron beneficiándose del *boom*, fuese o no estimable su contribución al software

18. Josh McHugh, «For the Love of Hacking», *Forbes*, 10 de agosto de 1998, pp. 94-100.

libre. Mediante el recurso a circunloquios, Raymond, O'Reilly, Perens y otras personas detrás del cambio de nombre habían logrado el reconocimiento del papel central del software libre en el éxito de Internet —y ahora su verdadero nombre podría ser célebre: código abierto.

No obstante, apenas nada cambió en el modo como se hacían las cosas. La compartición de código fuente, la redacción de licencias, la concepción de la apertura, la coordinación de proyectos —todo esto continuó como antes sin ninguna diferencia significativa entre aquellos que exhibían el heroico manto de la libertad y aquellos que vestían la pragmática túnica de la metodología. Con todo, a partir de entonces proliferaron los relatos: definiciones, distinciones, detalles y detracções llenaron el éter de Internet, abarcando desde las aspiraciones filosóficas del software libre hasta las paráboles de la ciencia como el «software de código abierto original». Los partidarios del software libre refinaron su mensaje en cuanto a los derechos legales, al tiempo que los defensores del código abierto refinaban sus reivindicaciones de agnosticismo político o sus aspiraciones no ideológicas a la «diversión». Todos estos relatos sirvieron para crear movimientos, para evangelizar y reivindicar y, como diría Eugen Leitl, para «corromper a mentes jóvenes» y convertirlas a la causa. El hecho de que existan diferentes narrativas para prácticas idénticas constituye un hecho beneficioso: sin importar *por qué* la gente piensa que hace lo que hace, todos ellos contribuyen sin embargo al mismo misterioso propósito.

¿Un movimiento?

Para la mayoría de espectadores, el software libre y los sistemas de código abierto parecen estar saturados de discusiones enloquecidas: las guerras de *flames* y las disputas, tanto en Internet como fuera de ella, parecen dominarlo todo. En caso de asistir a un congreso donde haya *geeks* presentes —especialmente los de perfil como Raymond, Stallman y Torvalds—, podría sospecharse que las detalladas prácticas del software libre están supervisadas por las intimidatorias e histriónicas excentricidades de unos pocos líderes carismáticos cuyas aspiraciones ideológicas derivarán en una oposición divergente, incompatible y cargada de emotividad que necesariamente adoptará formas específicas e incompatibles. Nada más lejos de la realidad, curiosamente: todo este ruido y furia apenas cambia lo que la gente hace, por más que se trate de un requisito de aprendizaje. Verdaderamente se trata de un asunto visto para sentencia.

De acuerdo con la mayor parte de la literatura científica, la función de un movimiento es narrar los objetivos compartidos y reclutar a nuevos miembros. ¿Pero es esto lo que ocurre con el software libre o el código abierto?¹⁹ Para empezar, *movimiento* es un término difícil, ya que no todos los miembros definirían su participación de ese modo. Richard Stallman sugiere que el software libre es un movimiento social, mientras que el sistema de código abierto es una metodología de desarrollo. De forma similar, algunos impulsores del código abierto lo perciben como una metodología pragmática y al software libre como una filosofía dogmática. Y por más que existan entidades específicas como la FSF y la Open Source Initiative, ellas no comprenden todo el software bajo esas denominaciones. El software libre como el código abierto no son ni compañías, ni organizaciones ni consorcios (ya que no hay organizaciones con las que asociarse); tampoco son nacionales, regionales o internacionales; no son «colectivos» porque no requieren o asumen ninguna afiliación —de hecho oír a alguien aseverar «Yo pertenezco» al software libre o al código abierto sonaría absurdo a cualquiera implicado en ellos. No son turbias bandas de *hackers*, *crackers* o ladrones que se dan cita en la oscuridad de la noche, esto es, no son ninguna organización «informal», ya que no hay un equivalente formal al que imitar o anular. Y tampoco son una multitud, pues una multitud es capaz de atraer a nuevos participantes que no tienen idea de cuál es el propósito que aquella persigue; además, las multitudes son temporales, mientras que los movimientos se prolongan en el tiempo. Puede que *movimiento* sea entonces el mejor término de todos ellos, por más que a diferencia de los movimientos sociales, cuya organización e impulso se alimentan de causas compartidas o se fragmentan por disputas ideológicas, el software libre y el código abierto comparten primero prácticas y luego ideologías.

19. Sobre movimientos sociales —la analogía más próxima, desarrollada hace tiempo— véase Gerlach y Hine, *People, Power, Change*, y Freeman y Johnson, *Waves of Protest*. Con todo, los movimientos de software libre y código abierto no poseen «causas» del tipo de las de los movimientos convencionales, más allá de la perpetuación del software libre y de código abierto (véase Coleman, «Political Agnosticism»; Chan, «Coding Free Software»). De modo similar, no existe una única metodología de desarrollo que solo abarque el código abierto. Los defensores del código abierto están más que dispuestos a excluir a aquellas personas u organizaciones que sigan la misma «metodología de desarrollo» pero *sin usar una licencia de software libre* —caso del tan ridiculizado programa de «código compartido» [«*shared-source*»] de Microsoft. La lista de licencias aprobadas tanto por la FSF como por la Open Source Initiative es sustancialmente la misma. Es más, las Guías de Software Libre de Debian y la «Definición de código abierto» son casi idénticas (compárese <http://www.gnu.org/philosophy/license-list.html> con <http://www.opensource.org/licenses/> [última consulta: 18 de abril de 2018]).

Es este hecho el que supone la confirmación más rotunda de que nos encontramos ante un público recursivo, un tipo de público que está tan implicado en los medios prácticos materiales para convertirse en público como en cualquier debate público dado.

El movimiento, en cuanto práctica de debate y discusión, se centra por tanto en los acuerdos fundamentales sobre los otros cuatro tipos de prácticas. La discusión y el debate tienen así una función específica: vincular prácticas divergentes conforme a un amplio consenso que trata de plasmar el *porqué* del software libre. ¿Por qué es diferente del desarrollo normal de software? ¿Por qué es necesario? ¿Por qué ahora? ¿Por qué lo crea la gente? ¿Por qué lo utiliza? ¿Puede preservarse y mejorarse? Ninguna de estas preguntas aborda el *cómo*: ¿Cómo debería difundirse el código fuente? ¿Cómo debería redactarse una licencia? ¿Quién debería encargarse de ello? Todos estos «cómo» cambian lenta y experimentalmente a través de la modulación cuidadosa de las prácticas, pero los «porqués» son turbulentos y a menudo desconcertantes. No obstante, las personas involucradas en el software libre —usuarios, desarrolladores, defensores y observadores— difícilmente podrían permanecer calladas ante esta cuestión, a pesar de la frecuente exigencia: «Cállate y muéstrame el código». ²⁰ El acto de «figurarse» el software libre también requiere la práctica de reflexionar sobre lo que es central para él y lo que queda al margen.

El movimiento, en cuanto práctica de discusión y debate, se compone de relatos. Se trata por tanto de una práctica narrativa: relatos populares cargados afectiva e intelectualmente que orientan a los participantes existentes hacia un problema concreto, desmienten otras historias, repelen ataques del exterior y atraen a nuevos miembros.²¹ Ello incluye proselitismo y evangelización (y ahí destacan a menudo los pasados usables de

20. El origen literal de esta «frecuente exigencia» se remonta a dos episodios de los orígenes del movimiento del software libre que viene relatando Kelty: el primero, acaso menos conocido, es la entrada de Eric Raymond «*Shut up and show them the code*» («Cállate y muéstrale el código»), expresión que el autor proponía el 28 de junio de 1999 para replicar a la «retórica» de Stallman que urgía a los defensores del código abierto a «hablar de libertad» en vez de centrarse en la «excelencia» de sus programas: <http://www.catb.org/esr/writings/shut-up-and-show-them.html>; el segundo, más célebre, remite al lacónico «*Talk is cheap. Show me the code*» («Hablar es fácil. Muéstrame el código») con que Linus Torvalds respondió el 25 de agosto de 2000 a las especulaciones de un desarrollador en la lista de distribución del núcleo de Linux.. <https://lkml.org/lkml/2000/8/25/132> [N. del E.]

21. Se trata, en los términos de la Teoría del Actor-Red, de un proceso de «enrolamiento» en el que los participantes encuentran modos de alinear —y desalinear— retóricamente sus intereses, aunque ello no constituya la sustancia de su interés. Véase Latour, *Science in Action* [ed. cast: *Ciencia en acción*]; Callon, «Some Elements of a Sociology of Translation».

las reformas protestantes, las singularidades, la rebelión y la iconoclastia), ya sea para la reforma de la legislación de propiedad intelectual o para la adopción de Linux en las trincheras del EEUU corporativo. Asimismo, ello incluye tanto una lealtad sentida hacia la justicia social como un agnosticismo político despojado de toda ideología.²² Cada vez que el software libre es presentado a alguien, que se habla de él en los medios, que se lo analiza en un trabajo académico o que se instala en un centro de trabajo, se recurre a un relato del software libre o del código abierto para explicar su propósito, su impulso y su temporalidad. En los extremos se encuentran los propios profetas y prosélitos: Eric Raymond describe el código abierto como un resultado evolutivamente necesario de la tendencia natural de las sociedades humanas hacia economías de la abundancia, mientras que Richard Stallman habla de una defensa de las libertades fundamentales de creatividad y expresión, usando diversas teorías filosóficas de la libertad, la justicia y la defensa de la autonomía.²³ Incluso los análisis académicos deben partir de una historia abreviada extraída de la propia autonarración de los *geeks* que crean o defienden el software libre.²⁴ De hecho, como nota metodológica, una de las razones por las que resulta tan sencillo rastrear todas estas historias y narrativas es porque a los *geeks* les encanta contar y, más importante aún, archivar dichos relatos —crear páginas web, definiciones, entradas de enciclopedia, diccionarios y minihistorias, así como guardar cada retal de correspondencia, cada riña y cada resolución relacionadas con sus actividades. Esta «*hybris arachivística*» da como resultado un tipo de campo de trabajo muy peculiar y específico: uno en el que la observación etnográfica «en el momento en que ocurren los hechos» no solo es posible por «estar presente» en ese momento, sino también por estar presente en los enormes y proliferantes archivos de los momentos pasados. Entender el movimiento como una entidad cambiante requiere volver constantemente la vista a sus promesas sobre el futuro y a las condiciones de su cumplimiento.

Los relatos del movimiento son también los relatos de un público recursivo. El hecho de que *movimiento* no termine de ser el término correcto evidencia un modo de aprehensión, de figurarse por qué tales prácticas tienen sentido para todos estos *geeks*, en este lugar y momento

22. Coleman, «Political Agnosticism»

23. Véase, respectivamente, Raymond, *The Cathedral and the Bazaar* [ed. cast.: «La catedral y el bazar»] y Williams, *Free as in Freedom*.

24. Por ejemplo, tanto Castells, *The Internet Galaxy*, [ed. cast.: *La galaxia Internet*] como Weber, *The Success of Open Source* ofrecen versiones del mismo relato de sus orígenes y desarrollo.

concretos; una práctica que no es tan diferente a la de mi propio desempeño etnográfico con ella. Nótese que tanto el software libre como el código abierto cuentan relatos acerca de movimiento(s): no están separados por la línea divisoria comercial-no comercial, pese a sí estarlo por nociones vagas y confusas acerca de sus objetivos últimos. El problema del público recursivo (o, en una expresión alternativa, del *mercado* recursivo) como imaginario social de orden moral y técnico es común a uno y otro como parte de sus prácticas. Por ende, las narraciones sobre «el movimiento» son relatos detallados acerca del orden moral y técnico que los *geeks* habitan, y están ligados a las funciones y el destino de Internet. Muchas veces dichos relatos constituyen ellos mismos prácticas de inclusión y exclusión (p.ej. «Esta no es una licencia de software libre» o «Este software no es un sistema abierto»); y otras veces son definiciones normativas sobre cómo *debería* ser el software libre. Sea como fuere, siempre son relatos que revelan las imaginaciones morales y técnicas compartidas que configuran el software libre como un público recursivo.

Conclusión

Antes de 1998 no había movimiento. Existía la FSF, con sus propósitos peculiares, y un surtido muy amplio de otros proyectos, personas, programas e ideas. Entonces, de forma repentina, en plena efervescencia del *boom* puntocom, surgió el software libre como movimiento. Súbitamente, se convirtió en un problema, un peligro, un empleo, un llamamiento, un dogma, una solución, una filosofía, una liberación, una metodología, un plan de negocios, un éxito y una alternativa. Súbitamente, se trataba del código abierto o del software libre, y se volvió necesario alinearse a uno u otro lado. A partir de 1998, estallaron los debates sobre las definiciones, al tiempo que proliferaban las denuncias, los manifiestos y la hagiografía periodística. Curiosamente, la creación de dos nombres permitió que la gente identificara *una única cosa*, pues ambos nombres se referían a prácticas, licencias, herramientas y organizaciones idénticas. El software libre y el código abierto compartían todo lo «material» pero diferían de forma acusada y profunda con respecto a su ideología. Stallman fue denunciado como un chiflado, un comunista, un idealista y un dogmático que lastraba la adopción empresarial del código abierto. Raymond y los usuarios del «código abierto» fueron acusados de traicionar los ideales de libertad y autonomía, de diluir los principios y la promesa del software libre, así como de ser títeres de la

dominación capitalista. Entretanto, ambos grupos se dedicaron a crear objetos —principalmente software— usando herramientas en las que coincidían, conceptos de apertura en los que coincidían, licencias en las que coincidían y esquemas organizativos en los que coincidían. Y sin embargo nunca hubo un debate tan feroz acerca de la definición del software libre.

Por un lado, la FSF privilegia la libertad y creatividad de los *geeks*, individuos implicados en prácticas de autorrepresentación [«*self-fashioning*»] a través de la creación de *software*. De este modo, otorga preeminencia a la reivindicación liberal de que sin libertad de expresión los individuos son despojados de su capacidad de autodeterminación. Por otro lado, el código abierto privilegia las organizaciones y los procesos, esto es, a los *geeks* que están implicados en la creación de empresas, entidades sin ánimo de lucro u organizaciones públicas y gubernamentales de un tipo u otro. De este modo, otorga preeminencia a la visión pragmática (o polimática) de que la obtención de resultados requiere principios flexibles y capacidad de negociación, y de que la práctica pública de construir y ejecutar artefactos debería separarse de la práctica privada de las creencias éticas y políticas. Ambas narrativas proporcionan a los *geeks* formas de dar sentido a una práctica que comparten en casi todos sus detalles, así como formas de comprender cómo el software libre o el código abierto se diferencian del desarrollo predominante de software privativo que domina sus horizontes. Tales narrativas convierten la participación y la compartición azarosas existentes antes de 1998 en las actuales prácticas coherentes y orientadas por objetivos, convirtiendo una clase en sí en una clase para sí, por usar una terminología mayoritariamente desafecta para los *geeks*.

Si dos ideologías radicalmente opuestas pueden sostener a personas implicadas en prácticas idénticas, entonces parece obvio que el verdadero espacio de la política y la protesta se sitúa al nivel de estas prácticas y de su surgimiento. Tales prácticas surgen en respuesta a una reorientación del poder y el saber, una reorientación en cierto modo ajena a las narrativas convencionales de libertad o a las reivindicaciones pragmáticas de necesidad metodológica o de innovación orientada por el mercado. Si dichas narraciones convencionales fueran suficientes, las prácticas serían meros trámites burocráticos y no las transformaciones radicales que representan.

IV. COMPARTICIÓN DE CÓDIGO FUENTE

El software libre no sería nada sin el código fuente compartido. La idea se halla inscrita en el propio nombre «código abierto», y a todas las licencias de software libre se les exige que su código fuente esté abierto a estudio, no «soldado herméticamente». Acaso irónicamente, el código fuente es el más material de los cinco componentes del software libre, a la vez un medio de expresión, como la escritura o el habla, y una herramienta que ejecuta acciones concretas. El código fuente constituye una mnemotecnia que hace de traductor entre los ilegibles flujos a la velocidad electrónica de nuestras máquinas y nuestra persistente capacidad para entenderlos y controlarlos parcialmente como agentes humanos. Muchos programadores y defensores del software libre sugieren que «la información quiere ser libre»¹ y que el acto de compartir es una condición natural de la vida humana, pero yo sostengo algo opuesto: el acto de compartir produce su propio tipo de orden moral y técnico, es decir, «la información hace que las personas quieran libertad» y la manera en que la quieren está vinculada a la forma en que esa información se crea y se difunde. En este capítulo exploró la sinuosa y contingente historia de cómo el código fuente y su compartición han llegado a adoptar las formas técnicas, legales y pedagógicas que tienen hoy, y cómo las normas de compartición han llegado a parecer tan naturales para *los geeks*.

El código fuente es esencial para el software libre debido a las formas históricamente específicas en las que ha llegado a compartirse, «portarse» y «bifurcarse». Nada en la naturaleza del código fuente requiere que sea compartido, sea por corporaciones para las que el

1. Véase nota 27 del capítulo I. [N. del E.]

secretismo y la protección celosa constituyen la norma, sea por académicos y *geeks* para los que el código fuente suele ser solo una expresión, o implementación, de una idea mayor digna de compartirse. Sin embargo, en los últimos treinta años, las normas de compartición de código fuente —normas técnicas, legales y pedagógicas— han devenido una práctica aparentemente natural. Dichas normas surgieron de los intentos de convertir el software en un producto, como la «desagregación» de software y hardware realizada por IBM en 1968, de definirlo y controlarlo jurídicamente mediante leyes de secretos comerciales, *copyright* y patentes, y de enseñar a los ingenieros cómo entenderlo y crearlo.

La historia de las normas de compartición de código fuente es también, y no por accidente, la historia del sistema operativo UNIX.² UNIX es un monstruoso híbrido académico-corporativo, un experimento sobre portabilidad y compartición cuyo impacto es amplia y reverencialmente reconocido por los *geeks*, pero subestimado de forma más general. La historia de UNIX demuestra los detalles sobre cómo el código ha llegado a compartirse técnica, legal y pedagógicamente. En términos técnicos, UNIX y el lenguaje de programación C en que se escribió demostraron varias ideas clave en la teoría y práctica de los sistemas operativos, y condujeron al extendido «porte» de UNIX a prácticamente cualquier tipo de hardware disponible en la década de los 70 en todo el mundo. En términos legales, la empresa propietaria de UNIX, AT&T, lo licenció de un modo amplio y liberal tanto en forma binaria como en código fuente. La definición legal de UNIX como producto, sin embargo, no era la misma que la definición técnica de UNIX como un experimento sobre sistemas operativos portables en continua evolución —una tensión que ha continuado a lo largo de su existencia. En términos pedagógicos, UNIX se convirtió en el paradigma de un «sistema operativo» y fue por ello portado no solo en el sentido técnico de una máquina a otra, sino de máquinas a mentes, a medida que los estudiantes de ingeniería informática que

2. La «compartición» de código fuente no es el único tipo de compartición entre *geeks* (p. ej., la compartición informal para comunicar ideas) y UNIX no es el único software compartido. Otros ejemplos que exhiben esta clase de proliferación (p.ej. el lenguaje de programación LISP y el sistema de composición textual TeX) son tan ubicuos hoy como UNIX. El reverso de mi argumento es que la venta produce un tipo diferente de orden: muchos productos que existían en número mucho mayor que UNIX han desaparecido desde entonces porque nunca fueron portados o bifurcados, y ahora forman parte de los museos y colecciones de informática muerta, si es que han sobrevivido para ello.

aprendían el significado de «sistema operativo» estudiaban los detalles del casi legalmente compartido código fuente de UNIX.³

La proliferación de UNIX fue también una empresa híbrida entre lo comercial y lo académico: no era ni un objeto de «dominio público» compartido exclusivamente entre académicos ni un producto comercial convencional. La proliferación tuvo lugar a través de nuevas formas de compartición académica así como a través de esquemas de licenciamiento restringidos por el peculiar estatuto de AT&T, un monopolio regulado que tenía vetado entrar en la industria informática antes de 1984. De este modo, la proliferación no fue una mera réplica: no se trataba de vender copias de UNIX, sino de un complejo entramado de fragmentos de código fuente compartidos y vueltos a compartir, así como de la re-implementación de un esquema conceptual elegante y simple. A medida que UNIX proliferaba, quedaba estabilizado por múltiples intervenciones: de académicos que buscaban mantenerlo completo y autocompatible a través de contribuciones al código fuente; de los abogados de AT&T que buscaban marcar límites que se ajustaran a las leyes, licencias, versiones y regulaciones; y de profesores universitarios que buscaban definirlo como un modelo de los conceptos básicos de la teoría de los sistemas operativos. En todas estas formas, UNIX era una especie de público recursivo primitivo que componía a personas para las que el sentido de su afiliación era el uso, modificación y estabilización de UNIX.

La otra cara de la proliferación es la diferenciación: la bifurcación. UNIX es admirado por su integridad como entidad conceptual ypreciado (u objeto de fascinación) por su extremadamente enmarañado árbol genealógico de portes y bifurcaciones: nuevas versiones de UNIX, algunas basadas directamente en el código fuente y otras no, algunas licenciadas directamente por AT&T, otras sublicenciadas o completamente independientes.

3. La historia de UNIX está aún por contar, y sin embargo se ha contado cientos de miles de veces. Cada *hacker*, programador, ingeniero informático y *geek* narra su versión de la historia de UNIX —un pasado usable. En consecuencia, las fuentes de este capítulo incluyen dichas versiones, oídas y registradas durante mi trabajo de campo, pero también fácilmente accesibles en obras académicas sobre software libre, las cuales participan entusiásticamente de esta narración de historia abreviada. Véase, por ejemplo, Steven Weber, *The Success of Open Source; Castells, The Internet Galaxy* [ed. cast.: *La galaxia Internet*]; Himanen, *The Hacker Ethic* [ed. cast.: *La ética del hacker*]; Benkler, *The Wealth of Networks* [ed. cast.: *La riqueza de las redes*]. Hasta la fecha no hay más que una historia detallada de UNIX—*A Quarter Century of UNIX*, de Peter Salus— sobre la que me apoyo extensamente. La tesis de Matt Ratto «The Pressure of Openness» también contiene una excelente historia analítica de los sucesos relatados en este capítulo.

La bifurcación, al igual que la mutación aleatoria, ha tenido efectos tanto positivos como negativos: por un lado, acabó creando versiones de UNIX que no eran compatibles entre sí (una especie de respuesta autoinmune), pero también permitió la fusión de UNIX y Arpanet, creando una situación en que los sistemas operativos de UNIX se convirtieron, no solo en el paradigma de los sistemas operativos, sino también en el paradigma de los ordenadores *conectados en red*, a través de su intersección con el desarrollo de los protocolos TCP/IP que componen el núcleo de Internet⁴. A mediados de los 80, UNIX era una especie de lugar de paso obligatorio para cualquier persona interesada en las redes, los sistemas operativos, Internet y especialmente los modos de crear, compartir y modificar código fuente —hasta el punto de que UNIX es famoso entre los *geeks* no solo como un sistema operativo sino como una filosofía, una respuesta a una pregunta muy antigua con nuevos ropajes: ¿cómo viviremos entre un nuevo mundo de máquinas, software y redes?

Antes del código fuente

En los comienzos de la maquinaria informática, no existía nada parecido al código fuente. A Alan Turing supuestamente le gustaba hablar con la máquina en binario. Grace Hopper, inventora de un pionero compilador, trabajó tan cerca del Harvard Mark I como pudo: accionando interruptores y conectando y desconectando relés que componían el «código» de lo que la máquina hacía. Semejante trabajo mecánico y meticuloso apenas es digno de los términos *lectura* y *escritura*; no había instrucciones GOTO, ni números de línea, tan solo cálculos que debían ser traducidos desde la escritura pseudo-matemática de ingenieros y computadores humanos a una configuración física o mecánica⁵. Escribir y leer código fuente y lenguajes de programación fue un desarrollo largo y lento que solo devino relativamente generalizado a mediados de los 70. Los llamados lenguajes de alto nivel comenzaron a aparecer a finales de los 50: FORTRAN, COBOL, Algol y los «compiladores» que permitían que los programas escritos en ellos se transformaran en las

4. La intersección de UNIX y TCP/IP se produjo en torno a 1980 y llevó al famoso desplazamiento del NCP (*Network Control Protocol*, Protocolo de Control de Red) al TCP/IP (*Transmission Control Protocol/Internet Protocol*, Protocolo de Control de Transmisión/Protocolo de Internet) que ocurrió el 1 de enero de 1983 (véase Salus, *Casting the Net*).

5. Light, «When Computers Were Women»; Grier, *When Computers Were Human*.

ilegibles representaciones mecánicas y valvulares de la máquina. Fue en esta época cuando surgieron los términos *lenguaje fuente* y *lenguaje destino* para designar la actividad de traducir entre lenguajes de alto y bajo nivel.⁶

Hay una cierta ironía respecto de los ordenadores que suele pasarse por alto: si damos crédito a las ubicuas proclamaciones de la potencia sin rival de los ordenadores, esta se basa en su programabilidad *general*, esto es, en que en principio son capaces de realizar cualquier cálculo, de lo que la llamada máquina universal de Turing proporciona la prueba matemática⁷. A pesar de la fuerza abstracta de tal certeza, lo cierto es que no vivimos en el mundo de El Ordenador —vivimos en un mundo de *ordenadores*. Los sistemas de hardware que los fabricantes crearon de la década de los 50 en adelante eran tan específicos e idiosincráticos que resultaba inconcebible que pudiera escribirse un programa para una máquina y luego simplemente ejecutarlo en otra. La «programación» se convirtió en una práctica hecha a medida, adaptada a cada nueva máquina, y por más que los programadores de una determinada máquina bien pudieran compartir programas entre ellos, no habrían visto mucho sentido en compartirlos con usuarios de otra máquina distinta. De la misma manera, los ingenieros informáticos compartían descripciones matemáticas de algoritmos e ideas para la automatización con tanto entusiasmo como celo empleaban las corporaciones en guardar los suyos, pero tales compartición y secretismo no se extendieron a la

6. Existe una abundante y creciente literatura académica sobre la historia del software: Wexelblat, *History of Programming Languages* y Bergin y Gibson, *History of Programming Languages 2* son recopilaciones de artículos de historiadores y participantes. Entre las obras históricas clave figuran Campbell-Kelly, *From Airline Reservations to Sonic the Hedgehog*; Akera y Nebeker, *From 0 to 1*; Hashagen, Keil-Slawik y Norberg, *History of Computing—Software Issues*; Donald A. MacKenzie, *Mechanizing Proof*. Michael Mahoney es quien más ha escrito con diferencia sobre la temprana historia del software; sus obras relevantes incluyen «The Roots of Software Engineering», «The Structures of Computation», «In Our Own Image» y «Finding a History for Software Engineering». Sobre UNIX en particular, hay una asombrosa escasez de literatura histórica. Martin Campbell-Kelly y William Aspray le dedican solo dos páginas en su historia general *Computer*. Ya en 1978, Ken Thompson y Dennis Ritchie reflexionaban sobre la «historia» de UNIX en «The UNIX Time-Sharing System: A Retrospective». Ritchie mantenía un sitio web [aún accesible] con una valiosa colección de documentos antiguos junto con sus propias reminiscencias (<http://cm.bell-labs.co/who/dmr/>). Mahoney también ha realizado entrevistas con los principales participantes en el desarrollo de UNIX en Laboratorios Bell. Estas entrevistas nunca han sido publicadas, pero me han servido de trasfondo para este capítulo (las entrevistas se hallan en los archivos personales de Mahoney).

7. Turing, «On Computable Numbers». Véase también Davis, *Engines of Logic*, para una explicación básica.

compartición del programa en sí mismo. La necesidad de «reescribir» un programa para cada máquina no era un simple accidente histórico, sino que venía determinada por las necesidades de los diseñadores e ingenieros, así como por las vicisitudes mercantiles de unas máquinas tan costosas.⁸

En los buenos viejos tiempos de los ordenadores del tamaño de habitaciones, los lenguajes que los humanos usaban para programar computadoras eran mnemotécnicos; no existían en el ordenador sino en un trozo de papel o en una hoja de código específicamente diseñada. Esta hoja de código otorgaba a los humanos que no eran Alan Turing una manera de seguir la pista de, compartir con otros humanos y pensar sistemáticamente sobre los invisibles cálculos a la velocidad de la luz de un dispositivo complicado. Tal mnemotecnia necesitaba ser «codificada» en tarjetas o cintas perforadas; y si los ingenieros discutían, lo hacían sobre hojas de papel que se acoplaban con cables, relés e interruptores —o luego con las impresiones de los diferentes códigos específicos de máquinas que representaban el programa y los datos.

Con la introducción de los lenguajes de programación, la distinción entre lenguaje «fuente» y lenguaje «destino» pasó al terreno de la práctica: los lenguajes fuente se «traducían» al ilegible lenguaje destino de la máquina. Tales lenguajes fuente de alto nivel eran todavía una suerte de mnemotecnia —ciertamente más fáciles de leer y escribir por humanos, mayormente sobre amarilleantes libretas de papel u hojas de código especiales—, pero también eran lo bastante estructurados como para poder introducir un lenguaje fuente en un ordenador y traducirlo al lenguaje destino que los diseñadores del equipo hubieran especificado. Los comandos y tarjetas de entrada y el código fuente precisaban una serie de acciones específicas para cada máquina: un lector particular de tarjetas o, más tarde, una perforadora de tarjetas con un «editor» especial para introducir los comandos. El código fuente que se introducía y traducía correctamente proporcionaba a la máquina un programa binario ensamblado que, de hecho, funcionaba (calculaba, operaba, controlaba...). Se trataba de una separación, una abstracción

8. La compartición de programas tiene sentido en este periodo solo en los términos de los grupos de usuarios como SHARE (IBM) y USE (DEC). Estos grupos de hecho estaban compartiendo el código fuente y los programas que habían escrito (véase Akera, «Volunteerism and the Fruits of Collaboration»), pero se constituyán en torno a máquinas y fabricantes específicos; la lealtad de marca y la personalización eran pues aspiraciones familiares, pero no así la compartición de código fuente entre ordenadores distintos.

que permitía una cierta división del trabajo entre los ingeniosos autores humanos y las rápidas y mecánicas máquinas de traducción.

Incluso tras la invención de los lenguajes de programación, la programación «en» un ordenador —sentarse frente a una pantalla brillante a hackear toda la noche— quedaba aún muy distante en el tiempo. Por ejemplo, hubo que esperar a 1969 para poder sentarse frente a un teclado, escribir el código fuente, instruir al ordenador para que lo compilase y finalmente ejecutar el programa —todo ello sin dejar el teclado—, una actividad que era del todo inimaginable en la época primigenia del «procesamiento por lotes».⁹ Muy pocos programadores trabajaron de esta manera antes de mediados de los 70, cuando los editores de texto que permitían a los programadores ver el texto en pantalla en lugar de en una hoja de papel comenzaron a proliferar¹⁰. A estas alturas estamos tan familiarizados con la imagen del hombre o la mujer sentados frente a la pantalla interactuando con su dispositivo que es prácticamente imposible imaginar cómo se llegó en primer término a una práctica tan aparentemente obvia —por medio de la lenta acumulación de las herramientas y técnicas para trabajar en un nuevo tipo de escritura— y cómo dicha práctica explotó en un Babel de lenguajes y máquinas que traicionaba la promesa de una máquina informática de propósito general.

La proliferación de diferentes máquinas con diferentes arquitecturas impulsó un deseo, especialmente entre los académicos, de estandarización de los lenguajes de programación, no tanto porque un lenguaje en particular fuera mejor que otro, sino porque parecía necesario para evitar inventar la rueda con cada nueva máquina que la mayoría de ingenieros y usuarios informáticos compartiera un corpus emergente de algoritmos, soluciones y técnicas de todo tipo. Algol, un lenguaje simplificado [«streamlined language»] adaptado a las representaciones algorítmicas y algebraicas, surgió a principios de los 60 como un candidato para la estandarización internacional. Otros lenguajes com-

9. Véase Waldrop, *The Dream Machine*, pp. 142-147.

10. En los 70 se creó un gran número de editores; El EMACS de Richard Stallman y el vi de Joy siguen siendo los más célebres. A Engelbart se le atribuye con generosidad en cierto modo excesiva la creación del ordenador interactivo, aunque el trabajo de Butler Lampson y Peter Deutsch en Berkeley, así como el del equipo Multics, Ken Thompson y otros sobre los primeros editores en pantalla son seguramente más sustanciales respecto de los problemas e ideas fundamentales para la manipulación de archivos de texto en pantalla. Esta historia permanece en gran medida sin documentar, excepto en la propia literatura de ingeniería informática. Sobre Engelbart, véase Bardini, *Bootstrapping*.

petían con diferentes fortalezas: FORTRAN y COBOL se orientaban al uso empresarial general y LISP al procesamiento simbólico. Al mismo tiempo, el deseo de un lenguaje «de alto nivel» estándar requería un bestiario de programas de traducción: compiladores, analizadores sintácticos y léxicos y otras herramientas destinadas a transformar el lenguaje de alto nivel (legible por humanos) en un lenguaje de bajo nivel específico para la máquina, es decir, en un lenguaje de máquina, en un lenguaje ensamblador y finalmente en los místicos ceros y unos que recorren nuestras máquinas. La idea de un lenguaje estándar y la necesidad de concebir herramientas de traducción específicas están en el origen del problema de la *portabilidad*: la habilidad de transferir software —no solo buenas ideas sino programas reales, escritos en un lenguaje estándar— de una máquina a otra.

La percepción era que un lenguaje fuente estándar abriría una vía para contrarrestar la proliferación de diferentes máquinas con arquitecturas sutilmente diferentes. El código fuente portable permitiría a los programadores imaginar sus programas como barcos, que arribarían a puertos de escala y atracarían en plataformas distintas, pero permaneciendo esencialmente móviles e inalterados por dichos puertos de escala. El código fuente portable se convirtió en el esperanto de los humanos que habían forjado su propio Babel de equipos informáticos tribales.

Mientras tanto, para la industria informática de los 60 el código fuente portable aparecía como un asunto irrelevante. El software y el hardware constituían las dos caras de una misma y extremadamente cara moneda —salvo a los ingenieros, a nadie le preocupaba en qué lenguaje se escribiera el código, siempre y cuando desarrollase la tarea en cuestión para los clientes. Cada nueva máquina necesitaba ser diferente, más rápida y más grande, al principio, y luego más pequeña, que la anterior. El ansia de diferenciar unas máquinas de otras no estaba guiada por una experimentalidad académica o una pureza estética, sino por una demanda de comerciabilidad, ventaja competitiva y transformación de las máquinas y el software en productos. Cada máquina tenía que hacer algo realmente bien y tenía que desarrollarse en secreto con el fin de desbancar los diseños e innovaciones de los competidores. En los 50 y 60, el software era un componente nuclear de este objeto comercializable; no era algo que en sí mismo se diferenciase o distribuyese separadamene —hasta la famosa decisión de IBM en 1968 de «desagregar» software y hardware.

Antes de los 70, los empleados de una corporación informática escribían el software en el seno de la propia empresa: la máquina era el producto y el software solo era un artículo extra en la factura. Con todo, IBM no fue la primera empresa en concebir el software como un producto independiente con su propio mercado, pues antes de ella dos compañías, Informatics y Applied Data Research, habían explorado las posibilidades de un mercado de software diferenciado¹¹. En particular, la empresa Informatics desarrolló el primer producto de software comercialmente exitoso, un sistema de gestión empresarial llamado Mark IV, que costaba 30 000 dólares en 1967. El presidente de Informatics, Walter Bauer, «recordó luego que los compradores potenciales se quedaron ‘pasmados’ ante el precio de Mark IV. En un mundo acostumbrado al software gratuito, un precio de 30 000 dólares era realmente elevado».¹²

La desagregación decidida por IBM supuso un parteaguas, el momento en que el código fuente «portable» se volvió una idea concebible, si no una realidad práctica, para muchas empresas informáticas.¹³ En lugar de proporcionar un paquete completo de hardware y software, IBM decidió diferenciar sus productos, vendiendo a sus clientes el software y el hardware por separado.¹⁴ Sin embargo, la portabilidad no era simplemente una cuestión técnica, sino también político-económica. La decisión de IBM fue impulsada tanto por el deseo de crear software corporativo que pudiera ejecutarse en todas sus máquinas (un objetivo central del famoso proyecto OS/360 supervisado y diagnosticado por Frederick Brooks) como por responder a una demanda antimonopolística interpuesta por el Departamento de Justicia estadounidense.¹⁵ Dicha demanda incluía como parte de sus alegaciones la insinuación de que la fuerte ligazón entre el

11. Véase Campbell-Kelly, *From Airline Reservations to Sonic the Hedgehog*.

12. *Ibid.*, p. 107.

13. Campbell-Kelly y Aspray, *Computer*, pp. 203-205.

14. En última instancia, el caso del Departamento de Justicia contra IBM usaría la agregación («*bundling*») como prueba de práctica monopolística, sumada a las alegaciones sobre la creación de las llamadas *Plug Compatible Machines* (Máquinas de Conectores Compatibles), dispositivos que fueron sometidos a ingeniería inversa mediante la meticulosa construcción tanto de la interfaz mecánica como del software que se comunicaría con los ordenadores centrales de IBM. Véase Franklin M. Fischer, *Folded, Spindled, and Mutilated*; Brock, *The Second Information Revolution*.

15. La historia de este proyecto y las lecciones que Brooks aprendió de él constituyen el tema de uno de los manuales de desarrollo informático más famosos, *The Mythical Man-Month*, de Frederick Brooks.

software y el hardware de la empresa representaba una forma de práctica monopolística, e inducía a IBM a considerar estrategias para «desagregar» sus productos.

En el mundo empresarial, sin embargo, la portabilidad significaba algo específico. Incluso si el software podía hacerse portable a nivel técnico —transferible entre dos máquinas distintas de IBM—, esto no constituía ninguna garantía de que fuera portable entre consumidores. El programa de contabilidad de una empresa, por ejemplo, podría no ajustarse a las prácticas de otra. Por tanto, la portabilidad venía dificultada tanto por la diversidad de las arquitecturas mecánicas como por la diversidad de la organización y las prácticas empresariales. De ahí que IBM y otros fabricantes no vieran ningún beneficio en la estandarización del código fuente, pues ello solo podría proporcionar ventaja a los competidores.¹⁶

Así pues, la portabilidad no era simplemente un problema técnico —el problema de ejecutar un programa en diversas arquitecturas— sino también una especie de problema político-económico. El significado de *producto* no siempre coincidió con el de *hardware* o *software*, sino que usualmente suponía una combinación de ambos. En esa primera etapa, los contornos de una competición sobre el significado de un código fuente *portable* o *compartible* son visibles, tanto en los desafíos técnicos de crear lenguajes de alto nivel como en los desafíos político-económicos a los que las corporaciones hacían frente al crear característicos productos patentados.

El sistema UNIX de tiempo compartido

Con este telón de fondo, la invención, el éxito y la proliferación del sistema operativo UNIX parecen bastante monstruosos, una aberración de prácticas tanto académicas como comerciales que debería haber fracasado en ambos terrenos en vez de convertirse en el sistema operativo portable más ampliamente usado de la historia y en el paradigma mismo de un «sistema operativo» en general. La historia de UNIX muestra cómo la portabilidad se hizo realidad y cómo a partir de aquí la particular práctica

16. La industria informática siempre ha dependido fuertemente de los secretos comerciales, mucho más que de las patentes o el *copyright*. La legislación de secretos comerciales produce asimismo su propia forma de orden, acceso y difusión, la cual se traspuso también a la industria informática inicial. Véase Kidder, *The Soul of a New Machine* para un relato clásico sobre el secretismo y la competencia en la industria informática.

de compartición del código fuente de UNIX se convirtió en una especie de estándar *de facto*.

UNIX fue escrito por primera vez en 1969 por Ken Thompson y Dennis Ritchie en los Laboratorios Bell de Murray Hill (Nueva Jersey). UNIX era el desenlace del proyecto Multics del MIT, que Laboratorios Bell había financiado parcialmente y al que se asignó a Ken Thompson. Multics fue uno de los primeros sistemas operativos de tiempo compartido completos, una plataforma de demostración de numerosas innovaciones en los sistemas de tiempo compartido (múltiples usuarios trabajando en el mismo ordenador).¹⁷ Hacia 1968, Laboratorios Bell retiró su apoyo al proyecto Multics y envió a Ken Thompson de vuelta a Murray Hill, donde él y Dennis Ritchie se vieron desprovistos de equipo, dinero y proyecto. Ambos eran especialistas en sistemas operativos, lenguajes de programación y arquitectura mecánica en un grupo de investigación que carecía de financiación y de mandato para dedicarse a estas áreas. Mediante el uso creativo de algunas máquinas desechadas, y en relativo aislamiento del resto del laboratorio, Thompson y Ritchie crearon, en un periodo aproximado de dos años, un sistema operativo completo, un lenguaje de programación llamado C y un montón de herramientas que siguen usándose pródigamente hoy. El nombre UNIX (originalmente UNICS: *Uniplexed Information and Computing Service*, Servicio de Computación e Información Uniplexado) era, entre otras cosas, un juego de palabras pueril: un Multics castrado.

La ausencia de un mandato económico o corporativo para la creatividad y el trabajo de Thompson y Ritchie no era inusual en Laboratorios Bell: los investigadores tenían libertad de trabajar en casi cualquier cosa, siempre y cuando poseyese algún tipo de vaga relación con los intereses de AT&T. No obstante, la falta de financiación para una máquina más potente sí que restringió el trabajo que Thompson y Ritchie pudieron llevar a cabo. En particular, influyó en el diseño del sistema, orientado hacia una unidad de control ultrarrreducida (un *kernel*) que gobernaba el funcionamiento básico de la máquina, y un conjunto expandible de herramientas pequeñas e independientes que hacían una cosa bien y que podían engarzarse para realizar tareas más complejas y potentes.¹⁸ Final-

17. Sobre tiempo compartido, véase Lee *et al.*, «Project MAC». Multics hace su aparición en casi todas las historias de la informática, siendo de lejos el mejor recurso al respecto la página web de Tom van Vleck <http://www.multicians.org/>.

18. Algunas innovaciones técnicas ampliamente admiradas (muchas de las cuales se tomaron prestadas de Multics) incluyen: el sistema de archivos jerárquico, el intérprete de

mente, y gracias a la ayuda de Joseph Ossana, Douglas McIlroy y otros, Thompson y Ritchie se las arreglaron para exigir un nuevo ordenador PDP-11/20 no basado en los méritos técnicos del sistema operativo UNIX en sí, sino en sus aplicaciones potenciales, especialmente en las vinculadas al grupo de preparación de texto, interesado en desarrollar herramientas de composición textual, tipográfica y de impresión, primordialmente con el propósito de crear aplicaciones patentadas, lo que para Laboratorios Bell, y más generalmente para AT&T, constituía obviamente un objetivo loable¹⁹.

UNIX fue único por muchas razones técnicas, pero también por una razón específicamente económica: nunca fue del todo académico ni del todo comercial. Martin Campbell-Kelly apunta que UNIX era un «sistema operativo no privativo de enorme trascendencia»,²⁰ aunque su uso de «no privativo», sin ser sorprendente, es incorrecto. Aunque a lo largo de los 80 y comienzos de los 90 el discurso empresarial contrapusiera habitualmente *abierto a privativo* (y UNIX, sin duda, era lo primero), el lapsus de Kelly marca claramente la confusión entre la propiedad y la distribución del software que permea tanto las concepciones populares como las académicas. Y es que UNIX sí que era privativo —Laboratorios Bell, y con ella Western Electric y AT&T, ostentaban su *copyright* y su plena propiedad, aunque no lo comercializaran o promocionaran exactamente. En lugar de eso, AT&T, permitió a particulares y empresas instalar UNIX y crear derivados similares a UNIX *por un canon de licencia muy reducido*. Hasta aproximadamente 1982, se concedieron muchas licencias de UNIX a académicos por una suma muy pequeña: normalmente se trataba de licencias exentas de derechos de autor con una cuota mínima de servicio (de entre 150 y 800 dólares).²¹ Estas

comandos para interactuar con el sistema, la decisión de tratar todo, incluidos los dispositivos externos, como el mismo tipo de entidad (un archivo), el operador de «tuberías» que permitía canalizar la salida de datos de una herramienta como datos de entrada para otra, facilitando la creación sencilla de tareas complejas a partir de herramientas simples.

19. Salus, *A Quarter Century of UNIX*, pp. 33-37.

20. Campbell-Kelly, *From Airline Reservations to Sonic the Hedgehog*, p. 143.

21. La página web de Ritchie contiene una copia de la licencia de 1974 (<http://cm.bell-labs.co/who/dmr/licenses.html>) y una serie de anuncios que ejemplifican el difícil posicionamiento de UNIX como producto comercial (<http://cm.bell-labs.com/cm/cs/who/dmr/unixad.html>). De acuerdo con Don Libes y Sandy Ressler: «Las licencias originales eran licencias de código fuente. [...] Las instituciones comerciales pagaban cánones del orden de 20 000 dólares. Si tenías más de un ordenador, tenías que comprar licencias del código binario para cada máquina adicional [no se permitía que copiaras el código fuente y lo instalaras] en la que quisieras instalar UNIX. El precio de estas licencias se fijó razonablemente en 8000 dólares,

licencias permitían a los investigadores hacer lo que quisieran con el software con tal de que lo mantuvieran en secreto: no podían distribuirlo o usarlo fuera de sus laboratorios universitarios (o usarlo para crear cualquier producto o proceso comercial), ni tampoco publicar ninguna parte del mismo. Como resultado, a lo largo de los 70 UNIX fue desarrollado por Thompson y Ritchie en el seno de Laboratorios Bell y por múltiples usuarios de todo el mundo de una manera relativamente informal. Laboratorios Bell siguió esta política tan liberal tanto porque era uno de los pocos centros académico-industriales de I+D como porque AT&T era un monopolio estatal que suministraba servicios telefónicos en EEUU y, como tal, tenía vetada la entrada directa en el mercado de software informático.²²

Este estatus fronterizo entre lo empresarial y lo académico tenía una doble implicación: por un lado, UNIX estaba resguardado de las demandas de los directivos y los mercados, lo que le permitió alcanzar la integridad conceptual que lo hizo tan atractivo para diseñadores y académicos; por el otro, AT&T trataba UNIX como un producto potencial en la incipiente industria de software, lo cual incluía nuevas cuestiones legales vinculadas a un cambiante régimen de propiedad intelectual, formas novedosas de mercadotecnia y distribución y nuevos métodos de desarrollo, mantenimiento y distribución del software.

A pesar de este estatus fronterizo, UNIX fue un éxito fenomenal. Las razones de la popularidad de este sistema operativo son múltiples: era ampliamente admirado por su estética, por su tamaño y por sus ingeniosas concepción y herramientas. Pero el hecho de que se difundiera tan extensa y rápidamente es también testimonio de la existencia de una comunidad entusiasta de académicos e ingenieros (y algunos *amateurs*) sobre la que el proyecto arrancó de forma autosostenida, usuarios a los que un sistema operativo potente, flexible, barato, modificable y rápido les pareció una suerte de revelación. En efecto, UNIX representaba una alternativa obvia para los complejos, mal documentados y defectuosos sistemas operativos que rutinariamente venían instalados por defecto

considerando que no podías revenderlas. Por otro lado, las instituciones educativas podían comprar licencias del código fuente por unos cientos de dólares —lo justo para cubrir los gastos de administración de Laboratorios Bell y el coste de las cintas» (*Life with UNIX*, pp. 20–21).

22. Según Salus, esta práctica de concesión de licencias también fue resultado directo de un decreto de consentimiento antimonopolístico dictado en 1956 por el Juez Thomas Meaney en el cual se exigía a AT&T revelar y conceder licencias sobre sus patentes por tarifas nominales (*A Quarter Century of UNIX*, p. 56); véase también Brock, *The Second Information Revolution*, pp. 116-120.

en las máquinas que compraban las universidades y centros de investigación. En otras palabras, «funcionaba».

Un aspecto clave de la popularidad de UNIX era la inclusión de su código fuente. Al conceder una licencia de UNIX, Laboratorios Bell normalmente proporcionaba una cinta que contenía la documentación (esto es, la documentación que formaba parte del sistema, no un manual de instrucciones técnicas externo a él), una versión binaria del software y el código fuente. La práctica de distribuir el código fuente alentó a las personas a mantenerlo, extenderlo, documentarlo —y también a compartir este trabajo con Thompson y Ritchie. Al hacer esto, los usuarios desarrollaron un interés en mantener y apoyar el proyecto precisamente porque este les daba la oportunidad y las herramientas para usar sus ordenadores creativa y flexiblemente. Semejante comunidad mundial de usuarios organizados primordialmente por su interés en el mantenimiento de un sistema operativo constituye un precedente del público recursivo, por más que confinada al mundo de los ingenieros informáticos e investigadores con acceso a máquinas aún relativamente caras. Como tal, UNIX no era solo un software cuasicomercial extensamente compartido (esto es, distribuido de un modo distinto del de los mercados minoristas basados en los precios), sino también el primero en incluir sistemáticamente el código fuente como parte de su distribución, lo que aumentaba su atractivo para académicos e ingenieros.²³

Durante la década de los 70, el canon de licencia reducido, la inclusión del código fuente y su integridad conceptual llevaron a que UNIX fuera portado a un importante número de máquinas distintas. En muchos sentidos, para los académicos implicarse en la creación y mejora de un sistema de vanguardia mediante la obtención de una licencia y el porte del software por sí mismos era tanto o más atractivo que recibirlo preinstalado, sin su código fuente, de una empresa. Peter Salus, por ejemplo, sugiere que la gente experimentó la ausencia de mantenimiento técnico por parte de Laboratorios Bell como una especie de acicate para desarrollar y compartir sus propios ajustes. Los

23. Incluso en el campo de la ingeniería informática, rara vez se compartía formalmente el código fuente, siendo más probable su presentación en forma de teoremas y pruebas, o en diversos lenguajes de alto nivel idealizados como el lenguaje MIX de Donald Knuth destinado a presentar algoritmos (Knuth, *The Art of Computer Programming*). Es mucho más probable hallar fragmentos de código fuente impresos en libros, manuales, guías y otras publicaciones profesionales orientadas a la instrucción de programadores.

medios por los que se compartía el código fuente, y las normas y prácticas de compartición, porte, bifurcación y modificación del código fuente se desarrollaron en este periodo como parte del desarrollo del propio UNIX —el diseño técnico del sistema facilita y en algunos casos refleja las normas y prácticas de la compartición que se desarrollaron: sistemas operativos y sistemas sociales.²⁴

Compartiendo UNIX

En el periodo que va de 1974 a 1977 la difusión y portabilidad de UNIX fueron fenomenales para un sistema operativo que carecía de un sistema formal de distribución y del apoyo oficial de la empresa a la que pertenecía, y que evolucionaba gradualmente a través de las contribuciones de personas de todo el mundo. En 1975 ya había formado un grupo de usuarios llamado USENIX.²⁵ Por la misma época UNIX se había extendido a Canadá, Europa, Australia y Japón, y al tiempo que se difundían de modo independiente numerosas herramientas y aplicaciones, estas se incluían en las frecuentes actualizaciones de Laboratorios Bell. Durante todo este tiempo, el departamento de licencias de AT&T trató de encontrar un equilibrio entre el permiso a la continuidad de dichas difusión e innovación y el intento de mantener el estatus de secreto comercial de su software. Llegado 1980, UNIX era sin duda el secreto comercial más amplia y profundamente difundido de la historia de la informática.

La manera en la que se daban la difusión y la contribución a UNIX no está bien documentada, pero incluye formas de compartición tanto técnicas como pedagógicas. Por el lado técnico, la distribución adoptó diversas formas, a la vez resistentes a los intentos de AT&T de controlarla y facilitadas por su inusualmente liberal política de licencias. Por el lado pedagógico, UNIX se convirtió enseguida en un objeto paradigmático para los estudiantes de informática precisamente por

24. Suele aludirse al desarrollo simultáneo del sistema operativo y de las normas para crearlo, compartirlo, documentarlo y extenderlo como la «filosofía UNIX». Dicha filosofía incluye la idea central de que se debería construir sobre la base de las ideas (y el software) de los demás (véase Gancarz, *The Unix Philosophy and Linux and the UNIX Philosophy*). Véase también Raymond, *The Art of UNIX Programming*.

25. Laboratorios Bell amenazó el naciente boletín de noticias *UNIX NEWS* con una demanda por infracción de marca registrada, por lo que «USENIX» supuso una concesión que evocaba el original USE, un grupo de usuarios de máquinas DEC, pero que evitaba usar explícitamente el nombre UNIX. Libes y Ressler, *Life with UNIX*, p. 9.

ser un sistema operativo funcional que incluía el código fuente y era lo bastante simple como para ser explorado en uno o dos semestres.

En su libro *A Quarter Century of UNIX*, Salus proporciona un par de relatos clave (de Ken Thompson y de Lou Katz) sobre cómo funcionaba exactamente la compartición técnica de UNIX, cómo puede distinguirse entre compartición, porte y bifurcación, y cómo toda esa actividad no era ni estrictamente legal ni deliberadamente ilegal en este contexto. Para empezar, Ken Thompson cuenta:

Lo primero que hay que comprender es que el mundo exterior funcionaba con versiones de UNIX (V4, V5, V6, V7), pero nosotros no. Nuestra visión era la de un *continuum*. Así, la versión V5 no era más que lo que teníamos en un momento dado, y probablemente se quedara obsoleta simplemente por la actividad requerida para ponerla en una forma exportable. Después de la V6, yo me disponía a irme a dar clases a Berkeley durante un año, así que me puse a preparar un sistema que llevarme allí. Dado que al final obtuve una versión casi publicable, hice un *diff* de la V6 [una cinta que solo contenía las diferencias entre la última versión publicada y la que Ken se iba a llevar]. De camino a Berkeley paré en Urbana-Champaign para saludar a Greg Chesson [...], le dejé la cinta *diff* y le dije que no me importaba si circulaba por ahí.²⁶

La necesidad de que una cinta magnética «circulara» marca la diferencia entre la década de los 70 y la actualidad: la distribución de software entrañaba tanto el transporte físico de los soportes *como* la copia digital de la información. El deseo de distribuir revisiones de errores (la cinta *diff*) encuentra eco en el posterior surgimiento del software libre: el hecho de que otros usuarios hubieran corregido errores y compartido esta labor con Thompson y Ritchie generó una obligación de hacer lo más visibles posible dichas correcciones, de modo que pudiesen ser a su vez portadas a nuevas máquinas. Por otra parte, Laboratorios Bell habría visto esto desde el prisma del desarrollo de software, que exigiría cada vez una nueva publicación, una renegociación del contrato y nuevos cánones de licencia. La noción de Thompson de un «*continuum*» en vez de una serie de publicaciones también marca la diferencia entre la idea de un conjunto común de objetos que evoluciona bajo la guía de

26. Salus, *A Quarter Century of Unix*, p. 138.

diversas personas en lugares lejanos y la idea del software como «producto» empaquetado, como un artículo de consumo económico, que ganaba ascendencia por esa misma época. Al decir «el mundo exterior», Thompson no solo se refiere a personas externas a su empresa sino a la visión del mundo de los abogados y comerciales Laboratorios Bell que optarían por crear una nueva versión. Para los abogados, la difusión del código fuente era problemática porque era necesario estabilizarlo, no tanto por razones comerciales como legales —para conceder una nueva licencia por cada nueva versión. La distribución de actualizaciones, ajustes y especialmente de nuevas herramientas y añadidos escritos por gente que no trabajaba para Laboratorios Bell embrolló la claridad judicial aunque al mismo tiempo reforzara la calidad técnica. Lou Katz lo explica así:

Un gran número de ajustes nos vinieron hechos, y en vez de publicarlos uno a uno, Ken preparó una cinta recopilatoria («los 50 ajustes») [la misma «cinta *diff*», presumiblemente]. Algunos eran bastante importantes, aunque no recuerdo ninguno en particular. *Sospecho que una fracción significativa de los ajustes fue realizada por personas ajenas a Bell.* Ken trató de distribuir la cinta, pero los abogados postergaron la distribución una y otra vez. Finalmente, y para gran disgusto, alguien «encontró una cinta en la Mountain Avenue» [la localización de Laboratorios Bell] que contenía todos los ajustes. Cuando los abogados se enteraron, llamaron a todos los titulares de licencia y les amenazaron con graves consecuencias si no destruían la cinta, tras intentar averiguar cómo les había llegado. Supongo que nadie iba a decirles cómo consiguió la cinta en realidad (yo al menos no lo hice).²⁷

La distribución de los ajustes no solo implicó una lucha de poder entre ingenieros y directivos, sino que de hecho vino claramente motivada por el hecho de que, como explica Katz, «una fracción significativa de los ajustes fue realizada por personas ajenas a Bell». Esto significaba dos cosas: primero, que había un claro incentivo a devolver el sistema actualizado a esas personas y a otras; segundo, que no era obvio que AT&T efectivamente poseyera o pudiera reclamar derechos sobre dichos ajustes —o que si lo hacía, necesitaba cubrir sus huellas legales, lo que quizá explica en parte las demoras y amenazas de los abogados,

27. *Ibid.*, el énfasis es mío.

que podían haber estado ganando tiempo para preparar una versión «legal», con los permisos apropiados.

Esta lucha debería contemplarse, no tanto como una pugna entre las fuerzas rebeldes del desarrollo de UNIX y el imperio del mal de los abogados y directivos, sino como una pugna entre dos maneras de estabilizar el objeto conocido como UNIX. Para los abogados, la estabilidad implicaba hallar modos de hacer pasar UNIX por un producto adaptado al marco legal existente y a las particulares demandas derivadas de pertenecer a un monopolio regulado que tenía prohibido competir libremente con otros fabricantes de equipos. En suma, la empresa estaba estrictamente obligada a rendir cuentas por todos los fragmentos de código, programas, ideas y contribuciones de su propiedad. Para los programadores, la estabilidad pasaba por redistribuir el sistema operativo más actualizado y por compartir todas las innovaciones con todos los usuarios de modo que estas también pudieran ser portables. Los abogados veían urgente hacer UNIX legalmente estable; los ingenieros veían urgente hacer UNIX técnicamente estable y compatible consigo mismo, esto es, evitar la *bifurcación* del sistema, la sentencia de muerte para la portabilidad. La tensión entre la consecución de estabilidad legal para el objeto y la promoción de su portabilidad y estabilidad técnicas es algo que se ha repetido a lo largo de la existencia de UNIX y sus derivados —y que posee ramificaciones en otras áreas también.

De este modo, la identidad y los límites de UNIX se conformaron intrincadamente a través de su compartición y distribución, las cuales produjeron su propia forma de orden moral y técnico. De inmediato surgieron preguntas inquietantes: las versiones que habían sido reparadas, extendidas y expandidas, ¿seguían siendo UNIX y, por ende, seguían bajo el control de AT&T? ¿O eran las diferencias lo bastante grandes como para que estuviera surgiendo otra cosa (no-UNIX)? Si una cinta llena de ajustes aportados por personas ajena a Laboratorios Bell estaba circulando entre los titulares de una licencia UNIX, y esos ajustes modificaban el sistema, ¿seguía siendo UNIX? ¿Seguía siéndolo en sentido legal, técnico o en ambos? Por más que estas preguntas puedan parecer relativamente escolásticas, la historia del desarrollo de UNIX sugiere algo mucho más interesante: pese a haberse introducido todas las modificaciones legales y técnicas posibles, el *concepto* de UNIX se ha mantenido extraordinariamente estable.

Portando UNIX

La portabilidad técnica solo explica parte del éxito de UNIX. Como recurso pedagógico, UNIX se convirtió rápidamente en una herramienta indispensable para los académicos de todo el mundo: a medida que se instalaba y desarrollaba, se enseñaba y aprendía. El hecho de que UNIX se difundiera primero en los departamentos universitarios de ciencias de la computación, y no en empresas, instituciones gubernamentales u ONGs, conllevó que también se volviera parte de la práctica pedagógica básica de una generación de programadores e investigadores. De este modo, a lo largo de los 70 y los 80, UNIX vino a exemplificar el propio concepto de sistema operativo, especialmente de los sistemas operativos multiusuario de tiempo compartido. Dos historias describen el porte de UNIX de las máquinas a las mentes e ilustran la práctica en su desarrollo y el modo en que se cruzó con los intentos técnicos y legales de estabilizar UNIX como un objeto: la historia de la obra *Commentary on UNIX 6th Edition* de John Lions y la historia de Andrew Tanenbaum y su Minix.

El desarrollo de un UNIX pedagógico dotó de una nueva estabilidad al concepto de UNIX en oposición a su estabilidad como un cuerpo de código fuente o como una entidad legal. El porte de UNIX fue tan exitoso que incluso en los casos en que una versión portada de UNIX *no comparte nada del código fuente original*, se la sigue considerando UNIX. La naturaleza monstruosa y promiscua de UNIX se ve del modo más claro en las historias de Lions y Tanenbaum, especialmente cuando se las compara con la integridad comercial, legal y técnica de algo como Microsoft Windows, que generalmente solo existe en un pequeño número de formas (NT, ME, XP, 95, 98, etc.) y cuyo código fuente está celosamente controlado, cercado por protección legal y distribuido exclusivamente a través de la venta y los paquetes de servicios para clientes o fabricantes de ordenadores personales. Aunque el uso de Windows esté mucho más extendido que el de UNIX, está lejos de convertirse en un objeto pedagógico paradigmático: su integridad es predominantemente legal, no técnica o pedagógica. O dicho en términos pedagógicos, Windows es a los peces lo que UNIX es a las clases de pesca.

La obra *Commentary* de Lions también es conocida como «el documento más fotocopiado de las ciencias de la computación». A principios de los 70 Lions era un investigador y profesor universitario en la Universidad de Nueva Gales del Sur que, tras leer el primer artículo de Ritchie y Thompson sobre UNIX, convenció a sus colegas

para comprar una licencia de AT&T.²⁸ Lions, al igual que muchos investigadores, estaba impresionado por la calidad del sistema e íntimamente familiarizado, como todos los usuarios de UNIX en ese periodo, con su código fuente —una necesidad para instalarlo, ejecutarlo o repararlo. Lions comenzó usando el sistema para dar sus clases sobre sistemas operativos, y en el transcurso de las mismas preparó una especie de libro de texto, que consistía en el código fuente completo de la versión 6 de UNIX (V6), junto con un comentario y una explicación línea a línea muy elaborados. El valor de este libro no puede subestimarse en modo alguno. El acceso a las máquinas y al software susceptibles de emplearse para comprender cómo funcionaba un sistema real era muy limitado: «Los ordenadores de verdad con sistemas operativos reales estaban guardados con llave en salas de máquinas y se dedicaban a procesar venticuatro horas al día. UNIX cambió eso».²⁹ Berny Goodheart, en una celebración del *Commentary* de Lions, reiteró esta idea de la utilidad práctica del código fuente y de las anotaciones al mismo:

Es importante entender la trascendencia del trabajo de John en esa época: para los estudiantes de informática de los 70, cuestiones complejas como proceso de programación, seguridad, sincronización, sistemas de archivo y otros conceptos estaban más allá de la comprensión normal y eran extremadamente difíciles de enseñar —simplemente no había disponible nada con la accesibilidad suficiente para los estudiantes que usar como caso de estudio. En lugar de ello, la disciplina de los estudiantes de informática se conseguía perforando tarjetas, recopilando papeles impresos doblados y demás. Básicamente, en aquella época un sistema operativo de ordenador era considerado como un enorme fragmento de código privativo inaccesible.³⁰

El libro de Lions representaba un documento único en el mundo de la informática, uno que contenía una especie de clave para aprender sobre un componente central del ordenador al que muy poca gente tenía acceso en los 70. Ello muestra cómo UNIX no solo se portó a

28. Ken Thompson y Dennis Ritchie, «The Unix Operating System», *Bell Systems Technical Journal*, 1974.

29. Greg Rose, citado en Lions, *Commentary*, sin página.

30. Lions, *Commentary*, sin página.

máquinas (que eran escasas) sino también a las mentes de jóvenes investigadores y estudiantes de programación (que eran abundantes). Varias generaciones tanto de académicos como de estudiantes de informática que acabarían trabajando en empresas de ordenadores o de software recibieron instrucción a partir de fotocopias del código fuente de UNIX, impregnadas de un aroma a tóner y a circulación ilícita: un sistema operativo distribuido en sentido textual.

Desafortunadamente, la distribución de *Commentary* se vio también legalmente restringida. Con la esperanza de poder mantener el estatuto de secreto comercial de UNIX, AT&T y Western Electric solo permitieron una circulación muy limitada del libro. Al principio, Lions obtuvo permiso para distribuir copias individuales solo a personas que ya poseyeran una licencia para UNIX V6; luego la propia empresa Laboratorios Bell distribuyó *Commentary* durante un breve periodo, pero solo a titulares de licencia y con la condición de no venderlo, distribuirlo o copiarlo. No obstante, casi todo el mundo parecía poseer una copia manoseada de enésima generación, tal y como atestigua Peter Reintjes:

Pronto entramos en posesión de lo que parecía una fotocopia de quinta generación y alguien cuyo nombre no referiré se pasó toda la noche en la fotocopiadora engendrando una sexta, un acto expresamente prohibido por una renuncia de responsabilidad cuidadosamente redactada en la primera página. Cuatro cosas extraordinarias se estaban produciendo a la misma vez: primera, habíamos descubierto el primer fragmento de software que resultaba inspirador y no enojoso; segunda, habíamos adquirido lo que equivalía a una crítica literaria de ese programa informático; tercera, estábamos haciendo el avance más relevante para nuestra educación informática al leer de primera mano un sistema operativo completo; y cuatro, estábamos infringiendo la ley.³¹

Así pues, estas generaciones de estudiantes de informática y académicos compartían un secreto —un secreto comercial convertido en un secreto a voces. Cada estudiante que aprendía las características esenciales del sistema operativo UNIX a partir de una fotocopia del *Commentary* de Lions, también se enteraba por su portada del intento de AT&T de controlar su distribución legal. El desarrollo paralelo de las fotocopiadoras

31. *Ibid.*

posee aquí una interesante resonancia: junto con las cintas de música caseras y la introducción del grabador de vídeo, las fotocopiadoras contribuyeron a impulsar las reformas de la legislación de *copyright* adoptadas en 1976.

Transcurridas más de tres décadas, y mucho después de que el código fuente que contenía hubiera sido reemplazado por completo, el *Commentary* de Lions sigue siendo ampliamente admirado por los *geeks*. Por más que el software libre haya cerrado el círculo al proporcionar a los estudiantes un sistema operativo real que se puede estudiar, enseñar, copiar e implementar legalmente, el tipo de «crítica literaria» que representa el trabajo de Lions sigue siendo extremadamente excepcional. Disponer de comentarios claros que apoyen la lectura de un código, incluso si está obsoleto, es una de las pocas maneras de entender verdaderamente los elementos de diseño y las inteligentes implementaciones que hicieron al sistema operativo UNIX tan diferente de sus predecesores e incluso de muchos de sus sucesores, de los cuales muy pocos, por no decir ninguno, han sido portados con tal éxito a las mentes de tantos estudiantes.

El *Commentary* de Lions contribuyó a la creación de una comunidad mundial de gente conectada entre sí por un cuerpo de código fuente, tanto en su forma implementada como en su forma textual, fotocopiada. Este naciente público recursivo no solo se concebía a sí mismo como perteneciente a una élite técnica constituida por la creación, comprensión y promoción de una herramienta técnica particular, sino que también se reconocía como «infractor de la ley», una comunidad constituida en oposición a las formas de poder que gobernaban la circulación, distribución, modificación y creación de las mismas herramientas que estaban aprendiendo a construir como parte de su vocación. La conexión material a su código fuente que compartían a escala mundial los *geeks* amantes de UNIX no constituye una mera experiencia técnica, sino también una social y legal.

Lions no fue el único investigador en reconocer que enseñar el código fuente era la ruta más rápida hacia su comprensión. La otra historia sobre la circulación del código fuente implica a Andrew Tanenbaum, un ingeniero informático muy respetado y autor de libros de texto estándar sobre arquitectura de computadores, sistemas operativos y redes.³² En la década de los 70 Tanenbaum también había usado

32. Los dos libros de texto más famosos de Tanenbaum son *Operating Systems* [ed. cast.: *Sistemas operativos: Diseño e implementación*] y *Computer Networks* [ed. cast.: *Redes de computadoras*], de los que se han publicado tres y cuatro ediciones respectivamente.

UNIX como herramienta didáctica en sus clases en la Vrije Universiteit de Ámsterdam. Al distribuirse su código fuente junto con el código binario, podía poner a sus estudiantes a explorar directamente las implementaciones del sistema, y a menudo usaba el código fuente y el libro de Lions en sus clases. Sin embargo, según su libro *Operating Systems: Design and Implementation* (1987), «cuando AT&T liberó la versión 7 [en torno a 1979], comenzó a darse cuenta de que UNIX era un producto comercial valioso, así que entregó la Versión 7 junto con una licencia que prohibía el estudio del código fuente en clase, a fin de evitar poner en peligro su situación de secreto comercial. Muchas universidades simplemente abandonaron el estudio de UNIX e impartieron solo teoría» (p. 13). Para Tanenbaum, esta era una alternativa inaceptable —pero también lo era, aparentemente, seguir infringiendo la ley enseñado UNIX en sus clases—, así que procedió a crear un sistema operativo similar a UNIX completamente nuevo que no usara ni una sola línea del código fuente de AT&T. A esta creación la llamó Minix. Se trataba de una versión básica destinada a ejecutarse en ordenadores personales (los PC de IBM) y a distribuirse junto con el libro *Operating Systems*, publicado por Prentice Hall.³³

El uso de Minix como herramienta educativa se generalizó tanto en la década de los 80 como el código fuente de Lions lo había hecho en los 70. Según Tanenbaum, el grupo de Usenet comp.os.minix había alcanzado 40 000 miembros hacia finales de los 80 y él recibía constantes sugerencias de cambios y mejoras del sistema operativo. Su propia dedicación a la enseñanza implicó que solo incorporara unas pocas de ellas, en un esfuerzo por mantener el sistema lo bastante simple como para poder imprimirla en un libro de texto y hacerlo comprensible para los estudiantes universitarios. Minix estaba disponible libremente como código fuente y era un sistema operativo plenamente operativo, incluso una alternativa potencial a UNIX ejecutable en un ordenador personal. He aquí un claro ejemplo de cómo la integridad conceptual de UNIX se transmitía a otra generación de estudiantes de informática: el libro de Tanenbaum no se llama «UNIX Operating Systems» [«Sistemas Operativos Unix»] sino *Operating Systems [Sistemas Operativos]*. La implicación obvia es que UNIX representaba el ejemplo más claro

33. Tanenbaum no fue la única persona en seguir esta ruta. El otro gigante reconocido de los libros de texto de ingeniería informática, Douglas Comer, creó Xinu y Xinu-PC (UNIX escrito al revés) en *Operating Systems Design* en 1984.

de los principios que deberían guiar la creación de cualquier sistema operativo: era, a todos los efectos, tecnología punta incluso veinte años después de su concepción original.

Minix no era un software comercial pero tampoco era software libre, pues Prentice Hall, la editorial de Tanenbaum, ostentaba su *copyright* y su control. Dado que no usaba ningún código fuente de AT&T, Minix también era legalmente independiente, un objeto legal en sí mismo. El hecho de que estuviera destinado a ser legalmente distinto pero conceptualmente fiel a UNIX es una clara indicación de los tipos de tensiones que gobiernan la creación y compartición de código fuente. La irónica apoteosis de Minix como el estándar pedagógico dorado para estudiar UNIX llegó en 1991-1992, cuando un joven Linus Torvalds creó una «bifurcación» de Minix, también reescrita desde cero, que con el tiempo se convertiría en la pieza paradigmática del software libre: Linux. El propósito de Tanenbaum era que Minix se mantuviera como un sistema operativo pedagógicamente útil —reducido, conciso e ilustrativo— mientras que Torvalds quería extender y expandir su propia versión de Minix para aprovechar plenamente las variantes de hardware producidas en los años 90. No obstante, ambos estaban comprometidos con la visibilidad y la compartición de código fuente como la ruta más rápida hacia la completa comprensión de los principios de los sistemas operativos.

Bifurcando UNIX

La necesidad de Tanenbaum de producir Minix estaba movida por el deseo de compartir el código fuente de UNIX con estudiantes, un deseo con el que AT&T estaba manifiestamente incómoda y que amenazaba el estatuto de secreto comercial de su propiedad. El hecho de que pueda llamarse a Minix una bifurcación de UNIX es un aspecto clave de la economía política de los sistemas operativos y los sistemas sociales. La *bifurcación* generalmente se refiere a la creación de un código fuente nuevo, modificado a partir de una base original de código fuente, dando así lugar a dos programas distintos de la misma ascendencia. Mientras que la modificación de un motor solo da lugar a un motor modificado, la modificación del código fuente implica diferenciación y reproducción, dada la facilidad con la que se puede copiar.

¿Cómo podría Minix —una completa reescritura— seguir considerándose el mismo objeto? Considerado únicamente desde la pers-

pectiva de la legislación de secretos comerciales, los dos objetos eran distintos, aunque desde la perspectiva del *copyright* quizás hubiera base para alegar una infracción, por más que AT&T no se apoyara tanto en el *copyright* como en el secreto comercial. Desde una perspectiva técnica, las funciones y procesos que cumple el software son los mismos, pero los medios por los cuales está codificado para cumplirlos son diferentes. Y desde un punto de vista pedagógico, los dos son idénticos —ejemplifican ciertas características fundamentales de un sistema operativo (estructura del sistema de archivos, paginación de memoria, gestión de procesos), siendo todo el resto optimización, o simple parafernalia. Entender la naturaleza de la bifurcación requiere también que UNIX sea entendido desde una perspectiva social, es decir, desde la perspectiva de un sistema operativo creado y modificado por usuarios-desarrolladores de todo el mundo de acuerdo con demandas particulares y parciales. Ello conforma la base para el surgimiento de un público recursivo robusto.

Uno de los ejemplos más importantes de la bifurcación del errante código fuente de UNIX y del desarrollo de la comunidad de co-desarrolladores de UNIX es la historia de la BSD (*Berkeley Software Distribution*, Distribución de Software de Berkeley) y su incorporación a los protocolos TCP/IP. En 1975 Ken Thompson se tomó un año sabático en su ciudad natal de Berkeley (California) y allí ayudó a miembros del departamento de ciencias de la computación en sus instalaciones de UNIX, llegando con la V6 y la cinta *diff* de los «50 ajustes». Ken había comenzado a trabajar en un compilador para el lenguaje de programación Pascal que se ejecutaría en UNIX, y su trabajo fue retomado por dos jóvenes estudiantes de posgrado: Bill Joy y Chuck Hartley. (Joy luego cofundaría Sun Microsystems, una de las compañías de terminales de trabajo basadas en UNIX más exitosas de la historia de la industria).

Joy, por encima de casi todos los demás, participó apasionadamente en la distribución informal del código fuente. Con un sistema Pascal popular y bien construido y un nuevo editor de texto llamado ex (más tarde vi), creó la BSD, un conjunto de herramientas que podía usarse en combinación con el sistema operativo UNIX. Se trataba de extensiones del sistema operativo UNIX original, pero no una versión reescrita completa que pudiera reemplazarlo. A decir de todos, Joy actuó como una especie de empresa unipersonal de distribución de software, haciendo cintas y enviándolas por correo, recibiendo pedidos y cobrando

facturas —todo ello además de crear el software.³⁴ Los usuarios de UNIX de todo el mundo pronto supieron de este valioso conjunto de extensiones al sistema, y en poco tiempo muchos diferenciaban ya entre AT&T UNIX y BSD UNIX.

Según Don Libes, Laboratorios Bell permitió a la Universidad de Berkeley distribuir sus extensiones de UNIX siempre y cuando los receptores también tuvieran una licencia de la empresa para el UNIX original (un acuerdo similar al que rigió para el *Commentary* de Lion)³⁵. Desde 1976 hasta 1981 aproximadamente, BSD se convirtió lentamente en una distribución independiente —de hecho, en una versión completa de UNIX— famosa por el editor vi y el compilador Pascal, pero también por la adición de memoria virtual y su implementación en las máquinas VAX de DEC.³⁶ Debería estar claro que el inusual estatuto cuasicomercial del UNIX de AT&T permitió esta situación de un modo que una empresa informática completamente comercial nunca habría hecho. Consideremos, por ejemplo, el hecho de que muchos usuarios de UNIX —entre ellos los estudiantes universitarios— esencialmente no podían saber si estaban usando un producto de AT&T o algo llamado BSD UNIX creado en Berkeley. El sistema operativo funcionaba exactamente del mismo modo y, excepto por la presencia de las notificaciones de *copyright* que ocasionalmente aparecían en pantalla, no hacía ningún alarde de su identidad de marca (eso vendría más tarde, en los 80). Mientras que un fabricante comercial de ordenadores solo habría permitido algo como BSD si se incorporase en y se distribuyese como un producto único, comercializable e identificable con un nombre ingenioso, en AT&T hicieron la vista gorda ante la proliferación y propagación de AT&T unix, lo cual dio pie a las bifurcaciones del proyecto: distintos cuerpos de código fuente, cada uno de los cuales representaba una variante de algo llamado UNIX.

34. McKusick, «Twenty Years of Berkeley Unix», p. 32.

35. Libes y Ressler, *Life with UNIX*, pp. 16–17.

36. Un posterior proceso judicial que enfrentó a la compañía SCO de Utah —que por entonces ostentaba los derechos legales sobre el código fuente original de UNIX— y a IBM suscitó una vez más la cuestión de cuánto del código original de UNIX hay en la distribución BSD. SCO alegó que IBM (y Linus Torvalds) insertaron código fuente de UNIX de su propiedad en el núcleo de Linux. Sin embargo, la trayectoria increíblemente enrevesada del código fuente «original» dificultó la constatación de tales alegaciones: el código se desarrolló en Laboratorios Bell, fue licenciado a múltiples universidades, usado como base para BSD, vendido a una versión anterior de la compañía SCO (entonces conocida como Santa Cruz Operation), que creó una versión llamada Xenix en colaboración con Microsoft. Véase el diagrama de Eric Lévénez en <http://www.levenez.com/unix/>. Para más detalles sobre este caso, véase www.groklaw.com.

A medida que se desarrollaba BSD, adquiría tipos de funcionalidad diferentes a las del UNIX del que fue engendrado. El desarrollo más relevante fue la inclusión de un código que le permitía conectar ordenadores a Arpanet, usando los protocolos TCP/IP diseñados por Vinton Cerf y Robert Kahn. Estos protocolos representaban una pieza clave de Arpanet, supervisada por la IPTO (*Information Processing and Techniques Office*, Oficina de Técnicas de Procesamiento de Información) de la DARPA (*Defense Advanced Research Projects Agency*, Agencia de Defensa de Proyectos de Investigación Avanzados) desde su concepción en 1967 hasta alrededor de 1977. El objetivo del conjunto TCP/IP era permitir la interconexión de diferentes redes, cada una con sus propias máquinas y límites administrativos.³⁷ Aunque hay un legado común —en la persona de J.C.R. Licklider— que vincula la imaginación del sistema operativo de tiempo compartido con la creación de la «red galáctica», Arpanet se desarrolló en sus inicios de modo completamente independiente a UNIX³⁸. En cuanto sistema operativo de tiempo compartido, UNIX estaba orientado a permitir la compartición de recursos en un único ordenador, ya fuera una unidad central o una minicomputadora, pero no estaba inicialmente destinado a ser conectado a una red de otras máquinas que ejecutaran UNIX, como ocurre hoy.³⁹ El objetivo de Arpanet, por contra, era explícitamente lograr la compartición de recursos localizados en diversas máquinas a través de diversas redes.

Para alcanzar los beneficios del TCP/IP, era necesario que los recursos estuvieran implementados en todos los diferentes sistemas operativos conectados a Arpanet —fueran cuales fueran el sistema operativo y la máquina que estuvieran en uso en cada uno de los nodos. Sin embargo, en torno a 1977, las máquinas usadas originalmente en la red estaban desfasadas y costaba cada vez más mantenerlas, de modo que, según Kirk McKusick, el mayor gasto al que se enfrentaban era el de portar el software del protocolo antiguo a nuevas máquinas. En consecuencia, IPTO decidió seguir en parte la estrategia de lograr una coordinación a nivel del sistema operativo y eligió UNIX como unas de las plataformas

37. Véase Vinton G. Cerf y Robert Kahn, «A Protocol for Packet Network Interconnection». Para un relato histórico, véase Abbate, *Inventing the Internet*; Norberg y O'Neill, *A History of the Information Techniques Processing Office*. Igualmente los capítulos 1 y 5 de este libro ofrecen más detalles sobre el papel de estos protocolos y del proceso RFC.

38. Waldrop, *The Dream Machine*, caps. 5 y 6.

39. Con la excepción de una herramienta no carente de importancia llamada Unix to Unix Copy Protocol, o uucp, que se usó ampliamente para transmitir datos por teléfono y que conformaría la base de la creación Usenet. Véase Hauben y Hauben, *Netizens*.

fundamentales en las que estandarizarse. En pocas palabras, habían visto la luz de la portabilidad. Alrededor de 1978, IPTO otorgó un contrato a Bolt, Beranek y Newman (BBN), una de las empresas contratistas de la Arpanet original, para integrar los protocolos TCP/IP en el sistema operativo UNIX.

Pero entonces sucedió algo extraño, según Salus:

BBN realizó y entregó a Berkeley un prototipo inicial. Bill [Joy] empezó inmediatamente a hackearlo porque solo ejecutaría una Ethernet a unos 56K/seg utilizando el 100% de la CPU en un 750... Bill lobotomizó el código e incrementó su rendimiento hasta un rango de 700KB/seg. Esto causó cierta consternación en BBN cuando aparecieron con su versión «finalizada», y Bill no la aceptó. A esto le sucedieron años de batallas acerca de qué versión se integraría en el sistema. Al final la versión de Berkeley fue la ganadora.⁴⁰

Aunque no está del todo claro, al parecer la intención de BBN era darle a Joy el código para incluirlo en la versión BSD de UNIX destinada a la distribución, y la intención de Joy y sus colaboradores era cooperar con Rob Gurwitz de BBN en una implementación final, pero Berkeley insistió en «mejorar» el código para que se ajustara más a sus necesidades, y aparentemente BBN no estuvo de acuerdo.⁴¹ Como resultado de esta refriega entre BSD y BBN surgió una genuina bifurcación: dos cuerpos de código que hacían lo mismo, compitiendo entre sí para convertirse en el estándar de implementación de TCP/IP en UNIX. He aquí, pues, un caso de compartición de código fuente que llevó a la creación de diferentes versiones de software —compartición sin colaboración. Algunos sitios usaron el código BBN, otros el código de Berkeley.

La bifurcación, sin embargo, no implica una divergencia permanente, y la continuidad de las mejoras, los portes y la compartición del software puede acarrear extrañas consecuencias cuando se da la bifurcación. Por un lado, existen *fragmentos específicos de código fuente*: estos deben ser identificables y exactos, y han de acompañarse de un aviso de *copyright*, como en el caso del código de Berkeley, famoso por la exhaustividad con que fue revisado por los miembros del consejo

40. Salus, *A Quarter Century of UNIX*, p. 161.

41. El *TCP/IP Digest* 1.6 de 11 de noviembre de 1981 contiene la explicación de Joy sobre las intenciones de Berkeley (Identificador del mensaje: anews.aucvax.5236).

rector de la Universidad de California, quienes permitieron una distribución muy liberal del código BSD con la condición de que mantuviera el aviso de *copyright*. Por otro lado, existen *recopilaciones de código* particulares que funcionan conjuntamente (por ejemplo, UNIX™, el UNIX aprobado de DARPA o, más tarde, el *Certified Open Source [sm]*) y que a menudo se identifican por un símbolo de marca registrada con la intención legal de diferenciar los productos, no de invocar la propiedad de un determinado producto.

La extraña consecuencia de todo ello fue esta: el código TCP/IP específico de Bill Joy no solo se incorporó a BSD UNIX sino también a otras versiones de UNIX, incluyendo el UNIX distribuido por AT&T (que originalmente había concedido una licencia de UNIX a Berkeley) sin el aviso de *copyright* de Berkeley. Este extravagante y enmarañado cúmulo de licencias y código derivó en un famoso cruce de demandas entre AT&T y Berkeley en el que las complejidades de esta situación quedaron solventadas.⁴² Un observador inocente que esperase que UNIX fuese una sola cosa podría sorprenderse al hallar que tal sistema adquiere formas diferentes por razones que son prácticamente imposibles de identificar, pero cuya causa última es clara: diferentes versiones de la compartición de código fuente en conflicto; diferentes ideas de orden moral y técnico que ocasionan complejas marañas de valor y código.

La bifurcación BSD de UNIX (y la sub-bifurcación del TCP/IP) fue solo una de las muchas por llegar. A principios de los 80, había emergido una proliferación de bifurcaciones de UNIX, a las que poco después seguiría un robusto movimiento de comercialización. Al mismo tiempo, la circulación del código fuente comenzó a ralentizarse, a medida que las corporaciones empezaron a competir mediante el añadido de características y la creación de hardware específicamente diseñado para ejecutar UNIX (como el terminal Sun Sparc y el sistema operativo Solaris, resultado de la comercialización que Joy hizo del BSD en los 80). La cuestión de cómo conseguir que todas estas versiones pudieran funcionar conjuntamente acabó convirtiéndose en el centro de las discusiones sobre sistemas abiertos que dominarían los sectores de terminales y redes del mercado informático desde principios de los 80 hasta 1993, cuando el doble éxito de Windows NT y la llegada de Internet a la conciencia pública cambiaron la suerte de la industria de UNIX.

42. Véase Andrew Leonard, «BSD Unix: Power to the People, from the Code», *Salon*, 16 de mayo de 2000: http://archive.salon.com/tech/fsp/2000/05/16/chapter_2_part_one/.

Un segundo y más importante efecto de la pugna entre BBN y BSD fue simplemente la adopción generalizada de los protocolos TCP/IP. Aproximadamente un 98% de los departamentos de ciencias de la computación de EEUU y muchos homólogos alrededor del mundo incorporaron los protocolos TCP/IP en sus sistemas UNIX y consiguieron un acceso instantáneo a Arpanet.⁴³ El hecho de que ocurriera cuando lo hizo es importante: pocos años más tarde, durante la era de la comercialización de UNIX, los fabricantes podrían muy bien no haber implementado estos protocolos de forma amplia (o, más probablemente, haberlo hecho de formas incompatibles y no estandarizadas), mientras que antes de 1983 los ingenieros informáticos de las universidades vieron plenos beneficios en hacerlo si con ello podían conectarse fácilmente a la mayor red singular de ordenadores del planeta. La implementación completa, ya operativa y relativamente estándar del TCP/IP en UNIX (y la capacidad de acceder al código fuente) dieron a estos protocolos una tremenda ventaja en términos de su supervivencia y éxito como la base de una red mundial y singular.

Conclusión

El sistema operativo UNIX no es solo un logro técnico, sino que supone la creación de un conjunto de normas para compartir código fuente en un entorno inusual: cuasicomercial, cuasiacadémico, conectado en red y de escala planetaria. La compartición del código fuente de UNIX se ha realizado de tres formas básicas: portando el código fuente (transfiriéndolo de una máquina a otra); enseñando el código fuente, es decir, «portándolo» a los estudiantes en un marco pedagógico donde el uso de un sistema operativo realmente funcional facilita la enseñanza de teoría y conceptos; y bifurcando el código fuente (modificando el código fuente existente para hacer algo nuevo o distinto). Este juego de proliferación y diferenciación es esencial para la identidad extraordinariamente estable de UNIX, pero tal identidad existe de múltiples formas: técnica (como un sistema operativo funcional y autocompatible), legal (como una versión circunscrita a una licencia y sujeta a la legislación comercial y de propiedad intelectual) y pedagógica (como un modelo conceptual, el paradigma de un sistema operativo). El código fuente así

43. Norberg y O'Neill, *A History of the Information Techniques Processing Office*, pp. 184–185. Estos autores citan la cifra de Comer, *Internetworking with TCP/IP*, p. 6.

compartido es esencialmente distinto de cualquier otro tipo de código fuente del mundo informático, ya sea académico o comercial, y suscita interrogantes problemáticos sobre estandarización, sobre control y auditoría y sobre legitimidad que no solo acucian al sistema UNIX sino también a Internet y a sus diversos protocolos «abiertos».

La compartición actual del código fuente en el software libre es como es debido a UNIX. Sin embargo, UNIX no es como es debido al genio inventivo de Thompson y Ritchie, o al *marketing* y la brillantez directiva de AT&T, sino debido a que *la compartición produce su propio tipo de orden*: sistemas operativos y sistemas sociales. El hecho de que los *geeks* estén habituados a hablar de «la filosofía UNIX» significa que UNIX no es solo un sistema operativo sino un modo de organizar las complejas relaciones entre vida y trabajo a través de medios técnicos; un modo de trazar y traspasar los límites entre lo académico, lo estético y lo comercial; y un modo de implementar ideas de orden moral y técnico. Es más, a medida que el código fuente llegue a incluir cada vez más actividades de comunicación y creación cotidianas —a medida que llegue a reemplazar la escritura y el pensamiento suplementario—, la genealogía de su portabilidad y la historia de su bifurcación iluminarán los tipos de orden surgidos de prácticas y tecnologías muy alejadas de los sistemas operativos —pero íntimamente ligadas a la filosofía UNIX.

V. CONCEPCIÓN DE SISTEMAS ABIERTOS

Lo bueno de los estándares es que hay muchísimos entre los que escoger.¹

La apertura es un concepto indómito. Mientras que el adjetivo *libre* tiende a la ambigüedad (¿libre como en libertad de expresión o como en barra libre?), *abierto* tiende a la ofuscación. Todo el mundo asegura que es abierto, todo el mundo tiene algo que compartir, todo el mundo coincide en que ser abiertos es lo que hay que hacer —al fin y al cabo, la apertura es la otra mitad del «código abierto»—, pero pese a toda su obviedad ser «abierto» es quizás el componente más complejo del software libre. Nunca queda muy claro si se trata de un fin o de un medio. Peor aún: en este caso lo opuesto de «abierto» (específicamente de «sistemas abiertos») no es cerrado, sino «privativo» —señal de la complicada imbricación de lo técnico, lo legal y lo comercial.

En este capítulo narro la historia de la disputa sobre el significado de «sistemas abiertos» desde 1980 a 1993, una disputa por crear una infraestructura simultáneamente moral y técnica en el seno de la industria informática.² La infraestructura en cuestión comprende componentes técnicos —el sistema operativo UNIX y los protocolos TCP/IP de Internet como sistemas abiertos— pero también «morales», incluyendo la demanda de estructuras de competencia justa y abierta, mercados antimonopolísticos y abiertos y procesos basados en estándares abiertos para desarrollar ordenadores y software de alta tecnología y

1. Citado en Libes y Ressler, *Life with UNIX*, 67, y también en Critchley y Batty, *Open Systems*, p. 17. La primera vez que oyó esto fue en una entrevista con Sean Doyle en 1998.

2. *Moral* en este uso indica el «orden moral y social» que exploré mediante el concepto de imaginarios sociales en el Capítulo 1. O, en el sentido de la Ilustración Escocesa de Adam Smith, apunta a la organización y relaciones de intercambio correctas entre humanos.

conectados en red en los años 80.³ Por *moral* entiendo imaginaciones del orden adecuado de la acción colectiva política y comercial. Haciendo referencia a algo más amplio que simplemente cómo deberían actuar los individuos, *moral* supone una visión de cómo la economía y la sociedad deberían ordenarse colectivamente.

La historia de los sistemas abiertos es asimismo la historia del punto ciego de dichos sistemas —punto ciego donde se halla la propiedad intelectual. La historia revela una tensión entre órdenes morales y técnicos incompatibles: por un lado, la promesa de múltiples fabricantes y corporaciones creando componentes interoperables y comercializándolos en un mercado abierto y heterogéneo; por el otro, un sistema de propiedad intelectual que fomentaba su celosa custodia y secretismo y que otorgaba estatuto monopolístico al código fuente, los diseños y las ideas con el fin de diferenciar los productos y promover la competencia. La tensión demostró ser irresoluble: sin código fuente compartido, por ejemplo, no es posible la interoperabilidad entre sistemas operativos; sin interoperabilidad, tampoco son posibles la interconexión y las aplicaciones portables; sin aplicaciones portables que puedan funcionar en cualquier sistema, los mercados abiertos son imposibles; y sin mercados abiertos, reina el poder del monopolio.

La estandarización estaba en el corazón de la disputa, pero nunca se resolvió quién la establecería y por qué medios. El sueño de los sistemas abiertos, perseguido en el marco de una industria totalmente carente de regulación, dio por resultado un complicado experimento de formas novedosas de estandarización y cooperación. La creación de un sistema operativo «estándar» basado en UNIX es la historia de un fracaso, una especie de «figuración» desbocada que derivó en un enorme consorcio de fabricantes de ordenadores intentando colaborar y competir entre sí a la misma vez. Mientras tanto, la creación exitosa de un protocolo de red «estándar» —conocido como el Modelo de Referencia OSI (*Open Systems Interconnection*, Interconexión de Sistemas Abiertos)— es una historia de fracaso que esconde un gran éxito: OSI quedó eclipsado en

3. Por supuesto, existe un discurso relativamente robusto sobre sistemas abiertos en biología, sociología, teoría de sistemas y cibernetica; no obstante, el significado de *sistemas abiertos* es más o menos completamente distinto de lo que *apertura* y *sistemas abiertos* vinieron a significar en la industria informática en el periodo comprendido entre la llegada del ordenador personal y la explosión de Internet (aproximadamente 1980–1993). Un solapamiento relevante entre estos dos significados puede encontrarse en el trabajo de Carl Hewitt en el Media Lab del MIT y en el interés por la «agórica» de K. Eric Drexler, Bernardo Huberman y Mark S. Miller. Véase Huberman, *The Ecology of Computation*.

el mismo periodo por la adopción rápida y *ad hoc* del TCP/IP, que usaba un proceso de estandarización radicalmente distinto y tuvo éxito por una serie de sorprendentes razones, permitiendo que Internet llegara a la forma que tomó en los 90 y ejemplificando en última instancia el imaginario moral-técnico de un público recursivo —y uno situado en el corazón de las prácticas del software libre.

La concepción de la apertura, que es la trama central de estas dos historias, se ha vuelto un componente esencial de la práctica y la potencia contemporáneas del software libre. Estas tempranas batallas crearon una especie de predisposición generalizada hacia el software libre en los 90, un reconocimiento del software libre como remoción del punto ciego de los sistemas abiertos y a la vez como explotación de su potencial. El ideal *geek* de apertura y orden moral-técnico (el que hizo de Napster un acontecimiento tan trascendente) se forjó en la era de los sistemas abiertos; sin esta concepción histórica concreta sobre cómo mantener la apertura en términos morales y técnicos, el público recursivo de los *geeks* no sería más que otra organización jerárquica cerrada —una corporación fallida— y no un público independiente que sirve de mecanismo de control sobre las formas de poder destructivo que dominaban la disputa sobre los sistemas abiertos.

Irremediablemente plurales

Big iron,⁴ silos, sistemas heredados, sistemas preconfigurados, dinosaurios, unidades centrales. Mirada en retrospectiva, la industria informática de los años 60 a los 80 se muestra retrógrada y cerrada, literalmente arrinconada, tal y como sugiere un antiguo anuncio de Intel (figura 3). Los observadores contemporáneos que muestran disgusto e impaciencia ante la forma que los ordenadores tomaron en esa época son sin excepción partidarios de los sistemas abiertos y contrarios a los sistemas privativos que «confinan» a los clientes a proveedores específicos y crean demandas artificiales de asistencia técnica, integración y

4. Una vez más dejamos en inglés este término debido a su especificidad, explicada así por Eric Raymond en la citada versión 4.4.7 del *Jargon File*.

Big iron: Ordenadores enormes, caros y ultrarrápidos. Usado generalmente para supercomputadoras dedicadas al cálculo numérico, pero también puede incluir a las grandes unidades centrales de IBM más convencionales. Término de aprobación. Compárese con «*heavy metal*», opóngase a «dinosaurio».

Disponible en: <http://www.catb.org/~esr/jargon/html/B/big-iron.html> [N. del E.]

gestión de recursos. Los sistemas abiertos (si se les permitiera florecer) resolverían todos estos problemas.

Figura 3

THE DIFFERENCE BETWEEN AN OPEN SYSTEM AND EVERYTHING ELSE.

One day you have a comet on the market. Next day, you're stuck in that comet. That's the technology trap so many computer systems back you into. Not ours. Our systems are open systems. They're based on industry standards. Many of which we created. And those standards are just one of the things that keep you from getting trapped, slowed down or otherwise beat out by your competition. Take for example, our new multi-user supermicro systems for OEMs. They're based on the world's most powerful (and popular) multitasking microprocessor, our iAPX 286. So you can look forward to exceptional price/performance in either real time or commercial applications. And, because the supermicro is an open system, you get configurability to spare. Which means you can take advantage of the products and applications already available from hundreds of MUL-TIBUS and independent software vendors (including ourselves). To add speech, graphics, Ethernet® networking, or whatever you need to make the sale. Easily. But that's just today. With our open systems, you're also never shut out of a market, even if you're a bit late getting in. With Intel, you'll always have the latest VLSI technology at hand. To make certain you always get a foot or two in the door. What's more, we can offer our Open Systems Maintenance program. Which covers your Intel system—from 82 worldwide service locations—even if the problem wasn't caused by one of our parts. Just what you'd expect from a billion-dollar-plus company. To find out more about the difference our systems can make for your business, ask for our open systems brochure. Just call 800-538-1876; in California, 800-672-1833. Or write Intel, Lit. Dept. Z-18, 3065 Bowers Ave., Santa Clara, CA 95051. You'll find that our open systems give you a powerful advantage. Especially when it comes time to close the sale.

intel

Los sistemas abiertos son la solución para que no te quedes arrinconado.

Anuncio de Intel, *Wall Street Journal*, 30 de mayo de 1984.

Dada la promesa de «un ordenador de propósito general», debería parecer en el mejor de los casos irónico que fuera necesaria la creación de sistemas abiertos. Pero dicho ordenador nunca se hizo realidad. No vivimos en el mundo de El Ordenador, sino en un mundo de ordenadores: una miríada de máquinas incompatibles y específicas. El diseño de máquinas (o «arquitecturas») especializadas fue, y sigue siendo, clave para una industria informática competitiva, la cual requería procesadores centrales, componentes y software asociado que pudieran ser claramente calificados y comercializados como productos distintivos: el DEC PDP-11 o el IBM 360 o el CDC 6600. Siguiendo el modelo fordista de producción automovilística, la misión de la industria informática era proporcionar funciones deseadas (cálculos científicos, contabilidad, gestión de reservas) insertas en una gran caja con un botón (o una gran cantidad de botones en cajas cada vez más pequeñas). A pesar de la posibilidad teórica, tales ordenadores no fueron diseñados para ejecutar cualquier función sino más bien para realizar tipos específicos de cálculos extraordinariamente bien. Eran objetos adaptados a mercados particulares.

Por consiguiente, la estrategia de *marketing* fue extremadamente estable desde 1955 a 1980: identificar clientes con necesidades de cómputo, fabricar un ordenador que les sirviera, proveerlos de todo el equipo, software, asistencia técnica o periféricos que necesitaran para su tarea —y cobrarles una alta cantidad por todo ello. Hablando en términos organizativos, era una industria dominada por «IBM y los siete enanitos»: Hewlett-Packard, Honeywell, Control Data, General Electric, NCR, RCA, Univac y Burroughs, con empresas emergentes como DEC esperando entre bambalinas.

Hacia los años 80, sin embargo, ya se había dado una cierta inversión: los ordenadores proliferaban y se habían vuelto más pequeños y rápidos, con lo que los fabricantes de grandes ordenadores tenían cada vez más claro que lo más valioso para los usuarios era la información que generaban, y no las máquinas a través de las que lo hacían. Esta toma de conciencia, cuenta la historia, conduce a la demanda de intercambiabilidad, interoperabilidad, compartición de información y creación de redes; y asimismo presenta la pesadilla de los problemas de conversión entre una variedad desconcertante, heterogénea y rápidamente creciente de hardware, software, protocolos y sistemas. Como lo expresa una contribución a un congreso sobre evaluación de sistemas abiertos:

En cierto momento una gran empresa mirará a su alrededor y verá una enorme cantidad de equipo y software que no operan conjuntamente. Y lo que es más importante, la información almacenada en estas diversas plataformas no se comparte, lo que conduce a una duplicación innecesaria y una pérdida de beneficios.⁵

Los sistemas abiertos surgieron en los años 80 como el nombre de la solución a este problema: una aproximación al diseño de sistemas que, si fuese adoptada por todos los participantes, conduciría a máquinas integradas ampliamente interoperables que podrían enviar, almacenar, procesar y recibir la información de los usuarios. En términos de *marketing* y relaciones públicas, tales sistemas proporcionarían una «integración perfecta».

En teoría, los sistemas abiertos se reducían a una cuestión de adopción de estándares. Por ejemplo, si pudiera convencerse a todos los fabricantes de sistemas UNIX de adoptar el mismo estándar básico para el sistema operativo, de ahí se seguiría naturalmente una integración fluida a medida que todas las diversas aplicaciones pudieran escribirse una vez para ejecutarse en cualquier variante del sistema UNIX, sin importar qué compañía hubiera detrás. En realidad, tal estándar no era para nada obvio y resultaba difícil de crear y más aún de imponer. Como tal, el significado de *sistemas abiertos* era «irremediablemente plural», y el término pasó a significar una gama increíblemente diversa de cosas.

La «apertura» es precisamente el tipo de concepto que oscila entre el fin y los medios. ¿Es buena en sí misma, o es un medio para lograr algo más —y en tal caso qué? ¿Quién quiere conseguirla, y con qué propósito? ¿Es la apertura un objetivo? ¿O es un medio para alcanzar un objetivo diferente —digamos, la «interoperabilidad» o la «integración»? ¿De quién son estos objetivos y quién los establece? ¿Son los objetivos de las corporaciones diferentes o contradictorios respecto de los de los investigadores universitarios o funcionarios gubernamentales? ¿Existen grandes visiones centrales a las que en última instancia se subordinan las actividades de todos ellos?

Entre 1980 y 1993, ninguna persona, compañía o consorcio de la industria informática establecía explícitamente la apertura como el objetivo al que deberían aspirar organizaciones, corporaciones o pro-

5. Keves, «Open Systems Formal Evaluation Process», p. 87.

gramadores, pero por la misma regla de tres casi nadie discrepaba de la demanda de apertura. Como tal, dicha demanda aparece claramente como una especie de imperativo cultural que refleja un imaginario social de larga data arraigado en nociones democráticas liberales, versiones de un mercado libre e ideales sobre el libre intercambio de conocimiento, pero que afronta un cambio de condiciones técnicas que pone de relieve, y también en cuestión, las ideas morales de orden.

En los 80 todo el mundo parecía querer algún tipo de apertura, fueran fabricantes o clientes, y ello desde General Motors hasta las fuerzas armadas⁶. Los debates, tanto retóricos como técnicos, sobre el significado de los sistemas abiertos han producido un aluvión de escritos, mayoritariamente dirigidos a directores de tecnologías de información y a directores generales de información. Por ejemplo, Terry A. Critchley y K.C. Batty, autores de *Open Systems: The Reality* (1993), afirman haber recopilado más de cien definiciones de *sistemas abiertos*, cada una de las cuales enfatiza diferentes aspectos —desde la interoperabilidad de equipos heterogéneos hasta la compatibilidad de aplicaciones diferentes, la portabilidad de sistemas operativos, los estándares legítimos con definiciones de interfaz abierta, etc.—, incluyendo aquellas que privilegian las ideologías de libre mercado, como la definición de Bill Gates: «No hay nada más abierto que el mercado de PCs [...] Los usuarios pueden escoger el mejor y más reciente software». El abanico de significados era enorme y orientado en torno a múltiples ejes: qué, a quién, cómo y así sucesivamente. Los *sistemas abiertos* podían implicar que el código fuente era accesible o que solo lo eran las especificaciones o interfaces; podían significar «disponible para determinados terceros» o «disponible para todos, incluidos los competidores»; podían suponer autoedición, interfaces bien definidas e interfaces de programación de aplicaciones (APIs), o bien adhesión a los estándares establecidos por gobiernos y sociedades profesionales. Para los cínicos, simplemente quería decir que al departamento de *marketing* les gustaba la palabra *abierto* y la usaba mucho.

6. General Motors incitó un profundo interés en los sistemas abiertos con la creación, en 1985, de su MAP (*Manufacturing Automation Protocol*, Protocolo de Automatización de Fabricación), que se construyó sobre UNIX. En ese momento, General Motors era el segundo mayor comprador de equipos informáticos por detrás del Gobierno de EEUU. El Departamento de Defensa y la Fuerza Aérea de EEUU también adoptaron y requirieron tempranamente sistemas UNIX compatibles con POSIX.

Una parte de la definición, pese a todo, era tan coherente como extremadamente importante: lo contrario de «sistemas abiertos» no era «sistemas cerrados» sino «sistemas privativos». En industrias distintas de las de redes e informática la palabra *privativo* tendrá con toda probabilidad una connotación positiva, como en «nuestra exclusiva tecnología privativa». Pero en nuestro contexto el empleo de ese vocablo se volvió un anatema en los años 80 y 90; lo que supuestamente querían los clientes era un sistema que funcionara bien con otros sistemas, y tal sistema tenía por definición que ser abierto, pues ninguna compañía podría atender por sí misma todas las posibles necesidades de una empresa o institución gubernamental modernas. E incluso si así fuera, no debería permitírselo. Por ejemplo, «en el principio fue el verbo y el verbo era ‘privativo’». IBM marcó la pauta al suministrar máquinas que existían en espléndido aislamiento:

No podían manejarse con programas escritos para cualquier otro ordenador y tampoco podían comunicarse con las máquinas de la competencia. Si tu empresa comenzaba comprando ordenadores de diversos tamaños a IBM por ser la corporación mayor y mejor, pronto te hallarías tan confinado al Gigante Azul como los desdichados encadenados en una mazmorra medieval. Cuando un rival de IBM desvelaba un producto tecnológicamente avanzado, lo único que quedaba era suspirar, pues podían pasar años antes de que esa nueva tecnología se integrase en la línea de IBM.⁷

Con la excepción de IBM (y, hasta cierto punto, de sus competidores más cercanos: Hewlett-Packard, Burroughs y Unisys), las corporaciones informáticas de los años 80 buscaron distanciarse de tales soluciones privativas «medievales» (esta expresión también evoca las de los pasados usables de la Reforma Protestante comúnmente empleadas por los geeks). Empresas emergentes como Sun y Apollo reprobaron deliberadamente el modelo de IBM. Así, Bill Joy supuestamente afirmó de uno de los nuevos lanzamientos de IBM en los 80 que era «un dinosaurio pastando ‘con un camión bombeándole los fluidos corporales desde fuera’»⁸.

7. Paul Fusco, «The Gospel According to Joy», *New York Times*, 27 de marzo de 1988, *Sunday Magazine*, p. 28.

8. Véase la entrada «Dinosaur», *The Jargon File*, <http://catb.org/jargon/html/D/dinosaur.html>.

Con todo, los sistemas abiertos nunca fueron una solución simple: toda esa complejidad en hardware, software, componentes y periféricos solo podía resolverse presionando por el establecimiento de estándares —incluso de uno solo. O por expresarlo de otro modo, durante los 80 todo el mundo coincidía en que los sistemas abiertos eran una gran idea, pero nadie coincidía en *cuáles*. Como señala un conferenciante anónimo en *Open Systems: The Reality*:

Me costó mucho tiempo entender a que se refería (la industria) con la oposición abierto vs. privativo, pero finalmente lo averigüé: desde la perspectiva de un proveedor cualquiera, abierto quería decir «nuestros productos» y privativo «los productos de los demás».º

Para la mayoría de partidarios de los sistemas abiertos, la oposición entre *abierto* y *privativo* tenía cierta fuerza moral: indicaba que las corporaciones que apostaban por esto último estaban peligrosamente cerca de ser malvadas, inmorales, tal vez incluso monopolistas criminales. Así, por ejemplo, Adrian Gropper y Sean Doyle, directores de Amicas, se referían repetidamente a los grandes sistemas de información sanitaria a los que se enfrentaban en estos términos: los sistemas abiertos son el camino de la luz, no de la oscuridad. Aunque sin duda hay argumentos a favor de los sistemas cerrados —seguridad, privacidad, robustez, control—, la demanda de interoperabilidad no significa que dicha clausura se vaya a sacrificar.¹⁰ La clausura también era una opción. Es decir, la apertura de los sistemas era una cuestión de soberanía que implicaba que los clientes tienen el derecho, en sentido moral, de controlar un orden técnico cercado por estándares corporativos, el derecho de combinar una serie de piezas distintas de hardware y software adquiridas en un mercado abierto y de controlar la configuración por sí mismos —no una apertura forzada, sino el derecho a decidir por uno mismo si y cómo adoptar lo abierto o lo cerrado.

La idea de orden moral de los sistemas abiertos entra en conflicto, sin embargo, con la idea de orden moral de la propiedad intelectual:

9. Crichtley y Batty, *Open Systems*, p. 10.

10. Un excelente contrapunto aquí es el libro de Paul Edwards *The Closed World*, que demuestra claramente el atractivo de un sistema rigurosa y jerárquicamente controlado como el SAGE (*Semi-Automated Ground Environment*, Entorno de Tierra Semiautomatizado) del Departamento de Defensa contra el surgimiento de modelos de apertura más «verdes» («green worlds»).

el derecho, codificado legalmente, de reivindicar la propiedad y el control de determinados bits de código fuente, software y hardware. El llamamiento a y el mercado de sistemas abiertos nunca se concibieron como opuestos a la propiedad intelectual en sí, por más que la oposición entre abierto y privativo pareciera indicar una forma de reconocimiento subterráneo del papel de la propiedad intelectual. La cuestión nunca se abordó explícitamente. De los cientos de definiciones incluidas en *Open Systems*, solo una se aproxima a la inclusión de cuestiones legales:

Conferenciante en Interop '90 (parafraseado y tal vez apócrifo): «Si pides obtener acceso a una tecnología y la respuesta que recibes es una lista de precios, entonces esa tecnología es 'abierta'. Si lo que recibes es una carta de un abogado, entonces no es 'abierta'.¹¹

Aquí la apertura no se equipara con la libertad para copiar y modificar, sino con la libertad para *comprar* el acceso a cualquier aspecto de un sistema sin tener que firmar un contrato, un acuerdo de confidencialidad o cualquier otro documento legal que no sea un cheque. Las reglas de juego de la competencia no se ponen en discusión: el sistema existente de propiedad intelectual —un sistema que se expandió y reforzó en este periodo— era una condición *sine qua non* de la competencia.

La apertura así entendida comporta un mercado abierto donde es posible adquirir objetos estandarizados que no son ni oscuros ni secretos, pero que pueden ser examinados y juzgados —un mercado de «productos primarios», donde los productos tienen funciones, donde la calidad es comparable y conforma la base de una competencia vigorosa. Lo que esta noción implica es *libertad frente al control monopolístico* de las corporaciones sobre los productos, una libertad que es casi imposible de mantener cuando toda la industria se estructura en torno al control monopolístico de la propiedad intelectual a través de los secretos comerciales, las patentes y el *copyright*. El punto ciego esconde la contradicción entre una industria imaginada según el modelo de la fabricación de productos distintos y tangibles, y la realidad de una industria que oscila entre los productos y los servicios, al comerciar con una propiedad intelectual intangible cuyas fronteras e identidad se definen de hecho por su modo de intercambio, circulación

11. Crichtley y Batty, *Open Systems*, p. 13.

y compartición, como en el caso de la proliferación y diferenciación del sistema operativo UNIX.

En la industria informática de los 80 no había desacuerdo alguno sobre la necesidad de la propiedad intelectual, y tampoco se percibía contradicción alguna en las exigencias de apertura. De hecho, la apertura solo podía tener sentido si se construía sobre un sistema estable de propiedad intelectual que permitiera a los competidores mantener definiciones claras de las fronteras de sus productos. Pero la creación de componentes interoperables parecía demandar una relajación del secretismo y recelo necesarios para «proteger» la propiedad intelectual. Es más, para algunos observadores el problema de la apertura creó la oportunidad para las peores formas de lógica cínica, como en este ejemplo de *Who's Afraid of Big Blue?*, de Regis McKenna.

Los usuarios quieren entornos abiertos, así que más vale que los proveedores se atengan a ello. De hecho, es una buena idea apoyar los nuevos estándares desde primera hora. De esta manera, puedes ayudar a controlar su desarrollo. Es más, puedes llevarte el mérito de guiar el estándar. Apoyar los estándares es una manera de demostrar que estás de parte de los usuarios. Por otro lado, las compañías no pueden competir basándose únicamente en los estándares. Las compañías que viven de estándares pueden morir por estándares. Otras que se adhieran a esos mismos estándares podrían ganar basándose en una tecnología de fabricación superior. Si las empresas no hacen otra cosa que adherirse a los estándares, entonces todos los ordenadores se convertirán en productos primarios, y nadie será capaz de ganar dinero. Por tanto, las compañías deben mantener algo privativo, algo que diferencie sus productos.¹²

Según tal razonamiento, los sistemas abiertos equivaldrían a una regresión económica, un estado de competencia pura sobre la base de la superioridad fabril, y no sobre la base de la ventaja competitiva otorgada por el monopolio de la propiedad intelectual, el claro sello distintivo de la industria de alta tecnología.¹³ Se trataba de una ten-

12. McKenna, *Who's Afraid of Big Blue?*, p. 178, el énfasis es mío. McKenna prosigue sugiriendo que las compañías informáticas pueden diferenciarse añadiendo servicios, mejores interfaces o mayor fiabilidad —lo cual es irónicamente similar a los argumentos que la Open Source Initiative defendería diez años después.

13. Richard Stallman caracterizaba así el punto ciego, evocando la imagen medieval del desdichado encadenado: «Unix no da a los usuarios ninguna mayor libertad legal que Windows.

sión irresoluble entre el deseo de una infraestructura cooperativa de base mercantil y la estructura de un sistema de propiedad intelectual inadaptado a las realidades técnicas en las que operaban las compañías y los clientes —una tensión que revela la reorientación del saber y el poder con respecto a la creación, diseminación y modificación del saber.

Desde la perspectiva de la propiedad intelectual, las ideas, los diseños y el código fuente lo son todo —si una compañía tuviera que liberar el código fuente y permitir a otros proveedores basarse en él, ¿qué le quedaría entonces para vender exactamente? Los sistemas abiertos no suponían nada parecido a una informática libre, de código abierto o de dominio público. Pero el hecho de que la competencia requería cierta forma de colaboración resultaba igualmente obvio: se necesitaban software y sistemas de red estandarizados, mercados estandarizados y normas estandarizadas de innovación dentro de los límites de los estándares. En suma, el reto no era solo la creación de productos competitivos sino la creación de una *infraestructura* estándar que abordase las cuestiones técnicas de la disponibilidad, la adaptabilidad y la reusabilidad de componentes, así como las cuestiones morales de la organización adecuada de la competencia y la colaboración a través de diversos dominios: la ingeniería, la academia, la industria informática y las industrias que esta a su vez informatizaba. Lo que sigue es la historia de cómo UNIX entró en la refriega de los sistemas abiertos, una historia que revela la tensión entre la concepción de la apertura y las demandas de la propiedad intelectual.

Sistemas Abiertos 1: Sistemas Operativos

En 1980 UNIX era sin discusión la opción más obvia como sistema operativo estándar por una razón que de entrada parecía sencilla: podía ejecutarse en más de un tipo de hardware. Había sido instalado en máquinas DEC e IBM y con procesadores Intel y Motorola —un hecho

A lo que se refieren con ‘sistemas abiertos’ es a que puedes mezclar y adaptar componentes, de modo que puedas decidir llevar, digamos, una cadena de Sun en la pierna derecha y una cadena de otra compañía en la pierna izquierda, y acaso una cadena de una tercera compañía en el brazo derecho, y se supone que esto es mejor que tener que elegir entre cadenas de Sun para todos tus miembros o cadenas de Microsoft para todos tus miembros. No sé, a mí me da igual de quién sean las cadenas de cada miembro. Lo que quiero es no estar encadenado por nadie» («Richard Stallman: High School Misfit, Symbol of Free Software, MacArthur-certified Genius», entrevista con Michael Gross, Cambridge (Mass.), 1999, p. 5, <http://www.mgross.com/MoreThgsChng/interviews/stallman1.html>).

estimulante para muchos programadores profesionales, ingenieros informáticos de la universidad y administradores de sistemas, muchos de los cuales también consideraban que UNIX era el mejor diseñado de los sistemas operativos disponibles.

Sin embargo, había un problema (siempre lo hay): UNIX pertenecía a AT&T, y esta lo había licenciado a múltiples fabricantes a lo largo de los años, permitiendo a la vez que el código fuente circulase sin grandes reparos por todo el mundo y que fuese portado a una amplia variedad de arquitecturas informáticas diferentes. Tal proliferación, aunque fortuita, era un sueño hecho realidad: un único sistema interoperable que funcionaba en todo tipo de hardware. Desgraciadamente, la proliferación también disiparía ese sueño, al conllevar que a medida que se animaban los mercados de terminales de trabajo y sistemas operativos, las versiones existentes de UNIX se afianzaron como versiones distintas e incompatibles con funcionalidades e interfaces diferentes. A mediados de los 80 había múltiples iniciativas que competían por estandarizar UNIX, empresa que finalmente se fue a pique y llevó a las llamadas guerras de UNIX, en las que hubo «bandas» de proveedores (algunos presentes en ambos lados de la contienda) que se agruparon para promover estándares rivales. El relato de cómo ocurrió esto es instructivo, pues es una historia que se ha reiterado varias veces en la industria informática.¹⁴

Como sistema híbrido comercial-académico, UNIX nunca entró al mercado como algo único, sino que se licenció de modos variados en entornos diferentes, tanto académicos como comerciales, conteniendo elementos, herramientas y otras características que no necesariamente se habían originado en (o habían retorna a) Laboratorios Bell. A principios de los 80, la versión BSD de hecho competía con la de AT&T, incluso aunque la BSD derivara de una sublicencia —y no era la única. A finales de los 70 y comienzos de los 80 varias compañías habían obtenido licencias de UNIX de AT&T para emplearla en máquinas nuevas. Microsoft adquirió una (y la llamó Xenix, en vez de licenciar también el nombre UNIX) para instalarla en máquinas basadas en Intel. IBM, Unisys, Amdahl, Sun, DEC y Hewlett-Packard siguieron

14. Se puede relatar una historia similar sobre el surgimiento, a finales de los 60 y principios de los 70, de fabricantes de dispositivos «de conector compatible» [*«plug-compatible»*], periféricos que se conectaban a máquinas IBM (véase Takahashi, «The Rise and Fall of the Plug Compatible Manufacturers»). De modo similar, la historia de la compatibilidad de los navegadores y los estándares del W3C en los 90 supone otra recapitulación.

el ejemplo y crearon sus propias versiones y nombres: HP-UX, A/UX, AIX, Ultrix y similares. Dadas las reglas básicas del secreto comercial y de la propiedad intelectual, era necesario que todas y cada una de estas versiones licenciadas fueran legalmente distintas —si es que habían de competir entre ellas. Por más que «UNIX» se mantuviese conceptualmente puro en sentido académico o pedagógico, cada fabricante tendría no obstante que retocarlo, extenderlo u optimizarlo con el propósito de diferenciarse del resto. Al fin y al cabo, «si las empresas no hacen otra cosa que adherirse a los estándares, entonces todos los ordenadores se convertirán en productos primarios, y nadie será capaz de ganar dinero». ¹⁵

Por lo tanto, era improbable que cualquiera de estas compañías retornase los cambios que introducían a UNIX a un fondo común, y ciertamente no a AT&T, que con posterioridad a la desinversión forzada de 1984 acabó lanzando su propia versión comercial de UNIX, llamada UNIX System V. Con gran rapidez, el prometedor UNIX «abierto» de los 70 se volvió un cenagal de sistemas operativos alternativos, cada uno incompatible con el siguiente gracias a la adición de funcionalidades comercialmente distintivas y retoques específicos de hardware. Según Pamela Gray, «hacia mediados de los 80 había más de cien versiones activas» en torno a los tres líderes del mercado: el System V de AT&T, el Xenix de Microsoft/SCO y el BSD.¹⁶ En torno a 1984, las diferencias entre sistemas se habían hecho significativas —como en el caso de la adición de los protocolos TCP/IP, el editor de texto vi y el compilador de Pascal a BSD— y crearon no ya una diferenciación en términos de calidad sino también incompatibilidad en los niveles del software y de las redes.

Los diferentes sistemas tenían por supuesto comunidades de usuarios diferentes, dependiendo de quién fuera cliente de quién. Eric Raymond sugiere que a mediados de los 80 los *hackers*, programadores e ingenieros informáticos independientes seguían mayoritariamente las vicisitudes de BSD:

La línea divisoria se trazaba a grandes rasgos entre los de pelo largo y los de pelo corto; los programadores y técnicos tendían a alinearse con Berkeley y BSD, y los tipos más orientados a los negocios, con

15. McKenna, *Who's Afraid of Big Blue?*, p. 178.

16. Pamela Gray, *Open Systems*.

AT&T y el System V. A los de pelo largo, repitiendo un tema de los primeros días de UNIX de hacía diez años, les gustaba considerarse rebeldes contra un imperio corporativo: una pequeña empresa publicó un póster en el que aparecía un caza espacial parecido al Ala-X con el rótulo «BSD» que se alejaba a toda velocidad de un enorme logo de AT&T con forma de Estrella de la Muerte que quedaba destruido y en llamas.¹⁷

Así que pese a que para mediados de los 80 UNIX se había convertido en el sistema operativo estándar preferido para ordenadores de tiempo compartido, multiusuario y de alto rendimiento, UNIX no existía como tal. Los competidores del mercado de UNIX apenas podían esperar que el dueño del sistema, AT&T, lo estandarizase y a la vez compitiese con ellos, y el resto de los sistemas seguían siendo en cierto sentido legal derivados del original de AT&T. Es más, en sus panfletos de licencia, AT&T llegaba a insistir en que UNIX no era un nombre, sino un adjetivo, como en «el sistema UNIX»¹⁸.

La incipiente toma de conciencia de que la proliferación de sistemas no solo estaba expandiendo UNIX por el mundo sino también desgastándolo y desmoronándolo condujo a una serie de intentos cada vez más asombrosos y de perfil alto de «estandarizarlo». Dado que las tres ramas principales (BSD, que se convertiría en la predilecta de la industria como sistema operativo Solaris de Sun; Microsoft, más tarde SCO Xenix, y el System V de AT&T) surgieron todas ellas del mismo trabajo de AT&T y Berkeley realizado principalmente por Thompson, Ritchie y Joy, podría pensarse que la estandarización sería pan comido. Y nada más lejos de la realidad.

La figuración se desboca

La figuración del orden técnico y moral de los sistemas abiertos se desbocó aproximadamente entre 1986 y 1988, cuando había no menos de cuatro estándares internacionales rivales, representados por enormes consorcios de fabricantes de ordenadores (muchos de los cuales pertenecían a otros múltiples consorcios): POSIX, el consorcio

17. Eric Raymond, «Origins and History of Unix, 1969–1995», *The Art of UNIX Programming*, <http://www.faqs.org/docs/artu/ch02s01.html#id2880014>.

18. Libes y Ressler, *Life with UNIX*, p. 22. También recogido en Tanenbaum, «The UNIX Marketplace in 1987», p. 419.

X/Open, la OSF (*Open Source Foundation*, Fundación para el Software Abierto) y UNIX International. El punto ciego de los sistemas abiertos tuvo mucho que ver en este resultado demencial: los académicos, la industria y el gobierno no hallaban manera de ponerse de acuerdo sobre la estandarización. Una meta de dicha estandarización era surtir de opciones a los clientes; otra era permitir una competencia no restringida por medios «artificiales». Llegar a un cuerpo de código fuente estándar era imposible; una «definición de interfaz» estándar quedaba demasiado abierta a interpretación; los estándares gubernamentales y académicos eran demasiado complejos y costosos; no cabía fiarse de ningún estándar exclusivo de una compañía (porque tampoco cabía fiarse de que lo revelaran con antelación a sus propias innovaciones); y peor aún, los clientes seguían comprando y los distribuidores seguían despachando, y el mundo estaba cada vez más lleno de diversidad, no de estandarización.

UNIX *proliferó* rápidamente debido a la portabilidad, lo que llevó a que existieran múltiples versiones de un sistema operativo con un código fuente sustancialmente similar compartido por los académicos y licenciado por AT&T. Pero *se diferenció* igual de rápido debido a las bifurcaciones, según se iban añadiendo funcionalidades particulares a cada nueva versión portada. Algunas de estas funcionalidades se reincorporaron a la rama «principal» —aquella en la que trabajaban Thompson y Ritchie— pero el grueso de estas mutaciones se extendió azarosamente, bien mediante la compartición directa entre usuarios, bien mediante su implementación en versiones comerciales de reciente creación. Algunas funcionalidades no eran más que añadidos, pero otras podían extender el sistema de modos que posibilitaran ejecutar una aplicación en una versión pero no en otra.

La proliferación y diversificación de UNIX como sistema operativo tuvo efectos peculiares en el mercado emergente de UNIX como producto: las cuestiones técnicas comportaban otras de diseño y organización. El UNIX original llegó a ser como era debido a la peculiarísima estructura de la organización que lo creó y mantuvo: Laboratorios Bell y la comunidad mundial de usuarios y programadores. Los recién surgidos competidores, al concebir UNIX como un producto distinto del UNIX original, lo adoptaron precisamente por su portabilidad y por la promesa que suponían los sistemas abiertos como alternativa a las enormes unidades centrales. Pero a medida que UNIX se canalizaba hacia compañías ya existentes que tenían su estructura organizativa y

diseño propios, empezó a volverse incompatible consigo mismo, y el deseo de competencia en los sistemas abiertos demandaba iniciativas de estandarización de UNIX.

El primer paso en este sentido fue la creación de lo que se llamó «definición del interfaz», un estándar que enumeraba el conjunto mínimo de funciones que debería admitir cualquier versión de UNIX *en el nivel de interfaz*. De este modo, cualquier programador que escribiese una aplicación podría esperar que ella interactuase con cualquier versión de UNIX en cualquier máquina de la misma manera y obtuviese la misma respuesta (sin importar la implementación específica del sistema operativo o del código fuente que se usara). Idealmente las definiciones de interfaz, y sus extensiones, tendrían que publicarse y estar a libre disposición.

La definición del interfaz era un estándar que enfatizaba la portabilidad en el nivel de las aplicaciones, no del código fuente o del sistema operativo. Con ello se permitía que aplicaciones basadas en una versión de UNIX se instalasen y ejecutasesen en cualesquiera otras. El impulso de dicho estándar provino inicialmente de un grupo de usuarios de UNIX fundado por Bob Marsh en 1980 y llamado, según las convenciones de las jerarquías de archivos de la interfaz de UNIX, «/usr/group» (más tarde se lo renombró Uniforum). El estándar /usr/group/ de 1984 definía un conjunto de llamadas al sistema que, sin embargo, «fueron inmediatamente ignoradas y, a efectos prácticos, inservibles». ¹⁹ Parecía que el campo cambiaba demasiado rápido y que la proliferación e innovación de UNIX eran demasiado amplias para que un estándar así funcionase.

Con todo, el estándar /usr/group/ proporcionó un punto de partida para que organizaciones de estandarización más tradicionales —el IEEE (*Institute of Electrical and Electronics Engineers*, Instituto de Ingenieros Eléctricos y Electrónicos) y el ANSI (*American National Standards Institute*, Instituto Nacional Estadounidense de Estándares)— acometieran la tarea. Ambas instituciones tomaron dicho estándar como base para lo que sería el IEEE P1003 POSIX (*Portable Operating System Interface*, Interfaz de Sistema Operativo Portable). Durante los tres siguientes años, de 1984 a 1987, POSIX trabajaría diligentemente en la provisión de una definición de interfaz estándar para UNIX.

19. Libes y Ressler, *Life with UNIX*, p. 67.

En paralelo a este desarrollo, la versión de UNIX de AT&T se convirtió en la base de un estándar distinto, el SVID (*System V Interface Definition*, Definición de Interfaz del Sistema V), que intentó estandarizar un conjunto de funciones similar pero no idéntico al de los estándares /usr/group y POSIX. De este modo surgieron dos definiciones rivales de una interfaz estándar para un sistema que proliferaba rápidamente en cientos de minúsculos feudos de sistema operativo.²⁰ El peligro de que AT&T estableciese el estándar no se le escapaba a ninguno de sus competidores. Incluso si creaba una definición de interfaz rigurosamente abierta, la versión de UNIX de AT&T sería la primera en implementarla y continuamente tendría información privilegiada sobre cualquier cambio: si AT&T pretendía cambiar la implementación, podría modificar el estándar; si recibía demandas de modificación del estándar, podrían alterar su implementación antes de lanzar el nuevo.

En respuesta a esta amenaza, apareció un tercer participante en la carrera de los estándares: el consorcio X/Open, que aglutinaba a una variedad de fabricantes de ordenadores europeos (¡incluyendo a AT&T!) y cuya intención era desarrollar un estándar que comprendiera tanto SVID como POSIX. La iniciativa del X/Open nació de la inquietud europea por la posición dominante de IBM e incluía originalmente a Bull, Ericsson, ICL, Nixdorf, Olivetti, Philips y Siemens. Con arreglo a cierto gusto de los 80 por la integración de la actividad económica europea con respecto a Estados Unidos y Japón, estos fabricantes se agruparon tanto para distribuir en Europa un sistema operativo UNIX unificado (basado inicialmente en las versiones de Sun y BSD) como para intentar al mismo tiempo estandarizarlo.

X/Open representó una sutil transformación de las iniciativas de estandarización de la definición organizativa de los sistemas abiertos. Mientras que el estándar /usr/group fue desarrollado por usuarios individuales de UNIX, y POSIX por una asociación profesional reconocida (el IEEE), el grupo X/Open era un colectivo de compañías informáticas que se habían unido para fundar una entidad independiente que promoviese la causa de un UNIX estándar. Esta paradójica situación —la necesidad de compartir un estándar entre todos los competidores y la necesidad simultánea de mantener en secreto los detalles de ese produc-

20. Podría alegarse que una tercera definición, el estándar ANSI para el lenguaje de programación C, también cubría un terreno similar, lo que sin duda tendría que hacer para permitir que aplicaciones escritas en un sistema operativo fueran compiladas y ejecutadas en otro (véase Gray, *Open Systems*, pp. 55-58; Libes y Ressler, *Life with UNIX*, pp. 70-75).

to estandarizado para no perder ventaja— era considerada por muchos fabricantes (especialmente los europeos, con su larga experiencia del monopolio de IBM) como mutuamente destructiva. De ahí que la solución fuera emprender una especie de innovación *organizativa*, crear una nueva forma de estructura metacorporativa que pudiera posicionarse estratégicamente como interesada, al menos provisionalmente, en *colaborar*, y no competir, con otras empresas. De esta manera, las historias y promesas acerca de los sistemas abiertos fueron abriéndose camino desde los detalles del diseño técnico hasta los del diseño organizativo, y de ahí al orden moral de la competencia y la colaboración, del poder y la estrategia. Los «estándares» se convirtieron en productos que las compañías pretendían «vender» a su propia industria mediante la intermediación del consorcio.

En 1985 y 1986 el estado caótico de UNIX también se volvió frustrante para los principales fabricantes estadounidenses, especialmente para Sun Microsystems, que se había cimentado en la creación de un mercado de «terminales de trabajo» basadas en UNIX, ordenadores en red de gran potencia que podían competir al mismo tiempo con las unidades centrales y con los PCs. Fundada por Bill Joy, Vinod Khosla y Andreas Bechtolsheim, Sun se había convertido muy rápidamente en una compañía informática extraordinariamente exitosa. Las páginas de negocios de periódicos y revistas estaban ávidas por dilucidar si las terminales eran competidores viables de los PCs, en particular de los de IBM y Microsoft, así como del sistema operativo estándar *de facto*, DOS, para el que se escribió una variedad de aplicaciones extremadamente exitosas de ámbito empresarial, personal y doméstico.

Sun aprovechó la preocupación existente en torno a los sistemas abiertos, como evidencia el anuncio que publicó durante el verano de 1987 (véase figura 4). El anuncio juega sutilmente con dos preocupaciones: la primera apunta a los consumidores e insinúa que solo con Sun se puede alcanzar realmente la interoperabilidad entre todos los ordenadores de una empresa, no digamos ya de una red o de una industria; la segunda es más sutil y juega con los miedos de la propia industria informática, la ansiedad de que Sun pudiera fusionarse con una de las grandes compañías, como AT&T o Unisys, y arrinconar el mercado de los sistemas abiertos produciendo un estándar *de facto*.

Figuras 4a y 4 b

**Announcing
the biggest
merger in
the computer
business.**

**The merger
of every computer
in your business.**



La ansiedad de los sistemas abiertos en torno a las fusiones y la compatibilidad.

Anuncio de Sun Microsystems, *Wall Street Journal*, 9 de julio de 1987.

4a: «Anunciando la mayor fusión del mundo de los ordenadores»

4b: «La fusión de todos los ordenadores de su negocio»

De hecho, en octubre de 1987 Sun anunció que había alcanzado un acuerdo con AT&T: esta distribuiría una terminal de trabajo basada en la línea SPARC de Sun y adquiriría el 20% de esta compañía²¹. Como parte del anuncio, Sun y AT&T dejaron claro que pretendían fusionar dos de las versiones de UNIX que dominaban el mercado: el System V de AT&T y el Solaris derivado del BSD. Este movimiento asustó claramente al resto de fabricantes interesados en UNIX y en los sistemas abiertos, ya que sugería una especie de alineamiento de superpotencias que reestructuraría (y potencialmente dominaría) el mercado. Un artículo de 1988 del *New York Times* cita a un analista de la industria que califica la fusión como «un asunto preocupante en las más altas esferas de todas y cada una de las principales compañías informáticas de EEUU, y posiblemente del mundo», y sugiere que los fabricantes de la competencia «también temen que AT&T vaya gradualmente convirtiendo UNIX en un producto privativo, utilizable únicamente en sus máquinas o en las de Sun».²² La ansiedad en la industria era tan

21. «AT&T Deal with Sun Seen», *New York Times*, 19 de octubre de 1987, D8.

22. Thomas C. Hayesdallas, «AT&T's Unix Is a Hit at Last, and Other Companies Are Wary», *New York Times*, 24 de febrero de 1988, D8.

grande que en marzo de 1988 Unisys (un fabricante de ordenadores antes llamado Burroughs-Sperry) anunció que colaboraría con AT&T y Sun para implantar UNIX en sus unidades centrales y para hacer que sus aplicaciones comerciales se ejecutasesn en UNIX. Este anuncio equivalía a la admisión por parte de Unisys de que no habría futuro en la informática *privativa* de alto nivel —el negocio en el que había basado su reputación hasta el momento— a no ser que pudiera formar parte del consorcio que podría poseer el estándar.²³

En respuesta a esta jugada, percibida como una colusión, un grupo de compañías estadounidenses y europeas se aliaron para formar otra organización rival —una que se solapaba parcialmente con X/Open pero que ahora *incluía* a IBM—, llamada OSF. Constituida como organización sin ánimo de lucro, la OSF incluía a IBM, Digital Equipment, Hewlett-Packard, Bull, Nixdorf, Siemens y Apollo Computer (el competidor más directo de Sun en el mercado de las terminales). Su meta explícita era crear un «estándar rival» para UNIX que estaría disponible en el hardware que dichas empresas fabricaran (y que, según algunos reportajes periodísticos, se basaba en el AIX de IBM, que acabaría llamándose OSF/1). Al principio pareció que AT&T apoyaba a la fundación, al sugerir que si la OSF podía dar con un estándar, AT&T haría su System V compatible con él. Así pues, el de 1988 fue el verano del amor abierto. Todos y cada uno de los principales fabricantes del mundo formaban parte de un consorcio u otro, y algunos formaban parte de dos —cada uno de los cuales promovía un estándar diferente.

De todas las corporaciones, la que más hizo por significarse como originadora del concepto de sistemas abiertos fue Sun. Así, esta empresa se atribuía de manera muy amplia el éxito de la estandarización de los sistemas abiertos, como por ejemplo en un anuncio publicado en agosto de 1988 (figura 5, pág. siguiente), donde declaraba, entre otras cosas:

Pero aún hay más, pues esas cifras de ventas confirman la amplia aceptación de la idea global en que se sustenta Sun: la idea de los Sistemas Abiertos. Sistemas basados en estándares tan universalmente aceptados que permiten combinaciones de hardware y software de literalmente miles de proveedores independientes... Así que, por vez primera, ya no estás confinado en la compañía que fabricó tus ordenadores. Incluso si somos nosotros.

23. «Unisys Obtains Pacts for Unix Capabilities», *New York Times*, 10 de marzo de 1988, D4.

Figura 5

It pays to be open.

Fiscal Year	Sales (\$ Millions)
83	~100
84	~200
85	~400
86	~800
87	~1200
88	~1500

During the past year we delivered \$1 billion worth of Sun computers and services. Gratifying enough for a company that's just six years old. But what's more, those sales confirm a broad acceptance of the whole idea behind Sun. The Open Systems idea. Systems based on standards so universally accepted that they allow combinations of hardware and software from literally thousands of independent vendors. Standards such as the UNIX® Operating System, SPARC™ processors, NFS® networking software, and the OPEN LOOK™ user interface. So for the first time, you're no longer locked into the company who made your computers. Even if it's us. Not that we're concerned. In a free market, the best products win out. And nobody but Sun offers such powerful and cost-effective distributed computing solutions. Solutions based on state-of-the-art workstations, servers, and networking software. A Sun workstation moves data many times faster than a PC, and its graphics are stunningly better. A Sun server is as powerful as a supermini, but at a fraction of the cost. And Sun's networking software provides the entire system with unmatched ability to talk to any kind of computer. Transparently. Effortlessly. Desk to desk. Desk to mini. Desk to mainframe. Even continent to continent. It was our vision that this kind of distributed computing could bring unprecedented power not only to the individual, but to the workgroup, and the entire company—in ways that PCs, minis and mainframes never could. Power to access, share, and work with information to a greater degree than ever before. That is the very reason Sun's Open Systems are at work today throughout all industries. From finance to semiconductors, automotive to aviation. At the largest and smallest corporations in the world, all over the world. These companies saw the future the same way we did. Wide open.

SUN
microsystems
The Network Is The Computer™

Ser abierto compensa: la versión de Sun de unos sistemas abiertos rentables y de éxito.

Anuncio de Sun Microsystems en el *New York Times* del 2 de agosto de 1988.²⁴

24. Incluimos aquí la traducción completa del anuncio:

«Ser abierto compensa. Durante el año pasado Sun distribuyó ordenadores y servicios Sun por valor de 1000 millones de dólares. Es bastante gratificante para una compañía que solo tienes seis años. Pero aún hay más, pues esas cifras de ventas confirman la amplia aceptación de la idea global en que se sustenta Sun: la idea de los Sistemas Abiertos. Sistemas basados en estándares tan universalmente aceptados que permiten combinaciones de hardware y software de literalmente miles de proveedores independientes. Estándares tales como el sistema operativo UNIX, los procesadores SPARC™, el software de red NFS™,

El anuncio prosigue sugiriendo que «en un mercado libre, los mejores productos se llevan la palma», pese a que Sun jugaba a dos bandas en cada batalla de estandarización, cooperando tanto con AT&T como con la OSF. Pero en octubre de 1988, a Sun le había quedado claro que la idea aún no se había hecho «tan universal». Ese mes, AT&T y Sun se aliaron con otros diecisiete fabricantes para formar un consorcio rival: Unix International, una especie de «coalición de los dispuestos»²⁵ que respaldaría el System V de AT&T como único estándar abierto verdadero. En un anuncio a toda página en el día de Halloween de 1988 (figura 6) —publicado simultáneamente en el *New York Times*, el *Washington Post* y el *Wall Street Journal*—, la retórica acerca del éxito alcanzado se mantenía, pero ahora en vez de «la idea de los Sistemas Abiertos» era «tu demanda de soluciones basadas en el UNIX System V la que dio paso a la era de la arquitectura abierta». En vez de un estándar para todos los sistemas abiertos, ahora había una guerra de todos contra todos para asegurar a los clientes que habían hecho la elección adecuada, no de hardware o de software, sino de *estándar*.

y la interfaz de usuario OPEN LOOK™. Así que, por vez primera, ya no estás confinado en la compañía que fabricó tus ordenadores. Incluso si somos nosotros. Tampoco es que nos preocupe. En un mercado libre, los mejores productos se llevan la palma. Y nadie más que Sun ofrece soluciones de informática distribuida tan potentes y rentables. Soluciones que se basan en terminales, servidores y software de red de vanguardia. Una terminal Sun maneja datos varias veces más rápido que un PC, y sus gráficos son asombrosamente mejores; un servidor Sun es tan potente como un *supermini*, pero a una fracción de su coste; y el software de red de Sun proporciona al sistema entero una capacidad sin par para comunicarse con cualquier tipo de ordenador. Con transparencia. Sin esfuerzo. De escritorio a escritorio. De escritorio a *mini*. De escritorio a unidad central. Incluso de continente a continente. Fue una visión nuestra que esta clase de informática distribuida podría proporcionar una potencia sin precedentes no sólo al individuo, sino también al grupo de trabajo y a toda la empresa —y ello de modos inalcanzables para los PCs, *minis* y unidades centrales. Potencia para acceder, compartir y trabajar con la información a un grado mayor que nunca antes. He aquí la razón de que los sistemas abiertos de Sun sean hoy empleados en todas las industrias: desde la financiera a la de los semiconductores, de la automoción a la aviación. En las mayores corporaciones del mundo y en las más pequeñas, por todo el planeta. Estas compañías vieron el futuro del mismo modo que nosotros. Abierto de par en par. Sun Microsystems. La Red Es El Ordenador». [N. del E.]

25. La expresión «coalición de los dispuestos» (*«coalition of the willing»*) adquirió su sinistra fama con el empleo que de ella hizo la Administración de George W. Bush para orquestar una supuesta fuerza multinacional (a la que se sumó entusiasta el Gobierno español de José María Aznar) que le sirviera de amparo para su invasión unilateral de Irak en 2003 al margen de cualquier mandato de la ONU. [N. del E.]

A la proliferación de estándares y de consorcios de estandarización se la denomina a menudo las guerras de UNIX de finales de los 80, pero la creación de tales consorcios no trazaba fronteras bien demarcadas. Otra metáfora que al parecer alcanzó bastante popularidad en la prensa de la época fue la de la guerra de «bandas» (a la que sin duda contribuyó la creación de otro consorcio industrial llamado informalmente la Banda de los Nueve, que estuvo implicada en la disputa acerca de si en los PCs deberían instalarse buses MicroChannel o EISA). La idea de que diversas compañías formasen bandas para pelear entre sí, al estilo de los *Bloods* y los *Crips* —o quizás más bien de los *Jets* y los *Sharks* de *West Side Story*, solo que sin cantar—, era indudablemente una metáfora atractiva durante el punto álgido de la muy real y prominente guerra de bandas callejeras en Los Angeles. Ahora bien, como señalaba un artículo del *New York Times*, se trataba de bandas extrañas:

Dado que «apertura» y «cooperación» son las palabras en boga detrás de estas alianzas, la banda en cuestión a menudo le pide a su enemigo que se le una. Y con frecuencia este acepta, bien para que no parezca que se opone a la apertura, bien para tener controlado al grupo: IBM fue invitada a unirse a la corporación por los Sistemas Abiertos, aun cuando el claro aunque no declarado propósito de este grupo era debilitar la influencia de IBM en el mercado; AT&T negoció su incorporación a la OSF, pero las conversaciones se rompieron hace poco. Algunas compañías ven del todo coherente ser miembros de bandas rivales. [...] Unas diez compañías son miembros tanto de la OSF como de su archirrival, Unix International.²⁶

26. Andrew Pollack, «Computer Gangs Stake Out Turf», *New York Times*, 13 de diciembre de 1988, D1. Véase también Evelyn Richards, «Computer Firms Get a Taste of 'Gang Warfare'», *Washington Post*, 11 de diciembre de 1988, K1; Brit Hume, «IBM, Once the Bully on the Block, Faces a Tough New PC Gang», *Washington Post*, 3 de octubre de 1988, E24.

Figura 6

If you believe UNIX® System V is the open systems standard, you're not alone.

After all, it was your demand for UNIX System V-based solutions that ushered in the era of open architecture, and finally made open systems a reality.

Now you're solving problems today and building for tomorrow with the only true standard available now:

UNIX System V

The companies represented on this page are committed to providing open, UNIX System V-based solutions, and to furthering the development of the industry-standard UNIX System V operating environment.

This month, the one millionth UNIX system-based solution will be installed.

As independent software and systems suppliers continue to deliver the full business benefits of open systems, each one will take direction from you, the users, who have led the crusade for open systems, for choice, and for freedom.

Thanks to you, the era of open systems has arrived, and UNIX System V is the blueprint.

UNIX System V
OPEN FOR BUSINESS

Las Guerras de UNIX, Halloween de 1988: anuncio de UNIX International en el *Wall Street Journal* y el *New York Times* 31 de octubre de 1988.²⁷

27. Incluimos aquí la traducción completa del anuncio: «Si crees que UNIX System V es el estándar de los sistemas abiertos, no estás solo. Al fin y al cabo, fue tu demanda de soluciones basadas en el UNIX System V la que dio paso a la era de la arquitectura abierta y finalmente hizo realidad los sistemas abiertos. Ahora estás resolviendo problemas de hoy y construyendo para el mañana con el único estándar verdadero que se encuentra disponible hoy: UNIX System V. Las compañías representadas en esta página están comprometidas con la provisión de soluciones abiertas basadas en el UNIX System V y con el impulso del desarrollo del entorno operativo del estándar industrial UNIX System V. Este mes se instalará la millonésima solución

La proliferación de estos consorcios se puede entender de varias maneras. Podría alegarse que surgieron en un periodo —durante la Administración Reagan— en que el control antimonopolístico había disminuido hasta tal punto que las corporaciones informáticas no veían esta colusión como una actividad arriesgada respecto de las leyes *antitrust*. También podría alegarse que dichos consorcios representaban un reconocimiento de que la atención sobre el control del hardware (el sentido de *privativo*) se había desplazado hacia el control del «estándar abierto» por uno o varios fabricantes, es decir, que la competencia ya no se basaba en productos superiores, sino en «poseer el estándar». Resulta significativo que los consorcios industriales arrollasen rápidamente en los medios de comunicación a las iniciativas de ámbito nacional, como el estándar POSIX del IEEE, una indicación de que nadie esperaba que el gobierno, las organizaciones sin ánimo de lucro o las asociaciones profesionales universitarias fueran quienes resolviesen la disputa mediante la declaración de un estándar, sino que fuera la propia industria la que forjase uno, *de facto* o de otra manera. Y otro modo más de entender el surgimiento de estos consorcios es como una especie de vigilancia mutua del mercado, una especie de estrategia paranoide de enseñarse unos a otros lo justo para asegurarse de que nadie superara al resto y acabara con la frágil competencia existente.

Lo que esta proliferación de estándares y consorcios de UNIX representa más claramente, sin embargo, es el punto ciego de los sistemas abiertos: la dificultad de que exista colaboración y competencia al mismo tiempo en el contexto de unas normas sobre propiedad intelectual que captan de modo incompleto las características específicas e inusuales del software. Para los participantes de este mercado, la estructura de la propiedad intelectual era irrefutable —sin ella la mayoría de participantes asumía que cesaría la innovación y desaparecerían los incentivos. A pesar de que el secretismo acuciaba a la industria, sus clientes buscaban tanto la apertura como la compatibilidad. Estas exigencias contradictorias demostraron ser irresolubles.

basada en el sistema UNIX. Mientras los proveedores independientes de software y de sistemas continúan distribuyendo todos los beneficios empresariales de los sistemas abiertos, todos ellos tomarán la dirección que marquéis vosotros, los usuarios, que habéis liderado la cruzada por los sistemas abiertos, por la capacidad de elección y por la libertad. Gracias a vosotros, la era de los sistemas abiertos ya ha llegado, y UNIX System V es el plan de acción. UNIX System V. ABIERTO A LOS NEGOCIOS». [N. del E.]

Desenlace

Irónicamente, las guerras de UNIX no terminaron con el surgimiento de un ganador, sino con la reafirmación de la informática privativa: Microsoft Windows y Windows NT. En vez de una salida victoriosa para los sistemas abiertos que diera paso a la era de la integración fluida de componentes diversos, sucedió lo contrario: Microsoft se las arregló para hacerse con una parte enorme del mercado de la informática, tanto de escritorio como de alto rendimiento, haciendo valer su marca, la ubicuidad de DOS y la dependencia de los desarrolladores de aplicaciones del monstruo «Wintel» (Windows más procesadores Intel). Microsoft triunfó en gran parte por las mismas razones por las que fracasó el sueño de los sistemas abiertos: la estructura legal de la propiedad intelectual favorecía la existencia de un fuerte monopolio corporativo sobre un único producto de marca por encima de un frágil surtido de componentes «abiertos» y rivales. Los inversores y las corporaciones no podían esperar grandes ganancias de una industria de tipos majos que compartían el código fuente y hacían que los componentes funcionasen conjuntamente. Microsoft, en cambio, había decidido hacer eso de manera interna: no necesitaba formar consorcios o estandarizar nada si podía hacer valer su dominio del mercado para difundir su sistema operativo por todas partes. Se trató, como les gusta decir a los observadores de los estándares, del triunfo de la estandarización *de facto* sobre la *de iure*. Se trató, en suma, del retorno a los desdichados encadenados al monopolio de IBM —pero con un nuevo amo del calabozo.

El desenlace de la historia de los estándares de UNIX fue rudo: en 1993 AT&T vendió su UNIX System Labs (incluyendo todo el código fuente y los derechos originales) a Novell, que a su vez lo vendió a SCO dos años después. Novell también vendió (o transfirió) la marca comercial UNIX™ al grupo X/Open, que continuó pugnando por la estandarización, incluyendo una única especificación universal de UNIX. En 1996 X/Open y la OSF se fusionaron para formar el Open Group,²⁸ que acabó aunando fuerzas con el IEEE para convertir POSIX en una única especificación de UNIX en 2001. De ahí continuaron impulsando la visión original de los sistemas abiertos, aunque se cuidan mucho de emplear ese nombre o concepto, refiriéndose en su lugar al trabalenguas de marca registrada «*Boundaryless Information Flow*»

28. «What Is Unix?», The Unix System, http://www.unix.org/what_is_unix/history_timeline.html.

(«Flujo de Información Ilimitado») y empleando una retórica actualizada y nuevamente inescrutable:

El *Boundaryless Information Flow*, una representación abreviada de «acceso a información integrada para apoyar mejoras en procesos de negocio», representa un estado deseado de la infraestructura de una empresa y es específico de las necesidades de negocio de la organización.²⁹

El Open Group, como tantos otros protagonistas de la historia de los sistemas abiertos, reconoce el surgimiento del «código abierto» como un retorno al ahora único camino verdadero del flujo de información sin límites. Eric Raymond, por supuesto, ve continuidad y renovación (siendo nada desdeñable en ello su propia participación en el movimiento de código abierto), afirmando en su libro *Art of UNIX Programming*:

El Movimiento de Código Abierto está basándose en estos cimientos estables y está causando un resurgir del entusiasmo por la filosofía UNIX. En muchos sentidos puede verse el Código Abierto como la auténtica realización de los Sistemas Abiertos que asegurará que continúa ganando más y más fuerza.³⁰

Dicha continuidad, por supuesto, niega deliberadamente la centralidad del componente legal, como ya hicieron Raymond y la Open Source Initiative en 1998. La distinción entre un mercado robusto de sistemas operativos UNIX y una *infraestructura* estándar basada en UNIX sobre la que pudieran darse otros mercados y actividades sigue estando confusa incluso para los más cercanos al dinero y a las máquinas. Aún no existe, y muy bien podría no existir jamás.

El crecimiento del software libre en los 80 y 90 dependía de la apertura como concepto y componente que se figuró durante las guerras de UNIX. Fue en este periodo cuando la FSF (y otros grupos, de maneras diferentes) empezaron a reconocer la centralidad de la cuestión de la propiedad intelectual respecto del objetivo de generar una infraestruc-

29. «About the Open Group», The Open Group, <http://www.opengroup.org/overview/vision-mission.htm>.

30. «What Is Unix?», The Unix System, http://www.unix.org/what_is_unix/history_timeline.html.

tura para la creación exitosa de sistemas abiertos.³¹ El proyecto GNU (acrónimo recursivo de «*GNU's Not Unix*», «GNU No es Unix») en particular, pero también el X Windows System del MIT, los sistemas RPC (*Remote Procedure Call*, Llamada a Procedimiento Remoto) y NFS (*Network File System*, Sistema de Archivos de Red) creados por Sun, y herramientas como sendmail y BIND eran, cada cual a su modo, experimentos con modalidades alternativas de concesión de licencias que a finales de los 80 circularon ampliamente en diversas versiones de UNIX. Por tanto, la experiencia de los sistemas abiertos, si bien técnicamente fue un fracaso por lo que concierne a UNIX, supuso sin embargo un profundo aprendizaje para una generación entera de ingenieros, *hackers, geeks* y empresarios. Al igual que UNIX tuvo una vida pedagógica propia, quedando inculcado en la mente de los ingenieros como el paradigma de los sistemas operativos, los sistemas abiertos tuvieron en gran parte el mismo efecto, haciendo realidad una incipiente filosofía de apertura, interconexión, compatibilidad e interoperabilidad —en resumen, de *disponibilidad y modificabilidad*— que estaba en conflicto con las estructuras de propiedad intelectual existentes. Por expresarlo en términos freudianos: la neurosis de los sistemas abiertos no estaba curada, pero la estructura de su imposibilidad había quedado mucho más clara a todo el mundo. Como sistema operativo, UNIX no desapareció en absoluto —pero como mercado sí lo hizo.

Sistemas Abiertos 2: Redes

La lucha por estandarizar UNIX como plataforma para sistemas abiertos no fue la única en este ámbito. En paralelo a las guerras de UNIX, se libraba otra encarnizada «guerra religiosa». El intento de estandarizar las redes —en particular, los protocolos para la interconexión de redes múltiples, diversas y autónomas— también fue un aspecto clave de la historia de los sistemas abiertos en los 80.³² La guerra entre

31. Larry McVoy fue una de las primeras voces dentro de Sun que defendió como solución al problema de los sistemas abiertos la adopción del software libre. Larry McVoy, «The Sourceware Operating System Proposal», 9 de noviembre de 1993, <http://www.bitmover.com/lm/papers/srcos.html>.

32. La distinción entre un protocolo, una implementación y un estándar es importante: Los *protocolos* son descripciones de los términos precisos por los que dos ordenadores pueden comunicarse (como un diccionario y un manual para comunicarse). Una *implementación* es la creación de un programa informático que usa un protocolo (esto es, que efectúa realmente la comunicación; en consecuencia, dos implementaciones que usen el mismo protocolo de-

las familias de protocolos TCP/IP y OSI fue también una historia de fracaso y de éxito sorprendente: la historia de un estándar exitoso que contaba con aprobación internacional (los protocolos OSI) que quedó eclipsado por uno experimental de financiación militar (los protocolos TCP/IP) que ejemplificaba un proceso de estandarización alternativo e inusual. Los órdenes morales y técnicos expresados por OSI y TCP/IP se enclavan, como los de UNIX, en la frontera entre lo gubernamental, lo universitario y lo industrial, representando imaginarios sociales opuestos en los que el poder y la legitimidad se organizan de manera diferente y, como resultado, se expresan de manera diferente en la tecnología.

OSI y TCP/IP partieron con metas diferentes: OSI pretendía satisfacer a todo el mundo, ser el modelo completo y exhaustivo ante el que se convalidasen todas las implementaciones en competencia; TCP/IP, por el contrario, enfatizaba la interconexión fácil y robusta de redes diversas. TCP/IP es un protocolo desarrollado por arranque autosostenido³³ entre estándar e implementación, un modo ilustrado por el sistema RFC (*Requests for Comments*, Peticiones de Comentarios) que se desarrolló en paralelo como parte del proyecto Arpanet. OSI, por su parte, era un «modelo» o estándar de referencia desarrollado por organizaciones de estandarización de prestigio internacional.

A mediados de los 80 OSI iba camino de ser adoptado internacionalmente, pero hacia 1993 había quedado casi completamente eclipsado por TCP/IP. El éxito de TCP/IP es significativo por tres razones: 1) *disponibilidad* —el propio protocolo estaba disponible a través de la red y su desarrollo estaba abierto a todo el mundo, mientras que OSI era un estándar caro y burocráticamente confinado en el que la participación estaba restringida a representantes estatales y corporativos, organizados a través de la ISO (*International Organization for Standardization*, Organización Internacional para la Estandarización) de Ginebra; 2) *modificabilidad* —TCP/IP podía copiarse de cualquier implementación existente (como la versión BSD de UNIX) y mejorarse, mientras que OSI era un estándar complejo que disponía de pocas implementaciones disponibles para copiar; 3) *serendipia* —por la que brotaron nuevos usos que aprovechaban las dos características

berían ser capaces de compartir datos). Un *estándar* define qué protocolo debería usarse por qué ordenadores y con qué propósitos. Dicho estándar puede o no definir el protocolo, pero establecerá límites para los cambios introducidos en tal protocolo.

33. Véase nota 42 del Capítulo I. [N. del E.].

anteriores, incluyendo la «*killer app*» («aplicación revolucionaria») que fue la *World Wide Web*, construida para funcionar sobre redes basadas en TCP/IP ya existentes y que convenció a muchos fabricantes de implementar este protocolo en vez del OSI, o además de este.

El éxito de TCP/IP sobre OSI también fue significativo por ejemplificar la diferencia en los *procesos* de estandarización. El estándar OSI (como todos los estándares oficiales internacionales) se concibe y publica como una ayuda al crecimiento industrial: fue diseñado de acuerdo con las reglas básicas de la propiedad intelectual y como un intento de facilitar la expansión de los mercados en el ámbito de las redes. Así, OSI sería un estándar «neutral respecto de proveedores»: estos crearían sus propias implementaciones secretas validables por OSI de modo que pudiera esperarse que interoperaran correctamente con otros sistemas igualmente validados. En claro contraste, los protocolos TCP/IP no se publicaban (en ningún sentido convencional del término) y sus implementaciones tampoco eran validadas por ninguna organización internacional de estandarización legítima; en vez de eso, los protocolos están en sí representados por implementaciones que permiten conectarse a la misma red (donde están disponibles los mismos protocolos TCP/IP y sus implementaciones). El hecho de que alguien solo pueda acceder a la red si posee o realiza una implementación del protocolo es visto generalmente como la validación definitiva: eso es que funciona.³⁴ En este sentido, la lucha entre TCP/IP y OSI es indicativa de un conflicto muy familiar del siglo XX acerca del papel y el alcance de la planificación y regulación gubernamentales (frente a la actividad empresarial y la libertad individual), quizás representado del mejor modo por las figuras gemelas de Friedrich Hayek y Maynard Keynes. En esta historia, es la aversión de Hayek a la planificación y el subsiguiente privilegio del orden espontáneo la que acaba triunfando, no la visión paternalista keynesiana del gobierno como cuerpo neutral que absorbe o alienta los vaivenes del mercado.

34. Las ventajas de semejante red no planificada e impredecible han dado en identificarse en retrospectiva como un principio de diseño. Véase Gillespie, «Engineering a Principle» para un excelente análisis de la historia de las redes «extremo a extremo» o «estúpidas».

Arranque autosostenido de redes

La «guerra religiosa» entre TCP/IP y OSI se dio en un contexto de intensa competencia entre fabricantes de ordenadores y durante un periodo de vibrante experimentación con redes informáticas en todo el mundo. Como en la mayoría de avances informáticos, IBM fue uno de los primeros fabricantes en introducir un sistema de red para sus aparatos a principios de los 70: el SNA (*System Network Architecture*, Arquitectura de Red de Sistemas). DEC siguió su estela con su DECnet o DNA (*Digital Network Architecture*, Arquitectura de Red Digital), al igual que Univac con DCA (*Distributed Communications Architecture*, Arquitectura de Comunicación Distribuida), Burroughs con BNA (*Burroughs Network Architecture*, Arquitectura de Red de Burroughs) y demás. Estas arquitecturas, como los sistemas operativos privativos de la época, eran consideradas redes cerradas, redes que interconectaban un número centralizadamente planificado y especificado de máquinas del mismo tipo o del mismo fabricante. El objetivo de tales redes era, pues, establecer conexiones internas a una empresa, aunque ello implicase sistemas muy extensos geográficamente (de las sucursales a la sede central, por ejemplo). En suma, las redes también habían de ser productos.

Las décadas de los 70 y 80 vieron una experimentación extraordinariamente vibrante con redes académicas, militares y comerciales. Robert Metcalfe había desarrollado Ethernet en el Xerox PARC a mediados de los 70, y luego IBM creó una tecnología similar llamada «de paso de testigo en anillo» [«*token ring*»]. Llegados los 80, los militares descubrieron que Arpanet estaba usándose predominantemente por ingenieros informáticos y no solo para aplicaciones militares, y decidieron escindirla en MILNET y CSNET.³⁵ Los Servicios de Tablón de Anuncios, que conectaban vía módem los PCs para descargar archivos, aparecieron a finales de los 70, y de ahí surgió el muy exitoso experimento de Tom Jennings llamado FidoNet.³⁶ Ya en los 80, una red social ya existente de profesores universitarios de la Costa Este de EEUU puso en marcha una red relativamente exitosa llamada BITNET (*Because It's*

35. William Broad, «Global Network Split as Safeguard», *New York Times*, 5 de octubre de 1983, A13.

36. Véase el incomparable *BBS: The Documentary* (en DVD), dirigido por Jason Scott (Boston: Bovine Ignition Systems, 2005), <http://www.bbsdocumentary.com/>.

There Network, «Red Porque Está Ahí»).³⁷ El uucp (*Unix to Unix Copy Protocol*, Protocolo de Copia de Unix a Unix), que posibilitó el inicio de Usenet, se desarrolló a finales de los 70 y fue ampliamente utilizado hasta mediados de los 80 para interconectar ordenadores UNIX. En 1984 la NSF (*National Science Foundation*, Fundación Nacional de la Ciencia) lanzó un programa de financiación de investigaciones en el campo de las redes y creó las primeras grandes troncales de NSFNet, sucesora de CSNET y Arpanet³⁸.

En los 70 las compañías de telecomunicaciones y las empresas emergentes derivadas de las universidades experimentaron mucho con los llamados sistemas «videotex», de los que el más conocido y ampliamente implementado es el Minitel francés.³⁹ Dichos sistemas se diseñaron para usuarios concebidos como consumidores y a menudo proporcionaban en forma embrionaria muchos de los servicios hoy generalizados en Internet (desde comparadores de precios para comprar coche hasta guías telefónicas y pornografía).⁴⁰ A finales de los 70 los sistemas videotex estaban en proceso de estandarización por el CCITT (*Commité Consultative de Information, Technologie et Télécommunications*, Comité Consultivo de Información, Tecnología y Telecomunicaciones) de la UIT (Unión Internacional de Telecomunicaciones) de Ginebra. Esta iniciativa se combinaría con el trabajo de la ISO sobre OSI, que se había originado a partir de la labor realizada en Honeywell.⁴¹

Un rasgo importante unificaba casi todos estos experimentos: las redes de los fabricantes de ordenadores generalmente arrancaban de forma autosostenida sobre, o se superponían a, infraestructuras de telecomunicaciones preexistentes construidas por empresas en situación de monopolio regulado o de titularidad estatal. Inevitablemente, esta situación auguraba problemas, ya que mientras que los proveedores de telecomunicaciones son entidades altamente reguladas, la industria informática ha carecido casi por completo

37. Grier y Campbell, «A Social History of Bitnet and Listserv 1985-1991».

38. Sobre Usenet, véase Hauben y Hauben, *Netizens*. Véase también Pfaffenberger, «A Standing Wave in the Web of Our Communications».

39. Schmidt y Werle, *Coordinating Technology*, cap. 7.

40. Véase, por ejemplo, Martin, *Viewdata and the Information Society*.

41. Existe poca información sobre el desarrollo de los sistemas abiertos; con todo, puede encontrarse una breve nota de William Stallings, autor del que quizás sea el libro de texto más ampliamente empleado sobre conexión de redes, en «The Origins of OSI,» <http://williamstallings.com/Extras/OSI.html>.

de regulación desde su origen. Y dado que una parte cada vez más esencial del negocio informático conllevaba el transporte de señales mediante sistemas de telecomunicaciones de forma desregulada, la industria de las telecomunicaciones se sintió naturalmente en desventaja.⁴² No se trata de que estas compañías fueran lentas para responder a la necesidad de comunicación de datos, sino de que su capacidad para experimentar con productos y prácticas fuera del campo de la telefonía y la telegrafía estaba a menudo obstruida por recelos acerca de las leyes antimonopolísticas.⁴³ Por contra, la industria informática no regulada veía los titubeos de la industria de las telecomunicaciones (o de los proveedores nacionales), bien como inercia burocrática, bien como desesperados intentos por mantener el control y el poder sobre las redes existentes —aunque a ningún fabricante de ordenadores le entusiasmaba la idea de construir su propia red física habiendo ya tantas.

TCP/IP y OSI se han convertido en emblemas de la división entre el mundo de las telecomunicaciones y el de la informática, volviendo moneda corriente las metáforas de guerras religiosas, ajustes de cuentas o guerras frías.⁴⁴ Un relato particularmente pícaro de esta época se encuentra en la obra de Carl Malamud *Exploring the Internet: a Technical Travelogue*, que documenta las visitas (físicas) del autor a sitios de Internet por todo el globo, sus conversaciones (con cervezas de por medio) con investigadores sobre detalles técnicos de las redes que han creado, así como sus propias posturas de estilo típicamente *geeky* y ocasionalmente ofensivo sobre la diferencia cultural⁴⁵. Un subtema de este relato es la guerra religiosa entre Ginebra (particularmente la UIT) e Internet: Malamud cuenta la historia de cuando le pidió a la UIT que liberase las 19 000 páginas de su «libro azul» de estándares de Internet para facilitar su adopción y difusión.

La resistencia de la UIT y los intentos heroicos aunque algo quijotescos de Malamud son una parábola de los imaginarios morales y técnicos en torno a la apertura —de hecho, su relato se inspira específicamente en la lucha entre la UIT y la IEC por definir los estándares de Internet.

42. Brock, *The Second Information Revolution* es una buena fuente introductoria para este conflicto, al menos en sus contornos de políticas públicas. La FCC dictó dos resoluciones (conocidas como «Computer 1» y «Computer 2») que trataron de abordar este conflicto intentando definir qué pasaba por comunicación de voz y qué por comunicación de datos.

43. Brock, *The Second Information Revolution*, cap. 10.

44. Drake, «The Internet Religious War».

45. Malamud, *Exploring the Internet*; véase también Michael M. J. Fischer, «Worlding Cyberspace».

camente en el pasado usable de Giordano Bruno.⁴⁶ El proyecto «bruno» demuestra el abismo que hay entre dos modelos de legitimidad —los de la ISO y la UIT— en los que los estándares representan el consenso legal y legítimo de una industria regulada, aprobados por los Estados miembros, pagados y aplicados por los gobiernos, e implementados y seguidos por las corporaciones.

Lo opuesto a ISO es el estilo *ad hoc* y experimental de los investigadores de Arpanet e Internet, donde los estándares están libremente disponibles y las implementaciones representan el modo de lograr consenso, más que el resultado de dicho consenso. En realidad, tal oposición retórica dista mucho de ser absoluta: en Internet se usan muchos estándares ISO, y esta sigue siendo una organización poderosa y legítima. Pero el choque entre la industria establecida (de telecomunicaciones) y la emergente (de redes informáticas) supone un contexto importante para comprender la lucha entre OSI y TCP/IP.

La necesidad de que existan protocolos de red estándar está fuera de discusión: la interoperabilidad es el sustento básico de una red. No obstante, las metas de los protocolos OSI y TCP/IP diferían de modos importantes, con profundas implicaciones para la forma de dicha interoperabilidad. Los objetivos de OSI eran la completitud, el control y la exhaustividad. OSI nació de la industria de las telecomunicaciones, que tenía un largo historial afrontando las vicisitudes de enlazar redes y facilitar la comunicación por todo el planeta, un problema que requería un sólido proceso de consenso y negociación entre grandes y poderosas entidades gubernamentales, además de entre fabricantes y proveedores más pequeños. OSI tenía los pies bien plantados en las organizaciones internacionales de estandarización como la ISO y la UIT (esta última tan antigua como las propias telecomunicaciones, pues data de la década de 1860).

Incluso siendo a menudo objeto de mofa por ser lentos, burocráticos o engorrosos, los procesos de dichas organizaciones —basados en el consenso, el acuerdo internacional y la rigurosa especificación técnica— tienen una legitimidad fuera de cuestión. Los representantes nacionales y corporativos que acuden a los debates de la ISO y la UIT,

46. El pasado usable de Giordano Bruno es invocado por Malamud para señalar la naturaleza hermética de su propia aspiración a publicar abiertamente los estándares que la ISO se negaba a liberar. El destino de Bruno a manos de la Inquisición romana dependió en buena parte de su aceptación de la cosmología copernicana, por lo que, al igual que Galileo, ha sido una figura natural de las reivindicaciones revolucionarias durante la década de los 90.

así como quienes diseñan, redactan y votan sobre estos estándares, no suelen ser burócratas sino ingenieros y directivos directamente interesados en las necesidades de su clientela. Este proceso orientado al consenso supone que los estándares de la ISO y la UIT intentan satisfacer los objetivos de todos los miembros, y por tanto tienden a ser documentos muy extensos, complejos y de gran especificidad. Dichos estándares se venden generalmente a corporaciones y otros actores que necesiten usarlos, en vez de publicarse libremente, un hecho que hasta hace poco reflejaba su legitimidad, más que su falta de ella.

El conjunto TCP/IP, por el contrario, surgió en condiciones muy diferentes.⁴⁷ Estos protocolos eran parte de un proyecto de investigación experimental financiado por el Departamento de Defensa: Arpanet. Los protocolos iniciales de esta red (el NCP, *Network Control Protocol*, Protocolo de Control de Red) eran insuficientes, y el TCP/IP fue un experimento para interconectar dos «redes conmutadas de paquetes» diferentes: la terrestre de Arpanet y una de ondas de radio llamada Packet Radio⁴⁸. El problema al que se enfrentaban los diseñadores no era cómo satisfacer a todo el mundo, sino meramente cómo solucionar un problema específico: la interconexión de dos redes técnicamente distintas, cada una con sus límites administrativos autónomos, sin forzar a ninguna de ellas a abandonar su sistema ni su autonomía.

Hasta mediados de los 80, los protocolos TCP/IP estaban orientados decididamente a la investigación y no eran objeto de un interés comercial dominante. Su desarrollo reflejaba un núcleo esencial de objetivos compartidos por los investigadores y promovidos en última instancia por la agencia central de financiación, el Departamento de Defensa. Es habitual referirse al conjunto TCP/IP como los protocolos que posibilitan las redes conmutadas de paquetes, pero esto es correcto solo en parte: su verdadera innovación estribaba en el diseño de una «*inter-network*» («*inter-red*»), un sistema que interconectaría varias redes diversas y autónomas (fueran conmutadas de paquetes o de circuitos), sin requerir su transformación, rediseño o estandarización —en suma,

47. Abbate, *Inventing the Internet*; Salus, *Casting the Net*; Galloway, *Protocol*; y Brock, *The Second Information Revolution*. Para historias de los participantes, véase Kahn *et al.*, «The Evolution of the Internet as a Global Information System»; Clark, «The Design Philosophy of the DARPA Internet Protocols».

48. Kahn *et al.*, «The Evolution of the Internet as a Global Information System», pp. 134-140; Abbate, *Inventing the Internet*, pp. 114-136.

requiriendo solo la estandarización de la intercomunicación entre redes, no de las redes en sí. En el primer artículo que describía el protocolo, Robert Kahn y Vint Cerf justificaron la necesidad del TCP/IP de la siguiente manera:

Aunque hay que resolver muchos problemas diferentes y complejos en el diseño de una red individual commutada de paquetes, estos problemas se acrecientan manifiestamente cuando se interconectan redes distintas. Aquí surgen cuestiones que pueden no tener un equivalente directo en una red individual y que influyen claramente en la manera en la que puede tener lugar la comunicación entre redes.⁴⁹

La meta explícita del TCP/IP era, por tanto, la de compartir recursos informáticos, no necesariamente la de conectar a dos individuos o a dos compañías, o la de crear un mercado competitivo de redes o de software de red. La compartición entre tipos diferentes de redes implicaba permitir que cada una de ellas se desarrollase de manera autónoma (como mejor vieran sus creadores y mantenedores), pero sin sacrificar la capacidad de continuar compartiendo. Años más tarde, David Clark, ingeniero jefe de Internet durante varios años de la década de los 80, dio una explicación mucho más explícita de los objetivos que llevaron al TCP/IP. En particular, sugería que la principal meta integral no era simplemente compartir recursos sino «desarrollar una técnica efectiva para la utilización multiplexada de las redes interconectadas existentes», y declaraba de manera más explícita el problema del control al que se enfrentaban los diseñadores:

Las redes representan límites administrativos de control, y una ambición de este proyecto era lidiar con el problema de integrar diversas entidades separadamente administradas en una herramienta común.⁵⁰

Al situar en primer lugar el objetivo de la expansibilidad, el TCP/IP se diseñó con un tipo específico de simplicidad en mente: la prueba del éxito de los protocolos era simplemente su capacidad de conectar.

49. Kahn y Cerf, «A Protocol for Packet Network Intercommunication», p. 637.

50. Clark, «The Design Philosophy of the DARPA Internet Protocols», pp. 54-55.

Al marcarse metas diferentes, pues, OSI y TCP/IP diferían no solo en sus detalles técnicos sino también en lo tocante a su contexto y legitimidad, siendo el primero producto de las instituciones internacionales de normalización y el otro de experimentos de investigación con financiación militar. Las diferencias técnicas y organizativas implican diferentes *procesos* de estandarización, y es la peculiar naturaleza del proceso llamado RFC la que le confirió a TCP/IP una de sus características más distintivas. El sistema RFC es ampliamente reconocido como un resultado único y fortuito del proceso de investigación de Arpanet⁵¹. En una retrospectiva sobre los treinta años del sistema (publicado, naturalmente, como un RFC, el RFC 2555), Vint Cerf afirma: «Escondida en la historia de los RFC está la historia de las instituciones humanas para alcanzar el trabajo cooperativo». Luego prosigue describiendo su evolución a lo largo de los años:

Cuando los RFC se produjeron por primera vez, tenían un carácter casi decimonónico —un intercambio epistolar para debatir públicamente los méritos de diversas opciones de diseño para los protocolos de ARPANET. A medida que el correo electrónico y los tablones de anuncios surgieron del fértil tejido de la red, los lejanos participantes de este histórico diálogo empezaron a hacer un uso cada vez mayor de los medios en línea para desarrollar la discusión —reduciendo la necesidad de documentar el debate en los RFC y, en algunos aspectos, dejando a los historiadores un tanto empobrecidos en el proceso. Lentamente, los RFC se convirtieron en conclusiones más que en debates.⁵²

Asimismo, los RFC se convirtieron crecientemente en un sistema de debate *e implementación* en el que los participantes creaban software operativo como parte de un experimento de desarrollo del estándar, tras lo cual se daba más debate, luego quizás más implementación y, finalmente, un estándar. El proceso de los RFC era una manera de condensar el proceso de estandarización y de validación mediante implementación; es decir, la prueba de los sistemas abiertos consistía en la conexión exitosa de redes diversas, y la creación de un estándar se

51. Los RFC se archivan en muchas ubicaciones, pero el sitio oficial es RFC Editor, <http://www.rfc-editor.org/>.

52. RFC Editor, RFC 2555, p. 6.

volvió una suerte de certificación *de facto ex post facto* de tal demostración. Cualquier mejora ulterior del estándar dependía de que mejorase la implementación normalizada puesto que los estándares resultantes estaban libre y ampliamente disponibles:

Cualquier usuario podía solicitar un RFC por correo electrónico desde su servidor y se le enviaba automáticamente a su buzón de correo. [...] Los RFC también se compartían libremente con organismos oficiales de estandarización, fabricantes y proveedores, otros grupos de trabajo y universidades. Ningún RFC fue jamás restringido o reservado como confidencial. Esto no era moco de pavo si tomamos en consideración que los estaba financiando el Departamento de Defensa en pleno apogeo de la Guerra Fría.⁵³

Los protocolos OSI no estaban ni por asomo tan libremente disponibles. Esta irónica inversión —la transparencia de un programa de investigación militar frente a la opacidad de una organización internacional de estandarización con sede en Ginebra— contribuye en gran medida a explicar las razones por las que a los *geeks* les resulta tan atractiva la historia del éxito de los protocolos TCP/IP. No es que los *geeks* sean secretamente militaristas, sino que se deleitan con tales inversiones sorprendentes, especialmente cuando ilustran el tipo de solución *ad hoc* e ingeniosa a problemas de coordinación que realiza el proceso de los RFC. Dicho proceso no es la única alternativa al modelo de estandarización orientado al consenso promovido por las organizaciones internacionales de Ginebra, pero supone una respuesta específica a una reorientación del poder y el saber que quizá fuera más «intuitivamente obvia» para los creadores de Arpanet e Internet, con sus inusuales metas de diseño y contexto, de lo que habría sido para los proveedores de sistemas de telecomunicaciones con más de cien años de experiencia en conectar a la gente de maneras muy específicas y establecidas.

El éxito como fracaso

Hacia 1985 OSI era ya un estándar oficial, uno con una amplia aceptación entre ingenieros, entre entidades gubernamentales y militares

53. *Ibid.*, p. 11.

(con el estándar «GOSIP»⁵⁴) y entre diversos fabricantes, siendo el más importante de ellos General Motors, con su MAP (*Manufacturing Automation Protocol*, Protocolo de Automatización de Fabricación). En los libros de texto y manuales de finales de los 80 y principios de los 90, era rutinario referirse a OSI como el estándar inevitable —lo que viene a decir que tenía amplia legitimidad como el estándar que todo el mundo *debería* implementar—, pero existían pocas implementaciones del mismo. Muchos de los libros de texto sobre redes de fines de los 80, especialmente aquellos con un sesgo hacia la introducción teórica, ofrecen explicaciones detalladas del modelo de referencia OSI—sin duda una generación de estudiantes fue adiestrada para entender el mundo desde el punto de vista de OSI—, pero la ambivalencia continuó. Es más, el legado más persistente de la creación de los protocolos OSI no son los protocolos en sí (algunos de los cuales, como ASN.1, aún son de uso generalizado hoy) sino su modelo pedagógico de «la pila de 7 capas», tan ubicuo en las clases y libros de texto sobre redes como lo es UNIX en las clases sobre sistemas operativos.⁵⁵

A finales de los 80, sin embargo, la ambivalencia se tornó en confusión. Mientras OSI era ampliamente reconocido como el estándar oficial, TCP/IP empezó a asomar en cada vez más sistemas realmente existentes. Por ejemplo, Carl Sunshine afirma en su obra *Computer Network Architectures and Protocols*: «Llegado el final de la década de los 80, gran parte de la batalla parece terminada. El CCITT y la ISO han alineado sus esfuerzos y la comunidad de investigadores parece haberse rendido en gran medida a OSI». Pero inmediatamente añade:

Es irónico que al tiempo que se ha desarrollado el consenso de que OSI es efectivamente inevitable, el conjunto de protocolos TCP/IP haya alcanzado un despliegue generalizado y ahora opere como un

54. El acrónimo GOSIP (*Government Open Systems Interconnection Profile*, Perfil Gubernamental de Interconexión de Sistemas Abiertos) contiene un juego de palabras intraducible que combina a su vez el acrónimo del modelo de referencia OSI y el adjetivo inglés «*gossip*» («cotilla»). Para mayor complejidad, en inglés existe el término «*gossip protocol*» («protocolo cotilla») para aludir a protocolos que emulan la forma de comunicación de los rumores con el fin de dotar de mayores velocidad y redundancia a determinados procesos en redes muy grandes o con estructuras complejas. [N. del E.]

55. Esto puede verse claramente, por ejemplo, comparando las diversas ediciones de los principales libros de texto sobre redes informáticas: cf. Tanenbaum, *Computer Networks* (1^a ed., 1981), (2^a ed., 1988), (3^a ed., 1996), (4^a ed., 2003); Stallings, *Data and Computer Communications*, (1^a ed., 1985), (2^a ed., 1991), (3^a ed., 1994), (4^a ed., 1997), (5^a ed., 2004); y Comer, *Internetworking with TCP/IP* (cuatro ediciones entre 1991 y 1999).

estándar de interoperabilidad *de facto*. [...] Parece que los proveedores fueron incapaces de llevar productos OSI al mercado lo bastante rápido como para satisfacer la demanda de sistemas interoperables, y el TCP/IP estaba ahí para cubrir esa necesidad.⁵⁶

Cuántas más implementaciones aparecían, menos seguro parecía ser el estándar legítimo. Según muchos testimonios, las especificaciones de OSI eran difíciles de implementar, y los congresos anuales «Interop» de la industria de redes se volvieron escenario habitual de la guerra religiosa entre TCP/IP y OSI. El éxito del primero sobre el segundo refleja la reorientación del saber y el poder a la que el software libre también responde. Las razones de dicho éxito son sin duda complejas, pero la trascendencia del éxito del TCP/IP ilustra tres aspectos: disponibilidad, modificabilidad y serendipia.

Disponibilidad

Los propios estándares TCP/IP eran libres y estaban disponibles para cualquiera en las redes TCP/IP, lo que ejemplifica uno de los aspectos de un público recursivo: que la única prueba para participar en una interred basada en TCP/IP consiste en poseer o haber creado un dispositivo que implemente el protocolo. El acceso a las redes está supeditado a la interoperabilidad de las mismas. Los estándares no eran «publicados» en un sentido convencional, sino que se ponían a libre disposición a través de la propia red sin ninguna restricción explícita de propiedad intelectual y sin tarifas ni limitaciones sobre quién podía acceder a ellos. Por el contrario, los estándares ISO por lo general no circulan libremente, sino que se venden a precios relativamente altos, como una fuente de ingresos y según la teoría general de que solo las corporaciones y agencias gubernamentales legítimas necesitarían acceder a ellos.

Igualmente relacionado con el asunto de la disponibilidad está el hecho de que el *proceso* de estandarización que regía el TCP/IP estaba a su vez abierto a todo el mundo, ya fuese del ámbito corporativo, militar o académico. La estructura de gobierno del IETF (*Internet Engineering Task Force*, Grupo Especial de Ingeniería de Internet) y de la ISOC (*Internet Society*, Sociedad de Internet) permitía a cualquiera que dispusiese de los medios asistir a las reuniones del «grupo de trabajo» que decidiría qué estándares aprobar. Ciertamente ello no significa que

56. Sunshine, *Computer Network Architectures and Protocols*, p. 5.

los ingenieros y los responsables de los contratistas de defensa buscasen activamente a actores corporativos o que concibiesen el sistema como «público» de cualquier manera drástica; sin embargo, en comparación con el sistema vigente en la mayoría de organismos de estandarización (donde habitualmente se requiere que los miembros sean representantes de corporaciones o gobiernos), el IETF permitía que los individuos participasen en cuanto individuos.⁵⁷

Modificabilidad

Las implementaciones del TCP/IP estaban ampliamente disponibles, fruto del arranque autosostenido de máquina a máquina junto con el sistema operativo UNIX y otras herramientas (como por ejemplo la implementación del TCP/IP en BSD 4.2, la versión BSD de UNIX), e incluyendo generalmente el código fuente. Una implementación existente es un objeto mucho más expresivo y usable que una especificación, y aunque la ISO generalmente prepara implementaciones de referencia para tales estándares, en el caso del OSI había muchas menos implementaciones sobre las que trabajar o construir. Dado que ya existían múltiples implementaciones del TCP/IP, era fácil validarla: ¿funcionó tu implementación (modificada) con las otras implementaciones existentes? Por contra, el OSI proporcionaría una validación independiente, pero la validación *in situ* mediante la conexión con otras redes era mucho más complicada de alcanzar, al existir tan pocas redes OSI o ser de acceso restringido. Es mucho más fácil construir sobre una implementación existente y mejorarlala gradualmente —o incluso reescribirla por completo, utilizando sus defectos como plantilla (por decirlo de alguna manera)— que crear una implementación basada exclusivamente en un estándar. La presencia de los protocolos TCP/IP en BSD 4.2 no solo implicaba que la gente que instalase ese sistema operativo podría conectarse fácilmente a Internet, en un momento en el que tal cosa no era ni mucho menos normal, sino también que los fabricantes o los aficionados a cacharrear podrían examinar el BSD 4.2 como base para una implementación modificada o completamente nueva.

57. La estructura del IETF, de la IAB (*Internet Architecture Board*, Junta de Arquitectura de Internet) y de la ISOC se detalla en Comer, *Internetworking with TCP/IP*, pp. 8-13; también en Schmidt y Werle, *Coordinating Technology*, pp. 53-58.

Serendipia

Quizá lo más relevante de todo fue la aparición de aplicaciones muy difundidas y populares que eran dependientes del TCP/IP, lo que dio a estos protocolos una inercia de la que carecía el OSI, que contaba con relativamente pocas aplicaciones de este tipo. La más importante de aquellas, con diferencia, fue la *World Wide Web* (el protocolo http, el lenguaje de marcado HTML y sus implementaciones tanto en servidores, como libwww, como en clientes, como Mosaic y Netscape). Los componentes básicos de la Web se crearon para funcionar sobre las redes TCP/IP, al igual que otros servicios que ya se habían diseñado (ftp, telnet, gopher, archie, etc.). De este modo, Tim Berners-Lee, coinventor de la *World Wide Web*, también pudo basarse en la disponibilidad y la apertura del trabajo anterior para sus propios protocolos. Además, Berners-Lee y el CERN (*Conseil Européen pour la Recherche Nucléaire*, Consejo Europeo de Investigación Nuclear) liberaron su obra al dominio público de modo más o menos inmediato, permitiendo esencialmente que cualquiera hiciera lo que le apeteciese con el sistema que habían confeccionado⁵⁸. Desde la perspectiva de la tensión entre TCP/IP y OSI, la *World Wide Web* fue, por tanto, lo que los ingenieros llaman una «*killer app*» («aplicación revolucionaria»), dado que su existencia realmente llevó a individuos y corporaciones a tomar decisiones (a favor del TCP/IP) que puede que no hubieran tomado de otra manera.

Conclusión

La apertura y los sistemas abiertos son clave para comprender las prácticas del software libre: las batallas de los 80 establecieron el contexto para el software libre, dejando tras de sí una infraestructura parcialmente articulada de sistemas operativos, redes y mercados como resultado de la figuración de los sistemas abiertos. El fracaso en la creación de un sistema operativo UNIX estándar abrió la puerta al Windows NT de Microsoft, pero también preparó el escenario para el surgimiento y difusión del núcleo del sistema operativo Linux. El éxito de los protocolos TCP/IP forzó a muchos proyectos de red rivales a converger en un único estándar —y en una sola entidad, Internet— que llevaba

58. Identificador de mensaje: 673c43e160cia758@sluvca.slu.edu. Véase también Berners-Lee, *Weaving the Web*.

integrado en su seno un conjunto de objetivos que reflejan el orden técnico y moral del software libre.

Esta «infraestructura» es a la vez técnica (protocolos y estándares e implementaciones) y moral (expresando ideas acerca del orden y la organización adecuados para que las iniciativas comerciales provean potencia de cálculo, software y redes de alta tecnología). Al igual que con la invención de UNIX, la oposición comercial-no comercial (o sus sosias⁵⁹ público-privado, lucrativo-no lucrativo, capitalista-socialista, etc.) no capta el contexto. Las restricciones sobre la capacidad de colaborar, competir o retirarse *se dan sobre la marcha* aquí a través de las imaginaciones técnicas y morales de los actores implicados: desde los monstruos corporativos como IBM a las (en su momento) empresas emergentes como Sun, pasando por los académicos, *amateurs* y *geeks* independientes con intereses en el nuevo mundo de alta tecnología de las redes y el software.

La creación de un mercado de UNIX fracasó, como también lo hizo la creación de un estándar de red legítimo a escala internacional. Pero fueron tan solo fracasos locales que abrieron las puertas a nuevas formas de práctica comercial (ilustradas por Netscape y el *boom* puntocom) y a nuevos tipos de desavenencias político-técnicas (ICANN, IPv6 y la «neutralidad de la red»). Pero el punto ciego de los sistemas abiertos —la propiedad intelectual— que estaba en el núcleo de estos fracasos proporcionó también el ímpetu para que algunos *geeks*, empresarios y abogados empezasen a figurar los aspectos legales y económicos del software libre, dando así inicio a una vibrante experimentación con las licencias de *copyright* y con formas innovadoras de coordinación y colaboración basadas en los protocolos en rápida expansión de Internet.

59. Rescatamos el término «sosias», procedente de la comedia de Plauto *Anfitrión*, para traducir el original «*doppelgänger*» (en alemán literalmente, «que camina al lado», con el sentido de *doble* de una persona). [N. del E.]

VI. REDACCIÓN DE LICENCIAS DE *COPYRIGHT*

Para proteger sus derechos necesitamos algunas restricciones que prohíban a cualquiera negarle a usted estos derechos o pedirle que renuncie a ellos.

Preámbulo a la Licencia Pública General de GNU¹

El uso de licencias de *copyright* novedosas y no convencionales es, sin duda, el componente más ampliamente reconocido y exquisitamente refinado del software libre. La GNU GPL (*General Public License*, Licencia Pública General), redactada inicialmente por Richard Stallman, es descrita a menudo como un hermoso, inteligente y poderoso «*hack*» («golpe») a la legislación de propiedad intelectual —cuando no se la denuncia como un objeto viral e infeccioso que amenaza al entramado mismo de la economía y de la sociedad. El hecho mismo de que algo tan aburrido, tan arcano y tan legalista como una licencia de *copyright* pueda convertirse en objeto tanto de devoción reverencial como de bilioso desprecio significa que hay mucho más en juego que la mera letra pequeña.

A comienzos del siglo XXI, había cientos de licencias de software libre diferentes, cada una con sutiles diferencias legales y técnicas, así como una enorme literatura jurídica para explicar sus detalles, motivaciones e impacto.² Las licencias de software libre difieren de las licencias de *copyright* convencionales aplicadas a programas informáticos porque normalmente restringen solo las condiciones de distribución, mientras que los denominados EULA (*End-User License Agreement*, Acuerdos de Licencia de Usuario Final) que acompañan a la mayoría del software privativo restringen lo que los usuarios pueden *hacer* con el software.

1. Para todos los fragmentos de la GPL citados por Chris Kelty usamos la pionera traducción (no oficial) al castellano realizada por Jesús González-Barahona y Pedro de las Heras Quirós, disponible en: <http://gugs.sindominio.net/licencias/gples.html> [N. del E.]

2. La literatura jurídica sobre software libre se expande de forma constante y rápida y aborda una variedad de asuntos legales diferentes. Dos excelentes puntos de partida son Vetter, «The Collaborative Integrity of Open-Source Software» e «'Infectious' Open Source Software».

Etnográficamente hablando, las licencias aparecen por doquier en este campo y los *hackers* contemporáneos se cuentan entre los no-abogados más jurídicamente sofisticados del mundo. De hecho, el aprendizaje en el mundo del *hacking* es actualmente imposible, como ha mostrado Gabriella Coleman, sin un largo y profundo estudio de la legislación de propiedad intelectual.³

Pero, ¿cómo se llegó a esto? Como en el ejemplo de la compartición del código fuente de UNIX, las licencias de software libre suelen explicarse como reacción a la expansión de la legislación de propiedad intelectual y como resistencia a la rapacidad corporativa. El propio texto de la GPL parte de estas suposiciones: «Las licencias que cubren la mayor parte del software están diseñadas para quitarle a usted la libertad de compartirlo y cambiarlo».⁴ Pero a pesar de la rapacidad corporativa, la compartición y modificación del software no son en modo alguno actividades humanas naturales. Antes de defenderlas, las ideas de compartición y de propiedad común, así como su relación con la libertad deben siempre ser producidas mediante prácticas específicas de compartición. La GPL es un ejemplo preciso de cómo los *geeks* encajan las prácticas de compartición y modificación del software con los órdenes morales y técnicos —los imaginarios sociales— de la libertad y la autonomía. Se trata a la vez de un documento legal exquisitamente preciso y de la expresión de una idea de cómo el software debería hacerse disponible, compatible y modificable.

En este capítulo narro la historia de la creación de la GPL, la primera licencia de software libre, durante una controversia acerca de EMACS, un editor de texto ampliamente usado y respetado. Dicha controversia concernía a la reutilización de fragmentos de código fuente sujeto a *copyright* en una versión de EMACS portada a UNIX. Hay dos razones para volver a contar esta historia detenidamente. La primera es simplemente para articular los detalles del origen de la propia GPL como un componente central del software libre, detalles que deben entenderse en el contexto de la cambiante legislación de *copyright* y de las disputas sobre UNIX y los sistemas abiertos de la década de los 80. La segunda es que, por más habitual que sea considerar la historia de la GPL una historia de «ética *hacker*», la GPL no es una «expresión» de

3. Coleman, «The Social Construction of Freedom».

4. «The GNU General Public Licence, Version 2.0»: <http://www.gnu.org/licenses/old-licenses/gpl-2.0.html>.

dicha ética, como si la ética fuera el genotipo de un fenotipo legal. En contraste con la historia familiar de la ética *hacker*, explico cómo la GPL fue «figurada» en medio de la controversia sobre EMACS, cómo se conformó en respuesta a un complicado estado de cosas, tanto legal como técnico, y en un medio nuevo para todos los participantes: las listas de distribución de correo electrónico y las listas de discusión de Usenet y Arpanet.⁵

La historia de la creación de la GPL afirma en última instancia la ética *hacker*, no como un relato de la genialidad de dicha ética, sino como un acontecimiento históricamente específico con una duración y un contexto, como algo que surge en respuesta a la reorientación del saber y el poder y a través de la modulación activa de prácticas existentes entre actores tanto humanos como no humanos. Por más que los propios *hackers* puedan interpretar la ética *hacker* como un conjunto inmutable de normas morales, sus prácticas desmienten dicha creencia y demuestran cómo ética y normas pueden surgir súbita y bruscamente, sufrir repetidas transformaciones y bifurcarse en campos ideológicamente distintos (software libre vs. código abierto), aun cuando las prácticas permanezcan estables en relación con ellas. La ética *hacker* no desciende de las cumbres de la filosofía como el imperativo categórico —los *hackers* no tienen un Kant, ni tampoco lo quieren. Más bien, como ha sugerido Manuel Delanda, la filosofía del software libre es el hecho mismo del software libre, sus prácticas y sus *objetos*. Si existe una ética *hacker*, esta es el software libre en sí, el público recursivo en sí,

5. Todos los relatos existentes sobre la ética *hacker* proceden de dos fuentes: del propio Stallman y del vívido y atractivo capítulo sobre Stallman en la obra de Steven Levy *Hackers*. Ambas fuentes reconocen una prehistoria de tal ética. Levy la retrotrae al MIT Tech Model Railroad Club de la década de los 50, mientras que Stallman es más proclive a describirla como radicada en la revolución científica o incluso antes. Los relatos de los primeros *hackers* del MIT son declaradamente edénicos, y en ellos los *hackers* viven en un mundo de incontestada libertad y competencia académica —algo parecido a una comuna de escritores sin el alcohol ni las trifulcas. Existen historias sobre una impresora cuyo software necesitaba ser reparado pero solo estaba disponible bajo un acuerdo de confidencialidad; sobre una exigencia de usar contraseñas (que Stallman rechazó, eligiendo <return> como su contraseña y *hacceando* el sistema para animar al resto a hacer lo mismo); sobre una guerra de programación entre diferentes máquinas LISP; y sobre la sustitución del Incompatible Time-Sharing System con el sistema operativo de DEC TOPS-20 («Twenex»). Estas historias constituyen pasados usables repetidos a menudo, pero no son representativos. Las restricciones comerciales siempre han formado parte de la vida académica en informática e ingeniería: tanto el hardware como el software se adquirían necesariamente de fabricantes comerciales que solían mantener el control sobre ellos, por más que ofrecieran licencias «académicas» o «educativas».

lo cual va mucho más allá de una lista de normas.⁶ Al interpretarla de esta manera, se vuelve posible seguir la proliferación y diversificación de la ética *hacker* hacia nuevos y sorprendentes dominios, en vez de asumir su persistencia estática universal como un mero procedimiento que los *hackers* ejecutan.

Las licencias de software libre, una vez más con sentimiento

En una de sus conferencias sobre liberalismo de 1935, John Dewey afirmaba lo siguiente de Jeremy Bentham:

Fue, podríamos decir, el primer gran escarbador⁷ del campo del Derecho. [...] pero fue más que eso, siempre que veía un defecto, proponía un remedio. Fue un inventor en el campo del Derecho y de la administración, a la misma altura que cualquier inventor contemporáneo en el campo de la producción mecánica.⁸

El argumento de Dewey era que las reformas liberales atribuidas a Bentham no provinieron tanto de sus teorías cuanto de su implicación directa en la reforma administrativa y legal —de su experimentación. Al margen de si esta es o no la mejor interpretación posible de la influencia

6. Delanda, «Open Source».

7. El término original que emplea Dewey es «*muckraker*» (literalmente «removedor de estiércols»), surgido a comienzos del siglo XX en EEUU para (des)calificar a un grupo de periodistas de investigación progresistas (entre otros, Upton Sinclair, Ida Tarbell, Ray Stannard Baker, Lincoln Steffens o incluso el propio Joseph Pulitzer) que denunciaban con notable rotundidad y efectividad los desmanes de Wall Street y de las emergentes multinacionales, la corrupción del gobierno de EEUU, la explotación laboral o la miseria popular. El origen de este apelativo (que pronto se tornaría una suerte de elogio) se atribuye al presidente estadounidense Theodore Roosevelt, quien el 14 de abril de 1906 difundió el discurso «*The Man with the Muck Rake*», que equiparaba a dichos periodistas con «el hombre con el rastrillo de estiércol», en alusión a un personaje de *The Pilgrim's Progress* (1678), de John Bunyan, que renunciaba a la salvación divina por su empeño en escavar en la inmundicia. No me resisto a apostillar que la plena actualidad del «*muckraking*» viene avalada por las históricas filtraciones de Wikileaks, Julian Assange, Edward Snowden o Chelsea Manning (libre al fin), así como por la feroz represión a las mismas en forma de acusación (e incluso condena) a estos contemporáneos «alertadores» («*whistleblowers*») por el cargo de «colaboración con el enemigo»... en aplicación de la Ley de Espionaje aprobada en 1917 al poco de entrar EEUU en la I Guerra Mundial. A este respecto, una lectura imprescindible es Yochai Benkler, «A Free Irresponsible Press: Wikileaks and the Battle Over the Soul of the Networked Fourth Estate», *Harvard Civil Rights-Civil Liberties Law Review*, 46, 2 (2011): 311-397, disponible en: <https://dash.harvard.edu/bitstream/handle/1/10900863/benkler.pdf?sequence=1> [N. del E.]

8. Dewey, *Liberalism and Social Action*. [ed. cast.: *Liberalismo y Acción Social y otros ensayos*]

de Bentham, permite no obstante captar un componente importante de la reforma liberal de Europa y EEUU que también resulta clave en la historia del software libre: que la ruta para lograr el cambio pasa por la experimentación directa con el sistema jurídico y administrativo.

Una historia similar podría contarse de Richard Stallman, héroe *hacker* y fundador de la Fundación para el Software Libre, creador (entre otras muchas cosas) del compilador GNU C y de GNU EMACS, dos de las herramientas de software libre más extensamente usadas y probadas del mundo. Stallman es rutinariamente vilipendiado por sostener lo que muchos perciben como posiciones ideológicas «dogmáticas» o «intratables» sobre la libertad y el derecho de los individuos a hacer lo que deseen con el software. Si bien es indudablemente cierto que sus discursos y escritos claramente revelan un cierto fervor y fanatismo, sería un error suponer que sus discursos, ideas o demandas beligerantes respecto a la elección de palabras constituyen la verdadera esencia de su reforma. De hecho, son el software que ha creado y las licencias que ha escrito y reescrito los que representan la clave de su inventiva al estilo de Bentham. A diferencia de Bentham, sin embargo, Stallman no es un creador de estructuras legales y administrativas, sino un *hacker*.

La GPL de Stallman «*hackea*» la legislación federal de *copyright*, como suele señalarse. Y lo hace aprovechando los muy estrictos derechos otorgados por dicha norma para en realidad *desanudar* las restricciones normalmente asociadas a la propiedad. Así, ya que las leyes otorgan a los propietarios amplios poderes para crear restricciones, la GPL de Stallman contiene la restricción de que cualquiera puede utilizar el material licenciado, para cualquier propósito, siempre y cuando se ofrezca subsigüientemente la misma restricción. Los *hacks* (de los que toman el nombre los *hackers*) son, pues, soluciones ingeniosas a problemas o deficiencias en materia de tecnología; son rodeos, atajos ingeniosos que sacan provecho de características del sistema que pueden o no haber sido obvias para sus diseñadores. Los *hacks* abarcan desde lo puramente utilitario hasta lo juguetonamente absurdo, pero siempre dependen de un sistema o de un instrumento existente a través del cual alcanzan su objetivo. Llamar *hack* al software libre supone señalar que no sería nada sin la existencia de la legislación sobre propiedad intelectual, pues se basa en la estructura de la legislación de *copyright* estadounidense (USC § 17) con el fin de subvertirla. Las licencias de software libre son, en cierto sentido, inmanentes a las leyes de *copyright* —no hay nada ilegal ni aun legalmente arcano en lo que llevan a cabo—, pero sin embargo

existe una especie de persistente sensación de que este uso particular del *copyright* no era para lo que estaba pensada la ley.

Como todo programa informático desde las enmiendas de 1980 a la legislación de *copyright*, al software libre le es aplicable dicho *copyright* —es más, le es aplicable automáticamente por el hecho de escribirlo (ya no existe ningún requisito de registro). La legislación de *copyright* concede a los autores (o a sus empleadores) una serie de derechos estrictos sobre la dispensación de lo escrito: los derechos de copia, distribución y modificación de la obra.⁹ El *hack* del software libre consiste en hacer inmediatamente uso de ellos con el fin de abrogar los derechos concedidos a los programadores, otorgando así a todos los ulteriores licenciatarios los derechos de copia, distribución, modificación y uso del software bajo *copyright*. Algunas licencias, como la GPL, agregan la restricción adicional de que los licenciatarios deben ofrecer las mismas condiciones a cualesquiera ulteriores licenciatarios; otras no ponen tal restricción sobre los usos ulteriores. Así pues, si bien la legislación sugiere que los individuos necesitan estrictos derechos de propiedad y se los otorga, las licencias de software libre los anulan a todos los efectos en favor de otras actividades, como las de compartir, portar y bifurcar el software. Es por esta razón por la que se han ganado el nombre de «copyleft».¹⁰

Esta es, sin embargo, una conveniente descripción *a posteriori*. Ni Stallman ni ninguna otra persona partió con la intención de hackear la legislación de *copyright*. El *hack* de las licencias de software libre nació como respuesta a una complicada controversia acerca de un invento muy importante, una herramienta que a su vez permitió una invención llamada EMACS. La historia de la controversia es bien conocida entre los *hackers* y los *geeks*, pero no suele ser contada, y menos aún en detalle, fuera de estos reducidos círculos.¹¹

9. *Copyright Act of 1976*, Pub. L. No. 94–553, 90 Stat. 2541, aprobada el 19 de octubre de 1976; y *Copyright Amendments*, Pub. L. No. 96–517, 94 Stat. 3015, 3028 (enmendando las secciones §101 y §117, título 17, *United States Code*, relativas a los programas informáticos), aprobadas el 12 de diciembre de 1980. Todas las enmiendas aprobadas desde 1976 aparecen listadas en <http://www.copyright.gov/title17/92preface.html>.

10. La historia del *copyright* y del software es examinada en Litman, *Digital Copyright*; Cohen et al., *Copyright in a Global Information Economy*; y Merges, Menell y Lemley, *Intellectual Property in the New Technological Age*.

11. Véase Wayner, *Free for All*; Moody, *Rebel Code*; y Williams, *Free as in Freedom*. Aunque esta historia podría relatarse simplemente entrevistando a Stallman y a James Gosling, por seguir ambos vivos y activos en el mundo del software, he elegido contarla mediante un análisis detallado de los archivos de Usenet y Arpanet relativos a la controversia. Se trata de

EMACS, el editor de visualización en tiempo real extensible, personalizable y autodocumentado

EMACS es un editor de texto, y también algo parecido a una religión. Siendo uno de los dos editores de texto más famosos, es frecuentemente elogiado por sus fervientes usuarios y atacado por los detractores que prefieren a su competidor (el vi de Bill Joy, también creado a finales de los 70). EMACS es más que un mero instrumento de escritura de texto: para muchos programadores, era (y sigue siendo) la interfaz principal del sistema operativo. Por ejemplo, EMACS permite a los programadores no solo escribir un programa sino también depurarlo, compilarlo, ejecutarlo y enviarlo por correo electrónico a otro usuario, todo desde la misma interfaz. Más aún, permite a los usuarios escribir extensiones para EMACS de forma rápida y sencilla, extensiones que automatizan tareas frecuentes y a su vez se convierten en elementos esenciales del software. En suma, EMACS puede hacer casi cualquier cosa, pero también puede frustrar a casi cualquiera. El propio nombre está tomado de su muy admirada extensibilidad: EMACS es la abreviatura de «edición de macros» porque permite a los programadores grabar rápidamente una serie de comandos y agruparlos en una macro que puede ser invocada mediante una simple combinación de teclas. De hecho, fue uno de los primeros editores (si no el primero) en aprovechar teclas como ctrl y meta, como en el hoy ubicuo ctrl-S tan familiar a los usuarios de procesadores de texto no libres como Microsoft Word™.

Hemos de apreciar la innovación que representó EMACS: antes de la era del miniordenador dominada por UNIX, existían muy pocos programas para manipular texto directamente en pantalla. Concebir el código fuente como independiente de un programa ejecutado en una máquina significaba concebirlo primero como un código mecanografiado, impreso o garabateado que los programadores escrutaran en su forma más tangible, sobre el papel. Los editores que permitían a los programadores disponer del código en una pantalla frente a ellos, manipularlo directamente y guardar los cambios fueron una innovación de mediados y finales de los 60 que no se generalizaría hasta mediados

compensar entre una suerte de acceso en directo incompleto a un momento de la historia y las reelaboraciones probablemente revisionistas de aquellos que lo vivieron directamente. Todos los mensajes a los que me refiero se citan con su «Identificador de mensaje», lo cual debería permitir que cualquier persona interesada acceda a los mensajes originales a través de Google Groups (<http://groups.google.com>).

de los 70 (y ello solo entre los académicos más a la última y las empresas de informática). Junto con algunos pioneros como QED (creado originalmente por Butler Lampson y Peter Deutsch, y reescrito para UNIX por Ken Thompson), uno de los editores más famosos fue TECO (*text editor and corrector*, editor de texto y corrector), escrito por Dan Murphy para el ordenador PDP-1 de DEC entre 1962 y 1963. Con el paso de los años, TECO fue transformado (portado y extendido) para una amplia variedad de máquinas, incluidas las de Berkeley y el MIT, y para otros equipos y sistemas operativos de DEC. A principios de los 70, había una versión de TECO ejecutándose en el ITS (*Incompatible Time-sharing System*, Sistema de tiempo compartido incompatible), el sistema usado en el Laboratorio de Inteligencia Artificial (AI Lab) del MIT, la cual conformó la base de EMACS. (De ahí que el mismo EMACS se concibiera como una serie de macros para un editor distinto: *Editing MACroS for TECO*, Edición de MACroS para TECO).

Como en todos los proyectos ejecutados en el ITS del AI Lab, muchas personas contribuyeron a la extensión y mantenimiento de este editor (entre ellas, Guy Steele, Dave Moon, Richard Greenblatt y Charles Frankston), pero hay un claro reconocimiento de que Stallman hizo de EMACS lo que era. El primer memorándum del laboratorio sobre EMACS, de Eugene Ciccarelli, afirma:

Finalmente, de todas las personas que han contribuido al desarrollo de EMACS, y al del TECO que hay detrás de él, debemos una mención y agradecimiento especiales a Richard M. Stallman. No solo le dio a TECO la potencia y generalidad que tiene, sino que también recopiló las buenas ideas de muy diferentes paquetes de funciones de TECO, añadió una tremenda cantidad de ideas y entornos nuevos, y creó EMACS. Personalmente, uno de los mayores deleites de mi vida vocacional ha sido escribir funciones para Teco/EMACS; lo que hace que esto sea divertido y no penoso es el rico conjunto de herramientas con las que trabajar, casi todas las cuales tienen en alguna parte cinceladas las siglas «RMS».¹²

En este punto, en 1978, EMACS vivía en gran parte en el ITS, pero su reputación pronto se propagó, siendo portado al sistema operativo

12. Eugene Ciccarelli, «An Introduction to the EMACS Editor», MIT Artificial Intelligence Laboratory, *AI Lab Memo no. 447*, 1978, p. 2.

TOPS-20 (Twenex) de DEC y reescrito para Multics y para la máquina LISP del MIT, en la que se llamaba EINE (*Eine Is Not EMACS*, Eine No Es EMACS), tras el cual vino ZWEI (*Zwei Was Eine Initially*, Zwei Fue Eine Inicialmente).¹³

La proliferación de EMACS fue tan placentera como frustrante para Stallman, pues supuso que el trabajo se fragmentara en distintos proyectos, cada uno de ellos parecido a EMACS, en lugar de construir un único proyecto básico. Así, en un informe de 1981 Stallman afirmaba:

La proliferación de tales facsímiles superficiales de EMACS tiene un desafortunado efecto de confusión: se hace creer a sus usuarios, desconocedores de que están usando una imitación de EMACS sin haber visto nunca el EMACS en sí, que están disfrutando de todas las ventajas de EMACS. Dado que cualquier editor de texto en tiempo real supone una tremenda mejora con respecto a lo que probablemente tenían antes, los usuarios lo creen inmediatamente. Para evitar tal confusión, instamos a todo el mundo a que se refiera a una imitación no extensible de EMACS como un «*ersatz*¹⁴ EMACS».¹⁵

Así pues, aunque EMACS en su forma específica para el ITS era una creación de Stallman, la *idea* de EMACS o de cualquier «editor de texto de tiempo real» estaba proliferando de diferentes maneras y en diferentes máquinas. El porte de EMACS, al igual que el de UNIX, vino facilitado tanto por la integridad de su diseño conceptual como por su disponibilidad generalizada.

La frase «una imitación no extensible» capta la combinación de filosofía de diseño y filosofía moral que EMACS representaba. La extensibilidad no era meramente una característica útil para los usuarios individuales, sino también una manera de hacer que las mejoras de cada

13. Asumiendo los riesgos de explicar un chiste, pero con afán de explicitar aún más el carácter jocoso de estos acrónimos recursivos (nunca mejor dicho, en la medida en que van reconociendo a cada paso los hombros a los que se áupa un software para ver más lejos cada vez), apuntamos que en alemán «*eins*» significa «uno» (en este caso «*eine*» sería «una») al cual sigue el «*zwei*» («dos»). [N. del. E.]

14. En alemán en el original. «*Ersatz*» se traduciría aquí como «sucedáneo». [N. del. E.]

15. Richard Stallman, «EMACS: The Extensible, Customizable Self-documenting Display Editor», MIT Artificial Intelligence Laboratory, *AI Lab Memo no. 519a*, 26 de marzo de 1981, p. 19. También publicado como Richard M. Stallman, «EMACS: The Extensible, Customizable Self-documenting Display Editor», *Proceedings of the ACM SIGPLAN SIGOA Symposium on Text Manipulation*, 8-10 junio (ACM, 1981), pp. 147-156.

nuevo usuario estuvieran fácilmente disponibles del mismo modo para todos mediante la provisión a dichos usuarios de una manera estándar de añadir extensiones y aprender a usar las nuevas (la característica de «auto-documentación» del sistema). El programa tenía una integridad conceptual que se veía comprometida cuando se lo copiaba de modo imperfecto. EMACS tiene un diseño modular y extensible que por su propia naturaleza invita a los usuarios a contribuir a él, extenderlo y hacerle ejecutar todo tipo de tareas —una invitación a copiarlo y modificarlo literalmente, en lugar de solo imitarlo. Para Stallman, esto no constituía solo un diseño fantástico para un editor de texto sino una expresión de la forma en que siempre había hecho las cosas en el entorno de pequeña escala del AI Lab. La historia de las aspiraciones morales de Stallman enfatiza su resistencia al secretismo en la producción de software, y EMACS es, tanto en su diseño como en la distribución del mismo por parte de Stallman, un ejemplo de esta resistencia.

No todos compartían el sentido de orden comunal de Stallman, sin embargo. Con vistas a facilitar la extensión de EMACS mediante la compartición, Stallman comenzó lo que llamó la «comuna EMACS». Al final del informe de 1981 —«EMACS: The Extensible, Customizable Self-documenting Display Editor», datado el 26 de marzo) explicaba sus condiciones de distribución:

[EMACS] se distribuye sobre la base de la compartición comunitaria, lo que significa que todas las mejoras se me deben reenviar para su incorporación y distribución. Quienes estén interesados deberían contactar conmigo. De este modo también puede disponerse de más información sobre cómo funciona EMACS.¹⁶

En otro informe, concebido como un manual de usuario de EMACS, Stallman ofreció instrucciones más detalladas y ligeramente más pintorescas:

EMACS no cuesta nada, en lugar de eso, te unes a la comuna de compartición de software de EMACS. Las condiciones para ser miembros son el deber de reenviar cualquier mejora que hagas a EMACS, incluyendo cualesquiera bibliotecas que escribas, y el

16. Richard Stallman, «EMACS: The Extensible, Customizable Self-documenting Display Editor», MIT Artificial Intelligence Laboratory, *AI Lab Memo no. 519a*, 26 de marzo de 1981, p. 24.

deber de no redistribuir el sistema salvo exactamente como lo conseguiste, completo. (También puedes distribuir tus personalizaciones, *pero separadamente*). Por favor, no intentes conseguir una copia de EMACS, para ti mismo o para cualquier otra persona, descargándotela de tu sistema local. Casi seguro estará incompleta o será inconsistente. Es patético oír cómo páginas que recibieron copias incompletas desprovistas de las fuentes [el código fuente] me preguntan años después si las fuentes están disponibles. (Todas las fuentes se distribuyen y deberían estar en línea en cada página para que los usuarios puedan leerlas y copiar el código de ellas). Si deseas regalar una copia de EMACS, copia una cinta de distribución del MIT o mándame por correo una cinta y consigue una nueva.¹⁷

Dado que EMACS era tan ampliamente admirado y respetado, Stallman adquirió cierto poder sobre esta comuna. Si hubiera sido una herramienta obscura y no extensible, útil para un único propósito, nadie habría atendido a tales demandas, pero dado que EMACS constituía por naturaleza el tipo de herramienta que era tanto útil para todo tipo de tareas como personalizable para aquellas más específicas, Stallman no era la única persona que se beneficiaba de esta modalidad comunal. Es muy posible que dos sitios dispares hubieran necesitado la misma extensión de macros, y por tanto muchos de ellos podían ver fácilmente el beneficio social de reenviar las extensiones para su inclusión y de convertirse en una especie de co-desarrollador de un sistema tan potente. Como resultado, las peticiones de la comuna de EMACS, pese a ser inusuales y autocráticas, tenían un valor obvio para la multitud. En términos del concepto de público recursivo, EMACS era en sí la herramienta mediante la cual era posible que los usuarios extendieran EMACS, el medio de su afinidad; los usuarios tenían un incentivo natural para compartir sus contribuciones para que todos pudieran recibir el máximo beneficio.

Las condiciones del acuerdo de distribución de EMACS no eran del todo vinculantes legalmente; nada compelía a la participación excepto la reputación de Stallman, su autoridad intimidatoria o el deseo de reciprocidad de los usuarios. Por una parte, Stallman todavía no había optado por, o había sido forzado a, entender los detalles del sistema

17. Richard M. Stallman, «EMACS Manual for ITS Users», MIT Artificial Intelligence Laboratory, *AI Lab Memo no. 554*, 22 de octubre de 1981, p. 163.

legal, por lo que la comuna EMACS era la mejor alternativa. Por otro lado, el panorama de la legislación de propiedad intelectual era muy convulso por entonces y no estaba claro para nadie, ya fueran empresas o académicos, exactamente qué tipos de acuerdos legales serían legítimos (los cambios de la legislación de *copyright* de 1976 fueron de los más drásticos en setenta años, y una enmienda de 1980 hizo que el *copyright* fuera aplicable al software, todo ello sin que ningún proceso judicial hubiera dirimido tales cambios). El «acuerdo» de Stallman era un conjunto de normas informales que expresaban el sentido general de apoyo mutuo que caracterizaba tanto el diseño del sistema como la propia experiencia de Stallman en el AI Lab. Se trataba, pues, de una expresión de la manera en que Stallman esperaba que los otros se comportaran, y sus intentos de castigar o avergonzar a la gente equivalían a una aplicación informal de estas expectaciones. En este empeño, la pequeña escala de la comunidad jugó a favor de Stallman.

En su escala reducida, la comuna de Stallman estaba afrontando muchas de las mismas cuestiones que acuciaban los debates sobre sistemas abiertos que surgían por la misma época, cuestiones relativas a interoperabilidad, compartición de código fuente, estandarización, portabilidad y bifurcación. En especial, Stallman tenía una aguda conciencia del punto ciego de los sistemas abiertos: el conflicto de los órdenes morales y técnicos representados por la propiedad intelectual. Mientras que los proveedores de UNIX dejaron sin cuestionar las normas de propiedad intelectual y simplemente asumieron que eran las normas básicas del debate, Stallman las convirtió en la sustancia de su experimento y, como Bentham, se convirtió a resultas de ello en una suerte de escarbador legal.

El modelo comunal de Stallman no pudo impedir por completo el porte y la bifurcación del software. Pese a la petición de Stallman de que los imitadores se refirieran a sus versiones como un *ersatz* EMACS, pocos lo hicieron. En ausencia de amenazas legales por marca registrada, no se podía hacer gran cosa para impedir que la gente llamara EMACS a sus portes y bifurcaciones, un problema de éxito no muy diferente al de Kleenex o Xerox. Poca gente tomó las ideas principales del programa, las implementó en una imitación y después la llamó de alguna otra forma (EINE y ZWEI fueron excepciones). En el caso de UNIX, la proliferación de versiones bifurcadas del software no las hacía ser menos UNIX, incluso cuando AT&T insistía en la propiedad de la marca registrada. Pero con el paso del tiempo EMACS fue portado, bi-

furcado, reescrito, copiado o imitado en diferentes sistemas operativos y arquitecturas informáticas de universidades y empresas de todo el mundo, con el resultado de que en cinco o seis años muchas versiones de EMACS eran ampliamente usadas. Fue esta situación de adopción exitosa la que proporcionó el contexto para la controversia que se dio entre 1983 y 1985.

La Controversia

Brevemente, la controversia fue la siguiente: en 1983 James Gosling decidió vender su versión de EMACS —una versión para UNIX escrita en C llamada GOSMACS— a un proveedor de software comercial llamado Unipress. GOSMACS, la segunda implementación más famosa de EMACS (después de la del propio Stallman), fue escrita por Gosling cuando estaba matriculado en la Universidad de Carnegie Mellon. Durante años, la había distribuido por sí mismo y había administrado una lista de correo en Usenet donde respondía consultas y discutía las extensiones. Gosling había pedido explícitamente a la gente que no distribuyera el programa, sino que acudiese a él (o que enviase a los interesados directamente a él) para obtener versiones nuevas, lo que hacía de GOSMACS más una dictadura benevolente¹⁸ que una comuna. Gosling mantenía su autoridad, pero aceptaba con deferencia revisiones, depuraciones de errores y extensiones de parte de los usuarios, incorporándolas a las ediciones nuevas. El sistema de Stallman, por el contrario, permitía a los usuarios distribuir sus extensiones ellos mismos, además de incluirlas en el EMACS «oficial». Llegado 1983, Gosling había decidido que ya no podía seguir ocupándose de mantener y ofrecer asistencia técnica para GOSMACS —tarea que consideraba propia de una corporación.

Para Stallman, la decisión de Gosling de vender GOSMACS a Unipress fue un «sabotaje de software». Por más que Gosling hubiera tenido una responsabilidad sustancial en la creación de GOSMACS, Stallman se sentía en cierto modo propietario de ese *ersatz* —o, como mínimo, le molestaba que no existiera una versión no comercial de EMACS para UNIX. Así que el propio Stallman escribió una (como

18. En el ámbito del software libre y de código abierto, la popularización de las expresiones «dictadura benevolente» y «dictador benevolente» para caracterizar los modos de propiedad y toma de decisiones de algunos proyectos informáticos caracterizados por una suerte de liderazgo carismático suele atribuirse al artículo de Eric S. Raymond «Homesteading the Noosphere»: <http://firstmonday.org/article/view/621/542>. [N. del E.]

parte de un proyecto que anunció aproximadamente a la vez, llamado GNU [*GNU's Not Unix*, GNU No es Unix], para crear una versión completa de UNIX que no fuera de AT&T), la llamó GNU EMACS y la publicó bajo las mismas condiciones comunales de EMACS. El meollo del debate giró en torno al hecho de que Stallman empleó, aunque ostensiblemente con permiso, un pequeño fragmento del código de Gosling en su nueva versión, un hecho que llevó a mucha gente, incluidos los nuevos proveedores comerciales de EMACS, a poner el grito en el cielo. A continuación se sucedieron las recriminaciones y amenazas legales hasta que finalmente la controversia se resolvió cuando Stallman reescribió el código de la discordia, creando de esta manera una versión completamente «libre de Gosling» que terminó convirtiéndose en la versión estándar de EMACS para UNIX.

La historia suscita varias cuestiones con respecto al cambiante contexto legal del momento, en particular sobre la diferencia entre «la ley sobre el papel» y «la ley en acción»; es decir, la diferencia entre las acciones de *hackers* y entidades comerciales, asesorados por abogados y amigos con destrezas legales, y el texto y la interpretación de las leyes tal y como las redactan los legisladores y las interpretan los tribunales y abogados. Los asuntos legales abarcan el secreto comercial, las patentes y las marcas registradas, pero el *copyright* es especialmente significativo. En la época había tres cuestiones aún sin decidir: la aplicabilidad del *copyright* al software, la definición de qué cuenta como software y qué no, y el significado de las infracciones del *copyright*. Por más que la controversia no resolviera ninguno de estos asuntos (los dos primeros los solucionaría el Congreso y los tribunales; el tercero permanece un tanto nebuloso), sí que clarificó a Stallman las cuestiones legales lo bastante como para poder dejar atrás la comuna informal de EMACS y crear la primera versión de una licencia de software libre, la Licencia Pública General de GNU, cuya primera aparición data de 1985.

La decisión de Gosling de vender el programa, anunciada en abril de 1983, alimentó el creciente debate que se venía dando en la lista de distribución de GOSMACS, un grupo de Usenet llamado net.emacs. Como este grupo se reenviaba a Arpanet mediante una puerta de enlace mantenida por John Gilmore, de Sun Microsystems, una comunidad bastante grande de usuarios de EMACS estuvo al tanto del anuncio de Gosling. Antes de la declaración de este, había habido cierto debate sobre diferentes versiones de EMACS, incluyendo una ya «comercial» denominada CCA EMACS, escrita por Steve Zimmerman, de la *Computer*

Corporation of America (Corporación Informática de EEUU).¹⁹ Algunos lectores querían comparaciones entre CCA EMACS y GOSMACS, otros objetaban que era inapropiado hablar de una versión comercial en la lista: ¿era legítima semejante actividad o debería llevarse a cabo como parte de las actividades del servicio de asistencia de una compañía comercial? El anuncio de Gosling supuso por tanto toda una sorpresa, pues la suya ya era considerada la versión «no comercial».

Fecha: Martes 12 de abril de 1983 04:51:12

Asunto: EMACS se hace comercial

La versión de EMACS que escribí ahora está disponible comercialmente por medio de una empresa llamada Unipress. Ellos se encargarán del desarrollo y mantenimiento, y elaborarán un manual de verdad. EMACS estará disponible en muchas máquinas (ya se ejecuta en VAX con UNIX y VMS, con sistemas SUN, codatas y Microsoft Xenix). Junto a ello, lamento decir que ya no la voy a distribuir por más tiempo.

Se trata de un paso duro, pero siento que es necesario. Yo ya no puedo hacerme cargo del programa adecuadamente, pues tengo demasiadas demandas para el tiempo con el queuento. EMACS ha crecido hasta hacerse completamente inmanejable. Su popularidad ha vuelto imposible distribuirlo libremente: solo la tarea de grabar las cintas y meterlas en sobres es más de lo que puedo asumir.

La alternativa de abandonarlo al dominio público es inaceptable. Demasiados programas han acabado destruidos por esa vía.

Por favor, apoyad a esta gente. El esfuerzo que pueden permitirse aportar para hacerse cargo de EMACS será directamente proporcional al apoyo que reciban. Y sus precios son razonables.

JAMES²⁰

19. Allá por enero de 1983, Steve Zimmerman había anunciado que la compañía para la que trabajaba, CCA, había creado una versión comercial de EMACS llamada CCA EMACS (Identificador de mensaje: 385@yetti.uucp). Zimmerman no había escrito esta versión en su totalidad, sino que había tomado una versión escrita por Warren Montgomery en Laboratorios Bell (escrita para UNIX en un PDP-11s) y había creado la versión que los programadores de CCA estaban usando. Al parecer, Zimmerman había distribuido su creación por ftp al comienzo, pero cuando CCA determinó que podría ser de algún valor, decidieron explotarla comercialmente, en vez de dejar que Zimmerman la distribuyera «libremente.» Según el propio relato de Zimmerman, todo este procedimiento requería asegurarse de que no había quedado nada del código original de Warren Montgomery que era propiedad de Laboratorios Bell (Identificador de mensaje: 730@masscomp.uucp).

20. Identificador del mensaje de Gosling: bnews.sri-arpa.865.

Merece la pena prestar atención al mensaje de manera detenida: la tarea de distribuir las cintas se había hecho «inmanejable», y está claro que Gosling cree que el trabajo de mantenimiento, actualización y portabilidad (hacer el programa disponible en múltiples arquitecturas) debería acometerlo una empresa comercial. Es evidente que Gosling no entendía que su labor de creación y mantenimiento de EMACS había surgido de la compartición comunal de fragmentos de código —pese a haber hecho una labor sisífica para incorporar todos los cambios y sugerencias de sus usuarios—, pero sí que tenía desde hacía tiempo un compromiso por distribuirlo libremente, el cual propició que mucha gente aportara su granito de arena a GOSMACS.

«Libremente», sin embargo, no quería decir «de dominio público»,²¹ como queda claro por su afirmación de que «abandonarlo» al dominio público destruiría el programa. Se trata de una distinción importante que se les escapaba (y se les sigue escapando) a muchos miembros sofisticados de net.emacs. Aquí, *libre* significa libre de cobro, pero Gosling no tenía ninguna intención de dejar que la palabra insinuase que él no era el autor, propietario, mantenedor, distribuidor y único beneficiario de cualquiera que fuera el valor de GOSMACS. El *dominio público*, por el contrario, implicaba ceder todos estos derechos.²² Su decisión de vender GOSMACS a Unipress suponía transferir tales derechos a una compañía que pasaría a cobrar por todo el trabajo que previamente él había proporcionado sin coste alguno («libremente»). Tal distinción no le quedaba clara a todo el mundo: muchos consideraban que el hecho de que GOSMACS fuera gratuito implicaba que era de dominio público.²³ Una de ellas era ni más ni menos que Richard Stallman, quien se refirió al acto de Gosling como un «sabotaje de software» e instó a la gente a que dejara de usar la versión «semi-*ersatz*»²⁴ de Unipress.²⁵

21. El sentido de todo este párrafo en castellano debe entenderse con el trasfondo de la consabida polisemia de *free* en inglés que, parafraseando a Stallman una vez más, puede aludir tanto a «barra libre» (el sentido de gratuidad con el que Gosling emplea aquí el adjetivo) como a «libertad de expresión». [N. del E.]

22. El hilo que comienza con el mensaje con Identificador 969@sdcsvax.uucp contiene un ejemplo de debate en torno a la diferencia entre dominio público y software comercial.

23. En particular, un hilo que debate esto en detalle comienza con el mensaje con Identificador 172@encore.uucp e incluye los mensajes con los siguientes Identificadores 137@osu-eddie.UUCP, 1127@godot.uucp y 148@osu-eddie.uucp.

24. Véase nota 14 en este mismo capítulo. [N. del E.].

25. Identificador del mensaje: bnews.sri-arpa.988.

Para Stallman, el avance en la comercialización de EMACS, tanto por CCA como por Unipress, suponía un panorama frustrante. La comercialización de CCA había sido poco preocupante mientras GOSMACS permaneció libre, pero con el anuncio de Gosling ya no había ninguna versión de EMACS para UNIX disponible. Para Stallman, sin embargo, «libre» significaba algo más que «de dominio público» o «sin coste». La comuna de EMACS estaba diseñada para mantener el programa vivo y en constante crecimiento, además de para distribuirlo gratis: era un símbolo de administración comunitaria, en una comunidad que había incluido a Gosling hasta abril de 1983.

La desaparición de una versión de EMACS para UNIX, junto con el repentino interés por hacer de UNIX un sistema operativo comercializable, alimentaron el incipiente plan de Stallman de crear un sistema operativo UNIX completamente nuevo, no comercial y al margen de AT&T, que él distribuiría libremente a cualquiera que supiera usarlo. El 27 de septiembre de 1983 anunciaba así sus intenciones.²⁶

¡Por un Unix libre!

A partir del próximo Día de Acción de Gracias comenzaré a escribir un sistema de software completo compatible con Unix llamado GNU (que significa «Gnu No es Unix»), y lo distribuiré libremente para que todos puedan usarlo. Son muy necesarias las contribuciones de dinero, programas y equipos.

Sus justificaciones eran sencillas:

Por qué debo escribir GNU

Considero que la regla de oro exige que si a mí me gusta un programa, debo compartirlo con otras personas a quienes también les gusta. Mi conciencia no me permite firmar un acuerdo de confidencialidad o un acuerdo de licencia de software. Para poder seguir utilizando computadoras sin violar mis principios, he decidido reunir suficiente software libre para no tener que usar ningún programa que no sea libre.²⁷

26. Identificador del mensaje: 771@mit-eddie.uucp, anunciado en net.unix-wizards y net.usof.

27. Identificador del mensaje: 771@mit-eddie.uucp.

En ese punto, está claro que no había ninguna «licencia de software libre». Existía la palabra *libre*, pero no el término *dominio público*. Existía también la «regla de oro», así como una resistencia a los acuerdos de confidencialidad y las licencias en general, pero ciertamente no existía una concepción bien articulada del *copyleft* del software libre como una entidad distintiva legalmente. Y aun así, la intención de Stallman no era ni mucho menos «abandonarlo» al dominio público, como insinuaba Gosling. En vez de eso, es probable que Stallman pretendiera requerir que las mismas reglas de la comuna de EMACS se aplicaran al software libre, reglas que él podría controlar en gran parte supervisando (en un sentido no legal) a quiénes se enviaba o se vendía qué, y demandando (en forma de mensajes adjuntos al software) que cualquier modificación o mejora viniese en forma de donación. Fue durante este periodo de 1983 a 1985 cuando la comuna de EMACS mutó en la GPL, a medida que Stallman empezaba a añadir avisos de *copyright* y apéndices que explicitaban lo que la gente podía hacer con el software.²⁸

Sin embargo, al principio el proyecto GNU recibió escasa atención: mensajes sueltos en net.unix-wizards durante el transcurso de 1983 y 1984 preguntan periódicamente por su situación y cómo contactar con los responsables, a menudo en el contexto de debates sobre las prácticas de concesión de licencias de UNIX que AT&T estaba desplegando a medida que se desprendía de UNIX y empezaba a comercializar su propia versión del programa.²⁹ El plan original de Stallman para GNU era empezar por el núcleo del sistema operativo, el *kernel*, pero su extensa labor con EMACS y la repentina necesidad de un EMACS libre para UNIX le llevaron a empezar por ahí: de 1984 a 1985 él y otros empezaron a trabajar en una versión para UNIX de GNU EMACS. Las dos versiones comerciales de EMACS para dicho sistema operativo (la de CCA y la de Unipress) continuaron circulando y mejorando en paralelo. Mientras

28. Otras diversas personas parecen haber concebido un esquema similar por la misma época (si hemos de guiarnos por los archivos de Usenet), incluyendo a Guido Van Rossum (que luego se haría famoso por la creación del lenguaje de guiones Python). La siguiente nota aparece en el mensaje con Identificador 5568@mcvax.uucp:

/* Este software tiene *copyright* (c) Mathematical Centre, Amsterdam,

* 1983.

* Se concede permiso para usar y copiar este software, pero sin ánimo de *lucro,

* y siempre que estas mismas condiciones se impongan a cualquier persona

* que reciba o use el software.

*/

29. Por ejemplo, Identificador de mensaje: 6818@btl-tgr.arpa.

tanto, los usuarios de DEC empleaban la versión original libre y gratuita creada por Stallman. Y, como suele ocurrir, la vida siguió: Zimmerman dejó CCA en agosto de 1984 y Gosling se pasó a Sun, y ninguno de los dos siguió activamente implicado en el software que habían creado, dejándolo en manos de sus nuevos propietarios.

Para marzo de 1985, Stallman tenía una versión completa (la número 15) de GNU EMACS que se ejecutaba en la versión BSD 4.2 de UNIX (que Bill Joy había ayudado a crear y que se había llevado con él para conformar el núcleo de la versión de Sun), que a su vez se ejecutaba en los ordenadores VAX de DEC. Stallman anunció este software con su característico estilo extravagante, publicando en la revista mensual para programadores *Dr. Dobbs* un artículo titulado «El manifiesto GNU»³⁰.

El anuncio de Stallman de que estaba disponible una versión libre de EMACS para UNIX causó cierta preocupación entre los distribuidores comerciales. La principal preocupación era que GNU EMACS 15.34 contenía un código empleado para la visualización en pantalla de la aplicación que estaba marcado como «Copyright © James Gosling»³¹. El «descubrimiento» (nada complicado, dado que Stallman siempre distribuía el código fuente junto con el binario) de que dicho código había sido reutilizado por Stallman llevó a largos debates entre los usuarios de EMACS sobre asuntos como la mecánica del *copyright*, la naturaleza de las infracciones del mismo, la definición de software, el significado del dominio público, la diferencia entre patente, *copyright* y secreto comercial, y la mecánica de los permisos y de su concesión —en suma, un debate que se recapitularía repetidas veces en cada controversia sobre software y propiedad intelectual en el futuro.

La historia de la controversia revela que la estructura de los rumores en Usenet se asemeja un poco al juego infantil del teléfono roto, solo que todos los mensajes están archivados. GNU EMACS 15.34 fue lanzado en marzo de 1985, y hasta principios de junio no hubo referencia alguna a su situación legal, pero alrededor del 3 de junio empezaron a proliferar mensajes al respecto. La primera mención del tema no

30. Stallman, «The GNU Manifesto». Disponible en GNU's Not Unix!, <http://www.gnu.org/gnu/manifesto.html> [Versión en castellano: <https://www.gnu.org/gnu/manifesto.es.html>].

31. El principal archivo de la controversia se llamaba «*display.c*». Una versión modificada por Chris Torek aparece en *net.sources*, Identificador de mensaje: 424@umcp-cs.uucp. Un ejemplo distinto de un fragmento de código escrito por Gosling va acompañado de una nota que afirma haberlo declarado de dominio público, pero sin «incluir la infame cláusula anti-*copyright* de Stallman» (Identificador de mensaje: 78@tove.uucp).

apareció en net.emacs, sino en fa.info-vas —un grupo de noticias dedicado a hablar de los sistemas VAX («fa» viene de «*from Arpanet*», «de Arpanet»)—, e incluía un diálogo entre Ron Natalie y Marty Sasaki etiquetado como «GNU EMACS: ¿Cómo que dominio público?»:

FOO, no esperes que GNU EMACS de verdad sea de dominio público. UNIPRESS parece bastante molesta con que grandes porciones de él vengan marcadas con el *copyright* de James Gosling³². Este mensaje se reprodujo el 4 de junio en net.emacs con la siguiente adenda: «El trabajo de RMS³³ está basado en una versión del código de Gosling que existía antes de que Unipress lo adquiriera. Gosling había puesto ese código en el dominio público. Cualquier trabajo que parta de ese primer código de Gosling también es por consiguiente de dominio público.³⁴

A esta adenda le siguió entonces una exhaustiva respuesta de Zimmerman, cuyo CCA EMACS se había basado en el EMACS creado por Warren Montgomery en Laboratorios Bell pero había sido reescrito para evitar reutilizar el código, lo que puede explicar por qué su interpretación del asunto parece haber sido a la vez tan profunda y tan alarmante para él:

Esto va completamente en contra de las declaraciones públicas de Gosling. Antes de alcanzar el acuerdo con Unipress, la política de Gosling era que enviaría una copia gratis de su EMACS a cualquiera que la pidiera, pero no dio permiso (al menos públicamente) para que nadie más hiciera copias. Una vez que Unipress empezó a vender el EMACS de Gosling, él dejó de distribuir copias gratuitamente y siguió sin conceder permiso a nadie para que las hiciera; en vez de eso, sugirió a la gente que comprase EMACS a Unipress. Todas las versiones del EMACS de Gosling que él distribuyó llevan su aviso de *copyright*, y por tanto ninguna es de dominio público. Retirar los avisos de *copyright* sin permiso del autor es ilegal, por supuesto. Pues bien, una rápida comprobación de las fuentes de mi GNU EMACS muestra que, sin lugar a dudas, una serie de archivos tiene el aviso de *copyright* de Gosling. Lo que significa todo esto es que a no ser

32. Identificador de mensaje: 7773@ucbvax.arpa.

33. Abreviación habitual para referirse a Richard Matthew Stallman. [N. del E.]

34. Identificador de mensaje: 11400007@inmet.uucp.

que RMS consiguiera permiso por escrito de Gosling para distribuir su código, todas las copias de GNU EMACS constituyen una violación de la legislación de *copyright*. Todos aquellos que hagan copias, incluyendo a quienes permitan que otros las saquen de sus equipos, podrían exponerse a sanciones muy cuantiosas. Me parece que RMS haría bien en decírnos si tiene permiso por escrito de Gosling para hacer dichas copias. Si es así, ¿por qué no lo ha dicho antes (preferiblemente en la propia distribución), aclarando así un punto que podría generar una enorme confusión? Si no, ¿por qué ha seguido adelante y ha hecho que muchas, muchas personas puedan quedar sujetas a procesamiento criminal al recomendarles que distribuyeran el código sin advertirles siquiera de su responsabilidad legal? (Quienes distribuyan este código serían responsables incluso si alegan no haber visto los avisos de Gosling: el hecho de que los avisos estén ahí ya es suficiente. «La ignorancia de la ley no exime de su cumplimiento»).

Ahora bien, yo no tengo nada en contra del software libre; este es un país libre y la gente puede hacer lo que quiera. Se trata solo de que quienes distribuyen software libre harán bien en asegurarse de que tienen derecho legal para hacerlo, o que se preparen para afrontar las consecuencias. (9 de junio de 1985)³⁵

Stallman respondió al día siguiente:

No hay razón alguna para que nadie tenga miedo de usar o distribuir GNU EMACS. Es bien sabido que yo no creo que el software sea propiedad de nadie. No obstante, para el proyecto GNU decidí que era necesario obedecer la ley. Me he negado a considerar ningún código que no tuviera permiso para distribuir. Aproximadamente un 5% de GNU EMACS es similar (aunque bastante modificado) a una versión vieja del EMACS de Gosling. Yo lo estoy distribuyendo por Fen Labalme, quien recibió permiso de Gosling. Por lo tanto, es legal que yo lo haga. Para ser escrupulosamente legal, he puesto declaraciones que describen esta situación en el encabezamiento de los archivos correspondientes.

No veo nada de lo que deba avisar a la gente —excepto de que Zimmerman va a intentar amedrentarles.³⁶

35. Identificador de mensaje: 717@masscomp.uucp.

36. Identificador de mensaje: 4421@mit-eddie.uucp.

La original defensa de Stallman por haber usado el código de Gosling fue que tenía permiso para hacerlo: según él, Fen Labalme había obtenido permiso escrito para el código de visualización incluido en GNU EMACS 15.32 —no queda claro si para usarlo o para redistribuirlo. Según Stallman, en varios sitios (incluyendo la empresa donde trabajaba Labalme, Megatest) estaban usándose versiones de la versión de Labalme de la versión de Gosling de EMACS, y Stallman y Labalme consideraban que ello les dotaba de una posición legalmente defendible.³⁷

Durante las dos semanas siguientes, hubo un aluvión de mensajes que intentaban desentrañar e interpretar los problemas de *copyright*, propiedad, distribución y autoría. Gosling escribió para aclarar que GOSMACS jamás había sido de dominio público, pero que «desafortunadamente, dos mudanzas han dejado mis registros patas arriba» y por tanto guardaba silencio sobre la cuestión de si concedió permiso.³⁸ Es bastante probable que la afirmación de Gosling fuera estratégica: la concesión del permiso, de haber existido, podría haber enfadado a Unipress, que esperaba tener control exclusivo sobre la versión que se le había vendido; por la misma regla de tres, es igualmente probable que Gosling aprobara la recreación de Stallman pero no quisiera afirmarlo de ninguna manera legalmente perseguible. Entretanto, Zimmerman transmitió un mensaje anónimo que insinuaba que algunos abogados en alguna parte estimaban que el argumento de la «redistribución de tercera mano» simplemente «no había por dónde cogerlo».³⁹

La mayor preocupación de Stallman no era tanto la legalidad de sus propias acciones como la posibilidad de que la gente decidiese no usar el programa debido a amenazas legales (incluso si tales amenazas eran emitidas únicamente como rumores por ex-empleados de compañías que distribuían software escrito por ellos). Stallman quería que los usuarios no solo se sintieran seguros usando su programa sino que adoptasen su visión de que el software existe para ser compartido y mejorado, y que cualquier cosa que estorbe este proceso supone una pérdida para todos, lo cual hace necesaria una comuna de EMACS.

37. Identificador de mensaje: 4486@mit-eddie.uucp. Stallman también relata esta versión de los acontecimientos en «RMS Lecture at KTH (Sweden)», 30 de octubre de 1986, <http://www.gnu.org/philosophy/stallman-kth.html>.

38. Identificador de mensaje: 2334@sun.uucp.

39. Identificador de mensaje: 732@masscomp.uucp.

La base legal de Stallman para usar el código de Gosling pudo o no haber sido sólida. Zimmerman hizo cuanto pudo durante todo el transcurso del debate para explicar detalladamente qué tipo de permiso habrían necesitado Stallman y Labalme, echando mano de su experiencia con los abogados de CCA y con los Laboratorios Bell de AT&T, todo ello mientras reprendía a Stallman por no crear el código de visualización él mismo. Mientras tanto, Unipress emitió un mensaje oficial que decía:

UniPress quiere informar a la comunidad de que hay porciones del programa GNU EMACS que de ninguna manera son de dominio público, y que el uso y/o distribución de GNU EMACS no es necesariamente adecuado.⁴⁰

El ciertamente vago tono del mensaje dejó a la mayoría de la gente con la duda de qué quería decir —y de si Unipress tenía intención de demandar a alguien. Estratégicamente hablando, puede que la compañía desease estar a bien con los *hackers* y lectores de net.emacs, un público compuesto probablemente por muchos clientes potenciales. Además, si Gosling le había dado permiso a Stallman, entonces la propia Unipress se vería en un terreno jurídico incierto, al no poder amenazar firme y claramente a los usuarios de GNU EMACS con acciones legales. En cualquier caso, la cuestión de si se necesitaba o no permiso quedaba fuera de discusión —solo se discutía si había sido concedido.⁴¹

Sin embargo, a resultas de ello surgió un asunto legal más complicado, relativo al estatuto legal del código remitido a Gosling por terceras personas. Fen Labalme escribió un mensaje en net.emacs que, aunque no aclaraba la situación jurídica del código de Gosling (Labalme también fue incapaz de hallar el «permiso» que este le dio), sí que planteaba una cuestión relacionada: el hecho de que él y otros habían aportado contribuciones significativas a GOSMACS, que Gosling había incorporado a su versión, para luego vendérsela a Unipress sin permiso de aquellos.

40. Identificador de mensaje: 103@unipress.uucp.

41. Contemplada en retrospectiva, la postura de que el software podría ser de dominio público también parece legalmente incierta, dado que las modificaciones de 1976 a la Ley de Copyright estadounidense abolieron el requisito de registrar las obras y, por la misma razón, volvieron incierto el estatus del código remitido a Gosling e incorporado por este a GOSMACS.

Como uno de los ‘otros’ que ayudaron a poner EMACS [GOSMACS] a pleno rendimiento, me sentí desolado cuando Jim le vendió el editor a UniPress. Aquello me pareció una violación directa de la confianza que yo y otros habíamos depositado en él al enviarle nuestras mejoras, modificaciones y depuraciones de errores. Me preocupa especialmente la actitud mercenaria generalizada en torno a EMACS que ha desplazado la antaño orgullosa ética *hacker* que hubo en su día —EMACS es una herramienta que puede mejorarnos la vida a todos. ¡Ayudemos a que crezca!⁴²

La inferencia de Labalme, de la que quizá él mismo no se percató, es que puede que Gosling hubiera infringido los derechos de otros al venderle el código a Unipress, como deja claro un mensaje distinto de Joaquim Martillo: «Las diferencias entre la versión actual del EMACS de Unipress y el display.c de Gnu EMACS (un módulo de 19 páginas) son de aproximadamente un 80%. En todos los módulos que RMS usó con permiso de Fen LeBalme [*sic*] las diferencias son similares. ¡Unipress ya ni siquiera está usando el software en disputa! Eso sí, estos módulos contienen código que gente como Chris Torek y otros aportó cuando el emacs de Gosling era de dominio público. Debo preguntarme si estas personas hubieran contribuido de saber que el código que aportaban gratuitamente iba a terminar formando parte de un producto ajeno»⁴³.

En realidad, la ironía general de esta complicada situación no fue desde luego tan evidente como podía haber sido dado el tono emotivo de los debates: Stallman estaba usando código de Gosling basándose en el permiso que este había dado a Labalme, pero Labalme había escrito código para Gosling que este había comercializado sin decírselo —posiblemente, aunque no sea probable, el mismo código. Para más inri, todos ellos estaban creando software que había sido concebido originalmente por Stallman (aunque basado en ideas y trabajo de TECO, un editor escrito veinte años antes que EMACS), que ahora estaba ocupado reescribiendo el mismo software que Gosling había reescrito para UNIX. La «antaño orgullosa ética *hacker*» mencionada por Labalme equivaldría así no tanto a una creencia explícita en la compartición cuanto a una práctica desidiosa de aportar contribuciones y arreglos

42. Identificador de mensaje: 18@megatest. Nótese aquí el uso de «la antaño orgullosa ética *hacker*», que parece confirmar el sentimiento perpetuo de que la ética se ha visto comprometida.

43. Identificador de mensaje: 287@mir-athena.uucp.

sin documentarlos, de dar permisos orales de utilización y reutilización y de «perder» registros que pudieron o no existir —una empresa no precisamente muy noble.

Con todo, el 27 de junio de 1985 todo el debate jurídico se tornó fútil cuando Stallman anunció que reescribiría por completo el código de visualización de EMACS.

He decidido sustituir el código de Gosling en GNU EMACS, aunque sigo creyendo que Fen y yo tenemos permiso para distribuirlo, con el propósito de mantener la confianza de la gente en el proyecto GNU.

Llegué a esta decisión cuando me di cuenta, anoche, de que sabía cómo reescribir las partes que me habían parecido complicadas. Espero tener el trabajo listo para el fin de semana.⁴⁴

El 5 de julio, Stallman envío un mensaje que decía:

¡Celebremos nuestra independencia de Unipress!

La versión 16 de EMACS, 100% libre de Gosling, está siendo probada en varios lugares. Parece que funciona sólidamente en Vax, pero aún queda probarlo en algunas otras máquinas.⁴⁵

El hecho de que solo tardase una semana en escribir el código es un testimonio de la ampliamente reconocida capacidad de Stallman de crear grandes programas —sin que parezca indicativo de amenaza (legal) o urgencia algunas. Es más, aunque Unipress también parecía preocupada por su propia reputación, y a pesar de la insinuación de Stallman de que habían forzado toda la controversia, la empresa se tomó un mes para responder. En ese punto, el empleado de Unipress Mike Gallaher escribió para insistir, un poco a toro pasado, en que Unipress no tenía ninguna intención de demandar a nadie —siempre y cuando se usase la versión 16 libre de Gosling de EMACS o superiores.

UniPress no tiene nada en contra del proyecto Gnu. Me molesta que la gente tenga la impresión de que estamos intentando obstaculizarlo. De hecho, apenas si hemos dicho o hecho gran cosa, excepto señalar que el código Gnumacs llevaba el *copyright* de

44. Identificador de mensaje: 4559@mit-eddie.uucp.

45. Identificador de mensaje: 4605@mit-eddie.uucp.

James Gosling. No hemos hecho nada para impedir que nadie use Gnumacs, ni tenemos intención de hacerlo ahora que ya está «libre de Gosling» (versión 16.56).

Podéis considerar esto como una declaración oficial de la compañía: en la versión 16.56 de Gnumacs no hay absolutamente nada que pueda molestar a UniPress. Si teníais miedo de usarlo porque creíais que os íbamos a incordiar, no lo tengáis, sobre la base de la versión 16.56.⁴⁶

Tanto Stallman como Unipress recibieron diversos ataques y defensas por parte de observadores de la controversia. Mucha gente señaló que a Stallman debería atribuirse la «invención» de EMACS y que la cuestión de si estaba infringiendo su propia invención era en consecuencia irónica. Otros proclamaban la inocencia y el carácter moral de Unipress, que, según se aseguraba, estaba proveyendo un servicio (el mantenimiento de EMACS) más que el programa en sí.

Algunos lectores interpretaron el hecho de que Stallman hubiera reescrito el código de visualización, fuera o no por la presión de Unipress, como confirmación de las ideas expresadas en «El Manifiesto GNU», a saber, que el software comercial asfixia la innovación. Según esta lógica, precisamente porque Stallman se vio obligado a reescribir el código en vez de seguir trabajando sobre algo para lo que daba por hecho que tenía permiso, no hubo innovación, solo precaución inducida por el miedo.⁴⁷ Por otro lado, latiendo en este debate había un profundo sentimiento de propiedad relativo a lo que la gente había creado: muchas personas, no solo Stallman, Gosling y Zimmerman, habían contribuido a hacer de EMACS lo que era, y la mayoría lo había hecho en la suposición, legalmente correcta o no, de que no les arrebatarían su labor ni mucho menos de que otros sacarían tajada de ella.

Así pues, la venta de EMACS por parte de Gosling es de un orden diferente al de su participación en la administración comunal del programa. La distinción entre la creación y el mantenimiento del software es una ficción comercial motivada en gran parte por la estructura de la propiedad intelectual. Esto refleja la experiencia de los sistemas

46. Identificador de mensaje: 104@unipress.uucp.

47. Joaquim Martillo, Identificador de mensaje: 287@mit-athena.uucpp: «Intentar prohibir a RMS usar el código descartado de modo que deba dedicar tiempo a reinventar la rueda respalda su aseveración de que los ‘avaros del software’ están ralentizando el progreso en ingeniería informática».

abiertos: el mantenimiento del software puede suponer mejorarlo, y su mejora puede suponer incorporar el trabajo y las ideas originales de otros. Hacer esto según las normas de una estructura de propiedad intelectual cambiante obliga a elecciones distintas a cuando se actúa conforme a una ética *hacker* informal o a una «comuna» experimental. La mejora minúscula de un programador se convierte así en la contribución original de otro.

El contexto del *copyright*

La controversia de EMACS ocurrió en el periodo inmediatamente posterior a algunos de los mayores cambios en la legislación estadounidense de propiedad intelectual de los últimos ochenta años. Merece la pena enfatizar dos aspectos de este contexto: 1) las prácticas y el conocimiento sobre el Derecho cambian lentamente y no reflejan inmediatamente las modificaciones, ya sea en la ley o en las estrategias de los actores; 2) el Derecho estadounidense crea una estructura formal de incertidumbre donde la interacción entre legislación y jurisprudencia nunca está del todo determinada. En cuanto al primer aspecto, los programadores que crecieron en los 70 contemplaron unas prácticas comerciales dominadas por entero por el secreto comercial y la protección de patentes, y muy raramente por el *copyright*; de ahí que el desplazamiento hacia un empleo generalizado de la legislación de *copyright* (facilitado por las enmiendas de 1976 y 1980 a la Ley de *Copyright*) para proteger el software supusiera a su vez un desplazamiento en el modo de pensar del que muchos participantes, incluidos los más astutos legalmente, cayeron en la cuenta solo lentamente, por tratarse de un desplazamiento estratégico general, y no solo legal. En cuanto al segundo aspecto, las enmiendas a la legislación de *copyright* introducidas en 1976 y 1980 contenían una serie de incertidumbres que costaría más de una década resolver en la jurisprudencia en asuntos como la aplicabilidad del *copyright* a los programas informáticos, la definición de software y el significado de infringir el *copyright* de un programa, por no hablar del impacto de la codificación del uso justo [*fair use*]⁴⁸ y de la eliminación

48. A diferencia de legislaciones (vinculadas al llamado derecho continental) que estipulan al detalle los «límites» a los derechos de autor (véase, por ejemplo, el homónimo Título II del Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual), la ley de propiedad intelectual de EEUU prevé unas excepciones no tasadas regidas por la lógica del derecho consuetudinario (o *common law*). Dichas

del requisito de registrar las obras (asuntos que posiblemente pasaron desapercibidos hasta el cambio de milenio). Ambos aspectos prepararon el escenario para la controversia sobre EMACS y para la creación de la GPL por Stallman.

Hablando en términos legales, la controversia de EMACS giraba en torno al *copyright*, los permisos y los significados del dominio público y de la reutilización del software (y, aunque nunca se mencionó explícitamente, en torno al concepto de uso justo). La legislación sobre patentes de software y secretos comerciales no está directamente implicada, pero aun así configura el trasfondo de la controversia. Muchos de los participantes manifestaron una ortodoxia legal y convencional que sostenía que el software no era patentable, esto es, que los algoritmos, las ideas o las ecuaciones fundamentales quedan fuera del ámbito de las patentes, si bien el caso «Diamond contra Diehr» de 1981 es generalmente considerado el primer espaldarazo de los tribunales a la obligación de la Oficina Estadounidense de Patentes y Marcas Registradas de otorgar patentes sobre software.⁴⁹ El software, continuaba esta ortodoxia, estaba mejor protegido por la legislación de secretos comerciales (de jurisdicción estatal, no federal), la cual proporcionaba protección a cualquier obra intelectual que su propietario intentase razonablemente mantener en secreto. El estatuto de secreto comercial de UNIX, por ejemplo, significaba que todos los licenciatarios educativos a los que se dio el código fuente habían aceptado mantenerlo en secreto, aunque era manifiesto que estaba circulando por todo el mundo; por lo tanto, uno podía estar transgrediendo las normas del secreto comercial si le echaba un vistazo al código fuente (p. ej., mediante la firma de un

excepciones, para cuya definición *sobre la marcha* se articulan una serie de criterios (relativos a la naturaleza del uso en cuestión, al impacto de dicho uso en un mercado existente, a la porción de obra utilizada, etc.) delimitarían un espacio de «usos justos» [*fair uses*] de las obras protegidas que no requerirían permiso previo de los titulares de los derechos de autor (pese a lo cual en la práctica no es raro tener que acudir a defenderlos ante los tribunales estadounidenses, y ello quienes pueden permitírselo). En este sentido, es interesante confrontar lo aquí expuesto por Kelty respecto de «la codificación» del uso justo introducida en 1976 con su posterior alusión al controvertido encaje del «*sampling*» musical dentro del *fair use* (véase nota 19, Capítulo VIII). [N. del E.]

49. *Diamond V. Diehr*, 450 U.S. 175 (1981), en el que el Tribunal Supremo dictaminó que la oficina de patentes estaba obligada a otorgar patentes sobre programas informáticos. Resulta interesante señalar que mucho tiempo atrás ya se venían otorgando patentes de software, pero o bien no hallaron oposición o bien no se ejercitaron. Un excelente ejemplo es la patente 3,568,156, ostentada por Ken Thompson, sobre concordancia de patrones de expresión habituales, concedida en 1971.

acuerdo de confidencialidad o a través de alguien de la empresa que ofrecía acceso a él) y luego implementaba algo similar.

En cambio, la legislación de *copyright* rara vez se desplegaba en asuntos de producción de software. El primer registro de *copyright* de un programa se dio en 1964, pero la deseabilidad de recurrir a esta vía en vez de al secreto comercial no estuvo clara hasta bien entrados los 70.⁵⁰ Algunas corporaciones, como IBM, marcaban rutinariamente todo el código fuente con un símbolo de *copyright*; otras se limitaban a hacerlo en los binarios que distribuían o en los acuerdos de licencia. El caso del software del sistema operativo UNIX y sus derivados es particularmente azaroso, y la existencia de avisos de *copyright* de los autores varía mucho: una encuesta informal de Barry Gold identificó a James Gosling, Walter Tichy (autor de rcs) y la RAND Corporation como los únicos que habían etiquetado adecuadamente el código fuente con avisos de *copyright*.⁵¹ Gosling también fue el primero en registrar EMACS como software con *copyright* en 1983, mientras que Stallman registró GNU EMACS justo después de que se publicase la versión 15.34 en mayo de 1985.⁵²

La incertidumbre del paso de recurrir al secreto comercial a hacerlo al *copyright* queda clara en algunas de las declaraciones de Stallman acerca de la reutilización del código de Gosling. Dado que ni uno ni otro pretendían mantener sus programas en secreto de ninguna forma —ya fuera concediendo licencias o exigiendo tal secretismo a los usuarios—, no podía haber reivindicaciones de secreto comercial sobre ninguno de ellos. Aun así, era frecuente la preocupación sobre si alguien había visto cualquier fragmento de código (en especial de un sistema UNIX, amparado en el secreto comercial) y sobre si el código que otras personas hubiesen visto, reescrito o distribuido públicamente era por ende «de dominio público».⁵³ Pero, al mismo tiempo, a Stall-

50. Calvin Mooers, en su artículo de 1975 «Computer Software and Copyright», sugiere que la decisión de IBM sobre la desagregación abrió la puerta a considerar la protección del *copyright*.

51. Identificador de mensaje: 933@sdcrdcf.uucp.

52. El EMACS 264 de Gosling (Stallman copió el EMACS 84) está registrado en la Biblioteca del Congreso de EEUU, al igual que el GNU EMACS 15.34. El número de registro del EMACS de Gosling en la Biblioteca del Congreso es TX-3-407-458, con fecha de 1992. El número de registro de Stallman es TX-1-575-302, con fecha de mayo de 1985. Las fechas listadas resultan inciertas, no obstante, pues se dan periódicamente re-registros y actualizaciones.

53. Esto es particularmente confuso en el caso de «dbx». Identificadores de mensaje: 4437@mit-eddie.uucp, 6238@shasta.arpa y 730@masscomp.uucp

man le preocupaba que reescribir el código de visualización de Gosling resultara demasiado difícil:

Cualquier código de visualización se asemejaría considerablemente a ese otro, por la pura razón de que ejecutan la misma tarea. Sin tener una idea clara de cómo de diferentes tendrían que ser para tranquilizar a los usuarios, no puedo saber si la reescritura lo lograría. La ley no sirve para orientarnos en esto. [...] Escribir un código de visualización que sea significativamente distinto no es fácil.⁵⁴

La estrategia de Stallman para reescribir software, incluido su plan para el sistema operativo GNU, también implicaba «no mirar» el código de nadie más, para así asegurarse de no incurrir en ninguna violación del secreto comercial. Aunque estaba claro que el código de Gosling no era ningún secreto comercial, tampoco resultaba obvio que fuera «de dominio público», una suposición atribuible más bien a otros tipos de programas protegidos por tal secretismo. Según las normas de secreto comercial, la distribución pública de GOSMACS por parte de Gosling parecía dar luz verde a su reutilización, pero según la legislación de *copyright*, una ley de estricta responsabilidad, cualquier uso no autorizado supone una infracción, sin importar lo público que pueda haber sido el programa.⁵⁵

El hecho de la protección del secreto comercial fue, de todas formas, un aspecto importante de la controversia de EMACS: la versión que Warren Montgomery había creado en Laboratorios Bell (y en la que se basaría la de Zimmerman para CCA) *estaba* sujeta a dicha protección por parte de AT&T, en virtud de su distribución con UNIX y de un acuerdo de confidencialidad. En aquel momento AT&T aún se encontraba a un año de la desinversión y por consiguiente no podía acometer la explotación comercial del software. Cuando CCA pretendió comercializar la versión de UNIX que Zimmerman había basado en la

54. Identificador de mensaje: 4489@mit-eddie.uucp.

55. Una práctica normalizada hasta bien entrados los 80, e incluso después, fue la creación de las llamadas versiones de sala aséptica [*clean-room versions*] del software, esto es, la contratación de nuevos programadores y diseñadores que no habían visto el código infractor para reimplementarlo con el fin de evitar la aparición de violaciones del secreto comercial. La legislación de *copyright* es una estricta legislación de responsabilidad, lo que implica que la ignorancia no exime de culpa a los infractores, por lo que la práctica de la «ingeniería de sala aséptica» no parece haber tenido tanto éxito en el caso del *copyright*, dado que el significado de infracción aquí permanece nebuloso.

de Montgomery, se hizo necesario retirar cualquier código de AT&T para evitar infringir su estatuto de secreto comercial. A su vez, CCA ofrecía una distribución de su EMACS con el código binario y otra con el código fuente (costando la primera unos 1000 dólares y la segunda unos 7000) y recurrió al *copyright* en vez de al secreto comercial para evitar usos no autorizados de su software.⁵⁶

Así pues, la incertidumbre sobre el *copyright* era en parte reflejo de un cambio de estrategia en la industria del software, una especie de desarrollo irregular donde el *copyright* vino a reemplazar lenta e indiscriminadamente al secreto comercial como principal forma de protección de la propiedad intelectual. Este desplazamiento tuvo consecuencias sobre el modo en que los programadores no comerciales, los investigadores y los *amateurs* podían interpretar su propio trabajo, además de para las compañías cuyos abogados estaban enfrentándose a los mismos problemas. Por supuesto, el *copyright* y el secreto comercial no son mutuamente excluyentes, pero estructuran la necesidad de secretismo de maneras diferentes y construyen alegatos distintos sobre cuestiones como la similitud, la reutilización y la modificación.

Las modificaciones de 1976 a la legislación de *copyright* fueron por tanto extremadamente trascendentales a la hora de establecer una nueva serie de límites y posibilidades para las discusiones sobre propiedad intelectual, discusiones que originaron un tipo de incertidumbre diferente a la aparejada a un proceso de cambio de estrategia comercial: una incertidumbre estructural causada por la necesidad de que se generase jurisprudencia en torno a los cambios legislativos implementados por el Congreso.

La Ley de *Copyright* de 1976 introdujo una serie de cambios que llevaban gestándose unos diez años, organizados principalmente en torno a nuevas tecnologías como las fotocopiadoras, las grabaciones de audio

56. Identificador de mensaje: 730@masscomp.uucp. AT&T estaba menos preocupada por las infracciones del *copyright* que por el estatuto de sus secretos comerciales. Zimmerman cita una declaración (del mensaje con Identificador 108@emacs.uucp) que según él indica esto: «A partir de la versión 162.36z, CCA EMACS dejó de contener ningún código del EMACS del Sr. Montgomery, ni métodos o conceptos que solo conocerían los programadores familiarizados con cualquier versión de BTL EMACS [de Laboratorios Bell]. La declaración no mencionaba el *copyright*, pero daba a entender que CCA EMACS no contenía ningún secreto comercial de AT&T, preservando así el estatuto de secreto comercial de su software. El hecho de que EMACS fuera un diseño conceptual —un tipo particular de interfaz, un intérprete de LISP y su extensibilidad— muy ampliamente imitado no guardaba aparentemente ninguna relación con el estatuto legal de estos secretos.

domésticas y los nuevos aparatos de video: codificó los derechos de uso justo, eliminó el requisito de registrar las obras y expandió considerablemente el rango de materiales a los que era aplicable el *copyright*. Sin embargo, no aludió explícitamente al software, lo cual frustró a mucha gente de la industria informática, en particular de la joven industria de desarrollo de programas. A resultas de esta omisión, se encargó a la CONTU (*National Commission on New Technological Uses of Copyrighted Works*, Comisión Nacional sobre Nuevos Usos Tecnológicos de Obras con *Copyright*) que hiciera sugerencias de enmiendas a la ley en relación al software. Por consiguiente, no fue hasta 1980 cuando el Congreso estadounidense implementó estos cambios, incorporando el software al título 17 de la Ley de *Copyright* como algo a lo que podía legalmente aplicarse dicho *copyright*.⁵⁷

La enmienda de 1980 a la Ley de *Copyright* respondió a una de las tres cuestiones pendientes acerca de la aplicación del *copyright* al software: la sencilla pregunta de si el software era un material susceptible de protegerse mediante *copyright*. El Congreso contestó que sí, pero dejando sin definir qué constituía dicho «software». Durante los años 80, una serie de casos judiciales contribuyeron a especificar qué se entendía por tal cosa, incluyendo el código fuente, el código objeto (los archivos binarios), la visualización en pantalla y la salida de información, la experiencia de usuario [*look and feel*], así como el microcódigo y los microprogramas [*firmware*].⁵⁸ La pregunta final, cuya resolución aún tienen que afrontar los tribunales, consiste en cuánta similitud es constitutiva de una infracción en cada uno de estos casos. Las implicaciones de la codificación del uso justo y de la eliminación

57. Informe Final de la CONTU: <http://digital-law-online.info/CONTU/contu1.html> (última consulta: 18 de abril de 2018).

58. Los casos que determinan el significado de las enmiendas de 1976 y 1980 comienzan en torno a 1986: *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc., et al.*, Tribunal de Apelación del Tercer Circuito de EEUU, 4 de agosto de 1986, 797 F.2d 1222, 230 USPQ 481, afirmando que «la estructura (o secuencia y organización)» del software es susceptible de protección mediante *copyright*, no solo el código literal del programa; *Computer Associates International, Inc. v. Altai, Inc.*, Tribunal de Apelación del Segundo Circuito de EEUU, 22 de junio de 1992, 982 F.2d 693, 23 USPQ 2d 1241, alegando que la prueba de estructura de Whelan no era suficiente para determinar la infracción y proponía en consecuencia una prueba de «comparación-abstracción-filiación» a tres bandas; *Apple Computer, Inc. v. Microsoft Corp.*, Tribunal de Apelación del Noveno Circuito de EEUU, 1994, 35 F.3d 1435, hallando que la «metáfora del escritorio» usada en Macintosh y Windows no era idéntica y por ende no constituye una infracción; *Lotus Development Corporation v. Borland International, Inc.* (94-2003), 1996, 513 U.S. 233, dictaminando que no era aplicable el *copyright* a la «experiencia de usuario» (*look and feel*) de una interfaz de menú.

del requisito de registrar las obras continúan desplegándose incluso en nuestros días.

La controversia de EMACS se enfrenta a cada una de estas tres cuestiones. La creación inicial del programa por parte de Stallman se completó en condiciones en las que no estaba claro si el *copyright* era aplicable (es decir, antes de 1980). Stallman, por supuesto, no intentó poner *copyright* a las primeras versiones de EMACS, pero las enmiendas de 1976 eliminaron el requisito de registro, haciendo que todo lo escrito a partir de su entrada en vigor (1978) quedase automáticamente sujeto a *copyright*. El registro representaba solamente un esfuerzo adicional para reclamar la propiedad ante casos de supuesta infracción.

A lo largo de este periodo, la cuestión de si el *copyright* era aplicable —o de hecho aplicado— al software se respondía de manera diferente en cada caso: AT&T se apoyaba en el secreto comercial; Gosling, Unipress y CCA negociaron sobre material con *copyright*; y Stallman estaba experimentando con su «comuna». Aunque la incertidumbre obtuvo una respuesta legislativa con la enmienda de 1980, no todo el mundo captó inmediatamente este nuevo hecho o cambió sus prácticas de conformidad con él. Existen pruebas de sobra en el archivo de Usenet de que los cambios de 1976 se entendieron defectuosamente, en especial si lo comparamos con la sofisticación jurídica de los *hackers* de los 90 y del nuevo milenio. Aunque la ley cambió de nuevo en 1980, las prácticas lo hicieron con más lentitud, y las justificaciones cristalizaron en el contexto de experimentos como el de GNU EMACS.

Junto a ello, surgió una tensión entre el significado de código fuente y el de software. Por un lado estaba la cuestión de si el *copyright* era aplicable al código fuente o al binario, y por otro la de definir los *límites* del software en un contexto en el que toda aplicación depende de otra para poder funcionar siquiera. Por ejemplo, EMACS se construyó en su origen sobre la base de TECO, al cual se aludía tanto como editor cuanto como lenguaje de programación; incluso distinciones aparentemente obvias (como la de aplicación frente a lenguaje de programación) no tenían por qué estar siempre claras. Si EMACS era una aplicación escrita en TECO en cuanto lenguaje de programación, entonces parecería que EMACS debería tener su propio *copyright*, distinto al de cualquier otro programa escrito en TECO. Pero si EMACS era una extensión o modificación de TECO en cuanto editor, entonces parecería una obra derivada y requeriría el permiso explícito de los titulares del *copyright*.

Más aún, cada versión de EMACS, para ser EMACS, necesitaba de un intérprete de LISP con el fin de hacer que la interfaz extensible fuera similar en todas las versiones. Ahora bien, no todas las versiones usaban el mismo intérprete. Gosling, por ejemplo, empleaba uno llamado MOCLISP (mlisp en la versión de Unipress con marca registrada). La cuestión de si el intérprete de LISP era un componente esencial del programa o un «entorno» necesario para extender la aplicación también era incierta y no quedaba especificada en la ley. Si bien sería posible tratar ambos como elementos de software susceptibles de protección mediante *copyright*, también se los podía concebir como componentes necesarios sobre los que construir programas a los que fuera aplicable dicho *copyright*.⁵⁹

Pero eso no es todo, pues tanto las enmiendas de 1976 como las de 1980 no se pronuncian sobre el estatuto de *copyright* del código fuente frente al binario. Mientras que todas las versiones de EMACS se distribuían en binario, tanto Stallman como Gosling incluían el código fuente para permitir que los usuarios lo modificasen y extendiesen, pero discrepan sobre la forma de redistribución adecuada. El umbral entre modificar el software para uno mismo y la infracción de *copyright* aún no estaba claro y dependía del significado de «redistribución». Modificar el software para usarlo en un único ordenador podía ser necesario para conseguir ejecutarlo, pero en los primeros días de Arpanet la colocación inocente de ese código en el directorio público de un ordenador podía verse como una distribución masiva.⁶⁰

Por último, la cuestión de qué constituye una infracción estaba en el centro de esta controversia y no se resolvió por ley o por decisión judicial, sino simplemente reescribiendo el código para evitar el problema. El uso del código de Gosling por parte de Stallman, su alegación sobre un permiso de tercera mano, la presencia o ausencia de dicho permiso escrito, la venta de GOSMACS a Unipress cuando era

59. La relación entre la definición de (código) *fuente* y la de destino (*target*) provoca confusión en la legislación sobre software hasta hoy mismo, siendo uno de los ejemplos más extravagantes el caso de DeCSS. Véase Coleman, «The Social Construction of Freedom», cap. 1: Gallery of CSS Descramblers, <http://www.cs.cmu.edu/~dst/DeCSS/gallery/>.

60. Una interesante anécdota aquí es que el manual de EMACS fue asimismo publicado por la misma época que el EMACS 1.6 y estaba disponible como archivo TeX. Stallman también trató de gestionar el documento en papel de la misma manera (véase el mensaje con Identificador: 4734@mit-eddie.uucp, 19 de julio de 1985), y ello se convertiría mucho después en una cuestión diferente y más espinosa que derivaría en la GNU FDL (*Free Documentation License*, Licencia de Documentación Libre).

más que probable que contuviese código no escrito por Gosling sino registrado a su nombre —todas estas cuestiones complicaron el asunto de la definición de infracción hasta el punto de que la única opción viable que le quedaba a Stallman para continuar creando software era evitar por completo el uso de código ajeno. De hecho, la decisión de Stallman de usar el código de Gosling (que afirma haber cambiado en porciones significativas) podría haber quedado en nada si hubiese tomado la decisión nada ética e ilegal de no incluir de ningún modo el aviso de *copyright* (siguiendo la teoría de que se trataba de código original de Stallman, o de una imitación, más que de una porción del trabajo de Gosling). Es más, Chris Torak obtuvo permiso de Gosling para retirar su nombre y su *copyright* de la versión de display.c que había modificado sustancialmente, si bien Torak optó por no hacerlo: «La única razón por la que no lo hice es porque siento que debería atribuírsele el mérito de ser (cuando menos) la inspiración del código».⁶¹ Igualmente, es del todo probable que a Stallman le preocupase obedecer la ley y atribuir el mérito a quien correspondiese, y por ello dejase adjunto el aviso de *copyright* —un claro caso de difuminación de los significados de autoría y propiedad.

En suma, la interacción entre la nueva legislación y su interpretación en los tribunales o en la práctica representó una incertidumbre estructural que establecía restricciones novedosas al significado de *copyright*, y especialmente a las normas y formas del permiso y la reutilización. GNU EMACS 15.34 era la opción más segura —una versión completamente nueva que realizaba las mismas tareas, pero de una manera diferente, empleando algoritmos y código diferentes.

Pero incluso resolviendo la controversia, GNU EMACS planteó nuevos problemas a Stallman: ¿cómo sobreviviría la comuna de EMACS si no estaba claro que se pudiera usar legalmente el código ajeno, incluso si era aportado libremente? ¿Era legítima la acción de Gosling de vender a Unipress trabajo ajeno? ¿Sería capaz Stallman de hacer que se cumpliese lo contrario, a saber, de impedir que la gente comercializase el código de EMACS que le enviara? ¿Cómo evitaría Stallman la posibilidad futura de que sus propios voluntarios y colaboradores acabasen alegando que había infringido su *copyright*?

Hacia 1986, Stallman empezó a enviar una carta que registraba la transferencia formal del *copyright* a la FSF (la Fundación para el Software

61. Identificador de mensaje: 659@umcp-cs.uucp.

Libre que había creado a finales de 1985) y reconocía la igualdad de derechos de uso no exclusivo del software⁶². Aunque tal exigencia de expropiación de *copyright* podría parecer contraria a los objetivos del proyecto GNU, en el contexto del despliegue de la legislación de *copyright* y de la controversia en torno a GOSMACS tenía todo el sentido: tras haber sido acusado él mismo de no tener el permiso adecuado para usar material ajeno con *copyright* en su versión libre de GNU EMACS, Stallman tomó medidas para evitar que sucediera lo mismo en el futuro.

La interacción entre las cuestiones técnico-legales y las inquietudes «éticas» se reflejaba en asuntos prácticos como el miedo, la intimidación y las (mal)interpretaciones de sentido común acerca de la legislación de propiedad intelectual. Las amenazas veladas de Zimmerman sobre la responsabilidad legal se dirigían no solo a Stallman sino a cualquiera que usara el programa creado por él; quebrantar la ley era, para Zimmerman, una falta ética, no un problema de incertidumbre y cambio. Fuera correcta o no tal interpretación legal, vino a revelar los mecanismos por los cuales un nivel bajo de conocimiento minucioso de la ley —y de una ley fluctuante además (por no hablar de la reputación litigiosa del sistema jurídico de EEUU en todo el mundo)— parecía justificar a menudo la sensación de que comprar un programa era simplemente una opción menos arriesgada que adquirirlo gratis. Se daba por sentado que en estos casos serían las empresas, y no los clientes, los que deberían responder de las hipotéticas infracciones. Por la misma regla de tres, la súbita inquietud de los programadores (más que de los abogados) hacia la detallada mecánica del *copyright* conllevó que un gran número de personas se encontrase reivindicando ideas de sentido común, solo para acabar envueltos en una guerra de *flames* sobre lo que la legislación de *copyright* «dice realmente».

Tal debate ha continuado y crecido exponencialmente durante las últimas décadas, hasta el punto de que los *hackers* del software libre poseen ahora un conocimiento de la propiedad intelectual casi tan profundo como del código informático⁶³. Lejos de representar el triunfo de la ética *hacker*, la Licencia Pública General de GNU representa el resultado concreto y tangible de una conversación cultural de alcance relativamente grande cercada por leyes cambiantes, sentencias judicia-

62. Identificador de mensaje: 8605202356.aa12789@ucbvax.berkeley.edu.

63. Véase Coleman, «The Social Construction of Freedom», cap. 6, sobre el Proceso de Nuevo Miembro de Debian, para un ejemplo de cómo la iniciación a un proyecto de software libre enfatiza lo legal tanto como lo técnico, si no más.

les, prácticas tanto comerciales como académicas y experimentos con los límites y las formas de los nuevos medios y tecnologías.

Conclusión

El resto de la historia se relata rápido: Stallman dimitió del AI Lab del MIT y fundó la FSF en 1985; creó un gran número de herramientas nuevas, pero sin completar finalmente un sistema operativo UNIX, y publicó la versión 1.0 de la GPL en 1989. En 1990 se le concedió una «beca de genios» MacArthur, y durante esa década se involucró en varias batallas de perfil alto entre una nueva generación de *hackers*. Dichas controversias incluyeron el debate sobre la creación por Linus Torvalds de Linux (a la que Stallman insistía en que debía denominarse GNU/Linux), la bifurcación de EMACS en Xemacs y la propia participación —y exclusión— de Stallman en congresos y eventos dedicados al software libre.

Entre 1986 y 1990 la FSF y sus programas se hicieron extremadamente célebres entre los *geeks*. En esto tuvo mucho que ver la riqueza del software que produjeron y distribuyeron vía Usenet y Arpanet. Y a medida que el software circulaba y se refinaba, lo mismo sucedía con las nuevas restricciones legales y el proceso de enseñar a los usuarios a entender lo que podían y no podían hacer con el software —y por qué *no* era de dominio público.

Cada vez que se publicaba una nueva aplicación, venía acompañada de uno o más ficheros de texto que explicaban cuál era su estatuto legal. Al principio, había un fichero llamado DISTRIB, que contenía una explicación de los derechos que los nuevos propietarios tenían para modificar y redistribuir el software.⁶⁴ DISTRIB hacía referencia a un fichero llamado COPYING, que contenía el «aviso de permiso de copia de GNU EMACS», también conocido como la GNU EMACS GPL. La primera de estas licencias reflejaba que Richard Stallman era el titular del *copyright* (en 1985), pero en 1986 ya todas reconocían dicha titularidad a la FSF.

A medida que la FSF publicaba otras aplicaciones, se renombraba la licencia —GNU CC GPL, GNU Bison GPL, GNU GDB GPL, y así sucesivamente, todas ellas conteniendo esencialmente las mismas condiciones— en un fichero llamado COPYING, que debía distribuirse junto

64. Por ejemplo, véase el mensaje con Identificador: 5745@ucbvax.arpa.

con la aplicación. En 1988, después de que el software y las licencias hubieran alcanzado una disponibilidad considerablemente mayor, Stallman introdujo algunos cambios en la licencia que suavizaban algunas condiciones y especificaban otras.⁶⁵ Esta nueva versión se convertiría en la GNU GPL 1.0. Para cuando el software libre apareció en la conciencia pública a finales de los 90, la GPL había alcanzado la versión 2.0 y la FSF tenía su propio equipo legal.

Es frecuente interpretar la creación de la GPL y la FSF como expresiones de la ética *hacker* o del compromiso ideológico de Stallman con la libertad. Sin embargo, la historia de EMACS y los complejos detalles técnicos y legales que lo estructuran ilustran cómo la GPL es más que un simple *hack*: era una «comuna» legal novedosa y ordenada de forma privada. Se trataba de un espacio rigurosamente independiente pero que se insinuaba entre los cimientos existentes de normas y prácticas del mundo informático empresarial y universitario, y se cincelaba a partir de la sustancia resbaladiza y cambiante de las leyes de la propiedad intelectual. En un momento en el que los gigantes de la industria del software estaban pugnando por crear un tipo diferente de apertura —una que preservara e incluso reforzara las relaciones de propiedad intelectual existentes— este *hack* constituía una alternativa radical que enfatizaba la soberanía, no de un *statu quo* nacional o corporativo, sino de unos individuos caracterizados por autorrepresentación [*self-fashioning*] que buscaban desvincularse de dicha unidad nacional-empresarial. La creación de la GNU GPL no entrañaba un retorno a una edad dorada de comunidades a pequeña escala liberadas de las estructuras dominantes de la modernidad burocrática, sino la creación de algo nuevo a partir de dichas estructuras. En definitiva, se valía de dichas estructuras y enfatizaba no su destrucción sino su estabilidad —al menos hasta que dejaran de ser necesarias.

La trascendencia de la GPL se debe a su incrustación en y su surgimiento de la infraestructura legal y técnica. Tal práctica de reelaboración situada es lo que dota al software libre —y quizá a todas las formas de práctica de ingeniería creativa— su urdimbre y trama. La decisión de Stallman de dimitir del AI Lab y fundar la FSF es un buen ejemplo: ello permitió a Stallman no solo dedicar su energía al software libre sino también diferenciar formalmente las organizaciones, evitar cuando menos la amenaza potencial de que el MIT (que todavía le proporcio-

65. Identificador de mensaje: 8803031948.aa01085@venus.berkeley.edu.

naba despacho, equipo y conexión de red) pudiera decidir reclamar la propiedad de su obra. Podría pensarse que la ética *hacker* y la imagen de los individuos libres autodeterminados demandarían la ausencia total de organizaciones, pero en vez de ello requiere su proliferación y modulación. El mismo Stallman nunca fue tan puramente libre, pues se apoyaba en la esplendidez del AI Lab del MIT, sin el cual se habría visto sin despacho, ordenador, conexión de red y, de hecho, durante una temporada, sin casa.

La FSF representa un reconocimiento por parte de Stallman de que la independencia individual y comunal llegaría al precio de recurrir a una entidad legal y burocráticamente reconocible, diferenciada del MIT y responsable únicamente ante sí misma. La FSF adoptó una forma clásica: una organización sin ánimo de lucro con una jerarquía. Pero en los primeros años 90 un nuevo conjunto de experimentos empezaría a cuestionar el aspecto de dicha entidad. Las historias de Linux y Apache muestran cómo estas empresas a un tiempo dependían del trabajo de la FSF y se desmarcaban de la tradición jerárquica que ella representaba con el fin de innovar en formas de coordinación socio-técnicas similarmente situadas.

El editor de texto EMACS todavía es ampliamente usado y está portado prácticamente a cualquier sistema operativo concebible. La controversia con Unipress se ha desvanecido en la distancia, a medida que Stallman y la FSF se enfrentaban a controversias más recientes e intensas, pero la GPL se ha convertido en la licencia legal más ampliamente adoptada y minuciosamente escrutada. Más importante aún, la controversia sobre EMACS no fue ni mucho menos la única que erupcionó en la vida de los programadores de software; de hecho, involucrarse en tales debates se ha convertido virtualmente en un rito de paso para los jóvenes *geeks*, dado que es la única manera en que los detalles técnicos y legales a los que hacen frente pueden explorarse con la precisión requerida. No todos estos debates abocan a la reescritura completa del código fuente y hoy en día muchos de ellos conciernen al intento de convencer o evangelizar en pro de la publicación del código fuente mediante una licencia de software libre. La controversia acerca de EMACS fue de alguna manera una escena primigenia —y traumática, por supuesto— que determinó el resultado de muchas pugnas ulteriores al dar forma a la licencia del software libre y a sus usos.

VII. COORDINACIÓN DE COLABORACIONES

El componente final del software libre es la coordinación. Para muchos participantes y observadores, esta es la innovación central y la trascendencia esencial del código abierto: la posibilidad de incitar a un número potencialmente enorme de voluntarios a trabajar gratuitamente en un proyecto de software, aprovechando la ley de los grandes números, «la producción entre pares», «las economías del don» y las «economías sociales autoorganizadas».¹ La coordinación en el software libre es de un tipo distinto de la que surgió en la década de los 90, directamente de las discusiones sobre compartición de código fuente, concepción de sistemas abiertos y redacción de licencias de *copyright* —todas ellas precursoras necesarias de las prácticas de coordinación. Los relatos en torno a estas cuestiones hallan continuidad en los referidos al *kernel* (núcleo) del sistema operativo Linux, al servidor web Apache y a los sistemas SCM (*Source Code Management*, Gestión del Código Fuente). En conjunto, estos relatos revelan cómo funcionaba y qué forma adquiría la coordinación en los 90.

La coordinación es importante porque desmonta y resuelve la distinción entre formas técnicas y sociales en un significativo todo para

1. La investigación sobre coordinación en el software libre conforma el núcleo central de la literatura académica reciente. Dos de las obras más ampliamente leídas, «Coase's Penguin», de Yochai Benkler, y *The Success of Open Source*, de Steven Weber, se orientan hacia cuestiones de investigación clásicas sobre acción colectiva. Las obras «Cooking Pot Markets», de Rishab Ghosh, y *The Cathedral and the Bazaar* [ed. cast.: «La catedral y el bazar»], de Eric Raymond, establecen muchos de los términos del debate. El artículo de Josh Lerner y Jean Tirole «Some Simple Economics of Open Source» supuso una contribución temprana al mismo. Otras obras importantes sobre el tema son Feller *et al.*, *Perspectives on Free and Open Source Software*; Tuomi, *Networks of Innovation*; Von Hippel, *Democratizing Innovation*.

los participantes. Por un lado está la coordinación y gestión de las personas; por el otro la coordinación del código fuente y los parches, correcciones, informes de errores, versiones y distribuciones —pero en conjunto hay una significativa práctica tecnosocial de gestión, toma de decisiones y rendición de cuentas que conduce a la producción colaborativa de aplicaciones y redes complejas. Semejante coordinación no sería nada excepcional, sino esencialmente un remedio de prácticas corporativas de larga data en el campo de la ingeniería, excepto por un hecho clave: no tiene objetivos. La coordinación en el software libre privilegia la *adaptabilidad* sobre la *planificación*. Las implicaciones de esto van más allá del mero permiso para cualquier tipo de modificación: la estructura de la coordinación del software libre realmente prioriza una apertura generalizada al cambio por encima del seguimiento de planes, metas o ideales compartidos que vengan dictados o controlados por una jerarquía de individuos.²

Adaptabilidad no significa azar o anarquía, sin embargo; es una forma muy específica de resolver la tensión entre la curiosidad individual y el virtuosismo de los *hackers*, por un lado, y la coordinación colectiva necesaria para crear y usar aplicaciones y redes complejas, por el otro. Ningún hombre es una isla, pero ningún archipiélago es una nación, por así decirlo. La adaptabilidad preserva el «goce» y la «diversión» de programar sin sacrificar la cuidadosa ingeniería de un producto estable. Linux y Apache deben entenderse como los *resultados* de este tipo de coordinación: experimentos con la adaptabilidad que han funcionado, para sorpresa de muchos que venían insistiendo en que la complejidad requiere planificación y jerarquía. Las metas y la planificación son el dominio del gobierno —la práctica de fijar objetivos, orientar y definir el control—, pero la adaptabilidad es el dominio de la crítica [*critique*], y es por esto que el software libre es un público recursivo: se sitúa al margen del poder y ofrece una crítica potente en forma de alternativas operativas. No es el dominio de lo nuevo —después de todo Linux no

2. Sobre la distinción entre adaptabilidad y adaptación, véase Federico Iannacci, «The Linux Managing Model»: <http://opensource.mit.edu/papers/iannacci2.pdf>. Matt Ratto caracteriza la actividad de los desarrolladores del *kernel* de Linux como una «cultura de la re-elaboración» y un «diseño para el re-diseño», y capta los exquisitos detalles de tal práctica tanto en la codificación como en el debate entre desarrolladores, una actividad que él denomina «la presión de la apertura» que «da pie a una contradicción entre la necesidad de mantener una actividad colaborativa productiva y la necesidad simultánea de permanecer abiertos a nuevas direcciones de desarrollo» («The Pressure of Openness», pp. 112-138).

deja de ser una reescritura de UNIX—, sino el dominio de la dirección pública crítica y adaptativa de una iniciativa colectiva.

Linux y Apache son más que fragmentos de código, constituyen organizaciones de un tipo insólito. Mi afirmación de que son «públicos recursivos» es útil en tanto que da nombre a una práctica que no es ni corporativa ni académica, ni lucrativa ni no lucrativa, ni gubernamental ni no gubernamental. El concepto de público recursivo incluye dentro del espectro de la actividad política la creación, modificación y mantenimiento de software, redes y documentos legales. Mientras que en la mayoría de teorías un «público» es un conjunto de personas y un discurso que da forma expresiva a algún interés, el «público recursivo» pretende sugerir que los *geeks* no solo dan forma expresiva a un conjunto de intereses (p.ej., que el *software* debería ser libre o que los derechos de propiedad intelectual son demasiado expansivos) sino que también dan forma *infraestructural* concreta a los medios de la expresión misma. Linux y Apache son herramientas de creación de redes por las que pueden garantizarse expresiones de nuevos tipos y por las que puede proseguirse la experimentación infraestructural. Para los *geeks*, el *hackeo* y la programación son variantes de la libertad de expresión y de la libertad de reunión.

De UNIX a Minix a Linux

Linux y Apache son dos casos paradigmáticos del software libre en los 90, tanto para los *hackers* como para los estudiosos del software libre. Linux es el núcleo de un sistema operativo similar a UNIX, arrancado de modo autosostenido a partir del sistema operativo Minix creado por Andrew Tanenbaum.³ Apache es la continuación del proyecto original del NCSA (*National Center for Supercomputing Applications*, Centro Nacional para Aplicaciones de Supercomputación) para crear

3. A menudo se designa a Linux como un sistema operativo, a lo que Stallman se opone según la teoría de que un núcleo es solo una parte de un sistema operativo. Stallman sugiere referirse al conjunto como GNU/Linux para reflejar el uso de herramientas del sistema operativo GNU en combinación con el *kernel* Linux. Este ardid no demasiado sutil para apuntarse el tanto de Linux revela la complejidad de las distinciones. El núcleo está en el corazón de cientos de «distribuciones» diferentes —como Debian, Red Hat, SuSe y Ubuntu Linux—, todas las cuales usan herramientas de GNU pero a menudo son recopilaciones de programas más grandes que un simple sistema operativo. Todos los implicados parecen tener un sentido intuitivo de lo que es un sistema operativo (gracias al éxito pedagógico de UNIX), pero pocos pueden trazar una línea de separación firme en torno al objeto mismo.

un servidor web (el programa original de Rob McCool, llamado httpd), arrancado de modo autosostenido a partir de un grupo distribuido de personas que usaban y mejoraban dicho software.

Tanto Linux como Apache representan experimentos de coordinación, pues ambos desarrollaron sistemas de toma de decisiones mediante experimentación: un sistema de votación en el caso de Apache y una jerarquía estructurada de administradores, con Linus Torvalds como dictador benevolente,⁴ en el caso de Linux. Ambos proyectos exploraron también novedosas herramientas técnicas de coordinación, especialmente aplicaciones de control del código fuente como cvs (*Concurrent Versioning System*, Sistema de Versiones Concurrentes). Y ambos proyectos son también citados a modo de ejemplos de cómo la «diversión», el «goce» o el interés determinan la participación individual y de cómo es posible mantener y alentar la participación y el apoyo mutuo en vez de restringir el foco o eliminar posibles vías de participación.

Más allá de estos experimentos específicos, abordo en detalle los relatos sobre Linux y Apache porque ambos proyectos fueron activamente centrales en la construcción y expansión de la Internet de los 90 al permitir que un número masivo de sitios tanto corporativos como no corporativos instalasen y ejecutasen servidores en Internet de forma barata. De ser Linux y Apache meros proyectos de aficionados con unos cuantos miles de cacharreadores, en lugar de los componentes técnicos nucleares de una emergente red planetaria, probablemente no representarían el mismo tipo de transformación revolucionaria que acabaría etiquetándose como «movimiento» en el periodo 1998-99.

La creación por Linus Torvalds del *kernel* de Linux es frecuentemente citada como el primer ejemplo del verdadero modelo de desarrollo de «Código Abierto», convirtiéndose rápidamente en el más estudiado de todos los proyectos de software libre⁵. A raíz de su aparición a finales de

4. Véase nota 18 del Capítulo VI. [N. del E.]

5. Eric Raymond dirigió su atención a Linux de modo primordial en *The Cathedral and the Bazaar* [ed. cast.: «La catedral y el bazar»]. Muchos otros proyectos precedieron al *kernel* de Torvalds, no obstante, incluyendo las herramientas que conforman el núcleo tanto de UNIX como de Internet: la implementación de Paul Vixie del DNS (*Domain Name System*, Sistema de Nombres de Dominio) conocida como BIND; el sendmail de Eric Allman para enrutar los correos electrónicos; los lenguajes de guiones perl (creado por Larry Wall), python (por Guido von Rossum), y tcl/tk (por John Ousterhout); el proyecto de investigación de X Windows en el MIT; y las obras derivadas de los originales BSD UNIX, FreeBSD y OpenBSD. Sobre el modelo del FreeBSD, véase Jorgensen, «Putting It All in the Trunk» e «Incremental and Decentralized

1991, Linux pasó rápidamente de ser un pequeño núcleo que apenas funcionaba a un reemplazo completamente funcional de los diversos sistemas UNIX comerciales que habían surgido de las guerras de UNIX de los 80. Linux ha llegado a ser lo bastante versátil como para emplearse en PCs de escritorio con muy poca memoria y pequeñas CPU, así como en «clusters» («conglomerados de ordenadores») que proporcionan una potencia de procesamiento paralelo masiva.

Cuando Torvalds dio sus primeros pasos, fue bendecido con una ávida audiencia de *hackers* deseosa de ver un sistema UNIX ejecutándose en PCs, así como con un estilo personal de estimulación que produjo una respuesta enormemente positiva. A Torvalds se le atribuye frecuentemente la creación, a través de su «estilo de gestión», de una «nueva generación» de software libre —una más joven que la de Stallman y Raymond. Linus y Linux no son de hecho la causa de este cambio, sino el resultado de estar en el sitio adecuado en el momento adecuado y de reunir una serie de componentes ya existentes. De hecho, el título de la reflexión semi-autobiográfica de Torvalds sobre Linux —*Just for Fun: The Story of an Accidental Revolutionary* («Solo por diversión: la historia de un revolucionario accidental»)— capta bien parte del carácter de esta génesis.

La «diversión» aludida en el título refleja el privilegio de la adaptabilidad sobre la planificación. Los proyectos, herramientas, personas o códigos que eran divertidos eran aquellos que no venían dictados por normas e ideas preexistentes. La diversión, para los *geeks*, estaba asociada con la repentina disponibilidad, especialmente para estudiantes universitarios y *hackers* aficionados, de un mundo subterráneo en rápida expansión hecho de redes y de software —especialmente Usenet e Internet, pero también redes, entornos en línea y juegos específicamente universitarios, así como herramientas para navegar por información de todo tipo. Buena parte de esta actividad se daba sin el beneficio de ninguna teorización explícita, con la posible excepción del discurso sobre «comunidad» (que vio la luz en letra de molde en 1993 por

Integration in FreeBSD». La historia de la génesis de Linux está muy bien contada en Moody, *Rebel Code*, y Williams, *Free as in Freedom*; existen también numerosos artículos—disponibles a través de la Free/OpenSource Research Community, <http://freesoftware.mit.edu/>—que analizan la dinámica de desarrollo del núcleo de Linux. Véase especialmente Ratto, «Embedded Technical Expression» y «The Pressure of Openness». Buena parte de mi análisis sobre Linux lo he desarrollado leyendo los archivos de la Linux Kernel Mailing List (Lista de Distribución del Kernel de Linux): <http://lkml.org>. También existen sumarios anotados de las discusiones de dicha lista en <http://kerneltraffic.org>.

medio de Howard Rheingold y estuvo presente de forma incipiente en las páginas de *Wired* y *Mondo 2000*) que se desarrolló durante la mayor parte de los 90.⁶ Los últimos 80 y primeros 90 dieron pie a una vasta experimentación con las posibilidades colaborativas de Internet en cuanto medio. Algo especialmente atractivo de tal medio era que se había construido con herramientas libremente disponibles, las cuales a su vez estaban abiertas a la modificación y reutilización creativa. Semejante estilo reflejaba el entorno cuasiacadémico y cuasicomercial de su concepción, del que UNIX representaba un arquetipo —ni investigación pura divorciada del contexto comercial, ni enteramente el dominio de la rapacidad comercial y la propiedad intelectual.

La diversión incluía la creación de listas de distribución gracias a la difusión de software como list-serv y majordomo; la supervisión y el mantenimiento colaborativos de Usenet; la creación de MUDs (*Multi-User Dungeons*, Mazmorras Multiusuario) y MOOs (*Mud Object Orienteds*, MUDs orientados a objetos), que ofrecían a los aficionados a los videojuegos y a los geeks de Internet una forma de co-creación de entornos informáticos y de descubrimiento de muchos de los problemas de gestión y supervisión que surgían con su uso.⁷ También incluía la creciente panoplia de «servicios de información» que se construían sobre Internet, como archie, gopher, Veronica, WAIS, ftp, IRC —todos ellos necesarios para acceder a la creciente riqueza informativa de la comunidad subterránea que merodeaba por Internet. El sentido y la práctica de la coordinación en cada uno de estos proyectos estaban al alcance de cualquiera: algunos se organizaban estrictamente como proyectos universitarios de investigación (gopher), mientras que otros eran más fluidos y estaban abiertos a la participación e incluso al control de los miembros colaboradores (los MOOs y MUDs). Los aspectos relativos a las licencias eran explícitos en algunos casos, poco claros en otros y totalmente ignorados en el resto. Algunos proyectos tenían líderes autocráticos, mientras que otros experimentaban con todo el espectro que va de la democracia representativa al anarquismo.

Durante este periodo (aproximadamente de 1987 a 1993), la FSF alcanzó un estatus de culto mítico —primordialmente entre usuarios

6. Howard Rheingold, *The Virtual Community*. Sobre la prehistoria de este periodo y la localización cultural de algunas aspectos clave, véase Turner, *From Counterculture to Cyberculture*.

7. Las obras «A Rape in Cyberspace», de Julian Dibbell, y *Life on the Screen*, de Sherry Turkle, son dos ejemplos clásicos de las detalladas formas de vida y de creación ética colaborativa que preocupaban a los moradores de estos mundos.

de UNIX y EMACS. Parte de este estatus se debía a la superioridad de las herramientas que Stallman y sus colaboradores ya habían creado: el gcc (*Gnu C Compiler*, Compilador Gnu de C), GNU EMACS, el gdb (*GNU Debugger*, Depurador GNU), GNU Bison, así como montones de aplicaciones más pequeñas que reemplazaban a las versiones originales del UNIX de AT&T. La GNU GPL también había adquirido vida propia para entonces, ya madura como licencia y convertida en la opción *de facto* para aquellos comprometidos con el software libre y la FSF. Llegado 1991, no obstante, los rumores de la inminente aparición del reemplazo de Stallman para el sistema operativo UNIX habían empezando a sonar vacuos —habiéndose pasado ya seis años desde el anuncio público de su intención. En su mayoría los *hackers* eran escépticos respecto del proyecto de Stallman, por más que reconocieran el éxito de todas las otras herramientas necesarias para crear un sistema operativo plenamente funcional, y el propio Stallman estaba estancado en el desarrollo de un componente específico: el núcleo en sí, llamado GNU Hurd.

El proyecto de Linus Torvalds no se imaginó inicialmente como una contribución a la FSF: era el proyecto nocturno de un estudiante universitario de Helsinki para aprender todos los pormenores del relativamente nuevo microprocesador Intel 386/486. Torvalds, al igual que decenas de miles de estudiantes de ingeniería informática, estaba recibiendo clases de UNIX a través de la pedagogía del Minix de Andrew Tanenbaum, del Xinu-PC de Douglas Comer y de un puñado de otras versiones didácticas del estilo diseñadas para ejecutarse en los PCs de IBM. En paralelo a esta pedagogía de aula de los 80 vino la inevitable conexión a, merodeo y publicación en las listas de distribución de Usenet y Arpanet dedicadas a temás técnicos (y no técnicos) de todo tipo.⁸ Torvalds estaba suscrito, naturalmente, a comp.os.minix, el grupo de noticias de usuarios de Minix.

La inserción pedagógica de Linus Torvalds en el mundo de UNIX, Minix, la FSF y Usenet no debería ser subestimada, como es habitual en los relatos hagiográficos del sistema operativo Linux. Sin la relativa robustez de esta infraestructura u orden moral-técnico en cuyo seno era

8. El flujo anual de estudiantes a Usenet y Arpanet en septiembre le valió a este mes el título de «el mes más largo», debido a la necesidad de entrenar a los nuevos usuarios en los usos y etiqueta de los grupos de noticias. Ya entrada la década de los 90, cuando AOL permitió que los suscriptores accedieran a la jerarquía de Usenet, ello vino a conocerse como «septiembre eterno». Véase «September that Never Ended», *Jargon File*, <http://catb.org/~esr/jargon/html/S/September-that-never-ended.html>.

possible estar en el lugar adecuado en el momento adecuado, el proyecto desarrollado por Torvalds en sus noches de residencia universitaria habría llegado poco más lejos que eso —pero todas las piezas estaban dispuestas de modo que sus modestas metas se transformasen en algo mucho más significativo.

Consideremos su anuncio del 25 de agosto de 1991:

—Hola a todos los que estáis ahí fuera usando minix. Estoy haciendo un sistema operativo (libre) (solamente por afición, no será grande ni profesional como el gnu) para clones 386(486) AT. Este ha estado gestándose desde abril, y está comenzando a estar listo. Me gustaría recibir cualquier comentario sobre las cosas que gustan/disgustan en minix, ya que mi SO (Sistema Operativo) se le parece un poco (la misma disposición física del sistema de ficheros, debido a motivos prácticos, entre otras cosas). Actualmente he portado bash(1.08) y gcc(1.40), y las cosas parecen funcionar. Esto implica que conseguiré algo práctico dentro de unos meses, y me gustaría saber qué características quiere la mayoría de la gente. Cualquier sugerencia es bienvenida, pero no prometo que las vaya a implementar :-) Linus...

PS. —Sí. Está libre de cualquier código de minix, y tiene un sistema de ficheros multi-hilo. NO es portable (usa el cambio de tareas del 386, etc.), y probablemente nunca soporte otra cosa que no sean los discos duros AT, porque es todo lo que tengo :-(.⁹

El anuncio de Torvalds señala dónde encaja su proyecto dentro del contexto existente por entonces: «solamente por afición», nada «grande ni profesional como el gnu» (un comentario que sugiere la talla que habían alcanzado Stallman y la FSF, especialmente considerando que en realidad ambos eran cualquier cosa menos «grandes y profesionales»). Dicho anuncio se publicó en la lista de Minix y por ende se dirigía esencialmente a usuarios de dicho sistema operativo; pero Torvalds también se cuida en insistir que el sistema sería libre de coste, y su postdata indica además que estaría libre de código de Minix, al igual que Minix había sido purgado de código de AT&T.

9. Identificador de mensaje: 1991aug25.205708.9541@klaava.helsinki.fi. [Traducción al castellano extraída de Wikipedia: https://es.wikipedia.org/wiki/Historia_de_Linux]

Torvalds también menciona que ha portado «bash» y «gcc», programas creados y distribuidos por la FSF y herramientas esenciales para interactuar con el ordenador y compilar nuevas versiones del *kernel*. La decisión de Torvalds de usar estas aplicaciones en vez de escribir las suyas propias refleja tanto los límites de su proyecto (un núcleo de sistema operativo) como su satisfacción con la disponibilidad y reusabilidad del software distribuido con GPL.

Así pues, su sistema se basa en Minix tal y como Minix se había basado en UNIX —por superposición o arranque autosostenido antes que por una reescritura completamente diferente, esto es, antes que por convertirlo en un tipo diferente de sistema operativo. Y sin embargo hay claramente cierta inquietud acerca de la necesidad de crear algo que no sea Minix, en vez de limitarse a extender o «depurar» Minix. Esta inquietud es clave para interpretar lo que sucedió con Linux en 1991.

Desde su creación en 1984, el Minix de Tanenbaum estuvo siempre orientado a permitir que los estudiantes vieran y cambiaran el código fuente de Minix para así aprender cómo funcionaba un sistema operativo, pero no era software libre. Minix tenía *copyright* y era propiedad de Prentice Hall, que distribuía los libros de texto donde venía incluido. Tanenbaum planteó el alegato —similar al de Gosling respecto a Unipress— de que Prentice Hall estaba distribuyendo el sistema de modo mucho más amplio que si estuviera disponible solo en Internet:

Un argumento que no creo que todo el mundo aprecie es que hacer algo disponible por FTP no es necesariamente la vía para lograr su distribución más amplia. Internet sigue siendo un grupo altamente elitista. La mayoría de ordenadores NO está conectada a ella. [...] MINIX también es ampliamente usado en Europa del Este, Japón, Israel, Sudamérica, etc. La mayoría de esta gente nunca lo habría conseguido si no hubiera existido una compañía que lo vendiera.¹⁰

Según todos los indicios, Prentice Hall no era restrictiva en su política de sublicencias del sistema operativo si alguien quería crear una versión «mejorada» de Minix. De modo similar, la frecuente presencia de Tanenbaum en comp.os.minix atestiguaba su compromiso con la compartición de su conocimiento sobre el sistema con cualquiera que lo deseara —no solo con clientes. No obstante, el uso acentuado

10. Identificador de mensaje: 12595@star.cs.vu.nl.

por Torvalds del término *libre* y su decisión de no reutilizar nada del código de Minix supone un claro indicador de su deseo de construir un sistema completamente desembarazado de restricciones, basándose quizá en el sentido folclórico intuitivo de los peligros asociados a casos como el de EMACS.¹¹

Con todo, el aspecto más trascendente del mensaje inicial de Torvalds es su solicitud: «Me gustaría saber qué características quiere la mayoría de la gente. Cualquier sugerencia es bienvenida, pero no prometo que las vaya a implementar». El anuncio de Torvalds y el subsiguiente interés que generó revelan claramente las cuestiones de coordinación y organización que llegarían a ser características de Linux. La razón de que Torvalds consiguiera tantos colaboradores apasionados para Linux, desde el primer momento, era que los estaba liberando entusiásticamente de las manos de Tanenbaum.

Diseño y adaptabilidad

El papel de Tanenbaum en la historia de Linux es habitualmente el de hombre de paja —un viejo y malhumorado profesor de informática que se opone al joven revolucionario Torvalds. El propio Tanenbaum tenía cierta reputación revolucionaria, debida a que Minix se usaba en aulas de todo el mundo y podía instalarse en los PCs de IBM (algo que ningún otro proveedor comercial de UNIX había conseguido), pero a la vez era un blanco natural para gente como Torvalds: el profesor titular que patrocinaba la versión de manual de un sistema operativo. Así que a pesar de que un gran número de personas usaba o conocía Minix como un sistema operativo UNIX (el número estimado de suscriptores al grupo comp.os.minix era de 40 000), Tanenbaum mostraba un desinterés enfático en la colaboración o la depuración colaborativa, espe-

11. De hecho, inicialmente las condiciones de Torvalds para la distribución de Linux fueron más restrictivas que las de la GPL, incluyendo limitaciones a su distribución cobrando una cuota o los costes de tramitación. Torvalds acabó suavizando las restricciones y se pasó a la GPL en febrero de 1992. Las notas de distribución de Torvalds para Linux 0.12 declaran: «El *copyright* de Linux cambiará: He recibido un par de solicitudes para hacerlo compatible con el *copyleft* de GNU, eliminando la condición ‘No puedes distribuirlo a cambio de dinero’. Estoy de acuerdo con esto. Propongo que se cambie el *copyright* para ajustarlo a GNU —quedando pendiente de aprobación por las personas que han ayudado a escribir el código. Supongo que esto no supondrá un problema para nadie: Si tenéis quejas (‘Yo escribí ese código asumiendo que el *copyright* se mantendría igual’), escribidme un correo. De lo contrario, el *copyleft* de GNU entra en vigor el 1 de febrero. Si no conocéis la esencia del *copyright* de GNU —leedlo» (<http://www.kernel.org/pub/linux/kernel/Historic/old-versions/RELNOTES-0.12>).

cialmente si tal depuración suponía también la creación de extensiones y la adición de características que harían el sistema más grande y difícil de usar como una herramienta didáctica elemental. Para Tanenbaum este aspecto era crucial:

Continuamente me han llegado ofertas de memoria virtual, paginación de memoria, enlaces simbólicos, sistemas de ventanas y toda suerte de funcionalidades. Habitualmente las he declinado porque sigo intentando mantener el sistema lo bastante simple como para que los estudiantes lo comprendan. Puedes meter todas estas cosas en tu versión, pero yo no las pondré en la mía. Creo que es esta cuestión la que irrita a la gente que dice «Minix no es libre», y no su precio de 60 dólares.¹²

Así pues, por más que Tanenbaum compartiera las metas de la FSF (en tanto que claramente quería que la gente pudiera usar, actualizar, mejorar y aprender de los programas informáticos), no compartía la idea de tener a 40 000 desconocidos haciendo «mejoras» en su software. O por expresarlo de otro modo, las metas de Minix siguieron siendo las de un investigador y autor de libros de texto: mantener su utilidad en el aula y su precio lo bastante asequible como para ser ampliamente accesible y usable en el mayor número de ordenadores baratos.

Por contra, el proyecto «divertido» de Torvalds carecía de metas. Siendo un petulante estudiante de 19 años con apenas nada mejor que hacer (libros de texto que escribir, estudiantes, becas, proyectos de investigación o reuniones a las que acudir), Torvalds estaba deseoso de aceptar toda la ayuda disponible que pudiera encontrar para mejorar su proyecto. Y con 40 000 usuarios de Minix disponía de un grupo de colaboradores más o menos instantáneo. En contraste, la audiencia de que disponía Stallman para su EMACS a comienzos de los 80 se limitaba a unas cien computadoras distintas, lo cual podía traducirse en miles, pero desde luego no decenas de miles, de usuarios. La labor de Tanenbaum de crear una generación de estudiantes que no solo comprendían las interioridades de un sistema operativo sino, más específicamente, las del sistema operativo UNIX creó una enorme reserva de *hackers* de UNIX competentes y entusiastas. Fue esta labor de portar UNIX no solo a diversas máquinas sino también a una generación de mentes la que

12. Identificador de mensaje: 12667@star.cs.vu.nl.

preparó el escenario para este evento —y ello supone un componente esencial, aunque a menudo desestimado, del éxito de Linux.

Muchos relatos de la historia de Linux se centran en la lucha entre Torvalds y Tanenbaum, una lucha librada en comp.os.minix con el asunto: «Linux está obsoleto». ¹³ Tanenbaum alegaba que Torvalds estaba reinventando la rueda al escribir un sistema operativo que, desde la perspectiva de la programación de última generación, estaba ya obsoleto. Torvalds, por el contrario, afirmaba que era mejor hacer algo rápido y sucio¹⁴ que funcionase, invitar a colaboradores y luego preocuparse por hacerlo de última generación. Lejos de ilustrar una suerte de conservadurismo caduco por parte de Tanenbaum, el debate resalta la distinción entre las formas de coordinación y los significados de la colaboración. Para Tanenbaum, las metas de Minix eran, bien pedagógicas, bien académicas: enseñar lo esencial de un sistema operativo o explorar nuevas posibilidades en el diseño de sistemas operativos. Según este modelo, Linux no hacía ni una cosa ni la otra: no podía usarse en clase porque rápidamente se volvería demasiado complejo y repleto de funcionalidades para ser enseñado; y tampoco ampliaba las fronteras de la investigación por ser un sistema operativo obsoleto. Torvalds, por el contrario, carecía de metas. Lo que impulsaba su progreso era un compromiso con la diversión y con una noción en gran medida inarticulada de lo que les interesaba a él y a otros, definida desde el principio casi enteramente en contra de Minix y otros sistemas operativos libres, como FreeBSD. En este sentido, Linux solo pudo surgir del contexto —que fijó los límites de su diseño— de UNIX, los sistemas abiertos, Minix, GNU y BSD.

Tanto Tanenbaum como Torvalds operaban según un modelo de coordinación en el que una sola persona era la responsable última de todo el proyecto: Tanenbaum supervisó Minix y garantizó que permaneciera fiel a sus objetivos de servir a una audiencia pedagógica; y Tor-

13. Identificador de mensaje: 12595@star.cs.vu.nl. Las partes clave de la controversia fueron reimprimidas en Dibona *et al.*, *Open Sources*.

14. Cabe apuntar, remontándonos a los controvertidos orígenes de Microsoft, que «rápido y sucio» eran las mismas dos características que daban nombre a QDOS (*Quick and Dirty Operating System*; luego renombrado 86-DOS para su venta comercial). La historia dice que Microsoft, al verse incapaz de cubrir el acuerdo suscrito con IBM para suministrarle el sistema operativo de los PCs que el Gigante Azul lanzaría en 1981, compró QDOS a la empresa Seattle Computer Products por solo 50 000 dólares para luego revendérselo a IBM por un precio mucho mayor, ya convenientemente rebautizado como MS-DOS (*MicroSoft Disk-Operating System*, Sistema Operativo de Disco de Microsoft). [N. del E.]

valds supervisaría Linux, aunque él incorporaría tantas funcionalidades diferentes como quisieran o pudieran aportar los usuarios. Muy rápidamente —gracias a su reserva de 40 000 colaboradores potenciales— Torvalds se vería en la misma posición en la que estaba Tanenbaum, es decir, obligado a tomar decisiones sobre las metas de Linux y sobre qué mejoras incorporar y cuáles no. Lo que hace la historia de Linux tan interesante para los observadores es que aparentemente Torvalds no tomó ninguna decisión: aceptó casi todo.

Las metas y planes de Tanenbaum para Minix eran claros y formados de manera autocrática. Al fin y al cabo, el control, la jerarquía y la restricción son adecuados para el aula. Sin embargo, Torvalds quería hacer más: quería continuar aprendiendo y probando alternativas, y siendo Minix la única vía ampliamente accesible para hacerlo, su decisión de iniciar un camino separado empieza a tener sentido, siendo además que claramente no estaba solo en su deseo de explorar y extender lo que había aprendido. No obstante, Torvalds se encontró con el problema de coordinar un proyecto nuevo y de tomar decisiones similares sobre su rumbo. En este punto, Linux ha sido objeto de mucha reflexión desde una perspectiva tanto interna como externa. A pesar de las imágenes de Linux, bien como un bazar anárquico, bien como una dictadura autocrática, la realidad es más sutil: incluye una jerarquía de colaboradores, mantenedores y «lugartenientes de confianza», y un sentido sofisticado, informal e intuitivo de «buen gusto» adquirido a través de la lectura e incorporación del trabajo de los co-desarrolladores.

Aunque para Torvalds fue posible llevar individualmente las riendas durante los primeros años de Linux (aproximadamente entre 1991 y 1995), con el tiempo empezó a delegar parte de ese control en personas que tomarían decisiones acerca de los diferentes subcomponentes del *kernel*. De este modo, distribuyendo parte del trabajo de evaluación de los «parches» (fragmentos de código) aportados por voluntarios entre gente distinta de Torvalds, se hizo posible incorporar más de ellos. Esta jerarquía informal se transformó poco a poco en una formal, como indica Steven Weber:

La «concesión» final de autoridad *de facto* llegó cuando Torvalds comenzó a redireccionar públicamente los envíos relevantes a sus lugartenientes. En 1996 la estructura de decisión se volvió más formal con una explícita diferenciación entre ‘desarrolladores acreditados’

y «mantenedores». [...] Si esto suena mucho a una estructura de decisión jerárquica es porque es justamente eso —aunque una en la que la participación es estrictamente voluntaria.¹⁵

Casi todas las decisiones de Torvalds y sus lugartenientes eran de una sola clase: incorporar o no un fragmento de código remitido por un voluntario. Cada una de estas decisiones era técnicamente compleja: insertar el código, recompilar el *kernel*, probar si funciona o si genera errores, decidir si vale la pena mantenerlo, lanzar una nueva versión con un registro de los cambios aplicados. A pesar de que los diversos líderes oficiales disponían de la autoridad para efectuar tales cambios, la coordinación seguía siendo técnicamente informal. Dado que todos estaban trabajando sobre un mismo objeto técnico complejo, era necesario en última instancia que una persona (Torvalds) verificase la versión final, que contenía todos los subcomponentes, para garantizar que funcionaba sin fallos.

Semejantes decisiones tenían muy poco que ver con ningún tipo de objetivos o planes de diseño, tan solo con el hecho de si el parche remitido «funcionaba», un término que refleja criterios a un tiempo técnicos, estéticos, legales y de diseño que no están explícitamente registrados en ninguna parte del proyecto —de ahí el privilegio de la adaptabilidad sobre la planificación. En ningún momento se asignaron o solicitaron tales parches, aunque Torvalds es justamente famoso por animar a las personas a trabajar sobre problemas particulares, pero solo si ellas querían. A resultas de ello, el sistema mutó de modos sencillos e imprevistos, divergiendo de su original y supuestamente retrógrado diseño «monolítico» para adoptar una configuración novedosa que reflejaba los intereses de los voluntarios y los criterios implícitos de los líderes.

Hacia 1995-1996, Torvalds y sus lugartenientes afrontaban considerables desafíos en relación con la jerarquía y la toma de decisiones, al haber crecido el proyecto en tamaño y complejidad. La primera respuesta ampliamente recordada a la crisis en marcha en la dictadura benevolente de Linux fue la creación de los «módulos cargables del núcleo», concebidos como un modo de liberar parte de la presión constante para decidir qué parches se incorporarían al *kernel*. La decisión de modularizar Linux era simultáneamente técnica y social: la base

15. Steven Weber, *The Success of Open Source*, p. 164.

de código del software se reescribiría para que los módulos cargables externos se insertaran «sobre la marcha», en lugar de ser compilados conjuntamente en un gran fragmento binario; al mismo tiempo, ello suponía que la responsabilidad de garantizar que los módulos funcionaran se transfería a los creadores de los módulos. La decisión repudiaba la inicial oposición de Torvalds a Tanenbaum en el debate «monolítico vs. Micronúcleo» mediante la invitación a que los colaboradores separasen las funciones centrales y periféricas de un sistema operativo (por más que el núcleo de Linux permanezca monolítico en comparación con los micronúcleos clásicos). Al mismo tiempo, con ello se permitía una significativa proliferación de nuevas ideas y proyectos relacionados. En suma, esta decisión entrañó a la vez una contracción y una distribución de la jerarquía: ahora Linus estaba al cargo de un proyecto más limitado, pero más gente podía trabajar con él de acuerdo con unas normas técnicas y sociales de responsabilidad bien estructuradas.

La creación de dichos módulos cargables cambió la apariencia de Linux, pero no debido a ninguna planificación o decisión de diseño establecida de antemano. Esta opción es un ejemplo de la adaptabilidad privilegiada de Linux, que resuelve la tensión entre la curiosidad y el virtuosismo de los colaboradores individuales del proyecto y la necesidad de control jerárquico con el fin de gestionar la complejidad. La aspiración de adaptabilidad disuelve la distinción entre los medios técnicos de coordinación y los medios sociales de gestión. Se trata de producir un todo significativo con el que poder coordinar tanto a personas como a código —un logro vigorosamente defendido por los *hackers* del *kernel*.

La organización y estructura adaptables de Linux es descrita a menudo en términos evolutivos, como algo carente de propósito teleológico que responde a un entorno. De hecho, el propio Torvalds manifiesta una debilidad por este tipo de explicación.

Seamos honestos y admitamos que [Linux] no fue diseñado.

Sin duda, también hay algo de diseño —el diseño de UNIX creó un andamiaje para el sistema y, lo que es más importante, facilitó que la gente se comunicase porque ya tenía un *modelo* mental sobre cómo era el sistema, lo que supone que es mucho más fácil discutir los cambios.

Pero eso es como decir que sabes que vas a construir un coche con cuatro ruedas y faros delanteros —es cierto, pero la verdadera jodienda está en los detalles.

Y yo sé mejor que la mayoría que lo que imaginé hace diez años no tiene *nada* en común con lo que Linux es hoy. No hubo ciertamente ningún diseño premeditado en ello.¹⁶

La adaptabilidad no responde a las preguntas sobre el diseño inteligente. Por ejemplo, ¿por qué un coche tiene cuatro ruedas y dos faros delanteros? A menudo estos debates están polarizados: o bien los objetos técnicos remiten a un diseño, o bien son el resultado de mutaciones aleatorias. Lo que esta oposición pasa por alto es el hecho de que el diseño y la coordinación de la colaboración van de la mano, y que cada uno revela los límites y posibilidades del otro. Linux representa un ejemplo particular de tal problemática —uno que se ha convertido en el caso paradigmático del software libre—, pero ha habido muchos otros, incluyendo UNIX, para el que los ingenieros crearon un sistema que reflejaba la colaboración distribuida de usuarios de todo el mundo por más que los abogados intentaran que se aviniera a las normas legales sobre licencias y a los intereses prácticos sobre contabilidad y asistencia técnica.

Por privilegiar la adaptabilidad sobre la planificación, Linux es un público recursivo: sistemas operativos y sistemas sociales. Linux privilegia la apertura hacia nuevas direcciones, a todos los niveles: privilegia el derecho a proponer cambios por medio de su creación efectiva y del intento de convencer a otros para que los usen e incorporen; privilegia el derecho a bifurcar el software en tipos de sistemas nuevos y diferentes. Dado lo que privilegia, Linux acaba evolucionando de modo diferente al de aquellos sistemas cuyos diseño y vida están restringidos por una organización corporativa, por los estrictos principios de diseño de los ingenieros o por las definiciones legales o mercadotécnicas de un producto —en suma, por metas claras. Lo que hace importante esta distinción entre el principio de diseño orientado por objetivos y el principio de la adaptabilidad es su relación con la política. Los objetivos y la planificación son materia de negociación y consenso, o bien de una toma de decisiones autocrática, mientras que la adaptabilidad es el terreno de la crítica. Deberíamos recordar que Linux no es de ningún modo un intento de crear algo radicalmente nuevo, sino una reescritura de un sistema operativo UNIX, como señala

16. Citado en Zack Brown, «Kernel Traffic #146 for 17Dec2001», *Kernel Traffic*, http://www.kerneltraffic.org/kernel-traffic/kt20011217_146.html; también citado en Federico Iannacci, «The Linux Managing Model», <http://opensource.mit.edu/papers/iannacci2.pdf>.

Torvalds, si bien una reescritura que a través de la adaptación puede acabar convirtiéndose en algo nuevo.

Parcheo y Voto

El servidor web Apache y el Apache Group (hoy Apache Software Foundation) aportan un segundo ejemplo revelador del cómo y el porqué de la coordinación en el software libre de los 90. Como en el caso de Linux, el desarrollo del proyecto Apache ilustra cómo se privilegia la adaptabilidad sobre la planificación y, en particular, cómo este privilegio pretende resolver las tensiones existentes entre la curiosidad y el virtuosismo individuales y el control y la toma de decisiones colectivos. Asimismo estamos ante el relato de la progresiva evolución de la coordinación, los mecanismos simultáneamente técnicos y sociales para coordinar personas y código, parches y votos.

El proyecto Apache surgió de un grupo de usuarios del servidor web httpd (HyperText Transmission Protocol Daemon) original creado por Rob McCool en el NCSA, basado en el trabajo del proyecto World Wide Web de Tim Berners-Lee en el CERN. Este había redactado una especificación para la World Wide Web que incluía el lenguaje de marcado HTML, el protocolo de transmisión http y un conjunto de bibliotecas que implementaban el código conocido como libwww, el cual Berners-Lee había cedido al dominio público.¹⁷

El NCSA de la Universidad de Illinois en Urbana-Champaign tomó ambos proyectos www para ulteriormente crear tanto el primer navegador de uso amplio, Mosaic, dirigido por Marc Andreessen, como httpd. Httpd fue de dominio público hasta la versión 1.3. El desarrollo se ralentizó cuando McCool fue captado por Netscape, junto con el equipo que creó Mosaic. A principios de 1994, cuando la World Wide Web había comenzado a expandirse, muchos individuos y grupos ejecutaban servidores Web que usaban httpd; algunos de ellos habían creado extensiones y corregido errores. Entre ellos había desde investigadores universitarios hasta corporaciones como Wired Ventures, que lanzó la versión en línea de su revista (*HotWired.com*) en 1994. La mayoría de usuarios se comunicaba principalmente a través de Usenet, en los grupos de noticias comp.infosystems.www.*, donde compartían ex-

17. Identificador de mensaje: 673c43e160C1a758@sluvca.slu.edu. Véase también Berners-Lee, *Weaving the Web*.

periencias, instrucciones y actualizaciones del mismo modo que otros proyectos de software que se remontaban a los inicios de los grupos de noticias de Usenet y Arpanet.

Cuando el NCSA se mostró incapaz de responder a la mayoría de correcciones y extensiones propuestas, varios de los usuarios más activos de httpd empezaron a comunicarse a través de una lista de distribución llamada new-httpd en 1995. La lista era mantenida por Brian Behlendorf, el administrador web de HotWired, en un servidor que gestionaba llamado hyperreal; y sus participantes eran aquellos que habían depurado httpd, creado extensiones o añadido funcionalidades. La lista era el principal medio de asociación y comunicación para un grupo diverso de personas de distintos lugares del mundo. Durante el año siguiente, los participantes debatieron cuestiones relacionadas con la coordinación, la identidad de y los procesos involucrados en el parcheo del «nuevo» httpd, versión 1.3.¹⁸

Parchear un programa supone una actividad peculiar, similar a la depuración, pero más parecida a una forma de diseño *a posteriori*. El parcheo abarca el espectro de cambios que pueden realizarse: desde corregir agujeros y errores de seguridad y errores que impidan la compilación del software hasta mejorar sus características y rendimiento. Gran parte de los parches que inicialmente unieron a este grupo surgió de las necesidades que cada miembro individual tenía para desarrollar una función del servidor web. Estos parches no respondieron a ninguna decisión de diseño o planificación por parte del NCSA, de McCool o del propio grupo luego formado, pero la mayoría era lo bastante útil como para que todos ganasen con su uso, pues corregían problemas que cualquiera podía encontrarse. A resultas de ello, la necesidad de un lanzamiento coordinado del nuevo httpd era clave para el trabajo del grupo. Inicialmente esta nueva versión del servidor web del NCSA no tenía nombre, pero *apache* era un candidato persistente; el origen en cierto modo apócrifo del nombre proviene de que se trataba, en la expresión inglesa, de «*a patchy server*» («un servidor parcheado»).¹⁹

18. El Apache Group original incluía a Brian Behlendorf, Roy T. Fielding, Rob Harthill, David Robinson, Cliff Skolnick, Randy Terbush, Robert S. Thau, Andrew Wilson, Eric Hagberg, Frank Peters y Nicolas Pioch. La lista de distribución new-httpd acabó convirtiéndose en la lista de desarrolladores de Apache. Los archivos están disponibles en http://mail-archives.apache.org/mod_mbox/httpd-dev/ y citados en lo sucesivo como «Lista de distribución de desarrolladores de Apache», seguido de remitente, asunto, fecha y hora.

19. Para otra versión de la historia, véase Moody, *Rebel Code*, pp. 127-128. La historia oficial hace honor a las tribus indias apache por sus «superiores habilidades en estrategia bélica e

Al comienzo, en febrero y marzo de 1995, el ritmo de trabajo de los diversos miembros de new-*httpd* difería en gran medida, pero en general era extremadamente rápido. Incluso antes de que existiera una versión oficial del nuevo *httpd*, el grupo tuvo que enfrentarse a cuestiones de proceso, como Roy Fielding explicó posteriormente:

Apache empezó con un intento consciente de solventar primero las cuestiones de proceso, antes incluso de empezar el desarrollo, porque estaba claro desde el primer momento que un conjunto de voluntarios geográficamente distribuidos y sin ningún lazo organizativo tradicional requeriría un proceso de desarrollo único para alcanzar decisiones.²⁰

La necesidad de abordar el proceso surgió de modo más o menos orgánico, a medida que el grupo desarrollaba mecanismos para gestionar los distintos parches: asignándoles identificadores, probándolos e incorporándolos «a mano» a la base de código fuente principal. En este proceso, algunos miembros de la lista se hallaban ocasionalmente perdidos, confundidos por el proceso o por la eficiencia de otros miembros, como se ve en este mensaje de Andrew Wilson sobre la gestión de la lista de errores de Cliff Skolnick:

Cliff, [;] puedes concentrarte en obtener una copia actualizada de la lista de errores/mejoras, por favor [?]. Ya he perdido la pista de lo que demonios se supone que está pasando. Además, [;] cuál es el estatus de esta versión de Apache pre-pre-prelanzamiento [?] ¿Se trata de un previo o no es seguro que lo sea? POR ÚLTIMO, ¿es esta cosa pre-pre-etc. la misma cosa en la que se supone que Cliff está trabajando?

Pero qué *oño está pasando? ¡Ay, ay, ay! Andrew Wilson.²¹

inextinguible resistencia». Hay pruebas claramente visibles del interés de los miembros originales por el uso del nombre en los archivos del proyecto Apache. Véase esp. Lista de distribución de desarrolladores de Apache, Robert S. Thau, Subject: The political correctness question..., 22 de abril de 1995, 21:06 EDT.

20. Mockus, Fielding y Herbsleb, «Two Case Studies of Open Source Software Development», p. 3.

21. Lista de distribución de desarrolladores de Apache, Andrew Wilson, Asunto: Re: *httpd* patch B5 updated, 14 de marzo de 1995, 21:49 GMT.

A lo cual Rob Harthill repuso: «Está volviéndose un embrollo. Sigo pensando que deberíamos juntarnos todos para implementar los parches de uno en uno. Al ritmo (y a las horas) al que algunos trabajan, probablemente podríamos sacar adelante un par de parches al día... Si esto resulta aceptable para el resto del grupo, creo que deberíamos ordenar los parches y arrancar un proceso sistemático de discusión, implementación y pruebas».²²

Algunos miembros encontraron este ritmo de trabajo estimulante, mientras que otros apelaron a una ralentización o pausa para estudiar la situación. Cliff Skolnick creó un sistema para gestionar los parches y propuso que los miembros de la lista votasen para así determinar qué parches serían incluidos.²³ Rob Harthill fue el primero en votar.

Aquí tenéis mis votos para la actual lista de parches disponible en:
<http://www.hyperreal.com/httpd/patchgen/list.cgi>

Aplicaré un voto de

-1: Me da problemas

0: Aún no lo he probado (no he conseguido entenderlo o lo que sea)

+1: Lo he probado, me ha gustado y no me da problemas.

[Aquí Harthill proporciona una lista de votos para cada parche.]

Si veís sentido a este sistema de votación, usémoslo para cribar aquello con lo que estemos contentos. Un voto «-1» debería vetar cualquier parche. Parece que somos como 6 o 7 los que comentamos activamente los parches, así que sugeriría que una vez que un parche obtenga un voto de +4 (sin vetos), lo podamos añadir a una alpha.²⁴

Los votos de Harthill inmediatamente instigaron la discusión sobre varios parches, sucesivas votaciones y un debate sobre el proceso (p.ej., cuántos votos o vetos eran necesarios), todo entremezclado en un frenesí de mensajes de correo electrónico. El proceso de votación estaba lejos de ser perfecto, pero permitía cierto consenso sobre lo

22. Lista de distribución de desarrolladores de Apache, Rob Harthill, Subject: Re: httpd patch B5 updated, 14 de marzo de 1995, 15:10 MST.

23. Lista de distribución de desarrolladores de Apache, Cliff Skolnick, Subject: Process (please read), 15 March 1995, 3:11 PST; and Subject: Patch file format, 15 de marzo de 1995, 3:40 PST.

24. Lista de distribución de desarrolladores de Apache, Rob Harthill, Subject: patch list vote, 15 de marzo de 1995, 13:21:24 MST.

que «apache» llegaría a ser, esto es, sobre qué parches se incorporarían a un lanzamiento «oficial» (aunque no demasiado público): el de Apache 0.2, el 18 de marzo de 1995.²⁵ Sin un sistema de votación, el grupo de colaboradores podría haber continuado aplicando parches individualmente, cada uno en su propio contexto, corrigiendo los problemas que a cada cual le aquejaran, pero ignorando aquellos que resultasen irrelevantes o innecesarios en ese contexto. Con un proceso de votación, en contraste, podría emerger la convergencia sobre un nuevo httpd testado y aprobado. A medida que se refinaba el proceso, los miembros buscaron a un voluntario para recabar los votos, abrir y cerrar las votaciones una vez por semana y construir una nueva versión de Apache una vez finalizado el proceso. (Andrew Wilson fue el primer voluntario, ante lo que Cliff Skolnick replicó: «Supongo que la primera votación es votar a Andrew como el responsable de votaciones :-))»²⁶. El proceso de «parcheo y voto» surgido en las primeras etapas de Apache no era enteramente novedoso; muchos colaboradores señalaron que el proyecto FreeBSD usaba un proceso similar, algunos apuntaron la necesidad de un «coordinador de parches» y otros se preocuparon de que «usar parches se pone muy feo muy rápido».²⁷

La trascendencia del sistema de «parcheo y voto» residía en que representaba claramente la tensión entre el virtuosismo de los desarrolladores individuales y un proceso grupal orientado a la creación y el mantenimiento de un programa común. Se trataba de una manera de equilibrar la capacidad derivada de la pericia de cada individuo y el deseo común de ofrecer y promover un servidor web estable, libre de errores y de dominio público. Como Rob Fielding y otros lo describirían en retrospectiva, esta tensión formaba parte de la ventaja de Apache.

Aunque el Apache Group toma decisiones en conjunto, todo el trabajo real del proyecto lo llevan a cabo individuos. El grupo no escribe código, diseña soluciones, documenta productos o provee apoyo técnico a nuestros clientes; eso lo hacen los individuos. El

25. Lista de distribución de desarrolladores de Apache, Rob Harthill, Subject: apache-0.2 on hyperreal, 18 de marzo de 1995, 18:46 MST.

26. Lista de distribución de desarrolladores de Apache, Cliff Skolnick, Subject: Re: patch list vote, 21 de marzo de 1995, 2:47 PST.

27. Lista de distribución de desarrolladores de Apache, Paul Richards, Subject: Re: vote counting, 21 de marzo de 1995, 22:24 GMT.

grupo proporciona un entorno para la colaboración y una excelente prueba de fuego para ideas y código, pero la energía creativa necesaria para solventar un problema particular, rediseñar un fragmento del sistema o corregir un error dado es casi siempre aportada por voluntarios individuales que trabajan por su cuenta, con sus propios propósitos y no a instancias del grupo. Los competidores asumen erróneamente que Apache será incapaz de abordar tareas nuevas o inusuales debido a la percepción de que actuamos como un grupo en lugar de seguir a un único líder. Lo que no aciertan a ver es que, al permanecer abiertos a nuevos colaboradores, el grupo dispone de un suministro ilimitado de ideas innovadoras, y son los individuos decididos a perseguir sus propias ideas quienes constituyen la verdadera fuerza motriz de la innovación.²⁸

Aunque generalmente se presenta la apertura como la clave de las innovaciones de Apache, dicha afirmación es algo engañosa: los parches son solo eso, parches. Cualquier cambio a gran escala en el código no podría lograrse mediante la aplicación de parches, especialmente si cada parche debe someterse a una votación relativamente severa para ser incluido. La única forma de realizar cambios de gran envergadura —especialmente cambios que requieren iteraciones y pruebas para ponerlos a punto— es acometer diversas «ramas» de un proyecto o diferenciar entre lanzamientos internos y externos; en suma, bifurcar el proyecto temporalmente con la esperanza de que en poco tiempo se reintegre en su progenitor estable. Apache se encontró a las primeras de cambio con este problema al aparecer «Shambhala», la reescritura de httpd a cargo de Robert Thau.

Shambhala nunca fue del todo oficial: Thau lo llamaba su servidor «improvisado», considerándolo un proyecto de «garaje». Partió de su intento de reescribir httpd como un servidor que pudiera gestionar y procesar múltiples peticiones al mismo tiempo. En cuanto experimento, se trataba por entero de un proyecto propio de Thau, al que ocasionalmente aludía en la lista new-httpd: «Todavía estoy hackeando Shambala, manteniendo un perfil bajo hasta que funcione lo bastante bien como para hablar de él».²⁹ A mediados de junio de

28. Roy T. Fielding, «Shared Leadership in the Apache Project».

29. Lista de distribución de desarrolladores de Apache, Robert S. Thau, Subject: Re: 0.7.2b, 7 de junio de 1995, 17:27 EDT.

1995, Thau tenía una versión funcional que anunció a la lista, de modo bastante modesto, como «un proyecto de garaje para explorar algunas nuevas direcciones posibles que creí que ‘podría’ ser útil que el grupo siguiera».³⁰ Otro miembro de la lista, Randy Terbush, lo probó y le dedicó críticas entusiastas, y hacia finales de junio había dos usuarios proclamando sus virtudes. Pero dado que en realidad nunca había sido identificado oficialmente como bifurcación, o vía de desarrollo alternativa, ello llevó a Rob Harthill a preguntar: «Entonces, ¿cuál es la situación con respecto a Shambhala y Apache? ¿Aquellos que os habéis pasado allí abandonáis Apache y este proyecto? Si es así, ¿necesitáis una lista separada para debatir sobre Shambhala?».³¹

Harthill había asumido que la base de código de la NCSA estaba «probada y testada» y que Shambhala representaba una escisión, una bifurcación: «La cuestión es ¿deberíamos ir todos en una dirección, continuar como están las cosas, o Shambhala [*sic*] debe seguir su propio camino?».³² Su consulta puso de relieve en detalle el problema de comunicación: que Thau lo había planeado como un recambio «automático» del httpd de la NCSA, y que su intención era convertirlo en el núcleo del servidor Apache, si lograba hacerlo funcionar. A Harthill, que había pasado no poco tiempo trabajando duramente en el parcheo del código del servidor existente, esto no le entusiasmó, y explicitó así las principales cuestiones en juego.

Quizá fueron las frases escogidas por rts [Robert Thau], tales como «proyecto de garaje» y el nombre diferente que le ha puesto, quizás no leí sus correos con suficiente meticulosidad, quizá no eran lo bastante explícitos, lo que sea... Es una vergüenza que nadie de los que usan Shambhala (que debían haberse percatado de lo que sucedía) expusiese estas cuestiones semanas atrás. Solo puedo presumir que rst fue demasiado modesto como para impulsar Shambhala, o al menos una discusión al respecto, de una manera más vigorosa. Recuerdo haberme pronunciando en la línea de «Esto es lo que planeo hacer, detenedme si pensáis que no es buena idea». ¿Por qué

30. Lista de distribución de desarrolladores de Apache, Robert S. Thau, Subject: My Garage Project, 12 de junio de 1995, 21:14 GMT.

31. Lista de distribución de desarrolladores de Apache, Rob Harthill, Subject: Re: Shambhala, 30 de junio de 1995, 9:44 MDT.

32. Lista de distribución de desarrolladores de Apache, Rob Harthill, Subject: Re: Shambhala, 30 de junio de 1995, 14:50 MDT.

demonios nadie dijo nada?... ¿Alguien más tuvo la misma impresión que yo acerca del trabajo de rts? Vamos, gente, si queréis formar parte de este grupo, ¡colaborad!³³

El requerimiento de colaboración de Harthill parece sorprendente en el contexto de una lista de correo y un proyecto creados para facilitarla, pero estamos ante un requerimiento específico: colaborar trazando planes y compartiendo objetivos. En sus palabras está implícita la tensión entre un proyecto con planes y objetivos claros, un diseño global al que todo el mundo contribuye, en contraposición a una plataforma grupal sin metas claras que proporciona a los individuos un marco para probar alternativas. En sus palabras está implícito el espectro entre la depuración de un programa existente con una identidad estable y la reescritura de sus aspectos fundamentales para convertirlo en algo nuevo. El significado de la colaboración se bifurca aquí: de una parte, el privilegio del trabajo autónomo individual que es remitido a una revisión por pares grupal y luego incorporado; de la otra, el privilegio de un conjunto de metas compartidas a las que se subordinan las acciones y la labor de los individuos.³⁴

De hecho, el diseño mismo de Shambhala refleja el primer enfoque centrado en privilegiar la labor individual: al igual que UNIX y EMACS previamente, Shambhala fue diseñado como un sistema modular, del tipo que podría «volver parte del proceso [de parcheo y voto] obsoleto, al permitir que elementos no universalmente aplicables (p.ej., administración de bases de datos), controvertidos o simplemente inacabados sean no obstante ofrecidos como módulos opcionales».³⁵ Semejante diseño separa la plataforma principal de los experimentos individuales que se realizan sobre ella, en lugar de crear un diseño que sea modular en el sentido jerárquico de que cada colaborador se dedique a una sección asignada del proyecto en cuestión. Indudablemente, dicha plataforma principal requiere coordinación, pero las extensiones y modificaciones pueden darse sin necesidad de transformar el proyec-

33. Lista de distribución de desarrolladores de Apache, Rob Harthill, Subject: Re: Shambhala, 30 de junio de 1995, 16:48 GMT.

34. Gabriella Coleman capta esto agudamente en su discusión sobre la tensión entre el virtuosismo individual de los *hackers* y el populismo corporativo de grupos como Apache o, en su ejemplo, la distribución de Linux Debian. Véase Coleman, «The Social Construction of Freedom».

35. Lista de distribución de desarrolladores de Apache, Robert S. Thau, Subject: Re: Shambhala, 1 de julio de 1995, 14:42 EDT.

to completo.³⁶ Shambhala representa un cierto triunfo de la estética del «Cállate y muéstrame el código»:³⁷ la «modestia» de Thau es más bien un reconocimiento de que debería mantenerse callado hasta que «funcione lo bastante bien como para hablar de él», mientras que la respuesta de Harthill expresa la frustración por que nadie había hablado de lo que Thau planeaba antes siquiera de que lo intentara. La consecuencia fue que el trabajo de Harthill parecía vano, viéndose reemplazado por el trabajo de demostración de una dirección superior realizado por un *hacker* más virtuoso.

En el caso de Apache puede verse cómo la coordinación en el software libre no es una mera idea por añadidura o un rasgo necesario del trabajo distribuido, sino que de hecho se sitúa en el núcleo de la producción misma de software, gobernando las normas y formas de vida que determinan qué se entenderá por un buen programa, cómo progresará con respecto a un contexto y un trasfondo y de qué modo se esperará que la gente interactúe en torno al asunto de las decisiones de diseño. El privilegio de la adaptabilidad trae consigo una opción en el modo de colaboración: ello resuelve la tensión entre la creación de software agonística y competitiva, como la de Robert Thau con Shambhala, y la necesidad de coordinación colectiva de la complejidad, como la solicitud de colaboración de Harthill para reducir el trabajo duplicado o innecesario.

Revisar y consignar

Las formas técnicas y sociales que adoptan Linux y Apache son posibilitadas por las herramientas que construyen y usan, desde las de seguimiento de errores y listas de distribución hasta los servidores web y los *kernels* mismos. Una de ellas desempeña un papel muy especial en el surgimiento de estas organizaciones: los sistemas de Gestión del Código Fuente (SCM, por sus siglas en inglés *Source Code Management*). Se trata de herramientas para coordinar a personas y código; permiten que múltiples personas dispersas geográficamente trabajen simultáneamente sobre el mismo objeto, el mismo código fuente, y ello sin la necesidad de una supervisión coordinadora central y sin el riesgo de solaparse unas con otras. La historia de los SCM —especialmente

36. Una explicación ligeramente diferente del papel de la modularidad es discutida en Steven Weber, *The Success of Open Source*, pp. 173-175.

37. Véase nota 20 del Capítulo III. [N. del E.]

en el caso de Linux— ilustra también el problema de la profundidad recursiva: a saber, ¿sigue siendo libre el software libre si se crea con herramientas no libres?

Los SCM, como cvs (*Concurrent Versioning System*, Sistema de Versiones Concurrentes) y Subversion, se han convertido en herramientas extremadamente comunes para los programadores de software libre; es más, es raro encontrar un proyecto, incluso uno realizado individualmente, que no haga uso de estas herramientas. Su función básica es permitir a dos o más programadores trabajar en los mismos archivos al mismo tiempo y proveer retroalimentación sobre los conflictos entre sus modificaciones. Cuando el número de programadores crece considerablemente, una SCM puede convertirse en una herramienta para gestionar la complejidad: lleva un registro de quiénes han «revisado» archivos; permite a los usuarios bloquearlos si desean garantizar que nadie realice cambios al mismo tiempo; puede seguir y mostrar los cambios incompatibles que dos usuarios realicen sobre un mismo archivo; puede usarse para crear bifurcaciones «internas» o «ramas» que acaso sean incompatibles pero que sigan permitiendo a los programadores probar cosas nuevas y, si todo va bien, incorporar luego las ramas al tronco. De formas sofisticadas puede emplearse para «animar» cambios sucesivos de un fragmento de código, con el fin de visualizar su evolución.

Más allá de las funciones de mera coordinación, los SCM también se usan como forma de distribución; generalmente los SCM permiten que cualquiera revise el código, pero restringen quiénes pueden registrar o «consignar» el código. El resultado es que los usuarios pueden disponer de acceso instantáneo a la versión más actualizada de un programa, y los programadores pueden diferenciar entre versiones estables, que tienen pocos errores, e «inestables» o experimentales, que están en construcción y necesitarán la ayuda de usuarios dispuestos a probar y depurar las últimas variantes. Las herramientas SCM automatizan ciertos aspectos de coordinación, no solo reduciendo la labor que conlleva sino abriendo nuevas posibilidades de coordinación.

La genealogía de los SCM puede verse en el ejemplo de la creación de una cinta *diff* por Ken Thompson, quien la usó para distribuir cambios aportados a UNIX. En el punto en que Thompson veía UNIX como un espectro de cambios y el departamento legal de Laboratorios Bell veía una serie de versiones, las herramientas SCM combinan estos dos enfoques gestionando minuciosamente las revisiones, asignando a

cada cambio (a cada *diff*) un nuevo número de versión y guardando el historial de todos esos cambios de modo que puedan deshacerse con precisión con el fin de descubrir cuáles producen errores. Escrito por Douglas McIlroy, «*diff*» es a su vez un programa, una de las afamadas herramientas pequeñas de UNIX que hacen una cosa bien. Así, la aplicación *diff* compara dos archivos, línea a línea, e imprime las diferencias entre ellos en un formato estructurado (mostrando una serie de líneas con códigos que indican cambios, añadidos o supresiones). Dadas dos versiones de un texto, se puede ejecutar *diff* para encontrar las diferencias y realizar los cambios adecuados para sincronizarlos, tarea de otro modo tediosa y, dada la exactitud del código fuente, propensa a errores humanos. Un efecto secundario útil de *diff* (cuando se combina con un editor como *ed* o *EMACS*) es que cuando alguien efectúa un conjunto de cambios en un archivo y ejecuta *diff* tanto en el archivo original como en el modificado, el resultado (que contiene solo los cambios) puede usarse para reconstruir el archivo original a partir del modificado. De este modo *diff* permite una forma inteligente y economizadora de espacio de guardar todos los cambios jamás realizados a un archivo: en lugar de conservar copias completas de cada nueva versión, se guardan solo los cambios. En suma, permite el control de versiones. *diff* —y programas parecidos a él— se convierten en la base para gestionar la complejidad de numerosos programadores trabajando en el mismo texto al mismo tiempo.

Uno de los primeros intentos de formalizar un control de versiones fue el RCS (*Revision Control System*, Sistema de Control de Revisiones) de Walter Tichy, de 1985.³⁸ RCS llevaba un registro de las modificaciones a diferentes archivos usando *diff* y permitía a los programadores ver todos los cambios realizados al mismo. Sin embargo, RCS no podía realmente reflejar la diferencia entre el trabajo de un programador y de otro. En este sentido, todos los cambios eran iguales y cualquier interrogante que pudiera surgir sobre el motivo de una modificación quedaba sin respuesta.

Con el fin de añadir sofisticación a RCS, Dick Grune, de la Vrije Universiteit de Amsterdam, comenzó a escribir guiones [*scripts*] que usaban RCS como un sistema de control de versiones multiusuario accesible por Internet, sistema que acabaría convirtiéndose en el mencionado *Concurrent Versioning System*. *cvs* permitía a múltiples

38. Tichy, «RCS».

usuarios revisar una copia, introducir cambios y luego consignarlos, y comprobaría cuáles eran incompatibles, bien para impedirlos, bien para marcarlos. Finalmente, cvs adquirió máxima utilidad cuando los programadores pudieron usarlo de forma remota para revisar código fuente desde cualquier lugar de Internet. Ello permitió a la gente trabajar a velocidades diferentes, en momentos diferentes y en lugares diferentes, sin necesidad de una persona central encargada de revisar y comparar los cambios. Los cvs crearon una forma de control de versiones descentralizado para colaboraciones a gran escala; los desarrolladores podrían trabajar *offline* en el software, siempre en su versión más actualizada, y aun así seguir trabajando en el mismo objeto.

Tanto el proyecto Apache como el del *kernel* Linux usan SCM. En el caso de Apache el sistema original de parcheo y voto rápidamente empezó a colmar la paciencia, el tiempo y la energía de los participantes a medida que el número de colaboradores y parches comenzó a crecer. Desde el principio mismo del proyecto, el colaborador Paul Richards había urgido al grupo a utilizar cvs. Su dilatada experiencia con el sistema en el proyecto Free-BSD le convencía de que suponía una alternativa mejor que el sistema de parcheo y voto. Del resto de colaboradores, no obstante, pocos tenían una amplia experiencia con él, por lo que no fue hasta más de un año después de que Richards comenzara sus admoniciones que cvs acabó por adoptarse. Sin embargo, cvs no es un simple sustituto de un sistema de parcheo y voto, sino que necesita una organización diferente. Y Richards reconoció las contrapartidas. El sistema de parcheo y voto creaba un nivel muy alto de garantía de calidad y revisión por pares de los parches que la gente enviaba, mientras que el sistema cvs permitía que los individuos realizaran más cambios que podían no alcanzar el mismo nivel de garantía de calidad. Igualmente el sistema cvs permitía ramas —estable, en pruebas, experimental— con diferentes niveles de garantía de calidad, mientras que el sistema de parcheo y voto estaba inherentemente orientado hacia una versión final y estable. Tal y como expuso el caso de Shambhala, bajo el sistema de parcheo y voto las versiones experimentales permanecerían como proyectos de garaje no oficiales, en vez de figurar como ramas oficiales con gente responsable de consignar cambios.

Aunque en general los SCM funcionan bien para gestionar cambios incompatibles, pueden hacerlo solo hasta cierto punto. Y es que permitir a *cualquiera* consignar una modificación podría derivar en un

desastre caótico, tan complicado de desenredar como lo sería sin un SCM. En la práctica, por tanto, la mayoría de proyectos designó a un puñado de personas como dotadas del derecho a «consignar» cambios. El proyecto Apache, por ejemplo, mantuvo su sistema de voto, pero lo convirtió en un modo de votar a «consignatarios» en vez de los parches mismos. De este modo, los consignatarios de confianza —aquellos con el misterioso «buen gusto» o intuición técnica— se convirtieron en los miembros nucleares del grupo.

El *kernel* de Linux también se ha enfrentado con varios problemas en torno a los SCM y la gestión de la responsabilidad que implican. La historia del llamado árbol VGER y la creación de un nuevo SCM llamado Bitkeeper es ejemplar a este respecto.³⁹ Hacia 1997, los desarrolladores de Linux comenzaron a utilizar cvs para gestionar los cambios al código fuente, aunque no sin resistencia. Torvalds aún estaba a cargo de las modificaciones al árbol estable oficial, pero a medida que se incorporaron otros «lugartenientes», la complejidad de las modificaciones al *kernel* creció. Uno de dichos lugartenientes era Dave Miller, que mantenía un «espejo» del árbol estable del núcleo de Linux, el árbol VGER, en un servidor de la Universidad de Rutgers. En septiembre de 1998 estalló una lucha entre los desarrolladores del *kernel* por dos cuestiones relacionadas: una, el hecho de que Torvalds no lograba incorporar (parchear) las contribuciones que le habían reenviado algunas personas, incluyendo sus lugartenientes; y dos, resultado de lo anterior, el repositorio cvs VGER ya no estaba sincronizado con el árbol estable que mantenía Torvalds. La amenaza de que surgieran dos versiones diferentes de Linux estaba servida.

Lo que siguió, esmeradamente recogido por Moody en *Rebel Code*, fue una buena dosis de criterio que culminó en la famosa frase, pronunciada por Larry McVoy, «*Linux does not scale*» («Linux no ajusta su escala»). El significado de esta frase es que la capacidad de Linux para crecer y convertirse en un proyecto aún mayor de complejidad creciente, uno capaz de manejar una miríada de usos y funciones (de ampliarse «a escala»), queda constreñida por el hecho de que solo existe un Linus Torvalds. A decir de todos, Linus era y es excelente en lo que hace —pero solo existe un Linus. El peligro de esta situación es el peligro de una bifurcación. Una bifurcación

39. Véase Steven Weber, *The Success of Open Source*, pp. 117-119; Moody, *Rebel Code*, pp. 172-178. Véase también Shaikh y Cornford, «Version Management Tools».

supondría que una o más versiones nuevas proliferarían bajo nuevos liderazgos, una situación muy parecida a la que se dio con la expansión de UNIX. Tanto las licencias como los SCM están diseñados para facilitar esto, pero solo como último recurso. La bifurcación implica asimismo dilución y confusión —versiones de lo mismo en competencia e incompatibilidades potencialmente inmanejables.

Con todo, la bifurcación nunca llegó, pero tan solo porque Linus se fue de vacaciones y regresó renovado y listo para continuar y mostrarse más receptivo. Pero la crisis había sido real, y llevó a los desarrolladores a valorar nuevas formas de coordinación. Larry McVoy se ofreció a crear un nuevo formato de SCM que permitiese una respuesta mucho más flexible al problema que el árbol VGER representaba. Sin embargo, la solución que propuso, llamada Bitkeeper, generaría mucha más controversia de la que precipitó su creación.

McVoy era famoso en los círculos *geek* antes de Linux. En la última etapa de la era de los sistemas abiertos, y mientras trabajaba en Sun, había escrito un importante documento titulado «*The Sourceware Operating System Proposal*». En este informe interno de Sun Microsystems McVoy defendía que la compañía hiciera libremente disponible su versión de UNIX. Se trataba de un intento desesperado por salvar el sueño de los sistemas abiertos, así como de la primera propuesta de «pasarse al código abierto» surgida desde dentro de una compañía, muy parecida a los documentos que en 1998 instarían a Netscape a abrir el código de su software. A pesar de este compromiso temprano, McVoy decidió *no* crear Bitkeeper como un proyecto de software libre, sino como uno cuasiprivativo, lo cual suscitó un interrogante de plena centralidad en términos ideológicos: ¿se puede, o se debería, crear software libre usando herramientas no libres?

A un lado de la controversia se encontraban, naturalmente, Richard Stallman y quienes compartían su visión del software libre. Al otro se ubicaron pragmáticos como Torvalds que afirmaban carecer de objetivos y de compromisos «ideológicos» —solo un compromiso con la «diversión». La tensión puso de manifiesto el modo en que los públicos recursivos negocian y modulan los componentes esenciales del software libre desde dentro. Torvalds realizó una declaración muy contundente y clara acerca de este asunto, respondiendo a las críticas de Stallman sobre el uso de software no libre para crear software libre:

Francamente, no quiero que la gente use Linux por razones ideológicas. Para mí la ideología apesta. Este mundo sería un lugar mucho mejor si la gente tuviera menos ideología y se moviera mucho más diciendo «Hago esto porque es DIVERTIDO y porque puede que otros lo encuentren útil, no porque es mi religión».⁴⁰

Torvalds enfatiza el pragmatismo en lo relativo a la *coordinación*: la herramienta adecuada para una tarea es la herramienta correcta para dicha tarea. En lo relativo a las *licencias*, no obstante, tal pragmatismo no es de aplicación, y Torvalds siempre ha estado firmemente comprometido con la GPL, negándose a permitir la inclusión en el *kernel* de software sin dicha licencia. Este pragmatismo estratégico representa de hecho un reconocimiento de dónde podrían proponerse cambios experimentales, y dónde existen prácticas establecidas. La GPL era un documento estable, la compartición amplia de código fuente era una práctica estable, pero la coordinación de un proyecto usando SCM era, durante este periodo, algo aún fluido, y por ende Bitkeeper era una herramienta que merecía la pena usar mientras siguiera adecuándose al desarrollo de Linux. Torvalds estaba experimentando con el significado de la coordinación: ¿Podría usarse una herramienta no libre para crear software libre?

McVoy, por otro lado, se movía en terreno inestable. Estaba experimentando con el significado de las licencias de software libre, y creó para Bitkeeper tres licencias distintas en un intento de jugar a dos bandas: una licencia comercial para quienes compraran Bitkeeper, otra para quienes lo vendieran y otra para «usuarios gratuitos». Esta última permitía a los desarrolladores de Linux usar el programa gratuitamente —aunque les exigía usar la versión más reciente— y les prohibía trabajar simultáneamente en un proyecto de la competencia. Este intento de McVoy de nadar y guardar la ropa también creó en la comunidad de desarrolladores una enorme tensión, la cual se prolongó desde 2002, cuando Torvalds empezó a usar Bitkeeper en serio, hasta 2005, cuando anunció que dejaría de hacerlo.

Tal tensión provenía de dos fuentes: la primera era el debate entre desarrolladores acerca de la cuestión moral de usar software no libre

40. Linus Torvalds, «Re: [PATCH] Remove Bitkeeper Documentation from Linux Tree,» 20 April 2002, <http://www.uwsg.indiana.edu/hypermail/linux/kernel/0204.2/1018.html>. Quoted in Shaikh and Cornford, «Version Management Tools.»

para crear software libre. Dicha cuestión moral, como siempre, era también técnica, como revelaría la segunda fuente de tensión, las restricciones de la licencia.

El desarrollador Andrew Trigdell, famoso por su labor de ingeniería inversa de un protocolo de red de Microsoft en el marco de un proyecto llamado Samba, lanzó un proyecto para hacer lo propio con Bitkeeper a través del estudio de los metadatos que producía al usarse para el proyecto Linux. Al hacer esto, traspasó una línea establecida por las condiciones de la licencia experimental de McVoy: la licencia «gratuita siempre que no me copies». Los abogados aconsejaron a Trigdell guardar silencio sobre el tema al tiempo que Torvalds le reprendía públicamente por «destrucción deliberada» y por un defecto de carácter moral al tratar de aplicar ingeniería inversa a Bitkeeper. Bruce Perens defendió a Trigdell y censuró a Torvalds por su ética aparentemente contradictoria.⁴¹ McVoy nunca demandó a Trigdell, y Bitkeeper se ha mantenido como un renqueante proyecto comercial, puesto que, de modo muy similar a lo sucedido con la controversia de EMACS de 1985, la de Bitkeeper de 2005 acabó con la decisión de Torvalds de crear directamente su propio SCM, llamado git.

La historia del árbol VGER y de Bitkeeper ilustra tensiones comunes en el seno de los públicos recursivos, específicamente la profundidad del significado de *libre*. De un lado está Linux en sí, un proyecto de software libre ejemplar cuya disponibilidad es igualmente libre; del otro, sin embargo, está la capacidad de contribuir a este proceso, el cual está potencialmente constreñido por el uso de Bitkeeper. Siempre que la función de Bitkeeper esté completamente circunscrita —es decir, completamente planificada— puede no haber problema. No obstante, en el momento en que un usuario vea un modo de cambiar o mejorar el proceso, y no solo el *kernel* mismo, entonces las restricciones y limitaciones de Bitkeeper pueden entrar en juego. Aunque no está claro que Bitkeeper realmente impidiera algo, sí lo está el hecho de que los desarrolladores lo reconocieron claramente como un potencial freno a un compromiso generalizado con la adaptabilidad. O por expresarlo en clave de públicos recursivos, solo una *capa* es propiamente abierta, la del *kernel* mismo; la capa subyacente, el proceso de su construcción, no es libre en el mismo sentido. Resulta curioso que Torvalds —por lo

41. Andrew Orlowski, «‘Cool it, Linus’—Bruce Perens», *Register*, 15 de abril de 2005, http://www.theregister.co.uk/2005/04/15/perens_on_torvalds/page2.html.

demás el portavoz de la antiplanificación y la adaptabilidad— adoptara deliberadamente esta forma de restricción, pero de ningún modo resulta sorprendente que tal opción recibiera un rechazo colectivo.

La controversia de Bitkeeper puede interpretarse como una especie de experimento, una modulación de los tipos de licencias aceptables (por McVoy), por un lado, y de las formas aceptables de coordinación (la decisión de Torvalds de usar Bitkeeper), por el otro. El experimento fue un fracaso, pero un fracaso productivo, pues identificó una clase de software no libre cuyo uso no es seguro en el desarrollo de software libre: el SCM que coordina a la gente y el código que ella aporta. En términos de los públicos recursivos, el experimento identificó la profundidad de recursión adecuada. Por más que fuera posible crear software libre usando ciertos tipos de herramientas no libres, los SCM no están entre ellas; tanto el software creado como el software usado para crearlo han de ser libres.⁴²

La controversia de Bitkeeper ilustra una vez más que la adaptabilidad no tiene que ver con la invención radical sino con la crítica (*critique*) y la respuesta. Mientras que el diseño controlado y la planificación jerárquica representan el terreno del gobierno —el control a través de la fijación de objetivos y la orientación de un colectivo o proyecto—, la adaptabilidad privilegia la política propiamente dicha, esto es, la capacidad de criticar los diseños existentes y de proponer alternativas sin restricciones. La tensión entre la fijación de objetivos y la adaptabilidad también forma parte de la dominante ideología de la propiedad intelectual. Según ella, la legislación en esta materia promueve la invención de nuevos productos e ideas, pero restringe la reutilización o transformación de los ya existentes. Con ello la definición de dónde comienza la novedad supone una prueba esencial para dicha legislación. McVoy explicitó esta tensión en sus justificaciones sobre Bitkeeper:

«Puede que Richard [Stallman] quiera considerar el hecho de que el desarrollo de software *nuevo* es extremadamente caro. Él está

42. Debates similares han aparecido regularmente en torno al uso de compiladores no libres, depuradores no libres, entornos de desarrollo no libres, y así sucesivamente. Con todo, hay mucha gente que escribe y promueve software libre que se ejecuta en sistemas operativos como Macintosh y Windows, y además está la distinción entre herramientas y formatos. Así, por ejemplo, el uso de la aplicación privativa Adobe Photoshop para crear iconos está bien siempre que se haga en formatos estándar como PNG o JPG, y no en formatos privativos como GIF o photoshop.

muy orgulloso de su recopilación de software libre, pero se trata de una recopilación de re-implementaciones, sin ideas ni productos profundamente nuevos. [...] ¿Y si el modelo del software libre simplemente no puede asumir los costes de desarrollar nuevas ideas?»⁴³.

La novedad, tanto en el caso de Linux como en la legislación de propiedad intelectual de modo más general, está directamente relacionada con la interacción de la coordinación social y técnica: orientación por objetivos vs. adaptabilidad. El ideal de adaptabilidad promovido por Torvalds sugiere una alternativa radical a la ideología dominante de creación incorporada en los sistemas contemporáneos de propiedad intelectual. Si Linux es «nuevo», lo es a través de la adaptación y coordinación de un gran número de colaboradores que desafían el «diseño» de un sistema operativo desde abajo, y no desde arriba. En contraste, McVoy representa un imaginario moral de diseño en el que es imposible alcanzar la novedad sin inversiones extremadamente costosas en un diseño de arriba abajo, orientado por objetivos y *apolítico* —y es esta actividad la que el sistema de propiedad intelectual tiene proyectado recompensar. Con todo, tanto Linux como Bitkeeper se embarcan en un experimento, el de «figurarse» cuáles son los límites del software libre.

La coordinación es diseño

Muchos relatos populares sobre el software libre pasan muy rápido por los detalles de sus mecanismos para sugerir que es de algún modo inevitable u obvio que el software libre debería funcionar —un sistema autoorganizado y emergente que maneja la complejidad a través de contribuciones distribuidas de cientos de miles de personas. En *The Success of Open Source* Steven Weber señala que cuando la gente se refiere al código abierto como un sistema autoorganizado, usualmente quieren decir algo más parecido a «No comprendo cómo funciona».⁴⁴

Eric Raymond, por ejemplo, sugiere que el software libre es esencialmente el resultado emergente y autoorganizado de la «depuración colaborativa»: «Dados los globos oculares suficientes, todos los errores

43. Citado en Jeremy Andrews, «Interview: Larry McVoy», *Kernel Trap*, 28 de mayo de 2002, <http://Kerneltrap.org/node/222>.

44. Steven Weber, *The Success of Open Source*, p. 132.

saltarán a la vista».⁴⁵ Esta frase implica que el éxito esencial del software libre radica en la labor de depuración distribuida, aislada, y que el diseño y la planificación se dan en otro lugar (cuando un desarrollador «se rasca su comezón» o responde a una necesidad personal). En la superficie, tal distinción parece bastante obvia: diseñar es diseñar, depurar es eliminar errores del software, ¡y *voilà!* —surge el software libre. Llevada al extremo, se trata de una interpretación por la que solo los genios individuales son aptos para la planificación y el diseño, y si las condiciones iniciales se establecen adecuadamente, entonces la sabiduría colectiva llenará los detalles.

Sin embargo, la práctica y el significado reales de la depuración colectiva o colaborativa es increíblemente elástica. A veces depurar implica corregir un error; otras veces implica hacer que el software realice una tarea diferente o nueva. (Un chiste común, a menudo hecho a costa de Microsoft, capta algo de esta elasticidad: cuandoquiera que algo parece no funcionar correctamente, alguien suelta: «Eso es una funcionalidad, no un error»). Algunos programadores ven una decisión de diseño como un error estúpido y se ponen manos a la obra para corregirlo, mientras que otros se limitan a aprender a usar el software tal y como ha sido diseñado. La depuración puede significar algo tan simple como leer el código de otros y ayudarles a entender por qué no funciona; puede significar encontrar errores en los programas de otros; puede significar reproducir errores de forma fiable; puede significar precisar la causa del error en el código fuente; puede significar modificar el código para eliminar el error; o puede significar, llevada al límite, cambiar o recrear el software para que haga algo diferente o mejor⁴⁶. Para los académicos, la depuración puede ser una forma de construirse una carrera: «Encuentra un error. Escribe un artículo. Corrige el error. Escribe un artículo. Repite el proceso».⁴⁷ Para los vendedores de software comercial, en contraste, la depuración es parte de la batería de pruebas orientadas a optimizar un producto.

45. Raymond, *The Cathedral and the Bazaar* [ed. cast.: «La catedral y el bazar»].

46. Gabriella Coleman, en «The Social Construction of Freedom», proporciona un excelente ejemplo de la frustración de un programador con el modo *font-lock* en EMACS, algo que se sitúa a medio camino entre un error y una funcionalidad. La frustración del programador se orienta hacia la estupidez del diseño (y hacia los diseñadores involucrados en él), pero su solución no es una corrección, sino un rodeo —e ilustra cómo la depuración no siempre implica colaboración.

47. Dan Wallach, entrevista, 3 de octubre de 2003.

La coordinación en el software libre trata sobre cómo la adaptabilidad prevalece sobre la planificación. Es un modo de resolver la tensión entre el virtuosismo individual en la creación y el beneficio social en la labor compartida. Si todo el software fuera creado, mantenido y distribuido solo por individuos, la coordinación resultaría superflua y los programas entrarían de hecho en el dominio de la poesía. Pero incluso los casos paradigmáticos de creación virtuosa —el EMACS de Richard Stallman, el UNIX de Ken Thompson y Dennis Ritchie— representan claramente la necesidad de formas creativas de coordinación y la práctica fundamental de reutilizar, reelaborar, reescribir e imitar. UNIX no se creó *de novo*, sino que fue un intento de optimizar y reescribir Multics, a su vez un sistema que evolucionó a partir del Project MAC y de las primeras brumas de los sistemas de tiempo compartido y el *hackeo* informático.⁴⁸ EMACS, por su parte, fue una reelaboración del editor TECO. Ambos ejemplos resultan útiles para entender la evolución de los modos de coordinación y el espectro del diseño y la depuración.

UNIX fue inicialmente portado y compartido por medios que combinaban lo académico y lo comercial, a través de la participación activa de ingenieros informáticos que a un tiempo recibían actualizaciones de Thompson y Ritchie y les enviaban sus propias correcciones. Y todo ello sin que existiera ningún sistema formal para gestionar este proceso. Cuando Thompson se refiere a su interpretación de UNIX como un «espectro» y no una serie de versiones (V1, V2, etc.), la implicación es que el trabajo sobre UNIX era continuo, tanto en el seno de Laboratorios Bell como entre sus usuarios diseminados. El uso por parte de Thompson de la cinta *diff* encapsula el problema esencial de la coordinación: cómo recopilar y redistribuir las modificaciones que los usuarios realizan al sistema.

De modo similar, tanto la distribución de BSD de Bill Joy como la distribución de GOSMACS de James Gosling eran experimentos no corporativos *ad hoc* con la máxima «publicar rápido y a menudo». Estas estrategias de distribución tenían un propósito (más allá de satisfacer la demanda de los programas): la distribución frecuente de parches, correcciones y extensiones facilitaba el incordio de depurar los programas y satisfacía la demanda de nuevas funcionalidades y extensiones por parte de los usuarios (permitiéndoles realizar ambas por sí mismos). De haber seguido Thompson y Ritchie el modelo corporativo

48. En *The Dream Machine* Mitchell Waldrop detalla bien la historia familiar.

convencional de producción de software, sobre ellos habría recaído la responsabilidad de depurar y probar exhaustivamente el software que distribuían, y AT&T o Laboratorios Bell habrían sido asimismo responsables de generar todas las innovaciones y extensiones, basándose en el *marketing* y en la investigación de producto. Semejante enfoque habría sacrificado la adaptabilidad en favor de la planificación, pero el modelo de Thompson y Ritchie era diferente: tanto la extensión como la depuración del software se convirtieron en responsabilidades compartidas de los usuarios y los desarrolladores. La creación de EMACS por parte de Stallman siguió un patrón similar: dado que EMACS era por diseño extensible y pretendía satisfacer una miríada de necesidades imprevistas, sobre sus usuarios recaía la responsabilidad de atender dichas necesidades, y la compartición de sus extensiones y correcciones traía consigo un beneficio social obvio.

La capacidad de ver el desarrollo de software como un espectro implica algo más que el trabajo continuo sobre un producto; supone ver el producto mismo como algo fluido, construido a partir de ideas y productos previos y transformándose y diferenciándose en ideas y productos nuevos. Desde esta perspectiva, la depuración no está separada del diseño, sino que ambas forman parte del espectro de cambios y mejoras cuyos objetivos y dirección vienen gobernados por los mismos usuarios y desarrolladores, así como por los patrones de coordinación que ellos adoptan. Es en el espacio entre depuración y diseño que el software libre halla su nicho.

Conclusión: Experimentos y modulaciones

La coordinación es un componente esencial del software libre, frecuentemente identificada como el componente central. El software libre es el resultado de una complicada historia de experimentación y construcción, y las formas que la coordinación toma en él representan resultados específicos de dicha historia más larga. Tanto Apache como Linux son experimentos —no experimentos científicos *per se* sino experimentos sociales colectivos en los que hay tecnologías y herramientas legales complejas, sistemas de coordinación y gobierno, y órdenes morales y técnicos ya presentes.

El software libre es un sistema experimental, una práctica que cambia con los resultados de nuevos experimentos. No obstante, el privilegio de la adaptabilidad lo convierte en un tipo peculiar de ex-

perimento, uno no dirigido por objetivos, planes o control jerárquico, sino más parecido a lo que John Dewey sugería a lo largo de su obra: la praxis experimental de la ciencia extendida a la organización social del gobierno al servicio de la mejora de las condiciones de libertad. Lo que otorga trascendencia a esta experimentación es la centralidad del software libre —y específicamente de Linux y Apache— para la expansión experimental de Internet. En cuanto infraestructura o medio [*milieu*], Internet está modificando las condiciones de la organización social, la relación del saber con el poder y la orientación de la vida colectiva hacia las formas de gobierno. Por ello puede afirmarse que el software libre constituye el mejor ejemplo de un intento de hacer pública dicha transformación, de asegurarse de que usa las ventajas de la adaptabilidad como crítica para contrarrestar el poder de la planificación como control. El software libre, en cuanto público recursivo, procede por medio de la propuesta *y el ofrecimiento* de alternativas. Sería un poco como la versión kantiana de la ilustración: en tanto que los *geeks* hablan (o *hackean*) como *eruditos*, en un ámbito público, tienen derecho a proponer críticas y modificaciones de cualquier clase; pero en cuanto desisten de tal compromiso, se convierten en empleados privados o sirvientes del soberano, obligados por conciencia y poder a desempeñar los deberes de su puesto dado. Con todo, la constitución de un ámbito público no constituye una actividad universal sino históricamente específica: el software libre confronta la específica infraestructura técnica y legal contemporánea por medio de la cual es posible proponer críticas y ofrecer alternativas. Lo que resulta de ello es un público recursivo compuesto no solo de individuos que gobiernan sus propias acciones sino también de código y conceptos y licencias y formas de coordinación que transforman dichas acciones en formas de vida técnicas concretas y viables que resultan útiles para los habitantes del presente.

PARTE III MODULACIONES

El problema no puede solventarse discutiendo, sino empleando el método experimental, es decir, experimentando y coordinando esfuerzos. Las razones para llevar a cabo el ensayo no son de carácter abstracto u oculto. Residen en la confusión, la incertidumbre y los conflictos que caracterizan el mundo moderno. [E]l objetivo es avanzar, no retroceder, hasta que el método de la inteligencia rija y oriente las relaciones sociales.

JOHN DEWEY, *Liberalismo y Acción Social*: 118

VIII. «SI TRIUNFAMOS, DESAPARECEREMOS»

A principios de 2002, tras años de lectura y aprendizaje sobre código abierto y software libre, por fin tuve fin la oportunidad de cenar con el afamado liberal-libertario, amante de las armas y empresario fundador del código abierto Eric Raymond, autor de «*The Cathedral and the Bazaar*» y otras meditaciones antropológicas *amateur* sobre el software libre. Había venido a la Universidad Rice de Houston a dar una charla a instancias del CITI (*Computer and Information Technology Institute*, Instituto de Tecnología de la Computación y la Información), y mi mente se llenó de visiones de una confrontación mortal entre dos antropólogos frustrados. Me imaginé explicando punto por punto por qué sus referencias a la autoorganización y a la psicología evolutiva eran erróneas, y cómo la larga tradición de antropología económica contradecía básicamente todo lo que tenía que decir acerca del intercambio de dones. Lamentablemente, dos elementos conspiraron contra esta confrontación épica, por más que afectada.

En primer lugar, se daba la circunstancia (tan habitual en los encuentros de *geeks*) de que había solo una mujer presente en la cena; se trataba de una joven, quizás soltera, pero que no era una estudiante sino una *hacker* interesada por el software libre. Raymond fue a sentarse a su lado, se volvió hacia ella y, salvo algunas pausas excepcionales de un minuto, se dedicó a prodigarle toda su atención. En segundo lugar, yo estaba sentado junto a Richard Baraniuk y Brent Hendricks. De repente, Raymond aparecía como el pasado del software libre, sosteniendo los mismos argumentos y usando la misma retórica de sus publicaciones en línea, mientras que Baraniuk y Hendricks representaban su futuro, planteando cuestiones sobre la transformación (la *modulación*) del software libre en algo sorprendente y nuevo.

Baraniuk, profesor de ingeniería eléctrica y especialista en procesamiento digital de señales, y Hendricks, un consumado programador informático, habían iniciado un proyecto llamado Connexions, un «repositorio abierto de materiales educativos». Este proyecto en marcha de extensión de las ideas del software libre a la creación de materiales educativos —en concreto, de libros de texto— me resultó mucho más interesante que el filosofar *amateur* de Raymond.

En apariencia, tanto Rich como Brent estaban igual de entusiasmados por sentarse a mi lado, quizá porque estaba respondiendo a sus preguntas, a diferencia de Raymond, o quizá porque acababan de contratarme en la Universidad Rice, lo que significaba que podíamos plantearnos seriamente entrar en colaboración. Rich y Brent (junto con Jan Odegard, director del CITI y organizador de la cena) mostraron gran interés en conocer lo que yo podía aportar para ayudarles a comprender los aspectos «sociales» de lo que querían hacer con Connexions, y yo, por mi parte, estaba deseoso de enterarme de cómo habían concebido ellos su proyecto inspirado en el software libre: ¿Qué cosas habían dejado igual y cuáles habían cambiado en su propio experimento? Independientemente de lo que quisieran decir con «social» (y unas veces querían decir ético, otra veces legal, otras veces cultural, etc.), tenían claro que había ámbitos de conocimiento en los que se sentían cómodos (programación, gestión de proyectos, enseñanza y un tipo particular de investigación sobre informática e ingeniería eléctrica) y otros en los que no (las «normas» de la vida académica ajenas a sus disciplinas, la legislación de propiedad intelectual, la «cultura»). Aunque intenté explicar la naturaleza de mi propia especialización en teoría social, filosofía, historia e investigación etnográfica, las distinciones académicas eran mucho menos importantes que el hecho de que pudiera hacerles preguntas detalladas e incisivas sobre el proyecto, preguntas que les indicaban que yo debía de tener algo que aportar en los ámbitos que les faltaba por cubrir —en particular, en torno a la cuestión de si Connexions era lo mismo que el software libre y de las implicaciones que se podrían derivar de ello.

Raymond continuó con su cháchara y su cortejo y luego se marchó, quedando difuminada la trascendencia de su charla y de la cena posterior con él. Sin embargo, durante las siguientes semanas mantuve el contacto con Brent y Rich y acabé (sorprendentemente rápido) formando parte de su novedoso experimento.

Después del software libre

Mi no-encuentro con Raymond supone una especie de alegoría: una alegoría de lo que viene después del software libre. El entusiasmo que rodeó aquella cena no tenía que ver tanto con el software libre o el código abierto como con una cierta posibilidad, una especie de impulso genotípico del que el software libre aparecía como un fenotipo fosilizado y Connexions como uno vivo. Rich y Brent eran personas inmersas en la figuración de algo: estaban dedicados a la modulación de las prácticas del software libre. Por *modulación* me refiero a la exploración detallada de las prácticas concretas —el cómo— del software libre con el fin de preguntarse qué es posible cambiar y qué no para mantener algo (*la apertura?*) que nadie puede realmente llegar a tocar. Lo que me atrajo inmediatamente de Connexions fue su relación con el software libre, de una forma no metafórica o ideológica sino concreta, práctica y experimental, una relación que tenía más que ver con el *surgimiento de* que con la *reproducción de* formas. Pero la oposición entre surgimiento y reproducción inmediatamente plantea un interrogante que no difiere mucho del relativo a la identidad de las especies en la evolución: si el software libre deja de ser software, ¿qué es exactamente?

En la parte III me enfrento directamente a esta pregunta. De hecho, para hacerlo hacía falta la parte II, esto es, la descomposición analítica de las prácticas y las historias del software libre. Con el fin de responder a la pregunta «¿Connexions es software libre?» (o viceversa) era necesario repensar el software libre en sí como un experimento técnico colectivo, más que como la expresión de una ideología o cultura. Con todo, tanto la respuesta afirmativa como la negativa meramente llevan al interrogante «¿Qué es el software libre?». ¿Cuál es la *trascendencia cultural* de estas prácticas? El propósito del concepto de público recursivo es desvelar en parte la trascendencia tanto del software libre como de proyectos emergentes como Connexions; es asimismo ayudar a delinear cuándo estos proyectos emergentes se desgajan absolutamente (dejan de ser públicos) y cuándo no, centrándose para ello en el modo en que modulan los cinco componentes que dotan al software libre de su identidad contemporánea.

Connexions modula todos los componentes salvo el del movimiento (de momento no ha surgido ningún movimiento real de «libros de texto libres», por más que el movimiento de «acceso abierto» sea un

primo segundo bastante cercano).¹ Quizá la modulación más compleja ataÑe a la coordinacióN —con modificaciones en las prácticas de coordinacióN y colaboracióN en la creacióN de libros de texto académicos en particular, y más generalmente en la naturaleza de la colaboracióN y la coordinacióN del saber en la ciencia y la academia.

Connexions surgió del software libre y no, como cabría esperar, de la educación, la elaboración de libros de texto, la educación a distancia o cualquiera de las áreas actualmente vinculadas a la pedagogía. Es decir, las personas que se involucraron en el proyecto no llegaron a él intentando abordar un problema destacado de la educación y la enseñanza sino más bien a través de los problemas suscitados por el software libre y del interrogante sobre cómo dichos problemas afectaban a los libros de texto universitarios. De modo similar, un segundo proyecto, Creative Commons, también surgió de una implicación directa en y una exploración del software libre, y no de ningún movimiento legal o compromiso académico con la crítica a la legislación de propiedad intelectual ni, más importante aún, de ningún deseo de transformar la industria del entretenimiento. Ambos proyectos están resueltamente comprometidos con la experimentación de las prácticas dadas del software libre —con la puesta a prueba de sus límites y con su modificación allá donde se pueda— y es esto lo que los hace vibrantes, arriesgados y potencialmente reveladores como casos de estudio de lo que es un público recursivo.

1. En enero de 2005, cuando escribí la primera versión de este análisis, esta afirmación era cierta. En abril de 2006, la Fundación Hewlett había convocado al «movimiento» de Recursos Educativos Abiertos como algo que transformaría la producción y difusión de libros de texto como los creados por Connexions. De hecho, en el informe de Rich Baraniuk para dicha fundación puede leerse en el primer párrafo: «Un movimiento de base está a punto de prender a través del mundo académico. El *movimiento de educación abierta* se basa en un conjunto de intuiciones compartidas por un abanico notablemente amplio de académicos: que el conocimiento debería ser libre y estar abierto a su uso y reutilización; que la colaboración debería facilitarse, y no dificultarse; que la gente debería recibir reconocimiento y prestigio por contribuir a la educación y la investigación; y que los conceptos e ideas están vinculados de modos inusuales y sorprendentes y no de las formas lineales simples que presentan los libros de texto. La educación abierta promete transformar fundamentalmente el modo en que autores, instructores y estudiantes interactúan en todo el mundo» (Baraniuk y King, «Connexions»). (En una linda confirmación de cuán integrada puede volverse la participación en antropología, Baraniuk copió la segunda frase de algo que yo había escrito dos años antes como parte de una descripción de lo que yo creía que Connexions esperaba conseguir). El «movimiento» como tal apenas existe ya, pero su impulso forma claramente parte de las acciones que la Fundación Hewlett aspira a alcanzar.

Por más que ambas iniciativas se ocupen de disciplinas convencionales (los materiales educativos y las producciones culturales), lo hacen entrando en liza de modo divergente, armadas de una inquietud acerca del orden social y moral en que viven, así como de un ansia de transformarlo mediante la modulación del software libre. He aquí lo que liga tales proyectos a través de ámbitos sustantivos, el hecho de verse forzados a ser opositores, no porque quieran serlo (el movimiento llega al final) sino por entrar en los dominios de la educación y la industria cultural como advenedizos. En muchos sentidos se sienten intuitivamente consternados por el *statu quo* existente, y sus organizaciones, herramientas, licencias legales y movimientos se perciben como imaginarios de orden alternativos, especialmente en lo tocante a la libertad creativa y a la continuidad de la existencia de un procomún de saber académico. En la medida en que estos proyectos se mantienen en una relación divergente, propician que aparezca un público recursivo —algo que, en contraste, no interesa en lo más mínimo a la industria editorial y a la del entretenimiento, por obvios motivos financieros y políticos.

Historias de conexión

Me encuentro de nuevo en una cena. Esta vez en una sala de conferencias sin ventanas ubicada quizá en el sótano de un hotel, o acaso en las alturas. Abogados, académicos, activistas, expertos en políticas públicas y gente de la organización se abren paso de forma semientusiasta hacia la comida calentada en las mesas de vapor del hotel. Estoy intentando contar una historia al grupo que se ha reunido —una historia que he oído contar a Rich Baraniuk cien veces—, pero la estoy fastidiando. Rich siempre genera entusiastas miradas de asombro, bombillas que se iluminan por todas partes, un «¡Ajá!» casi imperceptible o un enérgico asentimiento de cabezas. Yo, por contra, la estoy complicando demasiado. Las caras y frentes se arrugan en señal de comprensión fallida, la gente fija su mirada en la comida bañada en salsa de la que está dando cuenta, sopesando la opción de darle otro bocado o de seguir escuchándome complicar un mundo ya de por sí complicado. Yo no haría esto si no fuera por que Rich está en un avión o en un taxi, retrasado por la nieve o por ingenieros, o quizás alojado en un hotel epónimo de otra ciudad. Entretanto, nuestra coorganizadora Laurie Racine se ha convencido de algún modo de que poseo el entusiasmo infantil necesario para cubrir

a Rich. Me siento halagado, pero me falta convicción. Pasados veinte minutos a ella también, y en el momento en que intento responder una pregunta me interrumpe afirmando: «Es realmente necesario que venga Rich. Realmente debería ser él quien cuente la historia».

Milagrosamente, Rich aparece y, antes siquiera de poder decir hola, es apremiado a que cuente su historia como es debido. Suspiro aliviado y rezo para no haber causado ningún daño irreparable y poder así volver a mi papel de personaje serio. Ahora puedo dejar que el superalterno hable por sí mismo. El inconveniente de la observación participante es que te pidan que participes en lo que en primer término habías albergado la esperanza de observar. Conozco bien la historia —la he oído cien veces. Pero lo que oigo, con los oídos afinados para las cuestiones académicas y asombrándome de algunas de las afirmaciones más extrañas que hace Rich, de algún modo no se corresponde con el oído para la ilustración que su encanto ensayado y juvenil transmite a quienes oyen la historia por vez primera. Se trata más bien de un oído sintonizado con las preguntas del porqué: ¿Por qué este proyecto? ¿Por qué ahora? E incluso, de un modo enrevesado, ¿por qué está la gente tan fascinada cuando él cuenta la historia? ¿Cómo podría yo contarla como Rich?

Rich es ingeniero, concretamente especialista en DSP (*Digital Signal Processing*, Procesamiento Digital de Señales), la ciencia de las señales. Según Rich, el DSP está en todas partes: en nuestros teléfonos móviles, nuestros coches, nuestros reproductores de CD y en toda clase de dispositivos. Es una disciplina matemática pero también tiene una vertiente intensamente práctica, y está conectada con toda clase de campos de conocimiento próximos. Es el tipo de disciplina que puede conectar el cálculo, la bioinformática, la física y la música. Las técnicas estadísticas y analíticas provienen de toda clase de investigaciones y van a parar a toda clase de dispositivos interesantes. Así pues, Rich se encuentra a menudo intentando enseñar a sus estudiantes a establecer este tipo de conexiones —a comprender que una transformada de Fourier no es solo un capítulo más en el cálculo, sino una herramienta para manipular señales, ya sea en bioinformática o en música.

Fue en torno a 1998 o 1999 cuando Rich decidió que le había llegado el turno de escribir un libro de texto sobre DSP, así que se dirigió al decano de ingeniería, Sidney Burriss, para contarle su idea. Burriss, también experto en DSP y miembro de toda la vida de la Universidad de Rice, le contestó algo así: «Rich, ¿por qué no te dedicas a algo útil?». Con ello quería decir: ya hay cien libros de texto sobre DSP, ¿por qué

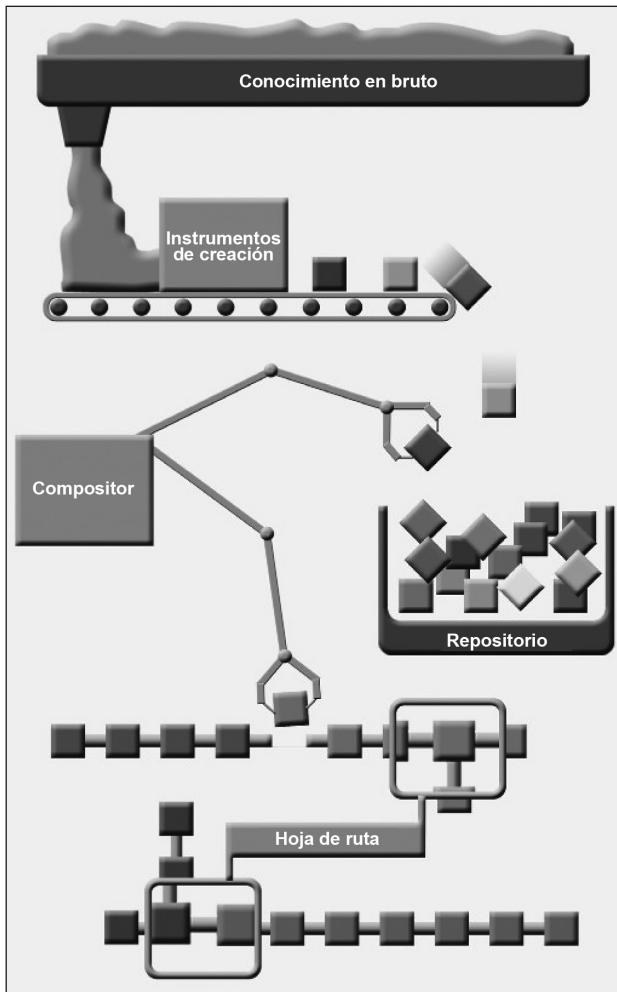
quieres escribir el que hace ciento uno? Burris animó a Rich a hacer algo más grande, algo lo bastante ambicioso como para poner la Universidad de Rice en el mapa. Menciono esto porque es importante señalar que incluso una universidad como Rice, con un profesorado y alumnado a la par de las principales universidades de ingeniería del país, percibe que no es respetada. Burris fue, y sigue siendo, un partidario inveterado de Connexions precisamente porque podría poner a Rice «en los libros de historia» por haber inventado algo verdaderamente novedoso.

Casi a la vez que la idea de Rich sobre el libro de texto, su grupo de investigación se estaba pasando a Linux y Rich empezaba a familiarizarse con el código abierto y la aparición de un sistema operativo completamente libre y creado en su totalidad por voluntarios. No está claro cuándo fue el «¡Ajá!» de Rich, pero el caso es que en un momento dado llegó a comprender que algo como Linux era de verdad posible. Sea como fuere, en cierto punto Rich tuvo la idea de que su libro de texto podría ser de código abierto, es decir, no solo creado por él sino por investigadores en DSP de todo el mundo, y disponible para que cualquiera lo usara, modificara y mejorara a su conveniencia, igual que Linux. Junto con Brent Hendricks, Yan David Erlich y Ross Redstrom, todos los cuales, como buenos *geeks*, estaban profundamente familiarizados con la historia y las prácticas del software libre y de código abierto, Rich empezó a conceptualizar un sistema, pensando acerca de las modulaciones de diferentes componentes del software libre y de código abierto. La idea de un repositorio de libros de texto de software libre empezó lentamente a tomar forma.

Y he aquí que surgió Connexions: un «repositorio de contenido abierto con materiales educativos de alta calidad». Estos «libros de texto» muy rápidamente evolucionaron hacia algo más: «módulos» de contenido, algo nunca definido con precisión, pero que más o menos se corresponde con un pedacito de información enseñable, como dos o tres páginas de un libro de texto. Dichos módulos son mucho más fáciles de imaginar en ciencias como las matemáticas o la biología, donde los libros de texto suelen ser recopilaciones de varios autores, minuciosamente divididas en capítulos cortos con diagramas, ejercicios, teoremas o programas. Los módulos se prestan mucho menos bien a un modelo académico de humanidades o ciencias sociales basado en la lectura, discusión, crítica y comparación de textos —y este sesgo es un reflejo claro de lo que Brent, Ross y Rich conocían mejor en cuanto a enseñanza y escritura. De hecho, el frecuente recurso por parte de

Connexions a la imagen de una cadena de montaje a menudo confirma los peores miedos de los humanistas y educadores cuando se encuentran por primera vez con el proyecto. La imagen sugiere que el conocimiento viene en porciones preenvasadas y coloridamente etiquetadas para deleite de los estudiantes universitarios, en vez de caracterizarlo como una situación o un proceso.

Figura 7



Connexions como una fábrica. Ilustración de Jenn Drummond, Ross Reedstrom, Max Starkenberg y otros, 1999-2004. Reproducida con permiso.

La imagen de la fábrica (figura 7, pág. anterior) es un poco engañosa. El imaginario de Rich y Brent es de hecho mucho más amplio, lo cual evidencian cada vez que realizan una *demo* del proyecto, dan una charla o conversan sobre él en una fiesta. Parte de mi fracaso en transmitir entusiasmo cuando cuento la historia de Connexions se debe a que me salto los ejemplos, que es justo por donde empieza Rich: «Pongamos, dice, que eres estudiante de Cálculo I y, al mismo tiempo, de Introducción a Señales y Sistemas —nadie va a explicarte cómo las transformadas de Fourier forman un puente, o conexión, entre ellas». «Nuestros cerebros no están organizados de manera lineal, por capítulos», dice siempre Rich, «así que, ¿por qué sí lo están nuestros libros de texto?». ¿Cómo podemos dar a los estudiantes un modo de ver la conexión entre estadística y genética, entre arquitectura y biología o entre la ley de propiedad intelectual y la ingeniería química? Rich siempre está en busca de nuevos ejemplos: por ejemplo, una clase de música para niños que usa información de la física, o viceversa. La mayor esperanza de Rich es que cuantos más módulos haya en el pro-común de Connexions, más fantásticas y fascinantes podrían ser las posibilidades para tales conexiones novedosas —y naturales.

El software libre —y el código abierto en especial, a modo de sistemas de coordinación «autoorganizados» y distribuidos— ofrecen su particular promesa para responder a los desafíos de la enseñanza y el aprendizaje que Rich cree que afrontamos. Sin embargo, el compromiso de Rich no es con un cierto tipo de práctica pedagógica sino con los beneficios «sociales» o «comunitarios» derivados de que miles de personas trabajen «juntas» en un libro de texto. De hecho, Connexions no surgió de los ámbitos de la educación o la tecnología educativa, ni tampoco se alineaba con ninguna teoría del aprendizaje concreta (aunque Rich acabaría desarrollando una retórica de conocimiento en red, enlazado y conectado —de ahí el nombre Connexions— que a menudo usa para vender el proyecto). En la Universidad de Rice no hay facultad de educación ni una instancia particular donde encajar tal proyecto (como programas de formación de profesorado o requisitos administrativos sobre educación en línea). Lo que hace aún más difícil la tarea de venta de Rich es que el proyecto surgió casi a la vez que el sonado fracaso de los proyectos alimentados por la burbuja puntocom que buscaban expandir la educación universitaria a modo de educación *online*, educación a distancia y otros sistemas por los que ampliar el cuerpo de estudiantes que pagan su matrícula sin realmente invitarlos

al campus. El mayor de estos experimentos fracasados fue de lejos el proyecto de la Universidad de Columbia, que había alcanzado la fase de implementación en el momento en que estalló la burbuja en 2000.²

Así pues, Rich diseñó Connexions para ir más allá de una fábrica de conocimiento —sería una comunidad o una cultura que desarrollaría tipos de libros de texto profusamente asociativos y novedosos— y mucho más allá de la simple educación a distancia. Es más, Connexions no fue el único proyecto empeñado en desmarcarse de los peligros detectados en la educación a distancia. En abril de 2001, el MIT había anunciado que publicaría el contenido de todos sus cursos de manera gratuita en Internet en un proyecto llamado estratégicamente OCW (*OpenCourseWare*, Materiales de Curso Abiertos). Tal noticia solo podía atraer atención sobre el MIT, que explícitamente posicionó su anuncio como una especie de golpe de gracia a la idea de la educación a distancia, afirmando que sus estudiantes no pagan 35 000 dólares o más al año por «conocimiento» —que es gratis— sino por la experiencia de estar en el MIT. Dicho anuncio generó plenos beneficios desde el punto de vista de la reputación del MIT como generador y diseminador de conocimiento científico, pero el proyecto no surgió directamente de un interés por remediar el éxito del código abierto. Tal perspectiva la aportaría ulteriormente el profesor de informática Hal Abelson, cuya profunda comprensión de la historia y el crecimiento del software libre provenía de su implicación directa en él como miembro de toda la vida de la comunidad informática del MIT. Todo apunta a que OCW surgió del extraño resultado de un informe de comisión, encargado por el rector, sobre cómo debería posicionarse el MIT en el ámbito del «aprendizaje a distancia/electrónico». La sorprendente respuesta fue: No hagan nada, den libre acceso al contenido y en su lugar añadan valor a la experiencia de enseñanza e investigación en el campus.³

Por lo tanto, aunque tanto OCW como Connexions como la educación a distancia estaban ostensiblemente interesados en combinar la educación con las redes y el software, cada uno surgió de demandas diferentes y en lugares distintos. Mientras que la demanda de educación a distancia con fines lucrativos impulsó muchos intentos por todo el

2. Véase Chris Beam, «Fathom.com Shuts Down as Columbia Withdraws», *Columbia Spectator*, 27 de enero de 2003, <http://www.columbiaspectator.com/>. Véase también la ampliamente leída crítica de David Noble «Digital Diploma Mills».

3. «Provost Announces Formation of Council on Educational Technology», *MIT Tech Talk*, 29 de septiembre de 1999, <http://web.mit.edu/newsoffice/1999/council-0929.html>.

país, en el caso del OCW encalló, en gran parte porque el informe final del Consejo de Tecnología Educativa del MIT que recomendaba apostar por OCW se publicó al mismo tiempo que el primer desplome bursátil de abril de 2000. Tales preocupaciones no constituyeron un factor fundamental en el desarrollo de Connexions, lo cual no quiere decir que los problemas de financiación y sostenibilidad no hayan representado siempre preocupaciones importantes, tan solo que la génesis del proyecto no estaba en un nivel administrativo ni respondía a intereses acerca de la educación a distancia. El compromiso principal de Rich, Brent y Ross era con la transparencia y el éxito del código abierto en cuanto experimento de producción de software de modo masivo, distribuido, basado en Internet y colaborativo —su compromiso a este respecto ha sido, desde el primer momento, completamente inquebrantable. Con todo, el proyecto ha implicado modulaciones de las características esenciales del software libre. Tales modulaciones dependen, hasta cierto punto, del hecho de ser un proyecto surgido de las ideas y prácticas del software libre, y no (como es el caso de OCW) fundado a partir de objetivos contrapuestos (beneficios y libertad académica) y resultante en un uso estratégico de las relaciones públicas para incrementar el poder simbólico de la universidad por encima de su crecimiento financiero.

No obstante, cuando Rich relata la historia de Connexions no menciona nada de estos antecedentes. En lugar de ello, y como un buen narrador, espera a que las preguntas surjan por sí solas y deja que sus demostraciones las contesten. Normalmente alguien pregunta: «¿En qué se diferencia Connexions de OCW?». Y Rich siempre responde algo así: «Connexions trata de ‘comunidades’, de cambiar la manera en que los académicos colaboran y crean conocimiento, mientras que OCW no es más que un intento de transferir los cursos ya existentes a un formato web con el fin de hacer su contenido ampliamente accesible. Connexions es un experimento radical en la creación colaborativa de materiales educativos, un experimento que se basa en las ideas del código abierto y que en realidad engloba el proyecto OCW. En retrospectiva queda claro que OCW solo se interesaba en modular el significado del código fuente y la licencia legal, mientras que Connexions busca también modular la práctica de coordinación, en concreto respecto de los libros de texto académicos.

El relato de Rich sobre el origen de Connexions suele proseguir con una demostración del sistema en la que delinea los diversos conceptos técnicos, legales y educativos que lo distinguen. Connexions usa

un formato de documento normalizado, el XML (*eXtensible Mark-up Language*, Lenguaje de Marcado Extensible), y una licencia Creative Commons para cada módulo. Dicha licencia permite no solo copiar y distribuir la información, sino también modificarla e incluso usarla con fines comerciales (asunto que crea repetidas discusiones entre los miembros del equipo). El material abarca desde explicaciones detalladas de conceptos de DSP (naturalmente) hasta cursos de educación musical para todos los niveles preuniversitarios (el conjunto de módulos más popular). Algunos colaboradores han aportado cursos enteros, otros han creado algún módulo de cuando en cuando. Los colaboradores pueden organizar grupos de trabajo para gestionar la creación de módulos e invitar a otros usuarios a unirse a ellos. Connexions emplea un sistema de control de versiones para que todos los cambios queden registrados, de modo que si se modifica el módulo usado en una clase, la persona que lo utiliza para otra clase puede seguir contando con la versión anterior si lo desea. El número de soluciones detalladas e inteligentes incorporadas en el sistema nunca cesa de impresionar ampliamente a cualquiera que se tome el tiempo de echarle un vistazo.

Pero lo que siempre anima a la gente es la idea de una conexión aleatoria y flexible, la idea de que el autor de un libro de texto pueda basarse en el trabajo de cientos de personas que han contribuido previamente, para crear nuevas clases, nuevos módulos y conexiones creativas entre ellos, o yuxtaposiciones sorprendentes —desde biólogos que dan una clase de bioinformática y necesitan recordar a los estudiantes ciertas partes del cálculo sin requerir para ello un curso completo; a arquitectos que quieren un estudio para el análisis de formas biológicas, no necesariamente para realizar experimentos de biología sino para entender los edificios de modo diferente; a maestros de música que desean que su alumnado comprenda lo suficiente de física como para asimilar los conceptos de tono y timbre; a los físicos que necesitan un ejemplo concreto para explicar las ondas y la oscilación.

La idea de estas recombinaciones tan radicales resulta chocante para algunos (más frecuentemente para los académicos de humanidades y ciencias sociales que para los científicos o ingenieros, por razones que tienen que ver claramente con una ideología de habilidad creativa auténtica e individualizada). Las cuestiones derivadas —relativas a derechos de autor, plagio, control, uso no autorizado, abuso, equívocos, malinterpretaciones, difamación, etc.— generalmente aparecen con una fuerza y velocidad asombrosas. Si Rich estuviera intentando vender

una versión de «educación a distancia», el escepticismo y la sospecha rápidamente abrumarían al proyecto; pero siendo lo que es, Connexions invierte casi todas las expectativas que la gente ha desarrollado acerca de los libros de texto, la práctica didáctica, la colaboración y el *copyright*. La mayoría de las veces la gente abandona el debate convertida a la causa —sin duda ayudada por el don de Rich para la narrativa.

Modulaciones: Del software libre a Connexions

Connexions sorprende a la gente por algunas de las mismas razones por las que lo hace el software libre, surgiendo como surge directamente de las mismas prácticas y componentes. El software libre proporciona un patrón hecho de cinco componentes: código fuente compartido, un concepto de apertura, licencias *copyleft*, formas de coordinación y un movimiento o ideología. Connexions arranca con la idea de modular un «código fuente» compartido que no es software sino módulos de libros de texto que los académicos compartirán, portarán y bifurcarán. El experimento resultante posee también implicaciones para los otros cuatro componentes, las cuales llevan a nuevas preguntas, nuevas restricciones y nuevas ideas.

La modulación del código fuente introduce respecto del software libre una diferencia específica que puede generar confusión: Connexions es *tanto* un proyecto convencional de software libre *como* un experimento poco convencional basado en él. Naturalmente, hay mucho código fuente normal, es decir, diversos componentes de software que necesitan combinarse para permitir la creación de documentos digitales (los módulos) y para mostrar, almacenar, transmitir, archivar y medir la creación de módulos. La expectativa es que la creación y la gestión de dicho software funcionen más o menos como cualquier proyecto de software libre: se publica con licencias de software libre, se basa en estándares abiertos de diversa índole y se habilita para recibir contribuciones de otros usuarios y desarrolladores. El sistema para gestionar los módulos se basa él mismo en una variedad de otros componentes de software libre (y en un compromiso de utilizar únicamente dicho software). Connexions ha creado diversos componentes que, bien se publican como software libre convencional, bien se aportan a otros proyectos de software libre. La economía de las contribuciones y los lanzamientos entraña cierta complejidad: en cada decisión hay presentes cuestiones de apoyo y mantenimiento técnicos, así como de repu-

tación y reconocimiento. Y tal y como sucede en cualquier iniciativa de software libre, se invita a otras personas a contribuir al proyecto.⁴

Y al mismo tiempo está el «contenido», el término ubicuo para las creaciones digitales distintas del software. A diferencia de lo anterior, la creación de módulos de contenido (técticamente posible gracias al sistema del software) pretende funcionar *como* un proyecto de software libre donde, por ejemplo, un grupo de profesores de ingeniería pueda reunirse para colaborar en fragmentos de un libro de texto sobre DSP. El proyecto Connexions no engloba ni inicia tales colaboraciones, sino que más bien procede desde el supuesto de que dicha actividad ya está dándose y de que Connexions puede ofrecer una especie de plataforma alternativa —una *infraestructura* alternativa— de la que quienes elaboran libros de texto pueden servirse en lugar de la actual infraestructura editorial. Se da así por hecho que la infraestructura y el modelo técnico actuales de redacción de libros de texto son tales que impiden que la gente saque partido del modelo de desarrollo colaborativo del código abierto y convierten las obras académicas en «no libres». Los objetos de contenido que se comparten no son código fuente compilable, como el código fuente de C, sino documentos marcados con XML y llenos con contenido «educativo» que luego se «muestran», bien en papel, bien en pantalla.

El significado modulado de *código fuente* da lugar a todo tipo de nuevos interrogantes —específicamente respecto de los otros cuatro componentes. Así, por ejemplo, Connexions modula muy poco el componente relativo a la apertura: la mayoría de actores implicados es devota de los ideales de sistemas y estándares abiertos, en la medida en que se trata de un proyecto de software libre de tipo convencional. Connexions se basa en UNIX (Linux) e Internet, y sus líderes mantienen una devoción casi fanática por la apertura a todos los niveles: aplicaciones, lenguajes de programación, estándares, protocolos, lenguajes de marcado y herramientas de interfaz. Allá donde existe una solución abierta (en oposición a una privativa), su elección se impone sobre todas las demás (con una excepción digna de atención).⁵ James Boyle lo expresaba muy bien:

4. El software consiste en una recopilación de diferentes aplicaciones de código abierto improvisadamente combinadas para proporcionar la plataforma básica (los gestores de contenido Zope y Plone, la base de datos PostGresQL, el lenguaje de programación python y el programa de control de versiones cvs).

5. La excepción más significativa ha sido la cuestión de las herramientas para generar contenido en XML. Durante la mayor parte de la vida del proyecto *Connexions*, el lenguaje de

Siempre que sea posible, diseña el sistema para que funcione con contenido abierto, sobre protocolos abiertos, con una disponibilidad potencial para aceptar al mayor número posible de usuarios y la gama más amplia posible de modificaciones experimentales de aquellos usuarios que pueden por sí mismos determinar el desarrollo de la tecnología.⁶

Con respecto al contenido, la devoción por la apertura es casi idéntica, debido a que las editoriales de libros de texto convencionales «confinan» a sus clientes (el alumnado) mediante la creación de nuevas ediciones y de inútiles contenidos «ampliados», lo cual infla los precios y dificulta a los educadores la adaptación o personalización de sus propios cursos. En este sentido, la «apertura» se sirve del mismo razonamiento ya esgrimido en los 80: el aspecto más importante del proyecto es la información que la gente crea, y cualquier sistema privativo clausura el contenido e impide que este se transfiera o use en un contexto diferente.

En realidad, el compromiso con la apertura es tan firme que Rich y Brent a menudo afirman algo así como: «Si triunfamos, desapareceremos». Lo que desean no es convertirse en editores de libros de texto en línea famosos, sino en una *infraestructura* editorial famosa. Ser radicalmente abierto supone que cualquier competidor puede usar tu sistema —pero ello supone que *están usando tu sistema*, y ese es el objetivo. Ser abierto supone no solo compartir el «código fuente» (contenido y módulos) sino concebir formas de asegurar la apertura permanente de dicho contenido, es decir, crear un público recursivo dedicado al mantenimiento y modificabilidad del medio o infraestructura por la que se comunica. La apertura se impone sobre la «sostenibilidad» (esto es, la autoperpetuación de la viabilidad financiera de una organización particular), y allá donde no lo logra, el compromiso con la apertura ha quedado en peligro.

marcado XML ha estado claro y bien definido, pero no ha habido modo de escribir un módulo en XML, a no ser redactando directamente el texto y las etiquetas en un editor de texto. Para todos los posibles usuarios con muy pocas excepciones, esto se parece demasiado a programar y es experimentado como demasiado frustrante para merecer la pena. La solución (si bien temporal) fue alentar a los usuarios a recurrir a un editor de XML privativo (como el procesador word, pero capaz de crear contenido XML). De hecho, la devoción por la apertura del proyecto *Connexions* fue puesta a prueba por una de las decisiones más importantes tomadas por sus participantes: emprender la creación de un editor de texto XML de código abierto con el fin de proveer acceso a herramientas completamente abiertas para la creación de contenido completamente abierto.

6. Boyle, «Mertonianism Unbound», p. 14 [ed. cast.: «¿Mertonismo desencadenado?»].

En consecuencia, el compromiso con la apertura y la modulación del significado del código fuente crea implicaciones con respecto al significado de las licencias de software libre: ¿cubren tales licencias este tipo de contenido? ¿Son necesarias nuevas licencias? ¿Cómo deberían ser en tal caso? Connexions no fue en ningún caso el primer proyecto en estimular estos interrogantes acerca de la aplicabilidad de las licencias de software libre a textos y documentos. En el caso de EMACS y la GPL, por ejemplo, Richard Stallman ya se había enfrentado al problema de licenciar el manual al mismo tiempo que el código fuente del editor. Es más, tales cuestiones acabarían dando lugar a la GNU FDL (*Free Documentation License*, Licencia de Documentación Libre) específicamente pensada para los manuales de software. Debido a su preocupación al respecto, Stallman se enfrentaría durante los 90 con Tim O'Reilly, editor y presidente de O'Reilly Press, que llevaba tiempo publicando libros y manuales de programas de software libre. O'Reilly sostenía que los principios reflejados en las licencias de software libre no deberían aplicarse a los libros didácticos, ya que tales libros proporcionaban un servicio, un modo de que más gente aprenda cómo usar el software libre, y a su vez ampliaban el público de dicho software. Stallman argüía lo contrario: los manuales, al igual que el software al que sirven, necesitaban ser libremente modificables para seguir siendo útiles.

A finales de los 90, después de que el software libre y el código abierto hubieran salpicado los titulares de los medios dominantes, estaban en marcha numerosas tentativas para crear licencias que utilizaban como modelo el software libre pero que podían aplicarse a otros objetos. Una de las primeras y más generales fue la *Open Content License* (Licencia de Contenido Abierto), redactada por el investigador en tecnología educativa David Wiley con la intención de que sirviera para cualquier tipo de contenido: texto, fotografías digitales, películas, música, etc. Dichas licencias suscitan nuevas cuestiones, como por ejemplo: ¿es posible designar ciertas partes de un texto como «invariables» para evitar que sean modificadas y a la vez permitir que se cambien otras (el modelo finalmente adoptado por la GNU FDL)? ¿Cuál podría ser la relación entre el «original» y la versión modificada? ¿Puede esperarse que los autores originales incorporen sin más los cambios sugeridos? ¿Qué tipos de bifurcación son posibles? ¿En qué momento entran en juego los «derechos morales» de los autores (en relación con la «integridad» de una obra)?

Al mismo tiempo, la modulación del código fuente con el fin de incluir los libros de texto académicos posee implicaciones extremadamente complejas para el significado y el contexto de la coordinación: los académicos no redactan libros de texto del mismo modo que los programadores escriben código, así que ¿deberían coordinarse de la misma manera? La coordinación de un libro de texto o de un curso en Connexions requiere experimentos novedosos en la redacción de un libro de texto. ¿Se presta esta tarea a los estilos de trabajo académico, y en qué disciplinas y para qué tipo de proyectos? Para poder sacar provecho de la promesa de la creación distribuida y colaborativa sería necesario encontrar maneras de coordinar a los académicos.

De este modo, cuando Rich y Brent durante la cena reconocieron en mí a alguien que podría saber plantear estos temas, estaban admitiendo que el experimento que habían iniciado había creado cierta turbulencia en su comprensión del software libre y, a su vez, una necesidad de examinar los tipos de prácticas legales, culturales y sociales que estarían en juego.⁷

Modulaciones: De Connexions a Creative Commons

Estoy andando por un aparcamiento a 40 grados y con una humedad del 90%. Es primavera en Houston. Estoy buscando mi coche y no soy capaz de encontrarlo. James Boyle, autor de *Shamans, Software and Spleens* y distinguido profesor de derecho en la Universidad de Duke, me acompaña sin apartar su mirada de mí, sudando bajo su traje de lana y observando cómo busco mi coche bajo el sol abrasador. Su expresión se limita a decir: «Si no te destripo con mi puntero Palm Pilot, me voy a regodear contándole a tus amigos esta humillante historia cada vez que se me presente la oportunidad». Boyle es un hombre paciente con el tipo de humor escocés pícaro que puede hacerte sentir como su mejor amigo aun cuando sus historias acerca de la insensatez de la gente se desplieguen con perfecto tono cómico y acaben versando so-

7. El movimiento es el componente que permanece sin modular: no existe un movimiento de «libros de texto libres» asociado con Connexions, por más que muchos de los mismos debates que llevaron a una separación entre software libre y código abierto se den aquí también: la cuestión de si el término «free» («libre» o «gratis») es confuso, por ejemplo, o sobre el papel de los editores con ánimo de lucro o las empresas de libros de texto. Al final, la mayoría (si bien no todo) del equipo de Connexions y muchos de sus usuarios se contentan con usarlo como una herramienta útil para componer tipos novedosos de material educativo digital —no como un movimiento para la liberación del contenido educativo.

bre ti. Tras haberme reído por el camino con muchos de sus hilarantes relatos de las debilidades de mis semejantes, ahora caigo en la cuenta de que acabo de unirme a ellos en el teatro de las debilidades humanas de Boyle. Presiono repetidamente el botón del pánico de mi llavero, con la esperanza de encontrarme lo bastante cerca de mi coche como para que estalle en un frenesí de bocinazos y destellos que ponga fin a la humillación.

El día había comenzado bien. Boyle se había apretujado dentro de mi Volkswagen (es un tipo alto) y habíamos conducido hasta el campus, aparcado el coche en lo que sin duda parecía un espacio memorable a las 9 de la mañana y caminado alegremente a la reunión prevista —solo para comprobar que por error la habían programado para el día siguiente. Sin ser mi culpa, ahora ciertamente era mi problema. El ostensible propósito de la visita de Boyle era conocer al equipo de Connexions e informarse acerca de lo que hacían. El mismo Boyle había propuesto la visita, ya que tenía previsto pasar por Houston de todos modos. Mi intención había sido atosigarle con preguntas sobre las implicaciones políticas y las posibilidades a la hora de licenciar el contenido de Connexions y con comparaciones con el OCW del MIT y otros proyectos que Boyle conocía.

En vez de asistir a la reunión le llevé de vuelta a mi despacho, donde me enteré mejor de por qué estaba interesado en Connexions. El interés de Boyle no era enteramente altruista (ni tampoco estaba destinado a malgastar valiosos cuartos de hora en un aparcamiento bochornoso mientras yo buscaba mi utilitario). Lo que interesaba a Boyle era encontrar a un cuerpo de potenciales usuarios de Creative Commons, la organización sin ánimo de lucro que estaba creando junto a Larry Lessig, Hal Abelson, Michael Carroll, Eric Eldred y otros —en gran medida porque reconocía la necesidad de contar con un cuerpo de usuarios ya dispuesto para hacer funcionar Creative Commons. Dicho grupo previo era necesario tanto para dotar de legitimidad al proyecto como para permitir a sus fundadores comprender qué se necesitaba exactamente, en términos legales, para la creación de todo un conjunto nuevo de licencias similares a las del software libre.

Como organización y como movimiento, Creative Commons se había estado gestando durante varios años. En cierto modo, este proyecto representaba una simple modulación de la licencia del software libre: una ampliación del concepto de la licencia para cubrir otros tipos de contenidos. Pero el ímpetu que subyacía a esta iniciativa no se reducía

a un deseo de copiar y extender el software libre. En lugar de ello, toda la gente implicada en Creative Commons eran personas que durante muchos años habían estado lidiando con temas tales como la propiedad intelectual, la tecnología de la información y las nociones de procomún, dominio público y libertad de información. Boyle se dio a conocer con un libro sobre la construcción de la sociedad de la información a partir de sus estructuras legales (especialmente la propiedad intelectual). Eldred era un editor de obras de dominio público y el demandante principal en un proceso judicial que se elevó al Tribunal Supremo en 2002 para determinar si la reciente extensión de los límites de vigencia del *copyright* era constitucional. Abelson era un ingeniero informático con un activo interés en temas de privacidad, libertad y legislación sobre «fronteras electrónicas». Y Larry Lessig, originariamente interesado por el derecho constitucional y asistente del juez Richard Posner, era un autoproclamado investigador en ciberderecho que durante los 90 impulsó la explosión de interés por dicho campo, desarrollando gran parte de su tarea en el Centro Berkman de Internet y Sociedad de la Universidad de Harvard.

Con la excepción de Abelson —quien, además de ser un famoso informático, trabajó durante años en el mismo edificio por el que campaba Richard Stallman y presidió el comité que elaboró el informe recomendando el OCW—, ninguno de los miembros de Creative Commons se curtió en proyectos de software libre (eran abogados y activistas, primordialmente), y sin embargo la emergencia a la luz pública del código abierto en 1998 fue un acontecimiento cuyo sentido todos captaron de un modo más o menos instantáneo e intuitivo. Durante ese tiempo Lessig y los miembros del Centro Berkman lanzaron un proyecto de «derecho abierto» diseñado para imitar la colaboración basada en internet del código abierto con aquellos abogados que quisieran contribuir al caso Eldred. Así pues, Creative Commons se construyó tanto sobre un compromiso hacia la noción de creación colaborativa —el uso de Internet especialmente— como, de modo más general, sobre la capacidad de los individuos de trabajar conjuntamente para crear cosas nuevas, y en especial para coordinar dicha creación mediante el uso de novedosos acuerdos de licenciamiento.

Con todo, Creative Commons ofreció algo más que licencias. Formaba parte de un imaginario social de orden moral y técnico que se extendía más allá del software para incluir creaciones de todo tipo. De este modo, las nociones de libertad técnica y moral para hacer uso de

la «cultura» propia se hicieron cada vez más prominentes a medida que Larry Lessig se implicaba en cada vez más batallas con la industria del entretenimiento por el «control de la cultura». Pero para Lessig Creative Commons era un plan b. La ruta directa hacia la transformación de la estructura legal de la propiedad intelectual pasaba por el caso Eldred, un caso que generó un enorme impulso a lo largo de 2001 y 2002 y fue admitido a trámite por el Tribunal Supremo, que fijó la vista para octubre de 2002.

Una de las cosas que hizo destacable el caso fue la serie de extraños compañeros de cama que produjo: entre los economistas y juristas que apoyaron la derogación de la Ley «Sonny Bono» de Extensión del Plazo de *Copyright* estaban los archidefensores del libre mercado y galardonados con el Premio Nobel Milton Friedman, James Buchanan, Kenneth Arrow, Ronald Coase y George Akerlof. Tal y como Boyle señaló en un artículo, conservadores, progresistas y liberal-libertarios tenían todos ellos razones para estar a favor de restringir la expansión del *copyright*.⁸ Lessig y su equipo perdieron el caso y el Supremo ratificó esencialmente la interpretación constitucional del Congreso por la que «por tiempo limitado» simplemente implicaba que el periodo temporal fuera limitado, no que fuera corto.

Creative Commons representaba, pues, un enfoque de «puerta trasera»: si no se podían modificar las leyes, entonces debería facilitarse a la gente las herramientas necesarias para sortearlas. Comprender el modo en que se concibió Creative Commons requiere ver el proyecto como una modulación tanto de la noción de «código fuente» como de las «licencias de *copyright*». Ahora bien, tales modulaciones tuvieron lugar en el contexto de un sistema legal cambiante con el que ni Stallman ni sus usuarios de EMACS estaban familiarizados, un sistema legal que respondía a nuevas formas de software, redes y dispositivos. Por ejemplo, las modificaciones en la Ley de *Copyright* de 1976 crearon un efecto no intencionado del que Creative Commons acabó sacando partido. Al eliminar la exigencia de registrar las obras con *copyright* (esencialmente otorgando el *copyright* desde el momento en que la obra esté «fijada en un medio tangible»), la ley de *copyright* creó una situación en la que no había forma explícita de colocar intencionalmente una obra en el dominio público. En la práctica, un autor podía declarar que una obra suya estaba en el dominio público, pero desde

8. Boyle, «Conservatives and Intellectual Property».

una perspectiva legal el riesgo recaía por entero en quien quisiera hacer uso de dicha obra: copiarla, modificarla, venderla, etc. Con la explosión de interés en Internet, el problema se ramificó exponencialmente: se hizo imposible saber si alguien había subido a la Red un texto, una imagen, una canción o un video destinado a su uso por otros —*incluso si el autor declaraba explícitamente que estaba «en el dominio público»*. Por consiguiente, las licencias Creative Commons se concibieron y se posicionaron retóricamente como herramientas para explicitar exactamente qué usos de una obra específica podrían darse. Protegían los derechos de quienes buscaban hacer uso de la «cultura» (esto es, de materiales, ideas y trabajos cuya autoría era ajena), un enfoque que Lessig a menudo sintetizaba afirmando que «la cultura siempre se basa en el pasado».

Los antecedentes y el contexto del que surge Creative Commons fueron por supuesto mucho más complicados y tensos. Las preocupaciones iban desde los apuros de las bibliotecas universitarias ante los elevados precios de las revistas científicas, al problema de los realizadores de documentales incapaces de costearse los derechos de uso de imágenes o fragmentos de películas, o incluso de encontrar a sus titulares, pasando por las prominentes batallas por la comercialización de la música en línea, Napster y la RIAA. A lo largo de cuatro años, Lessig y los demás fundadores de Creative Commons abordarían todos estos temas en libros, incontables charlas, presentaciones y congresos por todo el mundo, de manera presencial o virtual, y ante audiencias que iban desde desarrolladores de software hasta empresarios, músicos, blogueros y científicos.

A menudo, el argumento a favor de Creative Commons recurre al concepto de cultura asediada por las industrias de contenidos. Una historia que Lessig disfruta contando —y que le he oído en varias ocasiones en que he asistido a sus charlas en congresos— es la de Mickey Mouse. Un rasgo interesante y cuasiconspirativo de la expansión de la legislación de propiedad intelectual en el siglo XX es que los límites de su vigencia parecen haberse extendido en torno al momento en que Mickey Mouse iba a pasar al dominio público. Ciento o no, el argumento que Lessig quiere plantear es que el Ratón no es la creación *ex novo* de la mente de Walt Disney que a la legislación de *copyright* le gusta presentar como tal, sino que se basa en el pasado de la cultura, en particular en Steamboat Willie, Charlie Chaplin, Krazy Kat y otros personajes del estilo, algunos tomados como inspiración, otros como

referencia explícita. La grandeza de la creación de Disney no procede de la mente del autor, sino de la cultura de la que surgió. Lessig a menudo ilustrará todo esto con videos e imágenes entremezclados con diapositivas cubiertas de un tipo de letra negra salido de una máquina de escribir y un estilo oratorio de ametralladora que te hace pensar que, o bien es un poeta *beat* frustrado, o bien se presenta a las elecciones, o quizás ambas cosas.⁹

Los libros y charlas de los defensores de Creative Commons también abundan en otros ejemplos de problemas ligados a la propiedad intelectual, historias sobre innovación bloqueada, creatividad sofocada y —el argumento más aterrador de todos (al menos para los economistas-juristas)— la ineficiencia provocada por la hiperexpansiva legislación de propiedad intelectual y el exceso de celo de las hordas de abogados corporativos.¹⁰ A menudo Lessig predica a los conversos (en lugares como el festival mediático South by Southwest Interactive y los congresos sobre código abierto de O'Reilly), y el público siempre se muestra indignado por el estado de cosas y ansioso por averiguar qué puede hacer al respecto. Con frecuencia la respuesta consiste en colaborar con Creative Commons. De hecho, al cabo de un par de años Creative Commons pasó rápidamente a ser más un movimiento (una modulación del movimiento de software libre/código abierto) que un experimento de redacción de licencias.

Aquella calurosa jornada de mayo de 2002, sin embargo, Creative Commons aún estaba en fase de desarrollo. Avanzado el día, Boyle tuvo por fin ocasión de reunirse con los miembros del proyecto

9. En este punto el paso del tiempo ha dado la razón a Chris Kelty, pues efectivamente Lawrence Lessig acabaría presentando su precandidatura a las primarias del Partido Demócrata para las elecciones presidenciales de EEUU de 2016: <https://lessig2016.us/> (última consulta: 11 de mayo de 2018). Obviamente, no era una candidatura al uso: fiel a cierto estilo *hacker* o, siguiendo a Kelty, *geek*, Lessig se postulaba como «no político» (no ya como independiente) y prometía que, de llegar a la Casa Blanca, se limitaría a implementar «El Plan» (consistente en aprobar una sola medida: la *Citizen Equality Act* o Ley de Igualdad de la Ciudadanía) y a continuación dejar el cargo a mitad de mandato. Y también obviamente (más aún tras las revelaciones de Wikileaks sobre las manipulaciones del Congreso Nacional Demócrata para inclinar la balanza a favor de la candidatura de Hillary Clinton, véase: <https://wikileaks.org/dnc-emails>), esta iniciativa quedó frustrada a los pocos meses de arrancar, como Lessig explicó luego en artículos como «Why I ran for President», publicado el 27 de enero de 2016 en *The New Yorker*: <https://www.newyorker.com/news/news-desk/why-i-ran-for-president> [N. del E.].

10. La producción de Lessig ha sido prodigiosa. Sus libros incluyen *Code and Other Laws of Cyber Space* [ed. cast.: *El código y otras leyes del ciberespacio*], *The Future of Ideas*, *Free Culture* [ed. cast.: *Por una cultura libre*], y *Code: Version 2.0* [ed. cast.: *El código 2.0*]. Asimismo ha escrito un gran número de artículos y entradas de blog (<http://www.lessig.org/blog/>).

Connexions. El equipo de Connexions ya se había percatado de que al llevar a cabo un proyecto experimental donde el software libre se usaba como patrón crearon la necesidad de nuevos tipos de licencia. Ante ello ya habían contactado con la asesoría legal de la Universidad de Rice, que, pese a su buena intención, adolecía por completo de una comprensión profunda del software libre y por ende se mostraba naturalmente suspicaz ante él. La presencia de Boyle y sus detalladas preguntas sobre el proyecto fueron como una revelación —una revelación de que ahí fuera ya había gente pensando precisamente en el problema que afrontaba el equipo de Connexions, y de que no era necesario que ellos lo resolvieron por sí mismos o instaran a la asesoría legal de la Universidad de Rice a redactar nuevas licencias de contenido abierto. Lo que Boyle ofreció fue la posibilidad de que Connexions, y yo mismo como intermediario, nos implicáramos en la planificación detallada y la redacción legal que estaba en marcha en Creative Commons. Al mismo tiempo, ello dotaba a Creative Commons de un «usuario pionero» de la licencia extremadamente voluntarioso y procedente de un rincón importante del mundo: la investigación y docencia académicas.¹¹ Mi tarea, una vez recuperado de la vergüenza de ser incapaz de encontrar mi coche, era organizar un taller en agosto en el que miembros de Creative Commons, Connexions, el OCW del MIT y otros proyectos del estilo serían invitados a hablar acerca de cuestiones de licencias.

Figuración participante

El taller que organicé en agosto de 2002 pretendía permitir que Creative Commons, Connexions y el proyecto OCW del MIT intentaran articular qué podría querer cada uno del otro. Estaba claro lo que Creative Commons quería: convencer a la mayor gente posible de usar sus licencias. Pero no lo estaba tanto lo que Connexions y OCW pudieran querer uno del otro, así como de Creative Commons. Dadas

11. En ese momento eran pocos los proyectos de esta índole en marcha, aunque había muchos en la etapa de planificación. Al cabo de un año, ya se había lanzado la PLoS (*Public Library of Science*, Biblioteca Pública de Ciencias), encabezada por Harold Varmus, el ex-director del estadounidense NIH (*National Institutes of Health*, Institutos Nacionales de Salud). Por entonces, sin embargo, el único proyecto académico de envergadura era el OCW del MIT, que, aunque ya había acordado usar licencias Creative Commons, había demandado una peculiar licencia específica.

las metas y trayectorias diferentes de los dos proyectos, sus necesidades de licencias diferían de modos sustanciales —hasta el punto de que la propia idea de usar la misma licencia fue, al menos temporalmente, considerada imposible por el MIT. Mientras que la principal preocupación de OCW era obtener permisos para colgar en la Web obras protegidas ya existentes, la de Connexions era más bien garantizar que las nuevas obras permanecieran disponibles y modificables.

En retrospectiva, este taller clarificó los novedosos interrogantes y problemas que surgieron del proceso de modulación de los componentes del software libre para diferentes campos, diferentes tipos de contenido y diferentes prácticas de colaboración y compartición. Desde entonces, mi propia implicación en esta actividad se ha orientado a resolver algunas de estas cuestiones en consonancia con una imaginación de apertura, una imaginación de orden social, que había aprendido de mi larga experiencia con *geeks*, y no de mis conocimientos putativos como antropólogo o investigador en estudios de ciencia. La ficción que en un principio había adoptado —que yo ponía sobre la mesa conocimientos académicos— se volvió cada vez más difícil de mantener a medida que me daba más cuenta de que era mi comprensión del software libre, adquirida a través de años y años de aprendizaje etnográfico, la que impulsaba mi implicación.

En efecto, la investigación que describo aquí apenas si fue emprendida como un proyecto de investigación. Me habría sido imposible concebirla como una actividad financiable con antelación a descubrirla; me habría sido imposible imaginar el curso de acontecimientos con el mínimo detalle necesario para redactar una propuesta de investigación como es debido. En lugar de ello, fue fruto de una reflexión y una participación que ya estaban en marcha y venían impulsadas en gran medida por intuición y cierta una sensibilidad hacia el problema representado por el software libre. Yo deseaba ayudar a figurar algo, deseaba ver cómo dicha «figuración» acontece. Por más que podría haber organizado un proyecto de investigación financiable en el que eligiera una iniciativa de software libre madura, articulara una serie de preguntas y dedicara tiempo a responderlas trabajando con ese grupo, semejante proyecto no habría dado respuesta a las preguntas que yo intentaba formular por esa época: ¿Qué le está pasando al software libre a medida que se difunde más allá del mundo de los *hackers* y el software? ¿Cómo está siendo modulado? ¿Qué tipo de límites se rompe cuando el software deja de ser el componente central? ¿Qué otros dominios

de pensamiento y práctica estaban o están «preparados» para recibir y comprender el software libre y sus implicaciones?¹²

Mi experiencia —mi observación participante— con Creative Commons, por tanto, se desarrolló primordialmente en mi papel de intermediario entre el proyecto Connexions (y, por ende, proyectos similares en marcha en otros lugares) y Creative Commons con respecto a la redacción de licencias. En muchos sentidos esta práctica detallada y específica fue el aspecto más desafiante e iluminador de mi participación, pero en retrospectiva representó una especie de pista falsa. No fue solo la modulación del significado del código fuente y de las licencias legales lo que diferenció estos proyectos sino que, de modo más importante, fue el significado de la colaboración, la reutilización, la coordinación y la práctica cultural de compartir y basarse en conocimiento previo lo que planteó los problemas más peliagudos.

Mi contacto en Creative Commons no eran ni James Boyle ni Larry Lessig sino Glenn Otris Brown, el director ejecutivo de la organización (allá por el verano de 2002). Conocí a Glenn por primera vez al teléfono, mientras trataba de explicarle de qué iba Connexions y por qué debería acompañarnos en Houston en agosto para debatir sobre las cuestiones de licencias relativas al material académico. Convencerle para venir a Texas resultó más fácil que explicarle Connexions (dada mi propensión a las complicaciones innecesarias), al ser Glenn nativo de Austin y haber estudiado en la Universidad de Texas antes de partir a la Facultad de Derecho de Harvard y recibir sus corruptoras influencias a manos de Lessig, Charlie Nesson y John Perry Barlow.

12. Puede que el hecho de que yo organizará un taller al que invité a «informantes» y al que ulteriormente me referí como investigación parezca incorrecto a algunas personas, tanto en el campo de la antropología como fuera de él. Pero se trata precisamente del tipo de ocasión que yo sostendría que ha devenido central a la problemática del método en la antropología cultural. Sobre este tema, véase Holmes y Marcus, «Cultures of Expertise and the Management of Globalization». Semejante participación estratégica y aparentemente *ad hoc* no nos excluye de intentar dilucidarla luego con el fin de comentar su valor y trascendencia, y especialmente de ofrecer una crítica al respecto. En esto consiste el intento de conseguir objetividad en ciencias sociales, una objetividad que va más allá de las nociones básicas de sesgo y efecto del observador tan comunes en este campo. La «objetividad» en un sentido social más amplio incluye la observación de los vínculos conceptuales que a un tiempo preceden a un taller como el citado (constituyendo la necesidad de que se organice) y lo suceden, independientemente de cualquier encuentro concreto. La complejidad de movilizar la objetividad en debates sobre el valor y la trascendencia de los fenómenos sociales y económicos fue bien articulada hace un siglo por Max Weber, y los problemas de método en el sentido planteado por él me parecen hoy no menos disputados que entonces. Véase Max Weber, «Objectivity in the Social Sciences.»

Glenn galvanizaba el proyecto. Con su trayectoria previa como abogado, destacando su agudo interés por la legislación de propiedad intelectual, y con su inveterado amor por la música de todo género, Glenn le ponía un increíble entusiasmo a su labor. Antes de unirse a Creative Commons, había sido asistente del Juez Stanley Marcus en el Undécimo Circuito de Apelaciones de Miami, donde trabajó en el llamado «caso Wind Done Gone».¹³ Su participación en el taller fue un experimento de su cosecha, pues por entonces trabajaba en un relato que contaría infinidad de veces y se volvería uno de los ejemplos clave del tipo de práctica que Creative Commons deseaba estimular.

Un artículo del *New York Times* describe cómo la banda The White Stripes había permitido a Steven McDonald, bajista de Redd Kross, incorporar una pista de bajo a las canciones que componían el álbum *White Blood Cells*. En un fragmento que acabaría convertido en una especie de mantra de Creative Commons, el artículo afirmaba:

El señor McDonald empezó a poner estas canciones en línea sin permiso de The White Stripes ni de su compañía discográfica. Durante el proyecto se encontró a Jack White, que le dio permiso verbal para continuar con su idea. Cuando te saltas los intermedios puede ser así de fácil.¹⁴

La facilidad con la que estos dos rockeros pudieron colaborar para crear una obra modificada (llamada, por supuesto, *Redd Blood Cells*) sin pasar por un estudio ni, lo más llamativo, por un bufete de abogados, era emblemática de la noción de que «la cultura se basa en el pasado» y de que dicha colaboración no tenía por qué ser difícil.

Glenn contó la historia con obvio y animado entusiasmo, concluyendo con la aseveración de que The White Stripes no necesitaba

13. *Suntrust v. Houghton Mifflin Co.*, Undécimo Circuito de Apelaciones de EEUU, 2001, 252 F. 3d 1165.

Se trata de un caso derivado de la demanda que los herederos de Margaret Mitchell, autora de la novela *Gone with the wind* (*Lo que el viento se llevó*), interpusieron contra Alice Randall, autora de la obra paródica *Wind done gone* (algo así como *El viento se lo llevó*), en la que se reescribe la historia de la obra original desde el punto de vista de Cynara, una esclava mestiza a la sazón hermanastra de Scarlett O'Hara. Para más detalles sobre el caso en castellano, puede leerse la nota recogida por *El Mundo* el 12 de mayo de 2002: <http://www.elmundo.es/elmundo/2002/05/11/cultura/1021127666.html> [N. del E.]

14. Neil Strauss, «An Uninvited Bassist Takes to the Internet», *New York Times*, 25 de agosto de 2002, sec. 2, 23.

renunciar a todos sus derechos para permitir esto, pero tampoco tenía que mantenerlos todos: en vez del «Todos los Derechos Reservados», sugería Glenn, podrían declarar «Algunos Derechos Reservados». El relato no solo logra captar el mensaje y las metas de Creative Commons, sino que también supone un sutil indicativo del tipo de papel dual que Glenn desempeñaba, en primer lugar como abogado, y en segundo como una especie de genio del *marketing* y emisario. La posibilidad de que solo existiera un puñado de personas al nivel de Glenn no pasó inadvertida a nadie, y su capacidad para intercambiar el lenguaje del Derecho y el del *marketing* populista no lucrativo era fenomenal.¹⁵

Durante el taller, los participantes tuvieron la oportunidad de resolver diferentes cuestiones relativas a la creación de licencias que se adecuaran al contenido académico: cuestiones de atribución y uso comercial, modificación y garantía; diferencias entre la legislación federal de *copyright* concerniente a las licencias y la legislación estatal concerniente a los contratos comerciales. Por más que para muchos el punto de partida fuera el software libre, no era el único. Existían al menos otros dos hilos amplios que sustentaban el debate y la comprensión general del estado de cosas al que se enfrentaban proyectos como Connexions y OCW. El primer hilo era el de las bibliotecas digitales, el hipertexto, la investigación sobre interacción persona-ordenador y la tecnología educativa. Estos proyectos y disciplinas a menudo toman como referencia común a dos pioneros, Douglas Engelbart y Theodore Nelson, y más probablemente a elementos como el programa HyperCard de Apple y a una variedad de experimentos sobre informática personal en la academia. Los debates y la historia que condujeron a la posibilidad de Connexiones son complejos y detallados, pero generalmente adolecen de atención a los detalles legales. Con la excepción de un puñado de gente de la biblioteconomía y las ciencias de la información que ha hecho del *copyright* «digital» una subespecialidad, pocos de esos proyectos han realizado un esfuerzo por entender, no digamos por incorporar, cuestiones de propiedad intelectual en su ámbito de competencia.

15. Es más, en un momento más autorreflexivo, una vez Glenn me escribió entusiasmado para explicarme que lo que él hacía era «intercambio de código» [«code-switching»] y que pensaba que los *geeks* que constantemente se involucraban en proyectos de tecnología, Derecho, música, videojuegos y demás serían casos de estudio excelentes para un estudio sobre intercambio de código por parte de antropólogos.

El otro hilo combina una serie de intereses más académicos que provienen de las disciplinas de la economía y la teoría legal: economía institucional, realismo jurídico crítico, Derecho y Economía —he aquí las designaciones escolásticas. Así, por ejemplo, tanto Boyle como Lessig son académicos: el primero no ha ejercido la abogacía y el segundo ha llevado unos pocos casos. Con todo, ambos son herederos de un pragmatismo legal y filosófico en el que el valor se mide por la transformación de las políticas públicas, no por la mera extensión o especificación de cuestiones conceptuales. Aunque ambos han escrito un gran número de complicados artículos teóricos (y Boyle es célebre en algunos ámbitos académicos por su libro *Shamans, Software and Spleens* y sus obras sobre autoría y legislación), ninguno de ellos, sospecho, sacrificaría la oportunidad de realizar un conjunto de cambios concretos en la práctica legal o política si tuvieran la ocasión. Esta idea llegó a mis oídos en una conversación que tuve cenando con Boyle y otras personas en la noche del lanzamiento de Creative Commons, en diciembre de 2002. Durante dicha conversación, Boyle dijo algo del estilo de «Realmente hemos *hecho* algo; no nos limitamos a sentarnos a escribir artículos y hablar sobre los peligros a los que nos enfrentamos —hemos hecho algo». Con ello se refería tanto a la organización como a las licencias legales que habían creado, y en este sentido Boyle cumple con creces los requisitos de un *geek* polímata cuya comprensión de la tecnología es la de una intervención en un estado de cosas ya constituido, intervención que demuestra su valor al ser creada e implementada, no al ser juzgada en el tribunal de las opiniones académicas.

De modo similar, el enfoque que Lessig imprime a sus escritos y charlas está indisolublemente orientado a transformar el modo en que la gente se aproxima a la legislación de propiedad intelectual y, de modo aún más general, el modo en que comprenden la relación entre sus derechos y su cultura.¹⁶ A un nivel académico, dicho enfoque impregna las enseñanzas legales y económicas (aunque, como ha puntualizado jocosamente, se trate de una «segunda» Escuela de Chicago) pero se centra más en la comprensión y manipulación de normas y costumbres (de la «cultura») que en el Derecho estrictamente concebido.¹⁷

Ambos pensadores se inspiran en un consenso económico algo heterodoxo extraído principalmente de la economía institucional y

16. Véase Kelty, «Punt to Culture».

17. Lessig, «The New Chicago School».

rutinariamente usado para elaborar argumentos de políticas públicas sobre la eficacia o eficiencia del sistema de propiedad intelectual. Asimismo se inspiran en un emergente consenso tendente a tratar el dominio público del mismo modo que los ecologistas trataron el medioambiente en los 60.¹⁸ Estos enfoques parten de una arraigada preocupación académica y política acerca del estatus y la naturaleza de los «bienes públicos», y no directamente del problema del software libre o de Internet. En cierto modo, la preocupación por los bienes públicos, el procomún, el dominio público y la acción colectiva forman parte de la misma «reorientación del poder y el saber» que identificó a lo largo de *Two Bits*: esto es, la legitimación de los medios de creación, comunicación y difusión del conocimiento. Con todo, la mayoría de expertos en economía institucional y políticas públicas está tan estupefacto ante el hecho del software libre como el resto del mundo, y su propósito ha sido cuadrar la interpretación existente de los bienes públicos y la acción colectiva con este nuevo fenómeno.¹⁹

Todos estos hilos conforman la trama del experimento orientado a modular los componentes del software libre para crear diferentes licencias que cubran una gama más amplia de objetos y atiendan a personas y organizaciones que no desarrollan software. Sin embargo, más que intentar proseguir las discusiones a nivel teórico, mi participación pretendía examinar cómo y qué discutían en la práctica quienes construían dichos experimentos, observar qué restricciones, argumentos, sorpresas o desconciertos surgían durante la creación tanto de nuevas licencias como de una nueva forma de autoría de material académico. Al igual que aquellos que estudian la «ciencia en acción» o la distinción entre «las leyes en los libros» y «las leyes en acción», yo buscaba observar las realidades de una práctica fuertemente determinada por marcos textuales y epistemológicos de diversas clases.²⁰

En mis años de participación en Connexions acabé contemplándolo como algo a medio camino entre un experimento natural y un experimento intelectual: el proyecto se desarrollaba de modo abierto

18. De ahí los conceptos de Boyle «Segundo Movimiento de Cercamiento» y «preservación del *copyright*» (véase Boyle, «The Second Enclosure Movement» [ed. cast.: «El segundo movimiento de cercamiento y la construcción del dominio público»]; Bollier, *Silent Theft*). Acaso la expresión más sofisticada y convincente del enfoque institucional-económico de la interpretación del software libre sea la obra de Yochai Benkler, especialmente «Sharing Nicely» y «Coase's Penguin». Véase también Benkler, *Wealth of Networks* [ed. cast.: *La riqueza de las redes*].

19. La obra de Steven Weber *The Success of Open Source* resulta ejemplar en este sentido.

20. Carrington y King, «Law and the Wisconsin Idea».

e invitaba a la participación de investigadores y docentes en activo (un experimento natural, por cuanto no era una empresa académica cerrada y orientada a establecer resultados específicos, sino un sistema funcional esencialmente irrestringido del que la gente podría y llegaría a depender), y sin embargo procedía mediante la realización de una serie de conjeturas estratégicas (un experimento intelectual) acerca de tres aspectos relacionados: (1) qué es (y será) posible hacer técnicamente; (2) qué es (y será) posible hacer legalmente; y (3) qué han hecho y hacen los investigadores y educadores en el normal transcurso de sus actividades.

Al mismo tiempo, este experimento dio forma a ciertas cuestiones legales que yo canalizaba a través de Creative Commons, asuntos que abarcaban desde interrogantes técnicos sobre estructuras de documentos digitales, requisitos de atribución y URLs hasta cuestiones relativas a derechos morales, derechos de renuncia y al significado de «modificación». La historia de la interacción entre Connexions y Creative Commons supuso para mí una lección sobre un modo particular de pensamiento legal que ha sido descrito en términos más académicos como la diferencia entre la tradición romana o, más aproximadamente, napoleónica de racionalismo legal y la tradición angloamericana de derecho consuetudinario.²¹ En suma, se trató de una experiencia práctica sobre la diferencia exacta entre el código legal y el código informático con respecto al modo en que esos dos elementos pueden hacerse flexibles o adaptativos.

21. En concreto, Glenn Brown sugería a Oliver Wendell Holmes como una especie de punto de origen tanto del realismo jurídico crítico como del Derecho y la Economía, una especie de filtro a través del cual los juristas obtenían tanto su particular Nietzsche como su liberalismo (véase Oliver Wendell Holmes, «The Path of the Law» [ed. cast.: *La senda del Derecho*]). La opinión de Glenn era que lo que él llamaba «despeje a la cultura» [«punting to culture»] (con lo que se refería a redactar leyes minimalistas que permitieran que los usos sociales completaran los detalles) descendía más o menos directamente del tipo de razonamiento legal encarnado por Holmes: «Nótese que si por algo [Holmes] es célebre en círculos legales es por su defensa de que las cuestiones de moralidad se eliminan del análisis legal y se dejen al campo de la ética. Esto es lo que hace de él el padrino tanto de los posners del mundo, como de los crits y de extraños híbridos como lessig» (Glenn Brown, comunicación personal, 11 de agosto de 2003).

IX. REUTILIZACIÓN, MODIFICACIÓN Y LA INEXISTENCIA DE NORMAS

El proyecto Connexions fue un experimento de modulación de las prácticas del software libre que no estuvo tanto inspirado cuanto basado en un patrón extraído de la experiencia de personas con cierta trayectoria en el software libre, incluido yo mismo. Pero, ¿cómo se usan exactamente esos patrones? ¿Qué se calca y qué se cambia? En lo tocante a la trascendencia cultural del software libre, ¿cuáles son las implicaciones de dichos cambios? ¿Mantienen la orientación de un público recursivo o son intentos de aplicar el software libre a otros intereses privados? Y si tienen éxito, ¿cuáles son las implicaciones para sus ámbitos de influencia: la educación, la investigación académica, el conocimiento científico y la producción cultural? ¿Qué efectos tienen estos cambios en las normas de trabajo y en el significado y la forma del conocimiento en dichos ámbitos?

En este capítulo exploró con detalle etnográfico cómo las modulaciones del software libre emprendidas por Connexions y Creative Commons se relacionan con los problemas de la reutilización, la modificación y las normas de la producción académica. Presento estos dos proyectos como respuestas a la reorientación contemporánea del saber y el poder; ambos son públicos recursivos al igual que el software libre, pero expanden la esfera de la práctica en nuevas direcciones, llevándola al mundo académico de los libros de texto y la investigación y a los dominios legales de la producción cultural más general.

En el proceso de «figurarse» lo que están haciendo, estos dos proyectos se topan con un fenómeno sorprendente: el cambiante significado de la *integridad* de una obra académica o creativa. Integridad no equivale a certeza. Mientras que la certeza es una problemática bien y frecuentemente estudiada en la filosofía de la ciencia y en los estudios

de ciencia, la integridad no lo es tanto. ¿Qué hace que una obra se mantenga como obra? ¿Qué hace que un hecho se mantenga como hecho? ¿Cómo alcanza estabilidad e identidad algo, cierto o no? Semejante integridad, de la que el paradigma máximo es el libro impreso, implica estabilidad. Pero Connexions y Creative Commons, a través de sus experimentos con el software libre, se enfrentan al problema de cómo estabilizar una obra en un contexto inestable: el propio de un código fuente compatible, una Internet abierta, licencias *copyleft* y nuevas formas de coordinación y colaboración.¹ El significado de integridad tendrá efectos importantes en la capacidad de constituir una política en torno a cualquier obra dada, ya sea una obra artística o una obra académica y científica. Los actores de Creative Commons y Connexions advierten esto y, en consecuencia, forman otra nueva instancia de un público recursivo, precisamente porque buscan modos de definir pública y abiertamente el significado de la integridad —y de hacer de la modificabilidad un aspecto irreversible del proceso de estabilización del saber.

Las modulaciones del software libre llevadas a cabo por Connexions y Creative Commons revelan dos cuestiones significativas. La primera es el problemático asunto del significado de la *reutilización*, relativo al uso de conceptos, ideas, escritos, artículos, libros y demás material ya existente para la creación de nuevos objetos de conocimiento. Del mismo modo que el código fuente del software se puede compartir, portar y bifurcar para crear nuevas versiones con nuevas funciones, y del mismo modo que el software y las personas se pueden coordinar de nuevas formas usando Internet, así también se puede proceder con el contenido académico y científico. Las implicaciones de esta comparación las exploraré en este capítulo. El ámbito central tanto de Connexions como de Creative Commons (y de buena parte de la práctica científica en general) consiste en que las nuevas obras se basan en obras previas. En las ciencias la noción de que el saber es acumulativo no está en cuestión, pero el modo exacto en

1. La teoría del actor-red es la que más se aproxima al tratamiento de tales asuntos «ontológicos» como, por ejemplo, los aviones en la obra *Aircraft Stories* de John Law, la enfermedad de la arteriosclerosis en *The Body Multiple* de Annemarie Mol, o la fecundación in vitro en *Making Parents* de Charis Thompson. El foco sobre la integridad aquí está estrechamente relacionado, pero apunta a revelar las características temporales de tipos de objetos de saber altamente modificables, como los libros de texto o las bases de datos, como en la obra *Memory Practices in the Sciences* de Geoffrey Bowker.

que dicho saber se acumula no está nada claro. Así, por más que pueda revelarse que la divisa «aupado a hombros de gigantes» esconde maquinaciones, tratos secretos y maniobras maquiavélicas del tipo más cobarde, el concepto mismo del conocimiento acumulativo es sólido: la construcción de un hecho, un resultado, una máquina o una teoría a partir de otras obras previas —esta forma de reutilización como progreso— no está en cuestión. Pero la práctica material real de escritura, publicación y reutilización de otros resultados y obras es algo que, hasta hace muy poco, ha quedado oculto a la vista o tan naturalizado que las normas de la práctica se vuelven casi invisibles para sus propios practicantes.

Esto plantea la otra cuestión central de este capítulo, la de la existencia o inexistencia de normas. Que un antropólogo cuestione si existen o no normas podría parecer un modo de teorizar su pérdida de trabajo: después de todo, si algo define a la antropología es la explícitación de las normas culturales. Pero el giro hacia las «prácticas» en la antropología y los estudios de ciencia ha supuesto en parte darle la espalda a las «normas» en su clásica formulación sociológica y específicamente mertoniana. La sugerencia de Robert Merton de que la ciencia ha estado gobernada por normas —desinterés, communalismo, escepticismo organizado, objetividad— ha sido repetida y categóricamente criticada por una generación de académicos de la sociología del conocimiento científico que señalan que, incluso si tales normas son afirmadas por los actores, a menudo quedan subvertidas en la práctica². Pero algo sorprendente ha ocurrido recientemente: aquellas normas mertonianas de la ciencia se han convertido *de facto* en los más o menos explícitos *objetivos en la práctica* de científicos, ingenieros y geeks tras la estela del software libre. Si las normas mertonianas no existen, entonces están siendo inventadas. Ello, por supuesto, suscita nuevos interrogantes: ¿Es posible *crear* normas? ¿Qué significaría esto exactamente? ¿De qué modo difieren las normas de las restricciones legales y técnicas? Tanto Connexions como Creative Commons plantean explícitamente estas preguntas y buscan modos de identificar, cambiar o trabajar con normas a medida que las comprenden en el contexto de la reutilización.

2. Merton, «The Normative Structure of Science» [ed. cast.: «La estructura normativa de la ciencia»].

Pizarras blancas: ¿Qué era la publicación?

Más de una vez, me he encontrado en una habitación con Rich Baraniuk, Brent Hendricks y otros varios empleados del proyecto Connexions, todos con la vista fija en una pizarra blanca sobre la que hay garabateados diversos temas y notas. Generalmente las notas presentan una especie de calidad de palimpsesto, debido a la variedad de conversaciones previas que ya hay ahí, reescritas en letra diminuta y precisa en una esquina o prácticamente borradas debajo de nuestro debate actual. Estas conversaciones vienen a menudo precipitadas por una serie de interrogantes que Brent, Ross Reedstrom y el equipo de desarrollo han encontrado a medida que construyen y refinan el sistema. Nunca son interrogantes simples. Cualquier visitante que contemplara la pizarra podría vislumbrar la peculiar locura que aqueja al proyecto, una mezcla de términos legales, términos técnicos y términos como *cultura académica o comunidades DSP*. A mí me consultan siempre que esta mezcla de términos comienza a preocupar a los desarrolladores en lo tocante a legalidad, cultura o a la relación entre una y otra. Generalmente me colocan en la posición de hablar, bien como abogado (lo cual se supone que no debo hacer, legalmente hablando), bien como antropólogo (lo cual hago principalmente en virtud de mi plaza en un departamento universitario de antropología). Rara vez lo que digo es aceptado con anuencia: Brent y Ross, como la mayoría de *hackers*, están increíblemente versados en los detalles de la legislación de propiedad intelectual y continuamente me corrigen cuando hago aseveraciones audaces pero no-del-todo-ciertas al respecto. Con todo, rara vez se sienten lo bastante versados como para tomar decisiones sobre temas legales por sí solos, y a menudo han recurrido a mí —ora como consultor reflexivo, ora como intermediario con Creative Commons.

Con este proceso, he llegado a comprender con el tiempo, se trata de figurarse algo. No es solo una cuestión de solventar problemas técnicos sobre los que yo pudiera tener algún conocimiento disciplinar específico. La figuración es modulación, es un trabajo con patrones. Cuando el software libre funciona como un patrón para proyectos como Connexions, lo hace literalmente, al permitirnos trazar una forma de práctica conocida (la del software libre) sobre un trasfondo menos conocido y aparentemente caótico para ver donde encajan las formas y donde no. Una manera muy buena de entender qué significa esto en un caso particular —esto es, de ver con mayor claridad las modulaciones

que Connexions ha realizado— es considerar la práctica y la institución de la publicación académica a través del patrón del software libre.

Consideremos los modos en que los académicos han entendido el significado y la trascendencia de la impresión y la publicación en el pasado, antes de Internet y de la reorientación contemporánea del saber y el poder. La lista de los ambiciosos historiadores y teóricos de la relación entre medios de comunicación y saber es larga: Lucien Febvre, Walter Ong, Marshall McLuhan, Jack Goody, Roger Chartier, Friedrich Kittler, Elizabeth Eisenstein y Adrian Johns, por nombrar algunos³. A excepción de Adrian Johns, no obstante, dicha historia no parte tanto de las prácticas convencionales, legales y formales de la publicación cuanto de las prácticas y la estructura materiales de los medios mismos, es decir, de la mecánica y tecnología del libro impreso⁴. Las teorías de Ong sobre alfabetización y oralidad, la re-teorización de Kittler sobre la estructura de la evolución de los medios, la antropología de Goody sobre los medios de contabilidad y escritura —todas ellas se centran en los medios tangibles como la variable dependiente de cambio. En contraste, *The Nature of the Book* de Johns deja al descubierto los contornos del masivo empeño implicado en la conversión del libro en una forma fiable y robusta para la propagación del saber a partir del siglo XVII.

Antes de la obra de Johns, los argumentos sobre la relación entre impresión y poder se agrupaban en dos campos: uno podía sobreestimar el papel de la impresión y de la imprenta sugiriendo que la «fijeza» de un texto y la creación de múltiples copias llevaban automáticamente a la difusión de ideas y al surgimiento de la ilustración. Alternativamente, uno podía subestimar el papel del libro sugiriendo que no era más que una forma mediática transparente cuyo efecto en la difusión o evaluación de ideas no era mayor ni menor que los manuscritos o la televisión. Johns señala en particular la influencia de la investigación

3. Véase Johns, *The Nature of the Book*; Eisenstein, *The Printing Press as an Agent of Change* [ed. cast. (abreviada): *La revolución de la imprenta en la Europa moderna*]; McLuhan, *The Gutenberg Galaxy* [ed. cast.: *La Galaxia Gutenberg*] y *Understanding Media* [ed. cast.: *Comprender los medios de comunicación*]; Febvre y Martin, *The Coming of the Book* [ed. cast.: *La aparición del libro*]; Ong, *Ramus, Method, and the Decay of Dialogue*; Chartier, *The Cultural Uses of Print in Early Modern France* y *The Order of Books* [ed. cast.: *El orden de los libros*]; Kittler, *Discourse Networks 1800/1900* y *Gramophone, Film, Typewriter*.

4. Existe menos comunicación entre los teóricos e historiadores del *copyright* y la autoría y los de los libros; los primeros son también fecundos en análisis, como Jaszi y Woodmansee, *The Construction of Authorship*; Mark Rose, *Authors and Owners*; St. Amour, *The Copywrights*; Vaidhyanathan, *Copyrights and Copywrongs*.

de Elizabeth Eisenstein sobre la imprenta (y a su vez la dependencia de Bruno Latour sobre ella), quien muy enfáticamente identificó las características de la obra impresa con los cambios culturales vistos como subsecuentes, incluyendo el éxito de la revolución científica y el método experimental⁵. Por ejemplo, Elizabeth Eisenstein defendió que la fijeza —el hecho de que un conjunto de libros impresos puedan ser copias exactas entre sí— implicaba diversas transformaciones en el conocimiento. Johns, sin embargo, se ha esforzado en demostrar cuán poco fiables se percibe a menudo que son los textos. ¿De qué fuentes provienen? ¿Son legítimos? ¿Tienen el respaldo o apoyo de académicos o de la corona? En suma, la fijeza puede implicar un conocimiento sólido solo si existe un sistema de evaluación ya establecido. Johns sugiere una inversión de esta noción actualmente de sentido común:

Podemos considerar la fijeza no como una cualidad *inherente* sino *transitiva*. [...] Podemos adoptar el principio de que la fijeza existe solo en tanto la gente la reconoce y actúa en consecuencia —y no de otro modo. La consecuencia de este cambio de perspectiva es que la cultura de la impresión en sí queda inmediatamente abierta al análisis, convertida en un *resultado* de múltiples representaciones, prácticas y conflictos, y no en la *causa* múltiple que se nos presenta a menudo. En contraposición a hablar de una «lógica de impresión» impuesta a la humanidad, este enfoque nos permite recuperar la construcción de diferentes culturas de impresión en circunstancias históricas particulares.⁶

La obra de Adrian Johns se centra en el elaborado y dificultoso trabajo cultural, social y económico desplegado en los siglos XVI y XVII para transformar el libro europeo en el tipo de autoridad por la que hoy se lo toma en todo el mundo. La creación y estandarización no solo de los libros sino de una *infraestructura editorial* implicó el tipo de cuidadosas ingeniería social, gestión de reputación y habilidades de distinción, exclusión y consenso que los estudios de la ciencia han explorado eficazmente en la ciencia y la ingeniería. De ahí que Johns

5. Eisenstein, *The Printing Press as an Agent of Change* [ed. cast. (abreviada): *La revolución de la imprenta en la Europa moderna*]. La obra de Eisenstein hace referencia directa a la tesis de McLuhan en *The Gutenberg Galaxy* [ed. cast.: *La Galaxia Gutenberg*], y Latour se apoya en estas y otras obras en «Drawing Things Together».

6. Johns, *The Nature of the Book*, pp. 19-20.

se centre en «la impresión-en-desarrollo» y en la relación de la cultura de la impresión de ese periodo con la fiabilidad del conocimiento. En lugar de realizar afirmaciones generales sobre la transformación del conocimiento producida por la impresión (inquietantemente similares en muchos aspectos a las realizadas acerca de Internet), Johns explora el choque de representaciones y prácticas necesario para crear la idea, ya en el siglo XX, de que realmente solo hay o hubo *una* cultura de la impresión.

Así pues, el problema de publicación que Connexions afronta no viene meramente ocasionado por la invención o propagación de Internet, mucho menos por el software libre. Más bien se trata de los problemas de producir estabilidad e integridad en condiciones técnicas, legales y sociales muy diferentes —un problema mucho más complejo incluso que el de las «diferentes culturas de impresión en circunstancias históricas particulares» de las que Johns habla respecto del libro. Connexions se enfrenta a dos retos: el de figurarse la diferencia que el presente introduce con respecto al pasado, y el de crear o modificar una infraestructura con vistas a satisfacer las demandas de un saber adecuadamente acreditado. Los libros de texto de Connexions por necesidad difieren de los convencionales, al consistir en documentos digitales o «módulos» que se vinculan y se hacen disponibles a través de la Web con una licencia Creative Commons que permite la libertad de uso, reutilización y modificación. Esta versión de la «publicación» claramente posee implicaciones para el significado de conceptos como autoría, propiedad, tutela, edición, validación, colaboración y verificación.

La aparición convencional de un libro —en librerías, por pedidos postales, en clubes de lectura, bibliotecas o universidades— era un acontecimiento que significaba, como el propio nombre sugiere, su aparición *pública* oficial en el mundo. Antes de dicho acontecimiento, el texto circulaba solo *en privado*, esto es, solo entre la relativamente reducida red de personas que podían efectuar copias de él o estaban implicados en las etapas de su redacción, edición, corrección, revisión, composición y demás. Con Internet, ese mismo texto puede hacerse inmediatamente disponible *en cada una de estas etapas* a tantos o más lectores potenciales. En la práctica, ello torna el acontecimiento de la publicación en algo *nocial* —el clic de un botón— más que en algo material altamente organizado. Aunque está claro que la práctica de la publicación ha quedado desnaturalizada o desestabilizada por la aparición de nuevas tecnologías de la información, ello difícilmente

implica que el trabajo de estabilización del significado de la publicación —y la resultante producción de saber acreditado— haya cesado. La parte intrincada consiste en entender cómo se usa el software libre como un patrón por el cual la autoridad de la publicación en la Galaxia Gutenberg se transforma en la autoridad de la publicación en el Universo Turing.

La publicación en Connexions

En el caso de Connexions existen básicamente tres etapas en la creación de contenido. La primera, por orden cronológico, alude a aquello que ocurre antes de que Connexions entre en juego, es decir, a las prácticas familiares de lo que yo llamaría *composición*, y no meramente redacción. Algún proyecto debe estar ya en marcha, acaso iniciado bajo las restricciones de y en la era del libro, acaso concebido como un libro digital o un libro de texto en línea, pero por el momento tan solo escrito en papel o guardado en un documento Word o en LaTeX, en el ordenador de algún investigador. Podría tratarse de un proyecto individual, como en el caso del plan inicial de Rich de escribir un libro de texto sobre DSP, o bien de un proyecto colaborativo a gran escala para escribir un libro de texto.

La segunda etapa es aquella en que el documento o conjunto de documentos es traducido, o «conexificado» [«*connexified*»] al sistema de marcado usado por Connexions, que es el XML (*eXtensible Markup Language*, Lenguaje de Marcado Extensible), concretamente un subconjunto de etiquetas especialmente adecuadas a libros de texto. Estas etiquetas «semánticas» (por ejemplo *<term>* [«término»]) remiten solo al significado del texto que enmarcan, y no a la «presentación» o apariencia sintáctica de lo enmarcado, dotando al documento de la estructura necesaria para ser transformado de diversas formas creativas. Dado que el XML solo está relacionado con el contenido, y no con la presentación (aludiéndose a él a veces como un lenguaje «agnóstico»), en Connexions el mismo documento puede ser modificado automáticamente para adoptar formas variadas, sea una presentación en pantalla en un navegador, un documento .pdf o una obra publicada por encargo que pueda imprimirse como un libro completo, con numeración consecutiva de páginas, notas al pie (en lugar de hipervínculos), portada, contraportada e índice. He aquí donde se halla gran parte de la magia técnica de Connexions.

Durante esta segunda etapa de marcado en XML, el documento no es del todo público, aunque está colgado en Internet dentro de lo que se llama un grupo de trabajo, donde solo pueden verlo quienes tengan acceso al grupo en cuestión (y las personas invitadas a colaborar). Solo cuando el documento está terminado y listo para su distribución pasará a la tercera etapa de «publicado» —aquella en la que cualquiera en Internet puede solicitar el documento XML y el programa lo mostrará, usando hojas de estilo o convertidores de software, como una página HTML, un documento .pdf imprimible o una sección de un curso más amplio. Sin embargo, la publicación aquí no significa integridad; de hecho, una de las ventajas cruciales de Connexions es que el documento se vuelve menos estable que el libro-objeto al que imita: puede actualizarse, cambiarse, corregirse, borrarse, copiarse y así sucesivamente, todo ello sin ninguno de los embrollos asociados a la modificación de un libro o artículo publicados. En efecto, la muy poderosa noción de fijeza teorizada por McLuhan y Eisenstein se vuelve aquí irrelevante. El hecho de que un documento haya sido imprimido (e imprimido como un libro) ya no significa que todas las copias serán iguales; es más, es muy posible que cambie de una hora para otra, dependiendo de cuánta gente contribuya (como en el caso del software libre, que puede pasar por revisiones y actualizaciones tanto o más rápido de lo que se tarda en descargar e instalar nuevas versiones). Con las entradas de Wikipedia extremadamente politizadas o activas, por ejemplo, es imposible llegar a un texto «final», aunque las dinámicas de revisión y contrarrevisión sí que sugieren esbozos para el surgimiento de ciertos tipos de estabilidad. Ahora bien, Connexions difiere de Wikipedia también con respecto a esta integridad, debido a la inserción de la segunda etapa, durante la cual un grupo autodefinido de personas puede trabajar en un texto no público antes de ejecutar cambios que puedan verse públicamente.

A estas alturas debería estar claro, dado el ejemplo de Connexions o de proyectos similares como Wikipedia, que el cambiante significado de «publicación» en la era de Internet posee implicaciones significativas, tanto prácticas (influyendo en el modo en que las personas pueden escribir y publicar sus obras) como legales (ajustándose con dificultad a las categorías establecidas para medios previos). La tangibilidad de un libro de texto queda bastante obviamente transformada por estos cambios, pero también lo hace la trascendencia cultural de la práctica de escribir un libro de texto. Y si los libros de texto se escriben de manera distinta, usando nuevas formas de colaboración y permitiendo nove-

dosos tipos de transformación, entonces la validación, la certificación y la estructura de autoridad de los libros de texto también cambian, invitando a nuevas formas de participación abierta y democrática en la escritura, la enseñanza y el aprendizaje. Ya no encontramos todas las prácticas establecidas de autoría, colaboración y publicación configuradas alrededor del mismo plano institucional y temporal (a saber, el libro y su infraestructura de publicación). En un sentido coloquial esto resulta obvio, por ejemplo, para cualquier músico actual: la grabación y el lanzamiento de una canción a millones de oyentes potenciales es ahora técnicamente posible para cualquiera, pero el modo en que ese hecho cambia la trascendencia cultural de la creación musical aún no está claro. Para la mayoría de artistas, la creación musical no ha cambiado demasiado con la introducción de herramientas digitales, pues las nuevas tecnologías de grabación y composición imitan en gran medida las prácticas discográficas que las precedieron (por ejemplo, un programa como *Garage Band* literalmente se presenta como una grabadora de cuatro pistas en pantalla). De manera similar, muchas de las prácticas de publicación digital se han dedicado a recrear algo que se parezca a la publicación tradicional.⁷

Como quizá era de esperar, al inicio del proyecto el equipo de Connexions invirtió un montón de tiempo en generar un sistema de creación de documentos pdf que, con solo pulsar un botón, imitara esencialmente la creación de un libro de texto convencional.⁸ Pero incluso este proceso ocasiona una transformación sutil: el concepto de «edición» se vuelve mucho más difícil de rastrear: mientras que un libro de texto convencional constituye una entidad estable que pasa por una serie de impresiones y ediciones, todas ellas marcadas en su

7. Sobre este tema, cfr. el estudio de Pablo Boczkowski sobre la digitalización de periódicos, *Digitizing the News*.

8. Lo cierto es que aquí «convencional» es un término bastante aproximado históricamente: el sistema crea un .pdf traduciendo el documento XML a un documento LaTeX, y de ahí a un documento .pdf. Durante unos veinte años LaTeX ha sido un lenguaje estándar de composición textual y tipográfica usado por algunos sectores de la industria editorial (principalmente ligados a las matemáticas, la ingeniería y la informática). De no ser por la existencia de este estándar del que arrancar de modo autosostenido, el proyecto Connexions habría afrontado un desafío considerablemente más difícil, pero buena parte de la infraestructura editorial ya se ha transformado parcialmente en un sistema informáticamente mediado y controlado cuyo producto final es un libro impreso. Más adelante en la vida de Connexions, el grupo se coordinó con una empresa emergente de edición por Internet llamada Qoop.com con el fin de dar el último paso para hacer los cursos de Connexions disponibles en forma de libros de texto de impresión por encargo y encuadernación en tela, incluyendo ISBNs y textos de contracubierta.

página de publicación, un documento de Connexions puede pasar por tantas versiones como veces desee un autor introducir cambios, sin que en todo ese tiempo sea necesario modificar la edición. A este respecto, la modulación del concepto de código fuente traduce las prácticas de actualización y «versionado» al ámbito de la escritura de libros de texto. Recordemos los casos que van desde el «*continuum*» de versiones de UNIX planteado por Ken Thompson hasta las complejas batallas en torno al control de versiones en los proyectos de Linux y Apache. En el caso de la escritura de código fuente, la exactitud requiere que el cambio incluso de un solo carácter sea rastreado y etiquetado como modificación de versión, mientras que una corrección ortográfica o una fe de erratas en un libro de texto convencional difícilmente crearía la necesidad de una nueva edición.

En el repositorio de Connexions se rastrea y registra todo cambio efectuado sobre un texto, pero la identidad del módulo no cambia. Así pues, las «ediciones» se han convertido en «versiones», mientras que un módulo sustancialmente revisado o modificado podría no requerir una nueva publicación sino más bien una bifurcación para crear otro módulo con una nueva identidad. En el campo de la publicación, las ediciones no son un rasgo del medio *per se*, sino que se hacen necesarias por las prácticas temporales y espaciales de la publicación como evento, aunque obviamente este proceso se haga visible solo a través del libro mismo. Del mismo modo, el versionado se usa ahora para gestionar un proceso, pero redunda en una configuración muy diferente del medio y del material disponible en él. Connexions traza el patrón de la producción de software (compartición, porte y bifurcación, así como las normas y formas de coordinación del software libre) directamente sobre las antiguas formas de publicación. Allí donde las prácticas encajan, no se da ningún cambio, y allí donde no lo hacen, son la reorientación del saber y el poder y el surgimiento de públicos recursivos los que sirven de guía para el desarrollo del sistema.

Desde la perspectiva jurídica, este paso de las ediciones a las versiones y bifurcaciones suscita preguntas inquietantes acerca de los límites y el estatus de una obra con *copyright*. Un rasgo peculiar de la legislación de derechos de autor es que ha de actualizarse cada vez que los *medios* cambian con el fin de armonizar ciertas prácticas antiguas con las nuevas posibilidades. De este modo, dispersas entre las normativas de *copyright* encontramos indicios de viejos nuevos medios: gramófonos, gramolas, televisión por cable, fotocopiadoras, programas

P2P de compartición de archivos, etc. Cada nueva forma de comunicación altera los supuestos de los medios pasados lo suficiente como para requerir una reevaluación del equilibrio putativo subyacente al mandato constitucional que dota de su inercia a la legislación de propiedad intelectual (de EEUU). Todo nuevo dispositivo ha de interpretarse en términos de creación, almacenamiento, distribución, producción, consumo y tangibilidad, con objeto de estimar los peligros que plantea a los derechos de inversores y artistas.

Dado que la legislación de *copyright* inscribe los medios específicos en las normativas por «codificación fija», se muestra cómoda ante, por ejemplo, las ediciones de libros o las grabaciones musicales. Pero en Connexions surgen interrogantes nuevos: ¿cuántos cambios constituyen una obra nueva y, por ende, demandan una nueva licencia de *copyright*? Si una licenciataria recibe una copia de una obra, ¿sobre qué versiones ostentará derechos tras introducir modificaciones? Debido a la complejidad del software implicado, aparecen asimismo cuestiones que la ley simplemente no es capaz de abordar (como tampoco lo fue a finales de los 70 con respecto a la definición de software): ¿es el documento XML equivalente al documento visible o debe incluirse también la hoja de estilo? ¿Dónde comienza el «contenido» y acaba el «software»? Hasta que las normativas incorporen estas nuevas tecnologías o bien se modifiquen para gobernar un proceso más general, y no medios particulares, estos interrogantes continuarán surgiendo como parte de la práctica de la escritura.

Esta desnaturalización de la noción de «publicación» es responsable de gran parte de la sorpresa y preocupación con que Connexions y proyectos similares son acogidos. Cuando he enseñado el sistema a académicos, a menudo han mostrado aburrimiento mezclado con miedo y frustración: «Nunca podrá reemplazar al libro». Por un lado, Connexions ha hecho un enorme esfuerzo para que sus productos se parezcan todo lo posible a libros convencionales; por otro, la ansiedad evidenciada está justificada, porque lo que Connexions busca reemplazar no es el libro, que no es más que tinta y papel, sino *todo el proceso de publicación*. El hecho de que no esté reemplazando el libro *per se* sino todo el proceso por el que los manuscritos se convierten en objetos sólidos y tangibles llamados libros es demasiado abrumador de contemplar para la mayoría de académicos —especialmente para quienes ya han dominado el proceso existente de escritura y creación de libros. El hecho de que el sistema legal esté edificado para salvaguardar

algo previo a y sin plena continuidad con la práctica de Connexions no hace sino aumentar la preocupación por que una transformación tan inmodesta y arriesgada ponga en peligro una práctica con siglos de estabilidad tras de sí. Connexions, sin embargo, no es la causa de la desestabilización; más bien se trata de la respuesta a o el reconocimiento de un problema, uno que no es nuevo sino que reaparece periódicamente: una reorientación del saber y el poder que incluye cuestiones de ilustración y racionalidad, de democracia y autogobierno, de principios liberales y problemas de autoridad y validación del conocimiento. Los momentos destacados de correlación no son la invención de la imprenta y de Internet, sino la lucha por hacer de los libros publicados una fuente de saber acreditado durante los siglos XVII y XVIII y la lucha por hallar modos de hacer lo mismo con Internet hoy.⁹

En muchos aspectos, los colaboradores de Connexions entienden el proyecto a la vez como una respuesta a las cambiantes relaciones entre saber y poder que reafirma los valores fundamentales de libertad académica y difusión del conocimiento, y como un experimento con, e incluso una radicalización de, los ideales del software libre y de la ciencia mertoniana. La transformación del significado de publicar implica una alteración fundamental del estatus, de la integridad del conocimiento. Tal transformación busca hacer del conocimiento (del conocimiento en papel, no en las mentes) algo vivo y en mutación constante, en oposición a algo estático y acabado. El hecho de que la publicación ya no signifique integridad —esto es, que ya no signifique un estado de fijeza que se supone en teoría (y a menudo en la práctica) que da cuenta de la fiabilidad de un texto— posee implicaciones sobre el modo en que el texto es usado, reutilizado, interpretado, valorado y dotado de confianza.¹⁰ Así, mientras que la forma tradicional del libro es la misma en todas sus versiones impresas, o bien sigue una práctica explícita de aparición en distintas ediciones (que incluyen nuevos prefacios y prólogos), un documento de Connexions podría

9. Véase Johns, *The Nature of the Book*; Warner, *The Letters of the Republic*.

10. Sobre la fijeza, véase la obra de Eisenstein *The Printing Press as an Agent of Change* [ed. cast. (abreviada): *La revolución de la imprenta en la Europa moderna*], la cual cita *The Gutenberg Galaxy* [ed. cast.: *La Galaxia Gutenberg*], de McLuhan. La estabilidad de los textos se ve también rutinariamente cuestionada por los investigadores textuales, especialmente aquellos que trabajan con manuscritos y complicadas variaciones textuales (para una excelente introducción, véase Bornstein y Williams, *Palimpsest*). La obra de Michel Foucault «What Is an Author?» [ed. cast.: «¿Qué es un autor?»] aborda una problemática relacionada pero divergente, y desatiende los hechos relativamente constatables de un medio cambiante.

muy bien cambiar de aspecto de una semana a otra o de un año a otro.¹¹ Y mientras que un libro de texto también podría modificarse significativamente para reflejar el cambiante estado del conocimiento en un campo dado, un objetivo explícito de Connexions es permitir que esto ocurra «en tiempo real», o lo que es lo mismo, permitir que los educadores actualicen los libros de texto tan rápido como producen conocimiento científico.¹²

Dichas implicaciones no se le escapan al equipo de Connexions, pero ni las entienden como objetivos ni como susceptibles de solución simple. Existe un cierto entusiasmo inmodesto, quizás incluso temerario, en torno a estas implicaciones, entusiasmo que puede asumir formas tanto polímatas como transhumanistas. Por ejemplo, la desestabilización del actual sistema editorial de libros de texto que Connexions representa es (según Rich) una forma más precisa de plasmar las conexiones entre conceptos que la del formato del libro de

11. Un prominente y reciente punto de comparación puede hallarse en la forma que adopta la «segunda edición» del libro de Lawrence Lessig *Code* [ed. cast.: *El Código y otras leyes del ciberespacio*], la cual se titula *Code: Version 2.0* [ed. cast.: *El Código 2.0*] (en el título se usa «versión», pero en el texto se habla de «edición»). El primer libro se publicó en 1999 («historia arcaica en tiempo de Internet») y Lessig convenció a la editorial de que lo hiciera disponible en forma de *wiki*, un sitio web colaborativo que puede ser directamente editado por cualquiera con acceso. Dicho *wiki* fue editado y actualizado por hordas de *geeks*, y luego «cerrado» y reeditado en una segunda edición con un nuevo prefacio. Se trata de un ejemplo de control de la colaboración especialmente estrecho: aunque el *wiki* y el libro estuvieran libremente disponibles, la modificación y transformación de los mismos no se equiparaba con una lógica simple de libre-para-todos. En lugar de ello, Lessig aprovechó su propia autoridad, su voz autoral y el poder de la editorial Basic Books para crear algo que se parece mucho a una segunda edición tradicional, por más que creada mediante procesos inimaginables diez años atrás.

12. La comparación más familiar es con Wikipedia, que se lanzó después de Connexions, pero que creció de modo mucho más rápido y dinámico, en gran medida debido al uso del sistema (una manzana de cierta discordia dentro del equipo de Connexions). Wikipedia ha sufrido ataques principalmente por su falta de fiabilidad. La sospecha y el miedo que rodean a Wikipedia son similares a los que afronta Connexions, si bien en el caso de las entradas de Wikipedia el compromiso con la apertura es obstinadamente meritocrático: cualquiera puede editar cualquier artículo en cualquier momento, y lo que importa no es cuán firmemente alguien se identifique como experto por rango, título, grado o experiencia —el conocimiento de una niña de doce años sobre la guerra del Peloponeso recibe el mismo acceso y estatus que el de un estudioso del clasicismo de ochenta años. Los individuos no son dueños de los artículos y todas las contribuciones son pseudónimas y difíciles de rastrear. El rango de calidad es por ello inmenso, y la prensa dominante se ha centrado ampliamente en si Wikipedia es más o menos fiable que las encyclopedias convencionales, y no en el proceso de producción de conocimiento. Véase, por ejemplo, George Johnson, «The Nitpicking of the Masses vs. the Authority of the Experts», *New York Times*, 3 de enero de 2006, Última Edición—Final, F2; Robert McHenry, «The Faith-based Encyclopedia», *TCS Daily*, 15 de noviembre de 2004, <http://www.techcentralstation.com/111504A.html>.

texto lineal. Por ende, Connexions representa un uso de la tecnología como intervención en un contexto de práctica existente. El hecho de que Connexions pudiera asimismo desestabilizar la fiabilidad o verosimilitud del saber académico es planteado a veces como un resultado inevitable del cambio técnico —algo con lo que el mundo entero, y no ya Connexions, debe aprender a lidiar.

Por decirlo de otro modo, el «objetivo» de Connexions nunca fue destruir el mundo editorial, sino que ha venido estructurado por el mismo tipo de imaginaciones de orden moral y técnico que permean el software libre y la construcción de Internet. En este sentido, Rich, Brent y demás son *geeks* en el mismo sentido que los *geeks* del software libre: ambos comparten un público recursivo consagrado a alcanzar un orden moral y técnico en el que la apertura y la modificabilidad sean valores nucleares («Si triunfamos, desaparecemos»). La implicación es que el modelo y la infraestructura existentes para la publicación de libros de texto es de un orden moral y técnico diferente, y de ahí que Connexions necesite innovar no solo la tecnología (el código fuente o la apertura del sistema) o los acuerdos legales (las licencias) sino también las mismas normas y formas de escritura de libros de texto (la coordinación y, a la postre, un movimiento). Si la publicación implicó una vez la aparición de textos definitivos fiables —aunque el saber allí contenido pudiera contestarse rutinariamente mediante la escritura de más textos y reseñas y críticas—, Connexions implica la desnaturalización, no del conocimiento *per se*, sino del proceso por el que dicho conocimiento es estabilizado y presentado como fiable, verosímil.

Una palabra clave para la transformación de la escritura de libros de texto es *comunidad*, como indica el lema del proyecto Connexions: «Compartiendo Conocimiento y Construyendo Comunidades». *Construir* implica que tales comunidades no existen aún y que la tecnología las hará posibles; sin embargo, Connexions partió de la presunción de que existe un estándar de prácticas y normas académicas de creación de materiales educativos. Como resultado, el papel de Connexions es tanto hacer posibles dichas prácticas y normas, facilitando una versión digital del libro de texto, como intervenir en ellas, creando un proceso de creación de libros de texto diferente. Las comunidades son tanto presumidas como deseadas. A veces son reales (un grupo de ingenieros de DSP, conectados en torno a Rich y otros que trabajan en su subespecialidad), y a veces son imaginadas (como cuando en el proceso de redactar una solicitud de proyecto de investigación afirmamos que el

componente más importante del éxito del proyecto es la «siembra» de comunidades académicas). Más aún, las comunidades no son audiencias ni consumidores, y a veces ni siquiera estudiantes o aprendices. Nos las imaginamos como productoras y usuarias activas y creativas de materiales didácticos, ya sea para la docencia o para la subsiguiente creación de más recursos de este tipo. La estructura de la comunidad tiene poco que ver con cuestiones de gobierno, solidaridad o pedagogía, y mucho más con un conjunto de relaciones que pueden darse con respecto a la elaboración de materiales didácticos —una comunidad de producción colaborativa o de depuración colaborativa, como en la modulación de formas de coordinación, modulada a su vez para incluir la actividad de crear materiales didácticos.

Agencia y estructura en Connexions

Uno de los debates de pizarra más animados que recuerdo haber tenido con Brent y Ross trató el tema de la diferencia entre los posibles «roles» que los usuarios de Connexions podrían ejercer y las repercusiones que ello podría tener tanto para las características técnicas del sistema como para las normas sociales que Connexions trata de mantener y reproducir. La mayoría de sistemas de software se contentan con designar simplemente «usuarios», una cuenta con nombre de usuario y contraseña genéricos a la que se puede otorgar una serie de permisos (y que tiene a sus espaldas una larga y robusta tradición en sistemas operativos informáticos y en investigación sobre seguridad). Los usuarios son usuarios, por más que puedan acceder a diferentes archivos y programas. Lo que necesitaba Connexions era un modo de designar que la misma persona pudiera tener dos roles *exógenos* diferentes: podría ser autora pero no poseedora del contenido, y viceversa. Por ejemplo, quizás la Universidad de Rice ostenta el *copyright* de una obra, pero a los autores se les reconoce su creación. Tal situación —conocida legalmente como «trabajo por encargo»— es rutinaria en algunas universidades y en la mayoría de corporaciones. Así, mientras que al autor generalmente se le da la libertad y autoridad para crear y modificar el texto como crea conveniente, la universidad se atribuye la propiedad del *copyright* para reservarse el derecho a explotar comercialmente la obra. Tal situación dista mucho de estar zanjada y, por supuesto, entraña tensiones políticas, pero el sistema de Connexions, con el fin de ser plenamente útil a todos, necesitaba darle cabida. Adoptar una posición política opositora

volvería el sistema inútil en demasiados casos o lo haría convertirse precisamente en el tipo de sistema sin autores ni reconocimiento que es Wikipedia —una vía no deseada por muchos académicos. En un mundo perfectamente abierto, cada uno de los módulos de Connexions podría tener idénticos autores y propietarios, pero el pragmatismo demanda que los dos roles se mantengan separados.

Además, hay mucha gente implicada en el día a día de la creación académica que no son ni autores ni propietarios: estudiantes de grado y posgrado, investigadores científicos, técnicos y demás personal en la vasta, disputada y compleja ecología académica. En algunas disciplinas, todas las contribuciones pueden obtener el reconocimiento de la autoría y algunas de ellas incluso pueden compartir la propiedad, pero a menudo muchos de los que hacen el trabajo solo son mencionados en los agradecimientos, o ni eso. Una vez más, pese a que el impulso de los creadores de Connexions podría ser nivelar el campo de juego y permitir solo un tipo de usuario, lo cierto es que los académicos simplemente no usarían un sistema semejante.¹³ He aquí cómo se presentó la necesidad de un rol del tipo «mantenedor» (que podría también incluir el de «editor»), el cual difería del de autor o propietario.

Estando Brent, Ross y yo con la mirada fija en la pizarra, el descubrimiento de la necesidad de múltiples roles exógenos nos golpeó como una especie de onda sísmica a cámara lenta. No se trataba solo de la necesidad de que el contenido tuviera asociadas diferentes etiquetas para mantener un registro de estas personas en una base de datos —algo más profundo entraba en juego: la legislación y la práctica de la autoría dictaban, hasta cierto punto, la forma que debería adoptar el propio software. De súbito, las preguntas aparecían preformateadas, digámoslo así, por la ley y ciertas clases de prácticas que habían sido normalizadas

13. La comparación con Wikipedia es de nuevo pertinente. Wikipedia está, moralmente hablando, y especialmente en la persona de su editor jefe, Jimbo Wales, totalmente consagrada a la igualdad basada en el mérito, con usuarios que no reciben ninguna designación especial más allá de la cantidad y calidad percibida del material que aportan. Los grados o posiciones especiales son un anatema. Se trata de un enfoque arquetípicamente estadounidense, de impulso antiintelectual, al estilo de Horatio Alger, en el que se hace *tabula rasa* y se da a los colaboradores la oportunidad de demostrar su valía con independencia de su experiencia. Connexions, en cambio, recurre a las filas de los intelectuales y académicos y busca reemplazar la infraestructura editorial. Wikipedia, por su parte, está únicamente interesada en crear una enciclopedia mejor, mostrando a este respecto un carácter transhumanista, al atribuir su singularidad y éxito a los avances de la tecnología (Internet, los *wikis*, las conexiones de banda ancha, Google). Por otro lado, Connexions es más polimata, consagrándose a intervenir en la práctica organizativa ya complejamente constituida de la investigación y la academia.

y por ende eran apenas visibles: ¿quién debería tener permiso para cambiar qué? ¿Quién tendrá permiso para añadir o eliminar autores? ¿A quién se le dejará hacer qué cambios, y quién tendrá el *derecho legal* y quién el derecho moral o consuetudinario para ello? ¿Qué repercusiones se seguirán de las decisiones que tomen los diseñadores y de las que nosotros presentemos a los autores o mantenedores?

Las licencias Creative Commons fueron clave para revelar muchas de estas cuestiones. En sí mismas eran modulaciones de las licencias propias del software libre, pero creadas teniendo en mente a artistas, músicos, académicos o productores cinematográficos. Sin ellas el contenido de Connexions quedaría sin licencia, acaso destinado al dominio público, pero en última instancia regido por normativas de *copyright* que no proveen respuestas claras a ninguna de estas preguntas, al estar concebidas para ocuparse de medios más antiguos y de un proceso de publicación diferente. El uso de licencias Creative Commons, por otro lado, implicaba que la situación del contenido de Connexions quedaba lo bastante bien definida, en un sentido legal, como para emplearse como una restricción en la definición de la estructura del sistema de software. La licencia misma nos proporcionaba el mapa del territorio al establecer los parámetros para aspectos como la distribución, la modificación, la atribución e incluso la exhibición, la lectura o la copia.

Por ejemplo, cuando autor y propietario son diferentes no resulta para nada obvio a quién debería atribuirse el mérito. Los autores, especialmente si son académicos, esperan recibir reconocimiento (a menudo es lo único que reciben) por un artículo o un libro de texto que han escrito, aunque las universidades suelen reservarse la propiedad de dichas publicaciones, la cual parecería conllevar un derecho legal a ser identificadas a la vez como propietarias y como autoras (p.ej., los informes de Forrester Research o de la UNESCO, que ocultan la identidad de sus autores). En ausencia de licencias de ningún tipo, tal situación carece de una solución obvia o depende por entero del contexto específico. Las licencias Creative Commons, en cambio, especificaban el significado de la atribución y el requisito de mantener el aviso de *copyright*, subrayando así un procedimiento que facilitaba a los diseñadores de Connexions restricciones fijas con las que medir cómo implementarían su sistema.

Un resultado positivo de tales restricciones es que permiten un tipo de flexibilidad institucional que no sería de otro modo posible. Tanto si una universidad insiste en expropiar el *copyright* como si permite

que los académicos lo conserven podrá usar Connexions. De este modo, Connexions es más «abierto» que las tradicionales editoriales de libros de texto porque posibilita la participación de un mayor número de colaboradores heterogéneos, pero también es más «abierto» que Wikipedia, que está ideológicamente comprometida con una única definición de autoría y propiedad (una creación colaborativa anónima y recíprocamente licenciada por autores que son también los propietarios de su obra). Mientras Wikipedia sostenga tal compromiso ideológico, no podrá ser empleada por instituciones que han tomado la decisión de operar como expropiadoras de contenido, o incluso en casos donde los autores voluntariamente permiten que otra persona reciba el mérito. Si los autores y los propietarios han de ser los mismos, entonces o bien se identifica al autor como propietario —algo ilegal en ciertos casos—, o bien se identifica al propietario como autor, una situación a la que ningún académico está dispuesto a someterse.

La necesidad de múltiples roles también reveló otros problemas peculiares e inquietantes, como la cuestión de otorgar una «identidad» a autores fallecidos hace mucho cuyas obras ya no están sujetas a *copyright*. Así, por ejemplo, se incluyó una pieza de A. E. Housman como módulo para una clase y, por más que esté claro que él es el autor, el *copyright* de la obra ya expiró, de modo que Housman ya no ostenta esos derechos (ni tampoco la sociedad que publicó su obra en 1921). Y sin embargo Connexions requiere que se asocie un *copyright* a cada módulo para así permitir licenciarlo de forma abierta. Este caso particular de un autor difunto requería dos interesantes intervenciones: alguien tenía que crear una cuenta para Housman y también publicar la obra como una «edición» u obra derivada sujeta a un nuevo *copyright*. En este caso los otros dos autores son Scott McGill y Christopher Kelty. En este contexto surgió una cuestión curiosa: ¿deberíamos figurar a la vez como autores y propietarios (y mantenedores), o solo como propietarios y mantenedores? Y si alguien usa el módulo en un nuevo contexto (a lo cual le da derecho la licencia), ¿tendrá que reconocer la autoría solo de Housman o también de McGill y Kelty? ¿Qué derechos de propiedad tienen McGill y Kelty sobre la versión digital del texto de dominio público de Housman?¹⁴

14. Un atributo aún más técnico atañía a la cuestión del orden de la autoría. Al principio los diseñadores decidieron que Connexions se limitara a mostrar a los autores en orden alfabético, una práctica adoptada en algunas disciplinas, como la informática. No obstante, en el caso del ejemplo de Housman ello derivó en lo que aparecía como un módulo cuyo autor

El debate sobre los roles circulaba fluidamente en torno a conceptos como la ley (y las licencias legales), las normas, la comunidad y la identidad. Brent, Ross y otros miembros de Connexions habían desarrollado una sofisticada imaginación acerca de cómo encajaría su sistema dentro de la ecología académica existente, constreñida al mismo tiempo tanto por objetivos estándar, como la usabilidad y la eficiencia, como por novedosas licencias e inquietudes legales acerca de las cambiantes prácticas de autores e investigadores. Por ejemplo, la cuestión de cómo *puede* usarse un módulo (técnicamente y legalmente) a menudo se confunde con, o cuesta desvincularla de, la cuestión de cómo *debería* usarse dicho módulo (técnicamente, legalmente o, más en general, «socialmente» —con un uso conformado por la comunidad que lo usa). Con el fin de darle sentido a esto, los programadores de Connexions y los participantes como yo somos propensos a usar el lenguaje de la costumbre y la norma, y la figura de la comunidad, como al aludir a «las normas consuetudinarias de una comunidad académica».

De la Ley y la Tecnología a la Norma

El significado de la publicación en Connexions y las preguntas acerca de los roles y su estatuto legal apropiado surgió de la preocupación fundamental por la reutilización, que es la principal modulación del software libre que Connexions lleva a cabo: la modulación del significado del código fuente para incluir la escritura de libros de texto. Lo que hace del código fuente un componente tan crucial del software libre es la forma en que se comparte y transforma, no los rasgos técnicos de ningún lenguaje o programa en particular. De ahí que la modulación del código fuente en Connexions no sea un mero intento de hacer de los libros de texto algo exacto, algorítmico o digital, sino un experimento para compartir la escritura de dichos libros de un modo similar al del software.

Dicha modulación también afecta a otros componentes: genera una demanda de apertura en la creación y difusión de libros de texto; demanda nuevos tipos de licencias de *copyright* (las licencias Creative Commons) e influye en el significado de la coordinación para los

principal era yo, quedando A.E. Housman en un lugar secundario. Y sin la capacidad de asignar explícitamente el orden de autoría, muchas disciplinas carecían de un modo de expresar sus convenciones a estos efectos. Como resultado, el sistema fue rediseñado para permitir que los usuarios también asignaran el orden de autoría.

académicos, abarcando desde formas de colaboración y co-creación explícitas a todo el espectro de usos y reutilizaciones que los académicos habitualmente hacen del trabajo de sus pares. Es esta modulación de la coordinación la que conduce a la segunda preocupación fundamental de Connexions: la existencia de «normas» de creación, uso, reutilización, publicación y difusión en el campo académico.

Dado que los programadores e ingenieros informáticos son propensos a pensar sobre las cosas de modos concretos, prácticos y detallados, los debates sobre creación, uso y difusión raramente se dan al nivel de la abstracción filosófica, sino que se desarrollan sobre pizarras blancas usando diagramas.

Figura 8

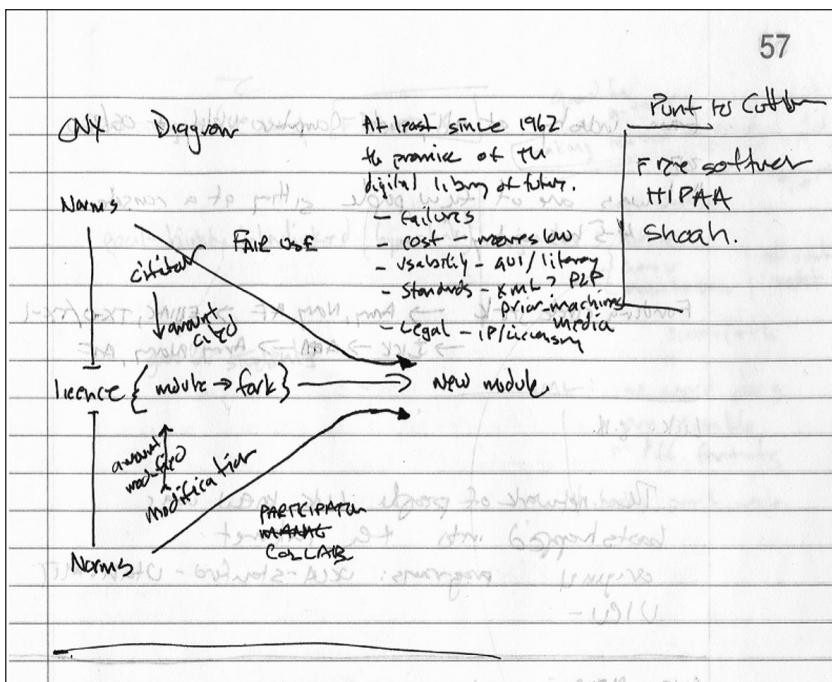
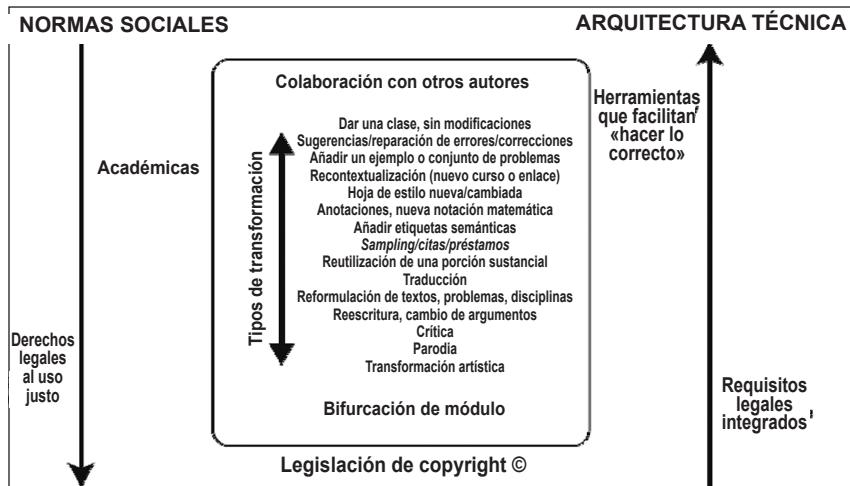


Diagrama de pizarra blanca: la cascada de la reutilización en Connexions. Concebido por Ross Reeds-trom, Brent Hendricks y Christopher Kelty. Transcrito en las notas de campo del autor, 2003.

El diagrama transscrito en la figura 8 vino precipitado por una pregunta bastante precisa: «¿Cuándo está la reutilización de un componente de un módulo (o de un módulo completo) regida por 'normas

académicas' y cuándo está sometida a las restricciones legales de las licencias?». Que alguien cite un fragmento de texto de un módulo en otro distinto se considera una práctica *normal* que, como tal, no debería suscitar preocupaciones acerca de los derechos y deberes legales de la bifurcación de dicho módulo (la creación de una nueva versión modificada, que quizás contenga solo el fragmento citado, es algo que las licencias legales permiten explícitamente). Pero ¿y si alguien toma prestadas, por decir algo, todas las ecuaciones de un módulo sobre teoría de la información y las usa para ilustrar una idea muy diferente en un módulo distinto? ¿Tiene esa persona un derecho normal o legal para ello? ¿Deberían citarse las ecuaciones? ¿Qué clase de cita debería realizarse? ¿Y si las ecuaciones resultan particularmente difíciles de marcar en el lenguaje MathML y por ello representan una significativa inversión de tiempo por parte del autor original? ¿Debería regirse esta actividad por la ley o por la normas?

Figura 9



El diagrama de pizarra transformado: formas de reutilización en Connexions.
Concebido por Christopher Kelty, 2004.

Existe una tendencia natural entre los *geeks* a responder estas preguntas únicamente con respecto a la ley; después de todo, esta presenta una alta codificación y una apariencia de autoridad en la materia. Con todo, a menudo no hay necesidad de involucrar a la ley, dado el supues-

to consenso (las «normas académicas») sobre cómo proceder, incluso si dichas normas entran en conflicto con la ley. Pero estas normas no están codificadas en ningún sitio, y ello provoca entre los *geeks* (y cada vez más entre los propios académicos) cierta incomodidad. Como en el caso del requisito de atribución, los límites marcados por una licencia escrita se perciben como mucho más estables y fiables que los derivados de la cultura, precisamente porque la cultura es aquello que permanece cuestionado y cuestionable. De ahí que la idea de crear una nueva «versión» de un texto resulte más fácil de entender cuando está claramente circunscrita como una «obra derivada» definida por la ley. En consecuencia, el software de Connexions se implementó de tal modo que el derecho legal a crear una obra derivada (a bifurcar un módulo) podría ejercerse pulsando un botón: con ello se crea automáticamente un módulo distinto, y este conserva los nombres de los autores y propietarios originales, pero incluyendo ahora también el nombre del nuevo autor tanto en calidad de autor como de mantenedor, lo cual le habilita a realizar tantos cambios como quiera.

Pero ¿es siempre necesaria la bifurcación? ¿Y si la obra derivada tan solo contiene algunas correcciones ortográficas e información ligeramente actualizada? ¿Por qué no cambiar el módulo existente (donde tales cambios se aproximarían más al lanzamiento de una nueva edición) en vez de crear una obra derivada definida legalmente? ¿Por qué no simplemente sugerir los cambios al autor original? ¿Por qué no *colaborar*? Mientras que una licencia legal otorga a la gente el derecho a hacer todas estas cosas sin siquiera consultar a la persona que licenció la obra, es muy posible que haya ocasiones en que tenga mucho más sentido ignorar dichos derechos en favor de otras normas. Las respuestas a estas preguntas dependen en gran medida del tipo y la intencionalidad de la reutilización. Una versión refinada del diagrama de la pizarra blanca, representado en la figura 9, pág. anterior, trata de captar los diversos tipos de reutilización y sus intersecciones con leyes, normas y tecnologías.

El centro del diagrama contiene una lista de diferentes tipos de reutilizaciones imaginables, dispuestas en orden descendente desde las menos hasta las más intervencionistas, lo cual implica que a medida que las transformaciones proyectadas se vuelven más drásticas, la probabilidad de colaborar con el autor original disminuye. La flecha de la izquierda indica la ruta legal desde las normas culturales a los usos justos [*fair uses*] protegidos independientemente de los deseos de los

autores; la flecha de la derecha indica la ruta técnica desde las restricciones legales integradas en las licencias hasta las herramientas de software que vuelven la colaboración (según normas supuestamente académicas) más fácil que su alternativa (ejercitar el derecho legal a realizar una obra derivada). Echando la vista atrás, me parece que las flechas a izquierda y derecha deberían realmente trazarse como un círculo que conecta leyes, tecnologías y normas en una cadena de influencia y restricción, pues en retrospectiva resulta claro que las normas de la práctica de la autoría han cambiado significativamente (o al menos se han explicitado) sobre la base de la existencia de licencias y la disponibilidad de distintos tipos de herramientas (como los blogs y Wikipedia).

El diagrama puede entenderse mejor como una forma de representar, para el propio proyecto Connexions (y sus fundadores), el experimento en curso con los componentes del software libre. Al modular el código fuente para incluir la escritura de libros de texto, Connexions visibilizó la necesidad de nuevas licencias de *copyright* adecuadas a este contenido; al basar el sistema en Internet y recurrir a estándares abiertos como XML y a componentes de código abierto, Connexions también moduló el concepto de apertura para incluir la publicación de libros de texto; y al posibilitar que el sistema fuera un repositorio abierto de módulos de libros de texto con licencia libre, Connexions visibilizó la variación en las condiciones de coordinación, no ya entre dos autores en colaboración, sino en el seno del sistema completo de publicación, cita, uso, reutilización, préstamo, inspiración, plagio, copia, emulación, etc. Tales cambios pueden o no arraigar, pues para muchos académicos plantean un incómodo desafío a un sistema de trabajo que se ha desarrollado durante siglos, mientras que para otros representan la eliminación de restricciones arbitrarias que impiden novedosas e innovadoras formas de asociación y creación de conocimiento posibilitadas en las últimas tres o cuatro décadas (y especialmente en los últimos años). Para algunos, estas modulaciones podrían conformar la base de una modulación final —un movimiento de libros de texto libres— pero por ahora tal movimiento no existe.

En el caso del código fuente informático compartido, una de las principales razones de su compartición era poder reutilizarlo: basarse en él, vincularlo, emplearlo de modos que hicieran más fácil la construcción de objetos más complejos... La misma filosofía de diseño de UNIX articula bien la necesidad de modularidad y reutilización, y la idea no es menos potente en otras áreas, como los libros de texto. Pero al

igual que la reutilización del software no es meramente un atributo de las características técnicas del software, la idea de «reutilizar» material académico posee todo tipo de implicaciones que no son meramente cuestiones de recombinación textual. La capacidad de compartir código fuente —y de crear programas complejos basados en él— requiere modulaciones tanto del significado legal del software, como en el caso de EMACS, como de su forma organizativa, como en el surgimiento de proyectos de software libre distintos de la Fundación para el Software Libre (el *kernel* de Linux, Perl, Apache, etc.).

En el caso de la reutilización de libros de texto (pero solo *después* del software libre), los problemas técnicos y legales que Connexions aborda están relativamente bien especificados: qué programas usar, la opción o no por XML, la necesidad de una excelente interfaz de usuario, etc. Por contra, el significado organizativo, cultural o práctico de la reutilización aún no está del todo claro (algo ilustrado por las figuras 8 y 9). De muchas maneras, el reconocimiento de que existen normas culturales entre los académicos supone un reflejo del (re) descubrimiento de las normas y la ética entre los *hackers* del software libre¹⁵. Pero la etiqueta «normas culturales» es un comodín para un problema que probablemente se entienda mejor como una mezcla de aspectos técnicos, organizativos y legales de carácter concreto y de imaginarios sociales de carácter más o menos abstracto a través de los cuales se comprende y se persigue un tipo particular de orden material —la creación de un público recursivo. ¿Cómo hacen los programadores, abogados, ingenieros y defensores del software libre (y los antropólogos) para «figurarse» cómo funcionan las normas? ¿Cómo se figuran modos de ponerlas en práctica o hacer uso de ellas? ¿Cómo se figuran modos de cambiarlas? ¿Cómo se figuran modos de crear normas nuevas? Lo hacen a través de las modulaciones de prácticas ya existentes, guiadas por imaginarios de orden moral y técnico. Connexions no tiende a convertirse en el software libre, sino que tiende a convertirse en un público recursivo con respecto a los libros de texto, la educación y la publicación de técnicas y conocimientos pedagógicos. La problemática de crear un público independiente y

15. Me refiero aquí al «descubrimiento» de Eric Raymond de que los *hackers* poseen normas tácitas que gobiernan sus acciones, además de las licencias legales y las prácticas técnicas en las que participan (véase Raymond, «Homesteading the Noosphere»). Para una crítica y contextualización de las normas y la ética de los *hackers*, véase Coleman, «The Social Construction of Freedom.»

autónomo constituye, pues, el sustrato tanto del software libre como de Connexions.

Hasta cierto punto, pues, la cuestión de la reutilización suscita multitud de preguntas sobre las fronteras y los límites de la academia. Brent, Ross y yo asumimos de inicio que las comunidades tienen tanto fronteras como normas, y que unas y otras están relacionadas. Pero a la vista está que esta no es una asunción segura. El uso del software no está ni técnica ni legalmente restringido a los académicos —es más, no hay manera factible de imponer tal restricción y seguir ofreciendo el programa en Internet—, y nadie implicado en él desea que lo esté. Con todo, existe una idea implícita de que las personas que aporten contenidos serán primordialmente académicos y educadores (al igual que se espera, pero no se exige, que los participantes en el software libre sean programadores). Como la figura 9 deja claro, es muy posible que se dé una tremenda variación en los tipos de reutilización que la gente desee hacer, incluso dentro de la academia. Los investigadores de humanidades, por ejemplo, son renuentes siquiera a imaginar que otros realicen obras derivadas a partir de artículos escritos por ellos, y solo pueden contemplar que su trabajo se use a la manera convencional de ser leído, citado y criticado. Por otro lado, bien pudiera ser que los investigadores de ingeniería, biología o informática encuentren placentera la idea o el acto de reutilizar, siempre que sea adecuadamente entendido como un «resultado científico» o un concepto debidamente estable sobre el que apoyarse¹⁶. La reutilización puede tener una gama de significados diferentes dependiendo no solo de si la emplean investigadores o académicos sino también dentro de ese mismo grupo tan heterogéneo.

El software de Connexions, sin embargo, no impone diferencias disciplinarias. Si acaso realiza afirmaciones muy contundentes y problemáticas sobre que el saber es el saber y que las restricciones disciplinarias son arbitrarias. Así, por ejemplo, si una bióloga desea transformar el artículo de un investigador literario sobre los tropos de Darwin para hacer que refleje la teoría de la evolución actual, podría hacerlo: ello es plenamente posible tanto legal como técnicamente. El investigador literario podría reaccionar de diversas formas, incluyendo la indignación por la malinterpretación o incomprendimiento de su obra o el placer de verla refinada. En este sentido Connexions se

16. El libro de Bruno Latour *Science in Action* [ed. cast: *Ciencia en acción*] defiende vigorosamente la centralidad de las «cajas negras» en ciencia e ingeniería precisamente por esta razón.

adhiere rigurosamente a sus ideas de apertura: ni propicia ni censura tales comportamientos.

En cambio, como sugiere la figura 9, la relación entre estos dos investigadores puede estar gobernada, bien por la especificación legal de los derechos contenidos en las licencias (un régimen legal de base privada dependiente de un régimen reglamentario nacional-global), bien por los medios consuetudinarios de colaboración posibilitados, o acaso potenciados, por las herramientas de software. El primero es el dominio del Estado, de la profesión legal y de un orden moral y técnico que, a falta de una palabra mejor, podríamos llamar modernidad. El segundo, en cambio, es el dominio de lo cultural, lo informal, lo práctico, lo interpersonal; este es el dominio de la ética (previa a su modernización, quizás) y de la *tradición*.

Si la figura 9 es una recapitulación de modernidad y tradición (¡qué mejor papel para un antropólogo!), entonces los presuntos límites en torno a las «comunidades» definen qué grupos poseen qué normas. Pero el diseño mismo de Connexions —su exactitud técnica y legal— inmediatamente hace que una variedad potencialmente enorme de tradiciones entren en conflicto entre sí. ¿Puede esperarse que la bióloga y el investigador literario ocupen el mismo universo de normas? ¿El hecho de ser académicos, empleados de una universidad o lectores de Darwin garantiza esta compartición de normas? ¿Cómo se vigilan los límites y se comunican y refuerzan las normas?

El problema de la reutilización suscita por tanto una cuestión mucho más amplia y compleja: ¿acaso existen realmente las normas? En particular, ¿existen de manera independiente de la específica práctica técnica, legal u organizativa en la que los grupos de personas tienen su existencia —fuera de la infraestructura coordinada de la academia y la ciencia? Y si Connexions suscita esta cuestión, ¿no puede también plantearse la misma pregunta respecto del elaborado sistema de profesiones, disciplinas y organizaciones que coordinan la producción académica de diferentes comunidades? ¿Hablamos aquí de normas, o se trata de prácticas «técnicas» y «legales»? ¿Qué diferencia marca la formalización? ¿Qué diferencia marca la burocratización?¹⁷

17. Debería mencionar, en mi defensa, que mis esfuerzos por hacer que mis informantes leyieran a Max Weber, Ferdinand Tönnies, Henry Maine o Emile Durkheim resultaron mucho menos exitosos que mi creación con Adobe Illustrator de lindos diagramas que hacían explícito el resurgimiento de cuestiones abordadas un siglo atrás. En todo caso, no fue por que no lo intentara.

La cuestión puede también plantearse de esta forma: ¿deberían entenderse las normas como constructos históricamente cambiantes o como características naturales de la conducta humana (patrones regulares o convenciones, que emergen *inevitablemente* allá donde interactúan los seres humanos)? ¿Son un rasgo de instituciones, leyes y tecnologías cambiantes, o se forman y persisten de la misma manera dondequiera que se congregue la gente? ¿Son las normas características de una «agencia calculadora», como lo expresa Michael Callon, o son características de la mente humana evolucionada, como defiende Marc Hauser?¹⁸

Ni una cosa ni la otra, he ahí la respuesta que mis informantes dan, en la práctica, respecto del modo de existencia de las normas culturales. Por un lado, en el proyecto Connexions la pregunta sobre el modo de existencia de las normas académicas queda sin respuesta; la asunción básica es que hay ciertas acciones que ni los límites legales ni las barreras técnicas llegan a capturar y restringir, y que corresponde a la gente que conoce o estudia las «comunidades» (a saber, los límites no legales y no técnicos) figurarse cuáles pueden ser esas acciones. Algunos días se entiende modestamente que el proyecto permite a los académicos hacer lo que hacen más rápido y mejor, pero sin alterar en lo fundamental ninguna de sus prácticas, instituciones o relaciones legales; otros días, sin embargo, se trata de un proyecto radicalmente transformador, que modifica el modo en que la gente piensa acerca de la creación de obras académicas, un proyecto que requiere educar a la gente y potencialmente «cambiar la cultura» del trabajo académico, incluyendo su tecnología, sus relaciones legales y sus prácticas.

En acusado contraste con esto (pese a su muy alto grado de simpatía), los principales miembros de Creative Commons responden la pregunta sobre la existencia de las normas de forma bastante distinta a como lo hacen los de Connexions: aseveran que las normas no solo cambian sino que son manipuladas y/o canalizadas mediante la modulación de prácticas técnicas y legales (he aquí la novedosa versión del Derecho y la Economía sobre la que se funda Creative Commons). Tal aserción deja escaso margen para las normas o la cultura; puede haber un profundo papel evolutivo en el seguimiento de las reglas o en la elección de conductas socialmente sancionadas por encima de otras socialmente inaceptables, pero la acción real ocurre en los dominios

18. Callon, *The Laws of the Markets*; Hauser, *Moral Minds* [ed. cast.: *La mente moral*]

legal y técnico. En Creative Commons la pregunta sobre la existencia de las normas se responde con firmeza mediante la frase acuñada por Glenn Brown: «Despeja a la cultura» [«*Punt to culture*»]. Para Creative Commons, las normas representan un sustrato prelegal y pretécnico sobre el que operan las licencias que ellos crean. Las normas *deben* existir para que la estrategia empleada en las licencias tenga sentido —como ilustra el siguiente relato.

Sobre la inexistencia de normas en la cultura de la no cultura

Más de una vez, me he encontrado al teléfono con Glenn Brown, con la mirada fija en mis notas, un diagrama o alguna inescrutable recopilación de jerga legal. Generalmente las conversaciones divagan desde refinados argumentos legales hasta la música y la política de Texas, pasando por los viajes de Glenn alrededor del mundo. A menudo estos temas vienen precipitados por alguna conversación previa o por la necesidad de Glenn de recordarse (y recordarme) que estamos inmersos en un proceso de creación. O de destrucción. Las suyas nunca son preguntas sencillas. Mientras que el proyecto Connexions partió de un repositorio de contenido académico necesitado de una licencia, Creative Commons partió de unas licencias necesitadas de unos tipos particulares de contenidos. Pero ambos proyectos requerían que los participantes ahondaran en los detalles tanto de las licencias como de la estructura del contenido digital, lo cual me situaba ante dichos proyectos como un intermediario idóneo para ayudar a explorar estas intersecciones. De este modo, mis conversaciones telefónicas con Glenn eran muy parecidas a las conversaciones de pizarra en Connexions: repletas de una mezcla de terminología técnica y legal y desarrolladas en gran medida con el fin de darle a Glenn la sensación de que había verificado sus planes con alguien supuestamente más conocedor del tema. He perdido la cuenta de las veces en que colgué el teléfono o abandoné la sala de reuniones preguntándome «¿Acabo de dar el visto bueno a alguna locura?». Y sin embargo rara vez he sentido que mis intervenciones sirvieran para algo más que para confirmar sospechas o desbaratar argumentos ya de por sí inestables.

En una conversación en particular —la conversación del «despeje a la cultura»— me vi desconcertado por una repentina comprensión del proceso de redacción de licencias legales y de los supuestos específicos sobre la conducta humana que tenían que estar presentes con vistas

a imaginar la creación de estas licencias o a garantizar que resulten beneficiosas a quienes las usen.

Estos debates (que a menudo incluían a otros juristas) se dieron en una especie de espacio hipotético de imaginación legal, un espacio altamente estructurado por conceptos, reglamentos y precedentes legales, y armonizado de modo extraordinariamente cuidadoso con los sutiles detalles de la semántica. Un aspecto esencial cuando se opera en este espacio de imaginación es la distinción entre la ley como una entidad semántica abstracta y la ley como un hecho práctico con el que la gente puede o no tener tratos. Qué duda cabe de que no todos los juristas operan así, pero la licencia para tal forma de pensar proviene nada menos que de una autoridad como Oliver Wendell Holmes, para quien «La senda del Derecho» iba siempre de la práctica a la regla abstracta, y no a la inversa.¹⁹ Tal oposición es inestable, pero la resalto aquí porque frecuentemente se la usaba como una *estrategia* para construir lenguaje jurídico preciso. La capacidad para imaginar la diferencia entre una regla abstracta que designa la legalidad y una regla encontrada en la práctica representaba un primer paso para ver cómo debería construirse el lenguaje de la regla.

Yo ayudé a redactar, leer y pensar las primeras licencias Creative Commons, y fue a través de esta experiencia que llegué a comprender cómo funciona la elaboración del lenguaje jurídico, y en especial cómo se relaciona el modo de existencia de normas culturales o sociales con dicha elaboración. Con todo, las licencias Creative Commons no constituyen una entidad legal familiar. Se trata de modulaciones de la licencia del software libre, pero difieren de ella de manera importante.

Las licencias Creative Commons permiten a los autores ceder el uso de su obra de una docena de formas diferentes —esto es, la misma licencia se presenta en distintas versiones. Es posible, por ejemplo, exigir atribución, prohibir la explotación comercial y permitir que se realicen y difundan obras derivadas o modificadas, o alguna combinación de todo lo anterior. Estas combinaciones diferentes efectivamente crean licencias diferentes, cada una de las cuales concede los derechos de propiedad intelectual en condiciones ligeramente diferentes. Pongamos por caso que Marshall Sahlins decide escribir un artículo sobre cómo Internet es cultural; a partir de ahí le añade a su artículo el *copyright* («© 2004 Marshall Sahlins»), exige que cualquier uso o copia

19. Oliver Wendell Holmes, «The Path of Law» [ed. cast.: *La senda del Derecho*].

del mismo mantenga el aviso de *copyright* y reconozca la autoría (que pueden no coincidir), y además permite el uso comercial del artículo. De este modo, sería legal que una editorial descargara el artículo de un servidor web basado en Linux y lo publicara en una recopilación sin tener que solicitar permiso, siempre y cuando el artículo permaneciera sin cambios y Sahlins figurara como autor de manera clara e inequívoca. La editorial no obtendría ningún derecho sobre la obra, y Sahlins tampoco cobraría derechos de autor por esa publicación. Si en cambio el autor hubiera especificado que solo permitía usos no comerciales, la editorial habría tenido que contactarle y acordar una licencia separada (las licencias Creative Commons son no exclusivas), en virtud de la cual Sahlins podría demandar una porción de los ingresos editoriales y que su nombre apareciera en la portada del libro.²⁰ Pero pongamos que en lugar de ello se trata de un joven científico que solo busca el reconocimiento y la aprobación de sus pares —entonces los derechos de autor quedarían en segundo plano frente a la máxima difusión de su trabajo. Creative Commons permite a los autores declarar, como lo expresan sus miembros, «Algunos derechos reservados» o incluso «Ningún derecho reservado».

Pero, ¿y si Sahlins hubiese elegido una licencia que permitiese modificar su obra? Ello implicaría que yo, Christopher Kelty, bien por acuerdo o bien por desacuerdo con su trabajo, podría descargar el artículo, reescribir grandes secciones del mismo, añadirle mi barroca e idiosincrásica aportación propia y escribir una sección que pretenda desacreditar (o, lo que vendría a ser lo mismo, aumentar) los argumentos de Sahlins. Entonces yo adquiriría el derecho legal a republicar el artículo como «© 2004 Marshall Sahlins, con modificaciones de © 2007 Christopher Kelty», en tanto identificara a Sahlins como el autor del artículo. La naturaleza o el alcance de las modificaciones no están limitados legalmente, pero tanto la versión original como la modificada se atribuirían legalmente a Sahlins (por más que él solo fuera titular del primer artículo).

A lo largo de varios intercambios de correos electrónicos, sesiones de *chat* y conversaciones telefónicas con Glenn, traje a colación este ejemplo y planteé la necesidad de que las licencias lo tuvieran en cuenta, pues me parecía plenamente posible que si yo pretendiera producir

20. En diciembre de 2006 Creative Commons anunció un conjunto de licencias que facilitaban el licenciamiento «subsiguiente» [«*follow up*» licensing] de una obra, especialmente si esta se publicó inicialmente con una licencia no comercial.

una obra modificada que distorsionara tanto el argumento original de Sahlins que este no quisiera que se le asociara con ella, él también debería disponer del derecho a repudiar su identificación como autor. Desde el punto de vista legal, Sahlins debería tener la capacidad de pedirme que retirara su nombre de todas las sucesivas versiones de mi tergiversación, preservando así su buen nombre y proporcionándome la libertad de continuar mancillando el mío sin remedio. Tras escrutar el asunto con el costoso despacho de abogados de Palo Alto que oficialmente se ocupaba del borrador de las licencias, elaboramos un texto que decía: «Si crea una Obra Derivada, al recibir notificación de cualquier Licenciador usted debe, en la medida de lo posible, retirar de la Obra Derivada cualquier referencia a tal Licenciador o al Autor Original, conforme a lo solicitado».

El grueso de nuestro debate se centró en la necesidad de la frase «en la medida de lo posible». Glenn me preguntó: «¿Cómo se supone que la autora original supervisará *todos* los usos posibles de su nombre? ¿Cómo ejecutará esta cláusula? ¿No va a resultar difícil retirar su nombre de todas las copias?». Glenn imaginaba una situación de adhesión estricta, en la que la presencia del nombre en el artículo fuese equivalente a la reputación del individuo, sin importar quién lo leyese realmente. A partir de esta teoría, hasta que no se extirpara todo rastro del nombre de la autora de cada uno de estos teratomas circulantes por el mundo, no podría haber paz ni descanso para la agraviada.

Hice una pausa y acto seguido solté la clase de suspiro que pretende insinuar que yo me había ganado mis conocimientos sobre cultura a través de una ardua investigación doctoral: «Probablemente no sea necesario ejecutar estrictamente esa cláusula en todos los casos —solo en los que sean significativos. Los académicos tendemos a respondernos entre nosotros solo en casos muy delimitados, escribiendo cartas al editor o enviando respuestas o refutaciones a la revista que publicó la obra. Velar realmente por una reputación conlleva mucho trabajo, y ello difiere de disciplina a disciplina. A veces puede que se necesite una acción drástica, aunque normalmente no. La gente sufre tantos usos indebidos y abusos de sus argumentos y su trabajo todo el tiempo que solo puede reaccionar cuando se enfrenta directamente a abusos graves. E incluso así, solo necesita responder en casos de críticas negativas o usos indebidos. Cuando un académico usa la obra de alguien de forma elogiosa, pero incorrecta, se suele considerar petulante (en el mejor de los casos) corregirle en público».

«En pocas palabras», dije, reclinándome en mi silla metido en el papel de experto, «se trata, ya sabes, venga —la ley no lo es *todo*, hay un montón de, ya sabes, reglas informales de urbanidad e historias que gobiernan ese tipo de cosas». Entonces Glenn afirmó: «Ah, de acuerdo, bueno, ahí es cuando despejamos a la cultura».

Cuando oí esta frase, me recliné tanto en la silla que caí de espaldas, jubilosamente pasmado. Glenn había conseguido captar lo que ningún trabajo de campo, por más amplio que fuera o más sujetos incluyera, podría haber captado. Cierta combinación de fútbol americano, un giro de Hobbes o Holmes y una vívida comprensión de qué es exactamente lo que estas licencias de *copyright* pretenden conseguir dotaron a esta frase de una luminosidad que generalmente asocio solo a las peleas de gallo balinesas. La frase encapsulaba, casi como un eslogan, una explicación muy precisa de lo que Creative Commons había emprendido. Lo que Glenn proponía con esta frase no era una teoría, sino una *estrategia* en la que una particular teoría de la cultura, por vaga que fuera, tenía su papel reservado.

Para quienes no estén familiarizados con el fútbol americano, un poco de contexto puede ser de ayuda. Cuando dos equipos se enfrentan en el terreno de juego, el equipo ofensivo dispone de cuatro intentos, llamados «*downs*», para mover el balón, bien avanzando diez yardas, bien consiguiendo un tanto en la zona de anotación. Los tres primeros intentos normalmente implican una o dos estrategias: correr o pasar, correr o pasar. En el cuarto intento, en cambio, el equipo ofensivo debe optar entre «jugársela» (llegando a la zona de anotación mediante carrera o pase), patear el balón para gol de campo (si se encuentra lo bastante cerca de la zona de anotación) o «despejar» [«*punt*»] el balón pateándolo hacia el otro equipo. El despeje es una opción algo decepcionante, porque implica ceder la posesión del balón al otro equipo, pero tiene la ventaja de hacerle retroceder tanto como sea posible, disminuyendo así la probabilidad de que anoten en su turno.

«Despejar a la cultura» sugiere, por tanto, que las licencias de *copyright* hacen tres intentos por restringir *legalmente* lo que los usuarios o consumidores de una obra pueden hacer con ella. Mediante el uso de las leyes federales vigentes sobre propiedad intelectual y de los reglamentos de licenciamiento y redacción de contratos, las licencias de *copyright* formulan lo que la gente puede y no puede hacer con una obra según la ley. Mientras que las licencias no *fuerzan* (no pueden hacerlo), en ningún sentido tangible, a hacer una cosa u otra, pueden usar el

lenguaje del Derecho y de los contratos para advertir a la gente, y acaso oblicuamente, para amenazarla. Si las licencias acaban quedándose en silencio ante un asunto —si no logran «anotar», para continuar con la analogía—, entonces es momento de «despejar» a la cultura. En vez de elaborar más leyes o llamar a la policía, la *estrategia* de la licencia se apoya en la cultura para cubrir los huecos con las propias interpretaciones de la gente sobre lo que es correcto e incorrecto, más allá del Derecho. Ello hace operativa una teoría de la cultura que enfatiza la soberanía de las costumbres no estatales y la diversidad de sistemas de normas culturales. Creative Commons preferiría que sus licencias se mantuvieran *legalmente* minimalistas. Y preferiría aún más asumir —de hecho, las licencias implícitamente lo requieren— la existencia robusta y potente de este variopinto, hetero-fisonómico y formidable oponente al Derecho que, sin uniforme ni mascota, se agazapa al otro extremo del campo, preparándose para, por así decirlo, pasarle por encima al Derecho.

De ahí que la «cultura» de Creative Commons aparezca como una vaga mixtura de muchas teorías familiares. La cultura es un conjunto no especificado pero meticulosamente articulado de normas dadas, evolucionadas, diseñadas, informales, practicadas, habituales, locales, sociales, civiles o históricas de las que se espera que gobiernen la conducta de los individuos en ausencia de un Estado, un tribunal, un rey o una fuerza policial en una de muy diversas escalas. No es monolítica (de hecho, mi aplomada explicación solo concernía a las normas de la «academia»), sino que asume una diversidad que trasciende la enumeración. Emplea elementos de relativismo —*cualquier* cultura debería ser capaz de superar a las reglas legales. No es una teoría biológica hereditaria, sino una que asume la contingencia histórica y las estructuras arbitrarias.

Ciertamente, sea lo que sea la cultura, es algo *distinto del Derecho*. El Derecho es, tomando prestada la famosa frase de Sharon Traweek, «una cultura de la no cultura» en este sentido. No son las prácticas culturales y normativas de los estudiosos del Derecho, jueces, abogados, legisladores y grupos de presión las que determinan qué forma adquirirán las leyes, sino su raciocinación cuidadosa, experta y *no cultural*. En este sentido, despejar a la cultura implica que las leyes son el resultado del designio humano, mientras que la cultura es el resultado de la actividad humana, pero no del designio humano. El derecho es sistemático y maleable, mientras que la cultura puede tener una estructura profunda

pero es inasequible al diseño humano. No obstante, la cultura puede canalizarse y rastrearse, acuciarse o guiarse, *por ley*.

En consonancia con ello, Lawrence Lessig, uno de los fundadores de Creative Commons, ha escrito extensamente acerca de la «regulación del significado social», usando casos como el uso o rechazo del cinturón de seguridad o la prohibición de fumar en lugares públicos. La decisión de no llevar cinturón de seguridad, por ejemplo, puede tener mucho más que ver con el significado contextual de abrochárselo (¿acaso no confías en el taxista?) que con la existencia del cinturón (o, ya puestos, de cinturones automáticos) o de leyes que demanden su uso. Según Lessig, lo mejor que puede hacer el Derecho frente a la costumbre es *cambiar el significado* de llevar el cinturón de seguridad: dar a su rechazo un significado deshonroso antes que honroso. Las licencias Creative Commons se basan en un supuesto similar: el Derecho es relativamente impotente frente a las costumbres artísticas y académicas profundamente arraigadas, así que lo mejor que pueden hacer las licencias es canalizar el *significado* de la compartición y la reutilización, del control o la infracción del *copyright*. Como Glenn explicaba en el contexto de un debate sobre una licencia que permitiría el *sampling* o muestreo musical:

Anticipamos que la frase «en adecuación al medio, género y nicho de mercado» podría generar cierta ansiedad, al dejar las cosas relativamente indefinidas. Pero la idea aquí es más metódica de lo que podría esperarse: la definición de «*sampling*» o «*collage*» varía a través de los diferentes medios. Más que intentar definir todos los supuestos posibles (incluyendo los aún no ocurridos)—lo que tendría el efecto de restringir los tipos de reutilizaciones a un conjunto limitado—, adoptamos el enfoque más *laissez faire*.

Esta suerte de deferencia a los valores comunitarios—piénsese en ello como un «despeje a la cultura»—es muy común en el derecho comercial y contractual cotidiano. La idea es que cuando los abogados tienen dificultades a la hora de definir los términos especializados de ciertas subculturas, deberían hacerse a un lado y permitir que dichas subculturas los elaboren. Probablemente no cause sorpresa que a Creative Commons le guste este tipo de noción.²¹

21. Mensaje de la lista de distribución cc-sampling, Glenn Brown, Asunto: Background: «As appropriate to the medium, genre, and market niche», 23 de mayo de 2003, <http://lists.ibiblio.org/pipermail/cc-sampling/2003-May/000004.html>.

Al igual que sucedía con la reutilización en Connexions, el *sampling* en el mundo de la música puede implicar una serie de significados consuetudinarios diferentes, acaso solapados, sobre qué es aceptable y qué no. Para Connexions, el truco estaba en diferenciar los casos en que la colaboración debería estimularse de aquellos en que el derecho legal al «*sampling*» —para bifurcar o crear una obra derivada— era la línea de acción apropiada. Para Creative Commons la misma estructura de las licencias intenta captar esta distinción como tal y permitir a los propios individuos tomar determinaciones sobre el significado del *sampling*.²²

Lo que hay en juego, pues, es la construcción tanto de tecnologías como de licencias legales que, como afirmarían Brent y Rich, «faciliten a los usuarios hacer lo correcto». «Lo correcto», sin embargo, es precisamente lo que se da por sobreentendido: el orden moral y técnico que guía el diseño tanto de las licencias como de las herramientas. A los usuarios de Connexions se les proporcionan herramientas que facilitan la cita, el reconocimiento, la atribución y ciertos tipos de reutilización, en lugar de herramientas que privilegian el anonimato o faciliten la proliferación o estimulen las colaboraciones no recíprocas. Análogamente, las licencias Creative Commons, siendo legalmente vinculantes, se crean con el propósito de cambiar las normas: promueven la atribución y la cita; promueven el uso justo, así como los usos claramente designados; están redactadas para dotar a los usuarios de

22. El *sampling* ofrece un ejemplo especialmente claro de cómo Creative Commons difiere de la práctica y la infraestructura existentes en la creación musical y la legislación de propiedad intelectual. En realidad, hace tiempo que la industria musical ha reconocido el *sampling* como algo que los músicos hacen y ha intentado abordarlo convirtiéndolo en una práctica económica explícita; de ahí que la industria musical estimule el *sampling* facilitando la venta de derechos para realizar dicho muestreo entre sellos discográficos y artistas. Las compañías discográficas negociarán el precio, la duración, la calidad y la cantidad del *sampling* y acordarán una tarifa.

Esta práctica se establece en oposición al supuesto, también codificado en el Derecho, de que el público tiene derecho a un uso justo del material sujeto a *copyright* sin que medio pago o permiso alguno. Podría parecer que el *sampling* de un fragmento musical entra en esta categoría de uso, excepto por el hecho de que uno de los criterios distintivos del uso justo es que no colisione con ningún mercado existente para tales usos, y el hecho de que la industria musical haya efectivamente creado un mercado para la compraventa de muestras supone que ahora el *sampling* queda habitualmente fuera de los usos justos codificados en la ley, eliminando así dicha práctica del ámbito del uso justo. Por otro lado, las licencias Creative Commons afirman que los titulares de derechos deberían poder designar su material como «apto para *sampling*», conceder permiso por anticipado y con esta práctica animar a otros a hacer lo propio. Así dan un significado «honroso» a la práctica del *sampling* gratuito, en vez del deshonroso creado por la industria. Por consiguiente, pasamos a una guerra en torno al significado de las normas, expresado en el lenguaje mezcla de jurídico-económico de Creative Commons y sus fundadores.

flexibilidad para decidir qué tipo de cosas deberían permitirse y cuáles no. Sin lugar a dudas, «lo correcto» lo es para algunos y no para otros —y por ende es político. Pero los criterios para definir lo correcto no son meramente políticos, sino que son lo que constituye de entrada la afinidad entre estos *geeks*, lo que hace de ellos un público recursivo. Ellos ven en estos instrumentos la posibilidad de la creación de públicos auténticos cuyo papel sea mantenerse al margen del poder, al margen de los mercados, y participar en la soberanía, y a través de esta participación producir libertad sin sacrificar la estabilidad.

Conclusión

¿Qué sucede cuando los *geeks* modulan las prácticas que constituyen el software libre? ¿Cuál es la intuición o la trascendencia cultural del software libre que lleva a la gente a querer emularlo y modularlo? Creative Commons y Connexions modulan las prácticas del software libre y las extienden de nuevas formas. Ambos proyectos alteran el significado del código fuente compartido para incluir la compartición de materiales distintos del software, e intentan aplicar las prácticas de redacción de licencias, coordinación y apertura a nuevos dominios. A un primer nivel, tal actividad es fascinante simplemente por lo que revela: en el caso de Connexions, revela el problema de determinar la integridad de una obra. ¿Cómo debería evaluarse la autoridad, estabilidad y fiabilidad del saber cuando es posible volver las obras permanentemente modificables? Se trata de una actividad que revela la complejidad del sistema de autorización y evaluación que se ha erigido en el pasado.

La intuición que Connexions y Creative Commons extraen del software libre gira en torno a la autoridad del saber, a una reorientación del saber y el poder que demanda una *respuesta*. Dicha respuesta ha de ser técnica y legal, por descontado, pero también ha de ser *pública* —una respuesta que defina pública y abiertamente el significado de integridad y haga de la modificabilidad un aspecto irreversible del proceso de estabilización del saber. Semejante aspiración es incompatible con la provisión de conocimiento estable por parte de actores privados a los que no se puede pedir cuentas, sean individuos, corporaciones o gobiernos, sean decisiones por decreto técnico. Siempre debe permanecer la posibilidad de que alguien pueda cuestionar, cambiar, reutilizar y modificar de acuerdo a sus necesidades.

CONCLUSIÓN.

LAS CONSECUENCIAS CULTURALES DEL SOFTWARE LIBRE

El software libre está cambiando. Todos sus aspectos aparecen hoy muy distintos a como eran cuando comencé mi trabajo, y en muchos sentidos el software libre aquí descrito no es aquel que los lectores encontrarán si acuden a Internet a buscarlo. Pero, ¿cómo podría ser de otro modo? Si la tesis que defiendo en *Two Bits* es mínimamente correcta, entonces la modulación debe estar dándose constantemente, pues la experimentación nunca persigue su propia conclusión. Una pregunta queda abierta, no obstante: al cambiar, ¿preservan el software libre y sus derivados la imaginación de orden moral y técnico que los creó? ¿Es el público recursivo algo que sobrevive, ordena o da sentido a estos cambios? ¿Existe el software libre para algo más que para sí mismo?

En *Two Bits* he explorado no solo la historia del software libre sino también la cuestión de cuál será la procedencia de tales cambios futuros. Yo defiendo una visión de continuidad en ciertas prácticas de la vida cotidiana precisamente porque Internet y el software libre permean dicha vida cotidiana de un modo destacado, y creciente. Todos los días surgen por doquier nuevos proyectos e ideas y herramientas y objetivos a partir de las prácticas que rastreo a través del software libre: Connexions y Creative Commons, el acceso abierto, la biología sintética de código abierto, la cultura libre, el acceso al conocimiento (a2k, por sus siglas en inglés), el refresco de código abierto OpenCola, el cine abierto, Science Commons, los negocios abiertos, el yoga de código abierto, la democracia de código abierto, los recursos educativos abiertos, el proyecto OLPC (*One Laptop Per Child*, Un Portátil Por Niño)¹ por no

1. Hoy ya prácticamente abandonado, véase: https://www.eldiario.es/tecnologia/OLPC-paises_en_desarrollo_0_461604619.html [N. del E.]

hablar de la proliferación de proyectos wiki-algo o la «producción entre iguales» de datos científicos o servicios al usuario —todos ellos nuevas respuestas a una reorientación del saber y el poder.² ¿Cómo podemos conocer la diferencia entre todas estas iniciativas? ¿Cómo podemos comprender su trascendencia y sus consecuencias respecto de la cultura? ¿Podemos distinguir entre proyectos que promueven una forma de esfera pública capaz de dirigir las acciones de nuestra sociedad frente a aquellos que favorecen el control corporativo, individual o jerárquico de la toma de decisiones?

A menudo, la primera respuesta a tales proyectos emergentes es centrarse en las promesas y la ideología de la gente implicada. Por un lado, proclamarse abierto o libre o público o democrático es algo que casi todo el mundo hace (incluyendo a improbables candidatos como las agencias de inteligencia de defensa de EEUU), y por ende deberíamos sospechar de toda declaración semejante.³ Aunque tales discusiones y proclamas ideológicas son importantes, sería un grave error centrarse solo en ellas. El «movimiento» —el aspecto ideológico, crítico o promisorio— es solo uno de los componentes del software libre y, de hecho, el último en llegar, después de que las otras prácticas se figuraran e hicieran legibles, replicables y modificables. Por otro lado, resulta fácil para los *geeks* y partidarios del software libre denunciar los proyectos emergentes diciendo: «Pero eso no es *realmente* código abierto o software libre». Y por más tentador que sea fijar la definición de software libre de una vez por todas con el fin de garantizar una línea divisoria nítida entre los verdaderos hijos y los advenedizos, hacerlo reduciría el software libre a una mera repetición sin diferencia y sacrificaría su atributo más potente y distintivo: su carácter adaptativo, emergente, *público*.

Pero, ¿qué preguntas deberíamos hacernos? ¿Dónde deberían centrar su atención los investigadores u observadores curiosos con el fin de ver si están o no ante un público recursivo? Muchas de estas preguntas son simples, de índole práctica: ¿están implicados a cualquier nivel el software y las redes? ¿Afirman los participantes comprender el software

2. Véase: <http://cnx.org>, <http://www.creativecommons.org>, <http://www.earlham.edu/~pteters/fos/overview.htm>, <http://www.biobricks.org>, <http://www.freebeer.org>, <http://freeculture.org>, [http://www.cptech.org/\[PAGE 348\]a2k](http://www.cptech.org/[PAGE 348]a2k), http://www.colawp.com/colas/400/cola467_recipe.html, <http://www.elephantdream.org>, <http://www.sciencecommons.org>, <http://www.plos.org>, <http://www.openbusiness.cc>, <http://www.yogaunity.org>, <http://osdproject.com>, <http://www.hewlett.org/Programs/Education/oer/> y <http://olpc.com>.

3. Véase Clive Thompson, «Open Source Spying», *New York Times Magazine*, 3 de diciembre de 2006, 54.

libre o el código abierto, ya sea en detalle o como fuente de inspiración? ¿Constituye la legislación de propiedad intelectual un problema clave? ¿Están los participantes intentando coordinarse a través de Internet y aprovechar las contribuciones voluntarias y autónomas de uno u otro tipo? Más específicamente, ¿están los participantes *modulando* alguna de estas prácticas? ¿Están pensando alguna cuestión en términos de código fuente, o de código fuente y código binario? ¿Están modificando o creando nuevas formas de licencias, contratos o acuerdos legales privados? ¿Están experimentando con formas de coordinar las acciones voluntarias de un gran número de gente irregularmente distribuida? ¿Existe entre quienes contribuyen una conciencia o una búsqueda activa de cuestiones de ideología, distinción, movimiento u oposición? ¿Son reconocidas estas prácticas como algo que crea la posibilidad de una afinidad, más que como meras prácticas «técnicas» arcanas que resultan demasiado complejas de comprender o apreciar?

En los últimos años, la cuestión del «software social» o «Web 2.0» ha dominado el circuito de congresos y debates de *geeks* y emprendedores: Wikipedia, MySpace, Flickr y Youtube, por nombrar algunos. Como ejemplo, hay montones de sitios musicales «sociales», con calificaciones colaborativas, compartición de música, descubrimiento de música, etc. Muchos de ellos usan directamente o toman su inspiración del software libre, y para todos la propiedad intelectual supone un problema central y dominante. Un factor clave de su novedad es el aprovechamiento y la coordinación de cantidades ingentes de personas de conformidad con líneas restringidas (a saber, preferencias musicales que guían el descubrimiento de música). Algunos incluso abogan o presionan a los gobiernos por un acceso (más) libre a la música digital. Sin embargo, aún no constituyen lo que yo identificaría como públicos recursivos: la mayoría de ellos son entidades comerciales cuyas estructura y especificaciones técnicas están celosamente protegidas y no abiertas a modificaciones. Por más que algunas de ellas puedan manejar contenido con licencias libres (por ejemplo, música con licencias Creative Commons), a pocas les interesa permitir que desconocidos participen, modulen o modifiquen el sistema como tal; lo que les interesa es posibilitar que los usuarios se vuelvan consumidores de formas cada vez más sofisticadas, y no necesariamente facilitar una cultura pública de la música. Quieren que la información y el conocimiento sean libres, por descontado, pero no necesariamente la infraestructura que hace disponible la información y posible el conocimiento. Tales entidades carecen del compromiso «recursivo».

En cambio, es más probable que cumplan estos criterios algunos segmentos del movimiento de acceso abierto. Como sugiere su apelativo, los participantes lo ven como un movimiento, no como una entidad corporativa o estatal, un movimiento basado en las prácticas *copyleft* y la modulación de las ideas de licenciamiento del software libre. El uso de datos científicos y las herramientas para dar sentido al acceso abierto están muy a menudo en el centro de la controversia en ciencia (algo frecuentemente reiterado por los estudios de ciencia y tecnología), de ahí que sea habitual la discusión no solo sobre la disponibilidad de los datos sino también sobre su reutilización, modificación y modulación. Proyectos como la BioBricks Foundation (biobricks.org) y organizaciones como la PLOS (*Public Library of Science*, Biblioteca Pública de Ciencia: plos.org) están comprometidos tanto con la disponibilidad como con ciertas formas de modificación colectiva. No obstante, el compromiso de convertirse en público recursivo suscita cuestiones sin precedentes acerca de la naturaleza de la calidad, fiabilidad e integridad de los datos y resultados científicos —interrogantes que reverberarán consecuentemente a través de las ciencias.

Yendo más allá, cuestiones como la defensa del «patrimonio tradicional», las licencias obligatorias de fármacos o las nuevas formas de «externalización distribuida» [«*crowdsourcing*»] en el mercado de trabajo también quedan abiertas a análisis en los términos que ofrezco en *Two Bits*⁴. Mundos virtuales como *Second Life*, «un mundo digital en 3D imaginado, creado y poseído por sus residentes», son cada vez más laboratorios para el mismo tipo de interrogantes suscitados aquí: tales mundos son mucho menos virtuales de lo que la mayoría de gente percibe, y los experimentos realizados en ellos son mucho más proclives a migrar al llamado mundo real antes de que nos demos cuenta —incluyendo experimentos tanto económicos como democráticos⁵. ¿Hasta dónde llegará *Second Life* en la facilitación de una esfera pública recursiva? ¿Puede sobrevivir como corporación y también como «mundo»? Y, por supuesto, queda la cuestión de la «blogosfera» como esfera pública, como un espacio de opinión y debate que está radicalmente abierto a

4. Véase especialmente Christen, «Tracking Properness» y «Gone Digital»; Brown, *Who Owns Native Culture?* y «Heritage as Property». El «*crowdsourcing*» encaja en otras formas vedosas de acuerdos laborales, que van desde la convencional externalización [«*outsourcing*»] y deslocalización [«*off-shoring*»] hasta recientes formas de cesión de trabajadores [«*bodyshopping*»] y «migración virtual» (véase Aneesh, *Virtual Migration*; Xiang, «*Global Bodyshopping*»).

5. Golub, «Copyright and Taboo»; Dibbell, *Play Money*.

las voces de un número masivo de personas. El *blogueo* desmiente la autopercepción del periodismo convencional como la esfera pública, pero en modo alguno es inmune a los mismos tipos de dinámicas y polarizaciones problemáticas, no más «racionales-críticas» que la FOX News y medios por el estilo.

Tales ejemplos deberían indicar hasta qué punto *Two Bits* se centra en un lapso de tiempo mucho más largo que el de los últimos años, y en asuntos de legitimidad política y cambio cultural mucho más amplios. Más que ofrecer prescripciones inmediatas de políticas públicas o buscar un cambio en la forma de pensar de la gente sobre algún asunto concreto, he abordado *Two Bits* como una obra de historia y antropología, reduciendo su aplicabilidad inmediata con la esperanza de que su utilidad sea más duradera. Los relatos que he contado se remontan al menos a cinco décadas, si no más. Si bien está claro que Internet tal y como la mayoría la conocemos surge a comienzos de los 90, ha estado «en preparación» desde al menos finales de los 50. A mis estudiantes —especialmente a los avezados *geeks* muy metidos en el software libre— les desconcierta enterarse de que los debates y pasados usables que están ensayando son refinamientos y variaciones de relatos tan o más viejos que sus padres. Es en esta estabilidad más profunda donde radica la trascendencia cultural del software libre: ¿qué diferencia introduce el software libre *hoy* con respecto al saber y al poder de *ayer*?

El software libre supone la respuesta a un problema, de modo muy similar a lo que supusieron la Royal Society en el siglo XVII, el surgimiento de una industria editorial en el siglo XVIII y las instituciones de la esfera pública en los siglos XVIII y XIX. Todos ellos respondían al desafío colectivo de crear regímenes de gobierno que requerían —y alentaban— un conocimiento empírico fiable como base de su legitimidad política. Tal legitimidad política no constituye un problema eterno o teórico; se trata de un problema de práctica constante en el mundo real para crear las infraestructuras por las que los individuos llegan a habitar y a comprender su propia forma de gobierno, ya sea ejercida por Estados, corporaciones o máquinas. Si el poder busca el consentimiento de los gobernados —y especialmente el consentimiento de tipo democrático y autónomo que se ha convertido en el ideal mundial dominante desde el siglo XVII—, debe también buscar cómo garantizar la estabilidad y fiabilidad del conocimiento sobre el que dicho consentimiento se apoya.

Los debates sobre la naturaleza y la historia de los públicos y las esferas públicas han servido como uno de los principales campos para este tipo de cuestionamientos pero, como espero haber mostrado aquí, este no es un asunto solo de esferas públicas sino de prácticas, tecnologías, leyes y movimientos; de iniciativas en marcha que se someten a modulación y experimentación de acuerdo con un imaginario social de orden tanto moral como técnico. El concepto de «público recursivo» no pretende reemplazar el de esfera pública. Tampoco es mi intención que resulte generalmente aplicable por los actores implicados ni realmente tampoco por muchos investigadores. No quisiera verlo repentinamente descubierto por doquier, sino principalmente a la hora de seguir la transformación, proliferación y diferenciación del software libre y sus derivados.

Llegados a este punto es posible conectar diversos hilos de las tres partes de *Two Bits*. Las detalladas descripciones del software libre y sus modulaciones dejan claro que (1) la razón de que Internet sea como es se debe a la labor de figuración del software libre, tanto antes como después de que este fuese reconocido como tal; (2) ni Internet ni el ordenador personal son la causa de una reorientación del saber y poder, sino que ambas son herramientas que vuelven posibles modulaciones de prácticas asentadas, modulaciones que revelan un problema mucho más antiguo relativo a la legitimidad de los medios de difusión y producción del saber; (3) el software libre no supone un posicionamiento ético, sino una respuesta práctica a la revelación de dichos problemas más antiguos; y (4) la mejor forma de entender esta respuesta es verla como un tipo de esfera pública, un público recursivo que es específico de las imaginaciones técnicas y morales de orden en el mundo contemporáneo de los *geeks*.

Ahora es posible regresar al significado práctico y político de la «singularidad» de Internet, esto es, al hecho de que solo hay una Internet. Esto no quiere decir que no existan otras redes, sino solo que Internet es una entidad singular y no un ejemplo de un tipo general. ¿Cómo es que Internet está abierta de la misma manera para todo el mundo, sea un individuo, una corporación o una entidad nacional? ¿Cómo se ha vuelto extensible (y, por extensión, defendible) por y para todo el mundo, sin importar su identidad, ubicación, contexto o grado de poder?

La singularidad de Internet es un hecho tanto ontológico como epistemológico; es una característica de las configuraciones técnicas

de Internet y los modos de ordenar en ella las acciones de humanos y máquinas mediante protocolos y software. Pero es también una característica de las imaginaciones técnicas y morales de la gente que construye, gestiona, habita y expande Internet. Ontológicamente, la creación y difusión de protocolos estandarizados y de novedosos procesos de estandarización constituyen la esencia de la historia. En el caso de Internet, las diferencias en los *procesos de estandarización* se revelan claramente en el famoso sistema RFC (*Requests for Comments*, Petición de Comentarios) de creación, distribución y modificación de protocolos de Internet. El sistema RFC, del mismo modo que la ISO (*International Organization for Standardization*, Organización Internacional de Estandarización) radicada en Ginebra, revelan las fallas de legitimidad internacional de sociedades complejas dependientes de redes, software y otras formas tecnológicamente avanzadas de producción, organización y gobierno del conocimiento. La legitimidad de los estándares posee una enorme trascendencia para la capacidad de los actores individuales de participar en sus propios públicos recursivos, ya sean públicos dedicados al software y a las redes u otros dedicados a la educación y el desarrollo. Pero al igual que la relación entre «la ley sobre el papel» y «la ley en acción», los estándares dependen de la acción coordinada y el orden de las prácticas humanas.

Es más, la línea aparentemente obvia entre un estándar legítimo y un producto comercializable basado en dichos estándares no comporta más que problemas. El caso de los sistemas abiertos en la industria informática de vanguardia de los 80 demuestra cómo la lógica de la estandarización no está nada claramente diferenciada de la lógica del mercado. Las batallas por los sistemas abiertos dieron lugar a novedosas formas de cooperación-dentro-de-la-competición que buscaban simultáneamente tanto la estandarización como la ventaja competitiva. Los sistemas abiertos fueron un intento por alcanzar un tipo de «singularidad» no solo para una red sino también para una infraestructura de mercado, y para ello buscaron formas de reformar en tandem las tecnologías y los mercados. Lo que ignoraron fue la estructura legal de la propiedad intelectual. El fracaso de los sistemas abiertos revela la centralidad del orden moral y técnico de la propiedad intelectual —tanto para la tecnología como para los mercados— y muestra cómo la dependencia de este imaginario vuelve literalmente imposible la estandarización de una infraestructura de mercado singular. Por contra, el éxito de Internet como infraestructura de mercado y como entidad

singular procede en parte de reconocer las limitaciones del sistema de propiedad intelectual —y durante los 90 el software libre supuso el principal campo experimental donde poner a prueba alternativas al mismo.

La singularidad de Internet descansa a su vez en una multiplicidad constraintuitiva: la multiplicidad del sistema operativo UNIX y sus miles de versiones e imitaciones y reimplementaciones. UNIX es un gran ejemplo de cómo pueden surgir novedosos e inesperados tipos de orden a partir de prácticas de alta tecnología: este sistema operativo no es ni un fenómeno académico (en forma de don) ni uno mercantil, sino un modelo de compartición híbrido que surgió a partir de un contexto técnico y legal muy inusual. UNIX demuestra cómo las prácticas de compartición estructuradas producen su propio tipo de orden. Contrariamente al consenso académico actual según el cual el software libre y sus derivados conforman una especie de «economía sumergida» (una economía «de compartición», una economía «de producción entre iguales» o una economía «no comercial»), UNIX nunca estuvo del todo fuera del mercado convencional. De ahí que los significados de compartición, distribución y rentabilidad remitan a su específico contexto técnico, legal y organizativo. Fue debido a la prohibición de comercializar UNIX impuesta a AT&T, al entusiasmo de sus usuarios por expandirlo y adaptarlo a sus propios usos con entusiasmo y también al entusiasmo de sus desarrolladores por alentar y asistir en dichas adaptaciones que UNIX proliferó y se diferenció de modos que pocos productos comerciales podían haber hecho. Pero UNIX nunca fue «libre» en ninguno de los sentidos posibles; más bien preparó el terreno, en combinación con otros sistemas abiertos, para lo que «libre» llegaría a significar en los 80 y 90. Se trató de un público recursivo naciente, el cual se enfrentaba a los retos técnicos y legales que llegarían a definir las prácticas del software libre. Sugerir que esto representa algún tipo de «afuera» respecto de un mercado económico operativo basado en dinero supone una percepción errónea de cuán transformativos de los mercados han sido UNIX e Internet (y el software libre). Es más, si algo representan UNIX y el software libre es una imaginación de cómo cambiar *toda una estructura de gobierno basada en el mercado* —no ya específicos mercados sectoriales— para incluir una forma de esfera pública, un control sobre el poder de la autoridad existente.

Por consiguiente, UNIX y los sistemas abiertos deberían contemplarse como fases iniciales de un experimento técnico colectivo de transfor-

mación de nuestras imaginaciones de orden, especialmente del orden moral de los públicos, los mercados y los pueblos autogobernados. En estos eventos quedan más patentes las continuidades y la gradualidad del cambio que cualquier súbita ruptura o continuidad que pudieran sugerir «la invención de Internet» o la aprobación de nuevas leyes de propiedad intelectual. La «reorientación del saber y el poder» tiene más de danza que de terremoto: estratificada en el tiempo, compleja en sus movimientos, adquiere una forma experimental cuyas huellas concretas son las redes, las infraestructuras, las máquinas, las leyes y los estándares que dejan tras de sí tales experimentos.

La disponibilidad, reusabilidad y modificabilidad están en el núcleo de dicha reorientación. Los experimentos de UNIX y los sistemas abiertos se habrían quedado en nada si no hubieran provocado una experimentación simultánea con la legislación de propiedad intelectual, de la que la licencia *copyleft* es la variable central y clave. La creación por Richard Stallman de GNU/EMACS y la controversia en torno a la propiedad que engendró fue en muchos aspectos un intento por resolver exactamente el mismo problema al que se enfrentaron los vendedores de UNIX y los partidarios de los sistemas abiertos: cómo incorporar extensibilidad al mercado del software —con la salvedad de que Stallman nunca lo vio como un mercado. Para él, el software era y es parte del propio ser humano, constitutivo de nuestra libertad misma y, por ende, inalienable. De ahí que la extensión del software, a través del apoyo mutuo colectivo, sea identificada con la vitalidad, el progreso o la autorrealización. Pero incluso para quienes insisten en ver el software como un mero producto el problema de la extensibilidad persiste. La estandarización, los procesos para acordar estándares y el acceso al mercado aparecen como problemas políticos tan pronto como se deniega la extensibilidad —y con ello la solución legal representada por el *copyleft* aparece como una opción, por más que suscite nuevos y turbadores interrogantes acerca de la naturaleza de la competencia y la rentabilidad.

Dichos nuevos interrogantes sobre la competencia y la rentabilidad han surgido de la masiva proliferación de formas comerciales y académicas híbridas, formas que traen aparejadas diferentes tradiciones de compartición, atribución, reputación, control, creación y difusión del saber y los productos que lo requieren. Las nuevas demandas económicas en la universidad —demasiado fácilmente etiquetadas como neoliberalización o privatización— reflejan las cambiantes demandas de una industria que viene a asemejarse más a las universidades, esto

es, que da más sin esperar nada a cambio, que divulga más y que coopera más. El desarrollo de UNIX, visto en detalle, es un síntoma de estos cambios, y el éxito del software libre aparece como un testigo inequívoco de los mismos.

La proliferación de formas comerciales-académicas híbridas en una era de modificabilidad y reusabilidad, en medio de los restos de estándares, procesos de estandarización y nuevos experimentos sobre propiedad intelectual, da lugar a un campo de juego donde se disputan mil partidas diferentes, todas las cuales giran en torno a una renovada experimentación con la coordinación, la colaboración, la adaptabilidad, el diseño, la evolución, los juegos y el jugar, los mundos y la creación de mundos. Estos juegos son indicativos del triunfo del amor estadounidense por el espíritu emprendedor y el experimentalismo, renunciando casi absolutamente a los ideales de la planificación y la jerarquía en favor de una especie de anarquismo integrado y técnica y legalmente complejo. Es aquí donde resurge la idea del público: la ambivalencia entre la renuncia absoluta al control y la absoluta desconfianza hacia el gobierno de unos pocos. Un público poderoso supone una respuesta, y una solución, siempre y cuando se mantenga fundamentalmente independiente del control por unos pocos. De ahí la aspiración, amplia y creciente, a constituir un público recursivo, el intento de mantener y extender los tipos de esferas públicas independientes, auténticas y autotélicas que la gente encuentra cuando llega a comprender cómo han evolucionado el software libre e Internet.

El movimiento de acceso abierto y ejemplos como Connexions representan intentos de mantener tales públicos. Algunos se conciben como baluartes contra la privatización invasora, otros como iniciativas novedosas e innovadoras, pero la mayoría comparte algunas de las prácticas concertadas a lo largo de la evolución del software libre e Internet. En lo concerniente a las publicaciones académicas y el acceso abierto, el movimiento ha reavivado debates sobre ética, normas y *método*. En ellos los ideales mertonianos reaparecen una vez más, si bien ahora no tanto como hechos del método científico sino como metas. El problema de estabilizar el saber colectivo ha pasado de ser un rasgo inherente de la ciencia a convertirse en un problema que precisa nuestra atención. La reorientación del saber y el poder y la proliferación de entidades comerciales-académicas híbridas en una era de dependencia masiva del conocimiento científico y la información conducen a un interrogante sobre la estabilización de dicho conocimiento.

Comprender cómo funciona el software libre y cómo se ha desarrollado junto a Internet y ciertas prácticas de crítica legal y cultural puede ser esencial para comprender los cimientos fiables de la producción y difusión del saber sobre los que seguimos buscando legitimar las formas de gobierno. Sin el software libre, acaso la única respuesta ante las continuas formas de exceso que asociamos con formas de gobierno ilegítimas, opacas e injustas sería un taciturno cinismo. Con él, estamos en posesión de un abanico de herramientas prácticas, respuestas estructuradas y modos inteligentes de abordar nuestra complejidad con vistas a las promesas de un imaginario compartido de gobierno legítimo y justo. Sin duda queda espacio para la crítica —y muchos investigadores la demandarán— pero la crítica académica tendrá que aprender cómo situarse, con mayor o menor facilidad, ante el software libre como *crítica*. El software libre puede también excluir, al igual que cualquier otro público o esfera pública, pero ello no supone, a mi entender, un motivo para resistirse sino para unirse a él. La alternativa sería no crear reglas, prácticas y procedimientos nuevos —es decir, quedarnos con lo que ya tenemos. El software libre no pertenece a los *geeks* ni tampoco es la única forma de convertirse en público, pero es una que tendrá un profundo efecto estructurante en cualquiera de las formas que estén por venir.

BIBLIOGRAFÍA

- ABBATE, Janet. *Inventing the Internet*. Cambridge (Mass.), MIT Press, 1999.
- Abbate, Janet y Brian Kahin (eds.). *Standards Policy for Information Infrastructure*. Cambridge (Mass.), MIT Press, 1995.
- ABELSON, Harold y SUSSMAN, Gerald J. *The Structure and Interpretation of Computer Programs*. Cambridge (Mass.), MIT Press, 1985.
- AKERA, Atsushi. «Volunteerism and the Fruits of Collaboration: The IBM User Group SHARE.» *Technology and Culture* 42.4 (octubre de 2001), pp. 710–736.
- Akera, Atsushi y NEBEKER, Frederik (eds.). *From 0 to 1: An Authoritative History of Modern Computing*. Nueva York, Oxford University Press, 2002.
- ANDERSON, Benedict. *Imagined Communities: Reflections on the Origins and Spread of Nationalism*. Londres, Verso, 1983 [ed. cast.: *Comunidades imaginadas. Reflexiones sobre el origen y la difusión del nacionalismo*, trad. por Eduardo L. Suárez. México D.F., Fondo de Cultura Económica, 1993].
- ANDERSON, Jane y BOWERY, Kathy. «The Imaginary Politics of Access to Knowledge.» Paper presented at the Contexts of Invention Conference, Cleveland, Ohio (20–23 de abril de 2006).
- ANEESH, A. *Virtual Migration: The Programming of Globalization*. Durham (N.C.), Duke University Press, 2006.
- ARENDT, Hannah. *The Human Condition*. 2d ed. University of Chicago Press, 1958. [ed. cast.: *La condición humana*, trad. por Ramón Gil Nogales. Barcelona, Paidós, 1993].
- BALKIN, Jack. *Cultural Software: A Theory of Ideology*. New Haven (Conn.), Yale University Press, 1998.
- BARANIUK, Richard y KING, W. Joseph. «Connexions: Sharing Knowledge and Building Communities.» *Sloan-C Review: Perspectives in Quality Online Education* 4.9 (septiembre de 2005), p. 8. <http://www.aln.org/publications/view/v4n9/coverv4n9.htm>.
- BARBROOK, Richard y CAMERON, Andy. «The California Ideology.» *Science as Culture* 26 (1996), pp. 44–72.
- BARDINI, Thierry. *Bootstrapping: Douglas Engelbart, Co-evolution and the Origins of Personal Computing*. Stanford (Calif.), Stanford University Press, 2001.

- BARLOW, John Perry. «The Economy of Ideas». *Wired* 2.3 (marzo de 1994). [ed. cast. «Vender vino sin botellas. La economía de la mente en la Red Global», trad. por Miquel Vidal. *Biblioweb de sindominio.net*, disponible en: <http://biblioweb.sindominio.net/telematica/barlow.pdf>].
- BARRY, Andrew. *Political Machines: Governing a Technological Society*. Londres, Athlone Press, 2001.
- BATTAGLIA, Deborah. «‘For Those Who Are Not Afraid of the Future’: Raëlian Clonehood in the Public Sphere». En *E.T. Culture: Anthropology in Outerspaces*, Deborah Battaglia (ed.), 149-179. Durham (N.C.), Duke University Press, 2005.
- BENKLER, Yochai. «Coase’s Penguin, or Linux and the Nature of the Firm.» *Yale Law Journal* 112.3 (2002), pp. 369-446.
- «Sharing Nicely: On Shareable Goods and the Emergence of Sharing as a Modality of Economic Production.» *Yale Law Journal* 114.2 (2004), pp. 273–358.
- *The Wealth of Networks: How Social Production Transforms Markets and Freedom*. New Haven (Conn.), Yale University Press, 2006. [ed. cast.: *La riqueza de las redes. Cómo la producción social transforma los mercados y la libertad*, ed. por Florencio Cabello y Andoni Alonso, trad. por Maryam Itatí Portillo *et al.* Barcelona, Icaria, 2012].
- BERGIN, Thomas J., Jr. y Richard, GIBSON G. Jr. (eds.). *History of Programming Languages 2*. Nueva York, Association for Computing Machinery Press, 1996.
- BERNERS-LEE, Tim, con Fischetti, Mark. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. San Francisco, Harper San Francisco, 1999.
- BIAGIOLI, Mario. *Galileo, Courtier: The Practice of Science in the Culture of Absolutism*. Chicago, University of Chicago Press, 1993.
- BOCZKOWSKI, Pablo. *Digitizing the News: Innovation in Online Newspapers*. Cambridge (Mass.), MIT Press, 2004.
- BOLLIER, David. *Silent Theft: The Private Plunder of Our Common Wealth*. Nueva York, Routledge, 2002.
- BORNSTEIN, George y WILLIAMS, Ralph, G. eds. *Palimpsest: Editorial Theory in the Humanities*. Ann Arbor, University of Michigan Press, 1993.
- BORSOOK, Paulina. *Cyberselfish: A Critical Romp through the Terribly Libertarian Culture of High Tech*. Nueva York, Public Affairs, 2000.
- BOWKER, Geoffrey. *Memory Practices in the Sciences*. Cambridge (Mass.), MIT Press, 2006.
- BOWKER, Geoffrey C. y LEIGH STAR, Susan. *Sorting Things Out: Classification and Its Consequences*. Cambridge (Mass.), MIT Press, 1999.
- BOYLE, James. «Conservatives and Intellectual Property». *Engage* 1 (abril de 2000), p. 83. <http://www.law.duke.edu/boylesite/Federalist.htm>.
- «Mertonianism Unbound? Imagining Free, Decentralized Access to Most

- Cultural and Scientific Material», en *Understanding Knowledge as a Common: From Theory to Practice*, Charlotte Hess y Elinor Ostrom (eds.), 123-144. Cambridge (Mass.), MIT Press, 2006. <http://www.james-boyle.com/mertonianism.pdf>. [ed. cast.: «¿Mertonismo desencadenado? Imaginar el acceso libre y descentralizado a la mayor parte del material cultural y científico», en Hess y Ostrom (eds.) *Los bienes comunes del conocimiento*, trad. por Pablo Carabajosa Pérez *et al.* Quito (Ecuador)/Madrid, IAEN/Traficantes de Sueños, 2016, pp. 143-162].
- «A Politics of Intellectual Property: Environmentalism for the Net?» *Duke Law Journal*, 47.1 (octubre de 1997), pp. 87-116.
 - (ed.). «The Public Domain». Special issue, *Law and Contemporary Problems*, 66.1-2 (invierno-primavera de 2003).
 - «The Second Enclosure Movement and the Construction of the Public Domain». En James Boyle (ed.), «The Public Domain», special issue, *Law and Contemporary Problems* 66.1-2 (invierno-primavera de 2003), pp. 33-74 [ed. cast.: «El segundo movimiento de cercamiento y la construcción del dominio público», trad. por Ariel Vercelli, en Beatriz Busaniche *et al.*, *Prohibido pensar, propiedad privada*. Córdoba (Argentina), Fundación Vía Libre, 2006, pp. 9-54. Disponible en: <http://www.vialibre.org.ar/2006/11/06/prohibido-pensar-propiedad-privada/>].
- BROCK, Gerald. *The Second Information Revolution*. Cambridge (Mass.), Harvard University Press, 2003.
- BROOKS, Frederick. *The Mythical Man-month: Essays on Software Engineering*. Reading (Mass.), Addison-Wesley, 1975.
- BROWN, Michael. «Heritage as Property», en *Property in Question: Value Transformation in the Global Economy*, Katherine Verdery y Caroline Humphrey (eds.), 49-68. Oxford, Berg, 2004.
- *Who Owns Native Culture?* Cambridge (Mass.), Harvard University Press, 2003.
- CALHOUN, Craig (ed.) *Habermas and the Public Sphere*. Cambridge (Mass.), MIT Press, 1992.
- CALLON, Michel. *The Laws of the Markets*. Londres, Blackwell, 1998.
- «Some Elements of a Sociology of Translation: Domestication of the Scallops and the Fishermen of St Brieuc Bay», en *Power, Action and Belief. A New Sociology of Knowledge*, John Law (ed.), 196-233. Londres, Routledge and Kegan Paul, 1986.
- CALLON, Michel; MÉADEL, Cécile y RABEHARISOA, Vololona. «The Economy of Qualities.» *Economy and Society* 31.2 (mayo de 2002), pp. 194-217.
- Campbell-Kelly, Martin. *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry*. Cambridge (Mass.), MIT Press, 2003.
- CAMPBELL-KELLY, Martin y ASPRAY, William. *Computer: A History of the Information Machine*. Nueva York, Basic Books, 1996.
- CARRINGTON, Paul D. y KING, Erika. «Law and the Wisconsin Idea». *Journal of Legal Education* 47 (1997), p. 297.

- CASTELLS, Manuel. *The Internet Galaxy: Reflections on the Internet, Business and Society*. Nueva York, Oxford University Press, 2001 [ed. cast.: *La galaxia Internet: reflexiones sobre Internet, empresa y sociedad*, trad. por Raúl Quintana. Barcelona, Plaza y Janés, 2001].
- . *The Rise of the Network Society*. Cambridge (Mass.), Blackwell, 1996 [ed. cast., *La era de la información: economía, sociedad y cultura. Vol. 1 La sociedad red*, trad. por Carmen Martínez Gimeno. Madrid, Alianza, 1997].
- CASTORIADIS, Cornelius. *The Imaginary Institution of Society*. Cambridge (Mass.), MIT Press, 1987 [ed. cast.: *La institución imaginaria de la sociedad*, trad. por Antoni Vicens y Marco-Aurelio Galmarini. Barcelona, Tusquets, 2013].
- CERF, Vinton G. y KAHN, Robert. «A Protocol for Packet Network Interconnection.» *IEEE Transactions on Communications* 22.5 (mayo de 1974), pp. 637–648.
- CHADWICK, Owen. *The Early Reformation on the Continent*. Oxford, Oxford University Press, 2001.
- CHAN, Anita. «Coding Free Software, Coding Free States: Free Software Legislation and the Politics of Code in Peru.» *Anthropological Quarterly* 77.3 (verano de 2004), pp. 531-545.
- CHARTIER, Roger. *The Cultural Uses of Print in Early Modern France*. Princeton, Princeton University Press, 1988.
- . *The Order of Books: Readers, Authors, and Libraries in Europe between the Fourteenth and Eighteenth Centuries*, trad. por Lydia G. Cochrane. Stanford (Calif.), Stanford University Press, 1994 [ed. cast.: *El orden de los libros*, trad. por Viviana Ackerman. Madrid, Gedisa, 2017].
- CHATTERJEE, Partha. «A Response to Taylor's 'Modes of Civil Society'». *Public Culture* 3.1 (1990, pp. 120–121.
- CHRISTEN, Kim. «Gone Digital: Aboriginal Remix and the Cultural Commons.» *International Journal of Cultural Property* 12 (agosto de 2005), pp. 315-345.
- . «Tracking Properness: Repackaging Culture in a Remote Australian Town». *Cultural Anthropology* 21.3 (agosto de 2006), pp. 416-446.
- CHRISTENSEN, Clayton. *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail*. Boston, Harvard Business School Press, 1997 [ed. cast.: *El dilema de los innovadores. Cuando las nuevas tecnologías pueden hacer fracasar a las grandes empresas*, trad. por Jorge Gorín. Buenos Aires, Granica, 1999].
- CHUN, Wendy y KYONG, Hui. *Control and Freedom: Power and Paranoia in the Age of Fiber Optics*. Cambridge (Mass.), MIT Press, 2006.
- CLARK, David. «The Design Philosophy of the DARPA Internet Protocols», 1988. En *Computer Communications: Architectures, Protocols, and Standards* (3^a ed.), William Stallings (ed.), pp. 54-62. Los Alamitos (Calif.), IEEE Computer Society Press, 1992.

- COHEN, Julie; PALLAS LOREN, Lydia; GANA OKEDIJI, Ruth y O'ROURKE Maureen (eds.). *Copyright in a Global Information Economy*. Aspen (Colo.), Aspen Law and Business Publishers, 2001.
- COLEMAN, E. Gabriella. «The Political Agnosticism of Free and Open Source Software and the Inadvertent Politics of Contrast». *Anthropological Quarterly*, 77.3 (verano de 2004), pp. 507–519.
- «The Social Construction of Freedom, Hackers, Ethics and the Liberal Tradition». *[Tesis doctoral]*, University of Chicago, 2005.
- COMAROFF, Jean y John Comaroff. *Ethnography and the Historical Imagination*. Boulder, Colo., Westview, 1992.
- COMER, Douglas E. *Internetworking with TCP/IP* (4^a ed.). Upper Saddle River, (N.J.), Prentice Hall, 2000 [ed. cast.: *Interconectividad de redes con TCP/IP. Vol II. Diseño e implementación* (3^a ed.), trad. por Sergio Kourchenko Barrena y Jorge Luis Gutiérrez. México D.F., Prentice Hall, 2000].
- *Operating System Design*. (1^a ed., 2 vols.). Englewood Cliffs (N.J.), Prentice Hall, 1984
- COOMBE, Rosemary y HERMAN, Andrew. «Rhetorical Virtues: Property, Speech, and the Commons on the World-Wide Web.» *Anthropological Quarterly*, 77.3 (verano de 2004), pp. 559–574.
- «Your Second Life? Goodwill and the Performativity of Intellectual Property in Online Digital Gaming.» *Cultural Studies* 20.2–3 (marzo–mayo de 2006), pp. 184–210.
- CRAIN, Patricia. *The Story of A: The Alphabetization of America from The New England Primer to The Scarlet Letter*. Stanford (Calif.), Stanford University Press, 2000.
- CRITCHLEY, Terry A. y BATTY, K. C. *Open Systems: The Reality*. Englewood Cliffs, N.J., Prentice Hall, 1993.
- CUSSINS, Charis. «Ontological Choreography: Agency through Objectification in Infertility Clinics». *Social Studies of Science* 26.3 (1996), pp. 575–610.
- DASTON, Lorraine, ed. *Biographies of Scientific Objects*. Chicago, University of Chicago Press, 2000.
- DAVIS, Martin. *Engines of Logic: Mathematicians and the Origin of the Computer*. W. W. Norton, 2001.
- DEAN, Jodi. «Why the Net Is Not a Public Sphere.» *Constellations* 10.1 (marzo de 2003), p. 95.
- DELANDA, Manuel. *Intensive Science and Virtual Philosophy*. Londres, Continuum Press, 2002.
- «Open Source: A Movement in Search of a Philosophy». Artículo presentado en el Institute for Advanced Study, Princeton (N.J.), 2001. <http://www.cddc.vt.edu/host/delanda/pages/opensource.htm>.
- *A Thousand Years of Non-linear History*. Nueva York, Zone Books, 1997 [ed. cast.: *Mil años de historia no lineal*, trad. por Carlos de Landa. Barcelona, Gedisa, 2012].

- DEWEY, John. *Freedom and Culture*. 1939 (reimp.), Amherst (N.Y.), Prometheus Books, 1989.
- *Liberalism and Social Action*. Nueva York, G. P. Putnam's Sons, 1935 [ed. cast.: *Liberalismo y Acción Social y otros ensayos*, ed. y trad. por J. Miguel Esteban Cloquell. Valencia, Edicions Alfons El Magnànim, 1996].
- *The Public and Its Problems*. Chicago, Swallow Press, 1927; reimpr., Sage Books / Swallow Press, 1954. [ed. cast.: *La opinión pública y sus problemas*, ed. por Ramón del Castillo, trad. por Roc Filella. Madrid, Morata, 2004].
- DIBBELL, Julian. *Play Money: Or, How I Quit My Day Job and Made Millions Trading Virtual Loot*. Nueva York, Basic Books, 2006.
- «A Rape in Cyberspace». *Village Voice* 38.51 (diciembre de 1993), p. 21.
- DIBONA, Chris, et al. *Open Sources: Voices from the Open Source Revolution*. Sebastopol (Calif.), O'Reilly Press, 1999.
- DiMAGGIO, Paul, Esther Hargittai, C. Celeste y S. Shafer. «From Unequal Access to Differentiated Use: A Literature Review and Agenda for Research on Digital Inequality», en *Social Inequality*, Kathryn Neckerman (ed.), 355-400. Nueva York, Russell Sage Foundation, 2004.
- DOWNEY, Gary L. *The Machine in Me: An Anthropologist Sits among Computer Engineers*. Londres, Routledge, 1998.
- DOYLE, Richard. *Wetwares: Experiments in Postvital Living*. Minneapolis, University of Minnesota Press, 2003.
- DRAKE, William. «The Internet Religious War». *Telecommunications Policy*, 17 (diciembre de 1993), pp. 643-649.
- DREYFUS, Hubert. *On the Internet*. Londres, Routledge, 2001.
- DUMIT, Joseph. *Picturing Personhood: Brain Scans and Biomedical Identity*. Princeton, Princeton University Press, 2004.
- EAGLETON, Terry. *Ideology: An Introduction*. Londres, Verso Books, 1991 [ed. cast.: *Ideología: Una introducción*, trad. por Jorge Vigil Rubio. Barcelona, Paidós Ibérica, 1997].
- *The Ideology of the Aesthetic*. Cambridge (Mass.), Blackwell, 1990 [ed. cast.: *La estética como ideología*, trad. por Germán Cano y Jorge Cano. Madrid, Trotta, 2006].
- EDWARDS, Paul N. *The Closed World: Computers and the Politics of Discourse in the Cold War*. Cambridge (Mass.), MIT Press, 1996.
- «Infrastructure and Modernity: Force, Time, and Social Organization in the History of Sociotechnical Systems», en *Modernity and Technology*, Thomas Misa, Philip Brey y Andrew Feenberg (eds.), 185-225. Cambridge (Mass.), MIT Press, 2003.
- EISENSTEIN, Elizabeth. *The Printing Press as an Agent of Change: Communications and Cultural Transformations in Early Modern Europe*. 2 vols. Cambridge, Cambridge University Press, 1979 [ed. cast. (abreviada): *La revolución de la imprenta en la Europa moderna*, trad. por Fernando Bouza Álvarez. Madrid, Akal, 1994].

- FAULKNER, W. «Dualisms, Hierarchies and Gender in Engineering». *Social Studies of Science* 30.5 (2000), pp. 759-792.
- FEBVRE, Lucien y MARTIN, Henri-Jean. *The Coming of the Book: The Impact of Printing 1450–1800*, trad. por David Gerard. 1958; reimpr., Londres, Verso, 1976 [ed. cast.: *La aparición del libro*, trad. por Agustín Millares Carlo. México D.F., Fondo de Cultura Económica, 2005].
- SELLER, Joseph; FITZGERALD, Brian; Hissam, Scott A. y LAKHANI, Karim R. (eds.). *Perspectives on Free and Open Source Software*. Cambridge (Mass.), MIT Press, 2005.
- FEYERABEND, Paul. *Against Method*. (3^a ed.) 1975. Londres, Verso Books, 1993 [ed. cast.: *Tratado contra el método. Esquema de una teoría anarquista del conocimiento* (1^a ed.), trad. por Diego Ribés. Madrid, Tecnos, 1997].
- FIELDING, Roy T. «Shared Leadership in the Apache Project». *Communications of the ACM*, 42.4 (abril de 1999), pp. 42-43.
- FISCHER, Franklin M. *Folded, Spindled, and Mutilated*. Cambridge (Mass.), MIT Press, 1983.
- FISCHER, Michael M. J. «Culture and Cultural Analysis as Experimental Systems». *Cultural Anthropology* 22.1 (febrero de 2007), pp. 1-65.
- *Emergent Forms of Life and the Anthropological Voice*. Durham (N.C.), Duke University Press, 2003.
- «Worlding Cyberspace», en *Critical Anthropology Now*, George Marcus (ed.), 245–304. Santa Fe (N.M.), School for Advanced Research Press, 1999.
- FLICHY, Patrice. *The Internet Imaginaire*, trad. por Liz Carey-Libbrecht. Cambridge (Mass.), MIT Press, 2007.
- FORTUN, Kim. *Advocating Bhopal: Environmentalism, Disaster, New Global Orders*. Chicago, University of Chicago Press, 2003.
- «Figuring Out Ethnography», en *Fieldwork Isn't What It Used to Be*, George Marcus y James Faubion (eds.). Ithaca (NY), Cornell University Press, 2009.
- FORTUN, Kim y FORTUN, Mike. «Scientific Imaginaries and Ethical Plateaus in Contemporary U.S. Toxicology». *American Anthropologist* 107.1 (2005), pp. 43–54.
- FOUCAULT, Michel. *La naissance de la biopolitique: Cours au Collège de France (1978-1979)*. Paris, Gallimard / Le Seuil, 2004. [ed. cast.: *Nacimiento de la biopolítica. Curso del Collège de France (1978-1979)*, trad. por Horacio Pons. Madrid, Akal, 2009]
- «What Is an Author?», en *The Foucault Reader*, P. Rabinow (ed.), 101-120. Nueva York, Pantheon Books, 1984. [ed. cast.: «¿Qué es un autor?», en *Entre filosofía y literatura. Obras esenciales*, vol. I, trad. por Miguel Morey. Barcelona, Paidós Ibérica, 1999, pp. 329–360]
- 1997. «What Is Enlightenment?», en *Ethics*, Paul Rabinow (ed.), pp. 303-317. Vol. 2 de *The Essential Works of Foucault 1954-1984*. Nueva York,

- New Press, 1997 [ed. cast.: «¿Qué es la Ilustración?», trad. por Antonio Campillo. *Daimon. Revista de Filosofía*, nº 7, 1993, pp. 5-18].
- FREEMAN, Carla. *High Tech and High Heels in the Global Economy*. Durham (N.C.), Duke University Press, 2000.
- FREEMAN, Jo y JOHNSON, Victoria, eds. *Waves of Protest: Social Movements since the Sixties*. Lanham (Md.), Rowman and Littlefield, 1999.
- GALISON, Peter. *How Experiments End*. Chicago, University of Chicago Press, 1987.
- *Image and Logic: The Material Culture of Microphysics*. Chicago, University of Chicago Press, 1997.
- GALLOWAY, Alexander. *Protocol, or How Control Exists after Decentralization*. Cambridge (Mass.), MIT Press, 2004.
- GANCARZ, Mike. *Linux and the UNIX Philosophy*. Boston, Digital Press, 2003.
- *The Unix Philosophy*. Boston, Digital Press, 1994.
- GAONKAR, Dilip. «Toward New Imaginaries: An Introduction». *Public Culture* 14.1 (2002), pp. 1-19.
- GEERTZ, Clifford. «Ideology as a Cultural System», en *The Interpretation of Cultures*, pp. 193-233. Nueva York, Basic Books, 1973 [ed. cast.: «La ideología como sistema cultural», en *La interpretación de las culturas*, trad. por Alberto L. Bixio, pp. 171-202. Madrid, Gedisa, 1989].
- GERLACH, Luther P. y Virginia H. Hine. *People, Power, Change: Movements of Social Transformation*. Indianapolis, Bobbs-Merrill, 1970.
- GHOSH, Rishab Ayer. «Cooking Pot Markets: An Economic Model for the Trade in Free Goods». *First Monday* 3.3 (1998). http://www.firstmonday.org/issues/issue3_3/ghosh/.
- GILLESPIE, Tarleton. «Engineering a Principle: 'End to End' in the Design of the Internet». *Social Studies of Science* 36.3 (2006), pp. 427-57.
- GOLUB, Alex. «Copyright and Taboo». *Anthropological Quarterly* 77.3 (2004), pp. 521-530.
- GRAY, Pamela. *Open Systems: A Business Strategy for the 1990s*. Londres, McGraw-Hill, 1991.
- GREEN, Ellen y Allison Adam. *Virtual Gender: Technology, Consumption and Identity*. Londres, Routledge, 2001.
- GREEN, Ian. *The Christian's ABCs: Catechisms and Catechizing in England c1530-1740*. Oxford, Oxford University Press, 1996.
- *Print and Protestantism in Early Modern England*. Oxford, Oxford University Press, 2000.
- GREEN, Sarah; HARVEY, Penny y KNOX, Hannah. «Scales of Place and Networks: An Ethnography of the Imperative to Connect through Information and Communication Technologies.» *Current Anthropology* 46.5 (diciembre de 2005), pp. 805-826.
- GRIER, David Alan. *When Computers Were Human*. Princeton, Princeton University Press, 2005.

- GRIER, David Alan y CAMPBELL, Mary. «A Social History of Bitnet and List-serv 1985-1991.» *IEEE Annals of the History of Computing* (abril–junio de 2000), pp. 32-41.
- GRINT, Keith y GILL, Rosalind. *The Gender-Technology Relation: Contemporary Theory and Research*. Londres, Taylor and Francis, 1995.
- HABERMAS, Jürgen. *The Structural Transformation of the Public Sphere: An Inquiry into a Category of Bourgeois Society*, trad. por Thomas Burger con la asistencia de Frederick Lawrence. Cambridge (Mass.), MIT Press, 1991 [ed. cast.: *Historia y crítica de la opinión pública: la transformación estructural de la vida pública*, trad. por Antoni Doménech y Rafael Grasa, Barcelona-México D.F., G. Gili, 1981].
- HAFNER, Katie. *Where Wizards Stay Up Late: The Origins of the Internet*. Nueva York, Simon and Schuster, 1998.
- HAMERLY, Jim y Tom Paquin, con Susan Walton. «Freeing the Source», en *Open Sources: Voices from the Open Source Revolution*, Chris Dibona et al. (eds.), 197–206. Sebastopol (Calif.), O'Reilly Press, 1999.
- HARDIN, Garrett. «The Tragedy of the Commons». *Science* 162 (1968), pp. 1, 243–248.
- HASHAGEN, Ulf; KEIL-SLAWIK, Reinhart y NORBERG, Arthur (eds.). *History of Computing-Software Issues*. Berlín, Springer Verlag, 2002.
- HAUBEN, Michael y HAUBEN, Rhonda. *Netizens: On the History and Impact of Usenet and the Internet*. Los Alamitos (Calif.), IEEE Computer Society Press, 1997.
- HAUSER, Marc. *Moral Minds: How Nature Designed Our Universal Sense of Right and Wrong*. Nueva York: Ecco Press, 2006 [ed. cast.: *La mente moral: cómo la naturaleza ha desarrollado nuestro sentido del bien y del mal*, trad. por Miguel Candel. Barcelona: Paidós Ibérica, 2008].
- HAYDEN, Cori. *When Nature Goes Public: The Making and Unmaking of Bioprospecting in Mexico*. Princeton, Princeton University Press, 2003.
- HAYEK, Friedrich A. *Law, Legislation and Liberty*. Vol. 1, *Rules and Order*. Chicago, University of Chicago Press, 1970 [ed. cast.: *Derecho, legislación y libertad: una nueva formulación de los principios liberales de la justicia y de la economía política. Vol. 1. Normas y orden*, (2^a ed.), trad. por Luis Reig Albiol. Madrid, Unión Editorial, 1985].
- HELMREICH, Stefan. *Silicon Second Nature: Culturing Artificial Life in a Digital World*. Berkeley, University of California Press, 1998.
- HERRING, Susan C. «Gender and Democracy in Computer-Mediated Communication», en *Computerization and Controversy: Value Conflicts and Social Choices*, Rob Kling y Charles Dunlop (eds.), pp. 476-489 (2^a ed.). Orlando, Academic Press, 1995.
- HESS, Charlotte y OSTROM, Elinor (eds.). *Understanding Knowledge as a Common: From Theory to Practice*. Cambridge (Mass.), MIT Press, 2006. [ed. cast.: *Los bienes comunes del conocimiento*, trad. por Pablo Carbajosa Pérez et al. Quito/Madrid, IAEN/Traficantes de Sueños, 2016].

- HIMANEN, Pekka. *The Hacker Ethic and the Spirit of the Information Age*. Nueva York, Random House, 2001. [ed. cast.: *La ética del hacker*, trad. por Ferrán Meler Ortí. Barcelona, Destino, 2002].
- HINE, Christine. *Virtual Ethnography*. Londres, Sage, 2000.
- HOLMES, Douglas y MARCUS, George. «Cultures of Expertise and the Management of Globalization: Toward the Re-Functioning of Ethnography.» In *Global Assemblages: Technology, Politics, and Ethics as Anthropological Problems*, ed. Aiwa Ong and Stephen J. Collier, 235–52. Boston: Blackwell, 2005.
- HOLMES, Oliver Wendell. «The Path of Law.» *Harvard Law Review* 10 (1897), p. 457 [ed. cast.: *La senda del Derecho*, trad. por José Ignacio Solar Cayón. Madrid, Marcial Pons, 2012].
- HOPKINS, Patrick D. (ed.). *Sex/Machine: Readings in Culture, Gender and Technology*. Bloomington, Indiana University Press, 1998.
- HUBERMAN, Bernardo A. (ed.). *The Ecology of Computation*. Amsterdam, North-Holland, 1988.
- HUXLEY, Julian. *New Bottles for New Wine: Essays*. Nueva York, Harper, 1957.
- JASZI, Peter y WOODMANSEE, Martha (eds.). *The Construction of Authorship: Textual Appropriation in Law and Literature*. Durham (N.C.), Duke University Press, 1994.
- JOHNS, Adrian. *The Nature of the Book: Print and Knowledge in the Making*. Chicago, University of Chicago Press, 1998.
- JORGENSEN, Neils. «Incremental and Decentralized Integration in FreeBSD», en *Perspectives on Free and Open Source Software*, Feller et al. (eds.), pp. 227–244. Cambridge (Mass.), MIT Press, 2004.
- «Putting It All in the Trunk: Incremental Software Development in the FreeBSD Open Source Project». *Information Systems Journal* 11.4 (2001), pp. 321–336.
- KAHN, Robert et al. «The Evolution of the Internet as a Global Information System.» *International Information and Libraries Review* 29 (1997), pp. 129–151.
- KAHN, Robert y CERF, Vint. «A Protocol for Packet Network Intercommunication.» *IEEE Transactions on Communications* Com-22.5 (mayo de 1974), pp. 637–644.
- KEATING, Peter y CAMBROSIO, Alberto. *Biomedical Platforms: Realigning the Normal and the Pathological in Late-twentieth-century Medicine*. Cambridge (Mass.), MIT Press, 2003.
- KELTY, Christopher (ed.). «Culture's Open Sources». *Anthropological Quarterly* 77.3 (verano de 2004), pp. 499–506. http://aq.gwu.edu/archive/table_summer04.htm.
- «Punt to Culture». *Anthropological Quarterly* 77.3 (verano de 2004), pp. 547–558.
- KENDALL, Lori. «'Oh No! I'm a NERD!' Hegemonic Masculinity on an Online Forum.» *Gender and Society* 14.2 (2000), pp. 256–274.

- KEVES, Brian William. «Open Systems Formal Evaluation Process». Artículo presentado en las Actas de la Asociación USENIX del Séptimo Congreso de Administración de Sistemas (LISA VII), Monterey (Calif.), pp. 1–5 (noviembre de 1993).
- KIDDER, Tracy. *The Soul of a New Machine*. Boston, Little, Brown, 1981.
- KIRKUP, Gill; JANES, Linda; WOODWARD, Kath y HOVENDEN, Fiona. *The Gendered Cyborg: A Reader*. Londres, Routledge, 2000.
- KITTNER, Friedrich. *Discourse Networks 1800/1900*, trad. por Michael Metteer, con Chris Cullens. 1985; reimpr., Stanford (Calif.). Stanford University Press, 1990.
- *Gramophone, Film, Typewriter*, trad. por Geoffry Winthrop-Young y Michael Wutz. 1986; reimpr., Stanford (Calif.), Stanford University Press, 1999.
- KLING, Rob. *Computerization and Controversy: Value Conflicts and Social Choices*. San Diego, Academic Press, 1996.
- KNUTH, Donald. *The Art of Computer Programming* (3^a ed.). Reading (Mass.), Addison-Wesley, 1997.
- KOHLER, Robert. *Lords of the Fly: Drosophila Genetics and the Experimental Life*. Chicago, University of Chicago Press, 1994.
- LACLAU, Ernesto y Chantal Mouffe. *Hegemony and Socialist Strategy*. Londres, Verso, 1985 [ed. cast.: *Hegemonía y estrategia socialista. Hacia una radicalización de la democracia*. Madrid, Siglo XXI Editores, 2015].
- LANDECKER, Hannah. *Culturing Life: How Cells Became Technologies*. Cambridge (Mass.), Harvard University Press, 2007.
- LATOUR, Bruno. «Drawing Things Together», en *Representation in Scientific Practice*, Michael Lynch y Steve Woolgar (eds.), pp. 19–68. Cambridge (Mass.), MIT Press, 1990.
- *Pandora's Hope: Essays on the Reality of Science Studies*. Cambridge (Mass.), Harvard University Press, 1999 [ed. cast.: *La esperanza de Pandora: ensayos sobre la realidad de los estudios de la ciencia*, trad. por Tomás Fernández Atíz. Barcelona, Gedisa, 2001].
- *Re-assembling the Social: An Introduction to Actor-Network Theory*. Oxford, Oxford University Press, 2005 [ed. cast.: *Reensamblar lo social: una introducción a la teoría del actor-red*, trad. por Gabriel Zadunaisky. Buenos Aires, Manantial, 2008].
- *Science in Action: How to Follow Scientists and Engineers through Society*. Cambridge (Mass.), Harvard University Press, 1987 [ed. cast.: *Ciencia en acción: cómo seguir a los científicos e ingenieros a través de la sociedad*, trad. por Eduardo Aibar, Roberto Méndez, Estela Ponisio. Barcelona, Labor, 1992].
- «What Rules of Method for the New Socio-scientific Experiments.» In *Experimental Cultures: Configurations between Science, Art and Technology 1830–1950, Conference Proceedings*, 123. Berlín, Max Planck Institute for the History of Science, 2001.

- LATOUR, Bruno y WEIBEL, Peter (eds.). *Making Things Public: Atmospheres of Democracy*. Cambridge (Mass.), MIT Press, 2005.
- LAW, John. *Aircraft Stories: Decentering the Object in Technoscience*. Durham (N.C.), Duke University Press, 2002.
- «Technology and Heterogeneous Engineering: The Case of Portuguese Expansion», en *The Social Construction of Technological Systems: New Directions in the Sociology and History of Technology*, W. E. Bijker, T. P. Hughes y T. J. Pinch (eds.), pp. 111–134. Cambridge (Mass.), MIT Press, 1987.
- LAW, John y HASSARD, John. *Actor Network Theory and After. Sociological Review Monograph*. Malden (Mass.), Blackwell 1999.
- LEACH, James; NAFUS, Dawn y KRIEGER, Berbard. *Gender: Integrated Report of Findings*. Free/Libre and Open Source Software: Policy Support (FLOSSPOLS) D 16, 2006. http://www.jamesleach.net/downloads/FLOSS-POLS-D16-Gender_Integrated_Report_of_Findings.pdf.
- LEE, J. A. N., R. M. Fano, A. L. Scherr, F. J. Corbato y V. A. Vyssotsky. «Project MAC.» *Annals of the History of Computing* 14.2 (1992), pp. 9-42.
- LERNER, Josh y TIROLE, Jean. «Some Simple Economics of Open Source.» *Industrial Economics* 50.2 (junio de 2002), pp. 197-234.
- LESSIG, Lawrence. *Code: Version 2.0*. Nueva York, Basic Books, 2006 [ed. cast.: *El Código 2.0*, ed. por Florencio Cabello, trad. por Maryam Itati Portillo et al. Madrid, Traficantes de Sueños, 2009].
- *Code and Other Laws of Cyber Space*. Nueva York, Basic Books, 1999. [ed. cast.: *El Código y otras leyes del ciberespacio*, trad. por Ernesto Alberola. Madrid, Taurus, 2001].
- *Free Culture: The Nature and Future of Creativity*. Nueva York, Penguin, 2003. [ed. cast.: *Por una cultura libre*, Javier Candeira (ed.), trad. por Antonio Córdoba/elastico.net. Madrid, Traficantes de Sueños, 2005].
- *The Future of Ideas: The Fate of the Commons in a Connected World*. Nueva York, Random House, 2001.
- «The New Chicago School.» *Legal Studies* 27.2 (1998), pp. 661–691.
- LEVY, Steven. *Hackers: Heroes of the Computer Revolution*. Nueva York, Basic Books, 1984.
- LIBES, Don y RESSLER, Sandy. *Life with UNIX: A Guide for Everyone*. Englewood Cliffs, (N.J.), Prentice Hall, 1989.
- LIGHT, Jennifer. «When Computers Were Women». *Technology and Culture*, 40.3 (julio de 1999), pp. 455-483.
- LIONS, John. *Lions'Commentary on UNIX 6th Edition with Source Code*. 1977; reimpr., San Jose, Peer to Peer Communications, 1996.
- LIPPmann, Walter. *The Phantom Public*. Nueva York, Macmillan, 1927 [ed. cast.: *Público fantasma*, trad. por César García Muñoz. Santander, Genueve Ediciones, 2011].
- LITMAN, Jessica. *Digital Copyright*. Nueva York, Prometheus Books, 2001.
- LIU, Alan. *The Laws of Cool: Knowledge Work and the Culture of Information*. Chicago, University of Chicago Press, 2004.

- MACKENZIE, Adrian. *Cutting Code: Software and Sociality*. Digital Formations Series. Nueva York, Peter Lang, 2005.
- MACKENZIE, Donald A. *Mechanizing Proof: Computing, Risk, and Trust*. Cambridge (Mass.), MIT Press, 2001.
- MAHONEY, Michael. «Finding a History for Software Engineering.» *Annals of the History of Computing* 26.1 (2004), pp. 8-19.
- «The Histories of Computing(s)». *Interdisciplinary Science Reviews* 30.2 (2005), pp. 119–135.
- «In Our Own Image: Creating the Computer», en *The Changing Image of the Sciences*, Ida Stamhuis, Teun Koetsier y Kees de Pater (eds.), pp. 9–28. Dordrecht, Kluwer Academic Publishers, 2002.
- «The Roots of Software Engineering.» *CWI Quarterly* 3.4 (1990), pp. 325-334.
- «The Structures of Computation», en *The First Computers: History and Architectures*, Raul Rojas y Ulf Hashagen (eds.), pp. 17-32. Cambridge (Mass.), MIT Press, 2000.
- MALAMUD, Carl. *Exploring the Internet: A Technical Travelogue*. Englewood Cliffs (N.J.), Prentice Hall, 1992.
- MANNHEIM, Karl. *Ideology and Utopia: Introduction to the Sociology of Knowledge*. Nueva York, Harcourt and Brace, 1946 [ed. cast.: *Ideología y utopía*, trad. por Salvador Echavarría, Madrid, Aguilar, 1973].
- MARCUS, George. *Ethnography through Thick and Thin*. Chicago, University of Chicago Press, 1998.
- MARCUS, George y CLIFFORD, James (eds). *Writing Culture: The Poetics and Politics of Ethnography*. Berkeley, University of California Press, 1986 [ed. cast.: *Retóricas de la antropología. Poética y política de la etnografía*, trad. por Jose Luis Moreno. Madrid, Júcar, 1991].
- MARCUS, George y FISCHER, Michael M. J. *Anthropology as Cultural Critique: An Experimental Moment in the Human Sciences*. Chicago, University of Chicago Press, 1986 [ed. cast.: *La antropología como crítica cultural. Un momento experimental en las ciencias humanas*, trad. por Eduardo Sinnott. Buenos Aires, Amorrortu, 2000].
- MARGOLIS, Jane y Allen Fisher. *Unlocking the Clubhouse: Women in Computing*. Cambridge (Mass.), MIT Press, 2002.
- MARTIN, James. *Viewdata and the Information Society*. Englewood Cliffs (N.J.), Prentice Hall, 1982.
- MATHESON, Peter. *The Imaginative World of the Reformation*. Edimburgo (Escocia) T and T Clark, 2000.
- MCKENNA, Regis. *Who's Afraid of Big Blue? How Companies Are Challenging IBM-and Winning*. Reading (Mass.), Addison-Wesley, 1989.
- MCKUSICK, M. Kirk. «Twenty Years of Berkeley Unix: From AT&T-owned to Freely Redistributable», en *Open Sources: Voices from the Open Source Revolution*, Chris Dibona *et al.* (eds.), pp. 31-46. ACM Sebastopol (Calif.), O'Reilly Press, 1999.

- MCLUHAN, Marshall. *The Gutenberg Galaxy: The Making of Typographic Man*. Toronto, University of Toronto Press, 1966 [ed. cast.: *La Galaxia Gutenberg*, trad. por Juan Novella, Barcelona, Círculo de Lectores, 1998].
- *Understanding Media: The Extensions of Man*. 1964; reimpr., Cambridge (Mass.), MIT Press, 1994 [ed. cast.: *Comprender los medios de comunicación. Las extensiones del ser humano*, trad. por Patrick Ducher. Barcelona, Paidós, 1996].
- MERGES, Robert; MENELL, Peter y LEMLEY, Mark (eds.). *Intellectual Property in the New Technological Age* (3^a ed.). Nueva York, Aspen Publishers, 2003.
- MERTON, Robert. «The Normative Structure of Science», en *The Sociology of Science: Theoretical and Empirical Investigations*, pp. 267-280. Chicago, University of Chicago Press, 1973 [ed. cast.: «La estructura normativa de la ciencia, en *La sociología de la ciencia: investigaciones teóricas y empíricas*, Norman W. Storer (comp.), Néstor Alberto Míguez (ed.), Madrid, Alianza, 1977】.
- MILLER, Daniel y SLATER, Don. *The Internet: An Ethnography*. Oxford, Berg, 2000.
- MISA, Thomas; BREY, Philip y FEENBERG, Andrew (eds.). *Modernity and Technology*. Cambridge (Mass.), MIT Press, 2003.
- MOCKUS, Audris; FIELDING, Roy T. y HERBSLEB, James. «Two Case Studies of Open Source Software Development: Apache and Mozilla». *ACM Transactions in Software Engineering and Methodology* 11.3 (2002), pp. 309–346.
- MOL, Annemarie. *The Body Multiple: Ontology in Medical Practice*. Durham (N.C.), Duke University Press, 2002.
- MOODY, Glyn. *Rebel Code: Inside Linux and the Open Source Revolution*. Cambridge (Mass.), Perseus, 2001.
- MOOERS, Calvin. «Computer Software and Copyright.» *Computer Surveys* 7.1 (marzo de 1975), pp. 45–72.
- MUELLER, Milton. *Ruling the Root: Internet Governance and the Taming of Cyberspace*. Cambridge (Mass.), MIT Press, 2004.
- NAUGHTON, John. *A Brief History of the Future: From Radio Days to Internet Years in a Lifetime*. Woodstock (N.Y.), Overlook Press, 2000.
- NOBLE, David. «Digital Diploma Mills: The Automation of Higher Education». *First Monday* 3.1 (5 de enero de 1998). http://www.firstmonday.org/issues/issue3_1/.
- NORBERG, Arthur L. y O'NEILL, Judy. *A History of the Information Techniques Processing Office of the Defense Advanced Research Projects Agency*. Minneapolis, Charles Babbage Institute, 1992.
- *Transforming Computer Technology: Information Processing for the Pentagon, 1962–1986*. Baltimore, Johns Hopkins University Press, 1996.
- ONG, Walter. *Ramus, Method, and the Decay of Dialogue: From the Art of Discourse to the Art of Reason*. 1983; reimpr., Chicago, University of Chicago Press, 2004.

- OSTROM, Elinor. *Governing the Commons: The Evolution of Institutions for Collective Action*. Cambridge (Mass.), Cambridge University Press, 1991. [ed. cast.: *El gobierno de los bienes comunes. La evolución de las instituciones de acción colectiva*, trad. por Corina de Iturbide Calvo y Adriana Sandoval. México D.F., UNAM/Centro Regional de Investigaciones Multidisciplinarias/Fondo de Cultura Económica, 2000].
- PERENS, Bruce. «The Open Source Definition», en *Open Sources: Voices from the Open Source Revolution*, Dibona et al. (eds.), pp. 171–188. Sebastopol (Calif.), O'Reilly Press, 1999. <http://perens.com/OSD.html> y <http://www.oreilly.com/catalog/opensources/book/perens.html>.
- PFAFFENBERGER, Bryan. «A Standing Wave in the Web of our Communications': USENet and the Socio-technical Construction of Cyberspace Values», en *From Usenet to CoWebs: Interacting with Social Information Spaces*, Christopher Lueg y Danyel Fisher (eds.), pp. 20–43. Londres, Springer, 2003.
- RABINOW, Paul. *Anthropos Today: Reflections on Modern Equipment*. Princeton, Princeton University Press, 2003.
- *Essays on the Anthropology of Reason*. Princeton, Princeton University Press, 1997.
- RATTO, Matt. «Embedded Technical Expression: Code and the Leveraging of Functionality». *Information Society* 21.3 (julio de 2005), pp. 205–13.
- «The Pressure of Openness: The Hybrid work of Linux Free/Open Source Kernel Developers» (tesis doctoral). University of California, San Diego, 2003.
- RAYMOND, Eric S. *The Art of UNIX Programming*. Boston, Addison-Wesley, 2004.
- *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. Sebastopol (Calif.), O'Reilly Press, 2001. Véase esp. «Homesteading the Noosphere», 79–135 [ed. cast.: «La catedral y el bazar», trad. por José Soto Pérez, *Biblioweb de sindomio.net*, disponible en: <http://biblioweb.sindominio.net/telematica/catedral.html>].
- ed. *The New Hackers' Dictionary*. (3^a ed.). Cambridge (Mass.), MIT Press, 1996.
- RHEINBERGER, Hans-Jörg. *Towards a History of Epistemic Things: Synthesizing Proteins in the Test Tube*. Stanford (Calif.), Stanford University Press, 1997.
- RHEINGOLD, Howard. *The Virtual Community: Homesteading on the Electronic Frontier*. ed. rev. 1993; (reimpresión), Cambridge (Mass.), MIT Press, 2000 [ed. cast.: *La comunidad virtual.*, trad. por Jose Ángel Álvarez. Madrid, Gedisa, 2009]
- RICOEUR, Paul. *Lectures on Ideology and Utopia*. Nueva York, Columbia University Press, 1986 [ed. cast.: *Ideologías y utopía*, George H. Taylor (comp.), trad. por Alberto L. Bixio. Madrid, Gedisa, 1989].
- RILES, Annelise. «Real Time: Unwinding Technocratic and Anthropological Knowledge». *American Ethnologist* 31.3 (agosto de 2004), pp. 392–405.

- RITCHIE, Dennis. «The UNIX Time-Sharing System: A Retrospective». *Bell System Technical Journal* 57.6, pt. 2 (julio-agosto de 1978). <http://cm.bell-labs.com/cm/cs/who/dmr/retroindex.html>.
- ROSE, Mark. *Authors and Owners: The Invention of Copyright*. Cambridge (Mass.), Harvard University Press, 1995.
- SALUS, Peter. *Casting the Net: From ARPANET to Internet and Beyond*. Reading (Mass.), Addison-Wesley, 1995.
- . *A Quarter Century of UNIX*. Reading (Mass.), Addison-Wesley, 1994.
- SCHMIDT, Susanne K. y WERLE, Raymund. *Coordinating Technology: Studies in the International Standardization of Telecommunications*. Cambridge (Mass.), MIT Press, 1998.
- SEGALLER, Stephen. *Nerds 2.0.1: A Brief History of the Internet*. Nueva York, TV Books, 1998.
- SHAIKH, Maha y CORNFORD, Tony. «Version Management Tools: CVS to BK in the Linux Kernel». Artículo presentado en el 21º Congreso Internacional sobre Ingeniería de Software - Taking Stock of the Bazaar: The Third Workshop on Open Source Software Engineering, Portland (Oregon), 3-10 de mayo de 2003.
- SHAPIN, Steven. *The Social History of Truth: Civility and Science in Seventeenth-Century England*. Chicago, University of Chicago Press, 1994.
- SHAPIN, Steven y SCHAFFER, Simon. *Leviathan and the Air Pump: Hobbes, Boyle and the Experimental Life*. Princeton, Princeton University Press, 1985.
- SMITH, Adam. *The Theory of Moral Sentiments*. 1759; (reimp.), Cambridge (Mass.), Cambridge University Press, 2002.
- STALLINGS, William. *Data and Computer Communications*. Londres, Macmillan, 1985.
- STALLMAN, Richard. «The GNU Manifesto». *Dr. Dobb's* 10.3 (marzo de 1985), pp. 30-35 [ed. cast.: «El Manifiesto GNU», en *Software libre para una sociedad libre*, ed. por Miquel Vidal, trad. por Jaron Rowan, Diego Sanz Paratcha y Laura Trinidad. Madrid, Traficantes de Sueños, 2004, pp. 45-58].
- ST. AMOUR, Paul K. *The Copywrights: Intellectual Property and the Literary Imagination*. Ithaca (N.Y.), Cornell University Press, 2003.
- STAR, Susan Leigh (ed.). *The Cultures of Computing*. Malden (Mass.), Blackwell, 1995.
- STAR, Susan Leigh y Karen Ruhleder. «Steps towards an Ecology of Infrastructure: Complex Problems in Design and Access for Large-scale Collaborative Systems». *Information Systems Research*, 7, 1996, pp. 111-133.
- STEPHENSON, Neal. *In the Beginning Was the Command Line*. Nueva York, Avon/Perennial, 1999 [ed. cast.: *En el principio fue... la línea de comandos*, ed. por Miquel Vidal, trad. por Asunción Álvarez. Madrid, Traficantes de Sueños, 2003].

- SUNSHINE, Carl. *Computer Network Architectures and Protocols* (2^a ed.). Nueva York, Plenum Press, 1989.
- TAKAHASHI, Shigeru. «The Rise and Fall of the Plug Compatible Manufacturers». *IEEE Annals of the History of Computing* (enero–marzo de 2005), pp. 4–16.
- TANENBAUM, Andrew. *Computer Networks* (1^a ed.). Upper Saddle River (N.J.), Prentice Hall, 1981. [ed. cast.: *Redes de computadoras* (4^a ed.), trad. por Elisa Núñez Ramos. México D.F., Pearson Education, 2003]
- *Operating Systems: Design and Implementation* (1^a ed.). Englewood Cliffs (N.J.), Prentice Hall, 1987 [ed. cast.: *Sistemas operativos: Diseño e implementación* (2^a ed.), trad. por Roberto Escalona. México D.F., Prentice-Hall, 1998].
 - «The UNIX Marketplace in 1987: Life, the UNIverse and Everything», en *Proceedings of the Summer 1987 USENIX Conference*, pp. 419–424. Phoenix (Ariz.), USENIX, 1987.
- TAYLOR, Charles. *Modern Social Imaginaries*. Durham (N.C.). Duke University Press, 2004 [ed. cast.: *Imaginarios sociales modernos*, trad. por Ramón Vilà Vernis. Barcelona, Paidós, 2006].
- «Modes of Civil Society» *Public Culture* 3.1 (1990), pp. 95–132.
 - *Multiculturalism and the Politics of Recognition*. Princeton (N.J.), Princeton University Press, 1992.
 - *Sources of the Self: The Making of the Modern Identity*. Cambridge (Mass.), Harvard University Press, 1989.
- THOMPSON, Charis. *Making Parents: The Ontological Choreography of Reproductive Technologies*. Cambridge (Mass.), MIT Press, 2005.
- THOMPSON, Ken y Dennis. «The UNIX Time-Sharing System.» *Communications of the ACM* 17.7 (julio de 1974), pp. 365–375.
- TICHY, Walter F. «RCS: A System for Version Control.» *Software: Practice and Experience* 15.7 (julio de 1985), pp. 637–654.
- TORVALDS, Linus, con David Diamond. *Just for Fun: The Story of an Accidental Revolutionary*. Nueva York, HarperCollins, 2002.
- TUOMI, Ilkka. *Networks of Innovation: Change and Meaning in the Age of the Internet*. Nueva York, Oxford University Press, 2002.
- TURING, Alan. «On Computable Numbers, with an Application to the Entscheidungsproblem» *Proceedings of the Londres Mathematical Society* 2.1 (1937), p. 230.
- TURKLE, Sherry. *Life on the Screen: Identity in the Age of the Internet*. Nueva York, Simon and Schuster, 1995 [ed. cast.: *La vida en la pantalla. La construcción de la identidad en la era de Internet*, trad. por Laura Trafi. Barcelona, Paidós Ibérica, 1997].
- *The Second Self: Computers and the Human Spirit*. Nueva York, Simon and Schuster, 1984.
- TURNER, Fred. *From Counterculture to Cyberculture: Stewart Brand, the Whole*

- Earth Network, and the Rise of Digital Utopianism*. Chicago, University of Chicago Press, 2006.
- «Where the Counterculture Met the New Economy». *Technology and Culture* 46.3 (julio de 2005), pp. 485–512.
- ULLMAN, Ellen. *The Bug: A Novel*. Nueva York, Nan A. Talese, 2003.
- *Close to the Machine: Technophilia and Its Discontents*. San Francisco: City Lights, 1997.
- VAIDHYANATHAN, Siva. *Copyrights and Copywrongs; The Rise of Intellectual Property and How It Threatens Creativity*. Nueva York, New York University Press, 2001.
- VETTER, Greg R. «The Collaborative Integrity of Open-Source Software». *Utah Law Review* 2004.2 (2004), pp. 563–700.
- «Infectious Open Source Software: Spreading Incentives or Promoting Resistance?» *Rutgers Law Journal* 36.1 (otoño de 2004), pp. 53–162.
- VINGE, Vernor. «The Coming Technological Singularity: How to Survive in the Post-Human Era» (1993). <http://www.rohan.sdsu.edu/faculty/vinge/misc/singularity.html> (último acceso: 18 de agosto de 2006).
- VON HIPPEL, Eric. *Democratizing Innovation*. Cambridge (Mass.), MIT Press, 2005.
- WAJCMAN, Judy. *Feminism Confronts Technology*. Cambridge, Polity, 1991.
- «Reflections on Gender and Technology Studies: In What State Is the Art?» *Social Studies of Science* 30.3 (2000), pp. 447–464.
- WALDROP, Mitchell. *The Dream Machine: J. C. R. Licklider and the Revolution that Made Computing Personal*. Nueva York, Viking, 2002.
- WALSH, John y BAYMA, Todd. «Computer Networks and Scientific Work.» *Social Studies of Science* 26.3 (agosto de 1996), pp. 661–703.
- WARNER, Michael. *The Letters of the Republic: Publication and the Public Sphere in Eighteenth-century America*. Cambridge (Mass.), Harvard University Press, 1990.
- «Publics and Counterpublics.» *Public Culture* 14.1 (2002), pp. 49–90.
- *Publics and Counterpublics*. Nueva York, Zone Books, 2003 [ed. cast: *Públicos y contrapúblicos*, trad. por Miguel Martínez-Lage. Barcelona, MACBA, 2008].
- WAYNER, Peter. *Free for All: How LINUX and the Free Software Movement Undercut the High-Tech Titans*. Nueva York, Harper Business, 2000.
- WEBER, Max. «Objectivity in the Social Sciences and Social Policy», en *The Methodology of the Social Sciences*, trad. y ed. por Edward Shils y Henry A. Finch, pp. 50-112. Nueva York, Free Press, 1949.
- WEBER, Steven. *The Success of Open Source*. Cambridge (Mass.), Harvard University Press, 2004.
- WEXELBLAT, Richard L. (ed.). *History of Programming Languages*. Nueva York, Academic Press, 1981.
- WILLIAMS, Sam. *Free as in Freedom: Richard Stallman's Crusade for Free Software*. Sebastopol (Calif.), O'Reilly Press, 2002.

- WILSON, Fiona. «Can't Compute, Won't Compute: Women's Participation in the Culture of Computing.» *New Technology, Work and Employment* 18.2 (2003), pp. 127–142.
- WILSON, Samuel M. y PETERSON, Leighton C. «The Anthropology of Online Communities.» *Annual Reviews of Anthropology* 31 (2002), pp. 449–467.
- XIANG, Biao. «*Global Bodyshopping: An Indian Labor System in the Information Technology Industry*. Princeton, Princeton University Press, 2006.
- ŽIŽEK, Slavoj (ed.). *Mapping Ideology*. Londres, Verso, 1994 [ed. cast.: *Ideología: Un mapa de la cuestión*, trad. por Cecilia Beltrame *et al.*. Buenos Aires, Fondo de Cultura Económica, 2003].

