ES **Exploraciones sobre la cultura del *software* libre o abierto como metáfora de la colaboración en la sociedad en red.**

EN **Explorations of the free or open *software* as a metaphor of society's collaboration in the Web.**

ITA **Esplorazioni sulla cultura del *software* libero / aperto come metafora della collaborazione nella società interconnessa.**

FRA **Explorations sur la culture du logiciel libre comme métaphore de la collaboration dans la société en réseaux.**

POR **Explorações sobre a cultura do *software* livre ou aberto como metáfora da colaboração na sociedade em rede.**

Luis Fernando Medina Cardona

Exploraciones sobre la cultura del *software* libre o abierto como metáfora de la colaboración en la sociedad en red

LUIS FERNANDO MEDINA CARDONA

Profesor Asociado, Escuela de Cine y TV, Universidad Nacional de Colombia.
E-mail: lfmedinac@unal.edu.co

RESUMEN (ESP)

Este artículo presenta un análisis sobre la cultura del software, en particular, del llamado software libre o abierto. Para ello establece como base la metáfora de la colaboración, es decir, revisa cómo su modelo de producción se ha convertido en una plantilla para otros procesos y disciplinas. Dado que el objeto de análisis es complejo y permea varios aspectos actuales, el abordaje metodológico es interdisciplinar y emplea la perspectiva de los estudios de software, con la ayuda de los estudios en ciencias, tecnología, sociedad, la ingeniería de software y la arqueología de medios. Así, se suministran varios ejemplos históricos de colaboración tanto en el software como en las redes de computadores. Finalmente, se discuten algunas definiciones como red sociotécnica, colectivo y los tres tipos de agencia que históricamente se han visto en el software. Particularmente, la agencia distribuida es de ayuda para contrastar la metáfora de la colaboración en un caso de estudio futuro representado por la cultura maker.

PALABRAS CLAVES: *estudios de software, estudios en ciencia y tecnología, software libre o de código abierto* FLOSS*, cultura maker, sociedad en red.*

ABSTRACT (ENG)

This article offers an analysis of software culture, more specifically of the so-called free or open software. To this effect, it establishes as its base the metaphor of collaboration, that is, it shows how its production model has become a template for other processes and disciplines. Given that the object of analysis is complex and permeates several present-day aspects, the methodological approach is interdisciplinary and uses the perspective of software studies, together with studies in science, technology, society, software engineering and media archeology. Thus, several historical examples of collaboration are offered, both in software and in computer webs. Finally, some definitions such as that of the socio-technical web, collective and the three types of agency historically present in software are discussed. More specifically, distributed agency is helpful for contrasting the collaboration metaphor in a future study case represented by the maker culture.

KEYWORDS: *software studies, studies in science and technology, free or open code software* FLOSS*, maker culture, web society.*

RIASSUNTI (ITA)

L'articolo presenta l'analisi sulla cultura del software, in particolare di quello conosciuto come libero oppure aperto. Sulla base della metafora della collaborazione, si riflette sul modello di produzione che è diventato sagoma per diversi processi e discipline. L'approccio metodologico è interdisciplinare, si serve della prospettiva non solo degli studi di software odierni, ma anche degli studi in scienze, tecnologia, società, l'ingegneria del software e l'archeologia dei media, visto che l'oggetto d'analisi è piuttosto complesso e riguarda diversi aspetti dell'attualità. Si presentano alcuni esempi storici di collaborazione così in software come in reti informatiche. Verso la fine, si considerano alcune definizioni: rete sociotecnica, collettivo e i tre tipi di agenzia che storicamente hanno esistito nei software. L'agenzia distribuita serve per contrastare la metafora della collaborazione in un caso di studio di futuro rappresentato dalla cultura maker

PAROLE CHIAVI: *studi di software, studi in scienza e tecnologia, software libero o di codici aperti* FLOSS*, cultura maker, società in rete.*

RÉSUMÉ (FRA)

Cet article présente une analyse au sujet de la culture du logiciel, en particulier le logiciel libre ou ouvert. Pour cela on prend comme base la métaphore de la collaboration et on examine comment son modèle de production est devenu un patron pour d'autres processus et disciplines. L'objet de l'analyse étant complexe et touchant à divers aspects actuels, l'approche méthodologique est interdisciplinaire, on adopte la perspective des études en informatique avec l'appui des études en science, technologie et société, du génie logiciel et de l'archéologie des médias. On expose divers exemples historiques de collaboration dans le logiciel et les réseaux d'ordinateurs. Enfin, on examine quelques définitions, comme réseau sociotechnique, collectif et les trois types d'architecture que le logiciel a connus dans son histoire. En particulier, l'architecture distribuée met en évidence la métaphore de la collaboration dans un cas d'étude futur représenté par la culture maker.

MOTS-CLÉS: *études de logiciel, études en science et technologie, logiciel libre ou à code source ouvert* FLOSS, *culture maker, société en réseaux.*

RESUMO (POR)

Este artigo apresenta uma análise sobre a cultura do software, particularmente, o denominado software livre ou aberto. Para isso, estabelece como base a metáfora da colaboração, isto é, revê a forma como seu modelo de produção tornou-se modelo para outros processos e disciplinas. Visto que objeto de análise é complexo e permeia várias questões atuais, a abordagem metodológica é interdisciplinar e emprega a perspectiva dos estudos de software, com base nos estudos em ciências, tecnologia, sociedade, a engenharia de software e arqueologia de mídia. Portanto, apresentam-se vários exemplos históricos de colaboração tanto no software quanto nas redes de computadores. Finalmente, discutem-se algumas definições como rede sociotécnica, coletivo e os três tipos de agência historicamente vistos no software. Particularmente, a agência distribuída é útil no contraste da metáfora da colaboração em um estudo de caso futuro representado pela cultura maker.

PALAVRAS-CHAVES: *estudos de software, estudos em ciência e tecnologia, software livre ou de código aberto* FLOSS, *cultura maker, sociedade em rede.*

Introducción: código y *software* como objeto de estudio

El mundo moderno está mediado por el *software*. Todos los días, varias de las tareas que muchas personas realizan (en el trabajo o en la vida privada) o que se llevan a cabo de manera transparente (como la semaforización) son controladas directamente por programas informáticos o por máquinas que en algún momento negociarán con un dispositivo controlado por un *software*. En pocas palabras, hoy en día, si una persona no está interactuando con el *software*, o incluso no se ve afectada indirectamente por él, las posibilidades de que esta persona viva completamente fuera de la sociedad global moderna son muy altas (Kitchin y Dodge, 2011). En consecuencia, el *software* como medio, que a su vez tiene la capacidad de simular varios otros medios de comunicación (Manovich, 2013), se ha convertido en un punto de interés académico que requiere un enfoque interdisciplinario. Sin embargo, el *software* sigue siendo un tema de difícil acceso, dada su naturaleza altamente técnica y la tendencia a destacar solamente sus impactos, dejando su lógica interna por fuera de los análisis.

Entre las actividades que se han beneficiado del *software* está la colaboración, tanto en su propio desarrollo como en las herramientas que usa. Por tanto, la colaboración en el proceso de desarrollo de *software* debe ser vista desde una perspectiva técnica y social, para con ello aportar al estudio de sus posibles consecuencias culturales en una sociedad que depende de las redes informáticas. En particular, el foco de interés es el llamado *software* libre o de código abierto, que puede considerarse como un concepto multicapa que describe al mismo tiempo un tipo específico de aplicación informática, así como un movimiento cultural y una metodología técnica para desarrollar *software*. El *software* libre o de código abierto, además de sus supuestas cualidades técnicas, ofrece varias ventajas para teorizar, como se discutirá más adelante. También tiene una fuerte carga política, de manera que es un tema álgido en debates actuales relacionados con el acceso a la tecnología, los derechos humanos, las libertades civiles en Internet, entre otros.

A pesar de esta motivación, el objetivo es estudiar la colaboración de forma más global —esto es, abordar la cultura del *software* en general— en lugar de solamente el denominado FLOSS[1]. Esto es una forma de señalar que solo a partir de los años ochenta del siglo XX se acuñó este último término, aunque los patrones, procedimientos y herramientas de colaboración han estado presentes desde el principio de la historia del *software* en sí mismo. Complementando los términos detrás de la motivación, *cultura* se utiliza para transmitir una comprensión más amplia de la aplicación informática que sobrepasa el ámbito técnico (sin dejarlo atrás) para evaluar las prácticas sociales relacionadas con las propiedades centrales del *software* construido colaborativamente. Por último, la *sociedad en red* encarna también una percepción de las redes informáticas como «metáfora» para describir los fenómenos modernos de interconexión y comunicación que muchas veces escapan a las definiciones informáticas. En este sentido y con respecto al título de este análisis, el uso de la palabra *metáfora* tiene dos propósitos. El primero es el coloquial, que se refiere a cómo la colaboración en el *software* se usa como un modelo cuya estructura, prácticas y conceptos son adaptados para explicar comportamientos colaborativos en otras disciplinas. Y el segundo apunta a la naturaleza lingüística y material que la operación de isomorfismo y correspondencia representada en la metáfora está en concordancia con el propio substrato lingüístico del *software* (representado en el código y otros artefactos textuales) y sus propias metáforas (por ejemplo, la programación orientada a objetos).

Un análisis completo de los temas expresados superaría el espacio disponible, por lo que se ha optado por una narrativa no secuencial donde se entrecruzan las posturas metodológicas y los resultados por igual. En este orden de ideas, este artículo se inscribe en la disciplina conocida como «estudios de *software*», trayendo con ello una serie de premisas como punto de partida. En la introducción se realiza un recuento general sobre este floreciente campo (Fuller, 2008), se dan dos argumentos que son útiles para el objetivo trazado de estudiar la colaboración mediada por el *software*. En primer lugar, los estudios de *software* son de naturaleza interdisciplinaria, donde la informática, la sociología de la tecnología, los estudios culturales, la teoría de los medios de comunicación, la ingeniería de *software*, entre otros enfoques, encuentran un lugar para dialogar y discutir sobre las aplicaciones informáticas y sus implicaciones. En segundo lugar, el *software* libre es especialmente interesante, teniendo en cuenta sus características abiertas, lo que facilita el análisis y la observación. El acceso al código fuente de las aplicaciones y su documentación, la abundancia de métricas de *software* que describen varios aspectos de los procesos e incluso el

1 **Acrónimo utilizado para denotar el *software* libre o de código abierto y que hace referencia a *free* [libre], *open source software*.**

estado de ánimo de los profesionales, pueden ser de gran utilidad para abordar el tema principal. Ambas premisas —constitutivas de los estudios de *software*— sustentan por ahora un enfoque dual en el que la sociología y la filosofía de la tecnología, por un lado, y la ingeniería de *software* de FLOSS, por el otro, pueden establecer un terreno común sobre el cual situarse. De esta manera, conceptos que provienen de contextos diferentes, pero relacionados progresivamente, se irán entramando para revelar tanto las estructuras subyacentes como la narrativa a elegir. Por ello, este artículo se ha dividido en tres secciones: la primera profundiza en lo que se entiende por *metáfora* y cómo los estudios de *software* se relacionan con otras disciplinas. La segunda y tercera exploran en clave de arqueología de medios la evidencia de colaboración en el tándem *software*-redes, previo a la aparición de la retórica actual basada en el FLOSS. Finalmente, a manera de conclusión, se presentan algunos de los conceptos elaborados, así como un caso de estudio en la llamada cultura *maker*.

LA METÁFORA DE COLABORACIÓN EN FLOSS Y SUS IMPLICACIONES

Como se ha descrito anteriormente, puede decirse que, en términos generales, el *software* libre o de código abierto (FLOSS) es el punto de partida moderno para entender conceptos como colaboración, código y desarrollo distribuido desde una perspectiva más amplia. *FLOSS* hace referencia al conjunto de programas que cumplen ciertos principios conocidos como las cuatro libertades (*software* libre) o nueve definiciones (código abierto)[2]. La identidad de dicho tipo de *software* puede resumirse usando la definición más básica que es la del *software libre* y que está sustentada en estos cuatro principios o libertades: el *software* puede ejecutarse sin ninguna restricción para cualquier propósito; la capacidad de adaptar el *software* dado el acceso para leer el código fuente; la libertad de distribuir copias de un programa y la libertad de modificar el código fuente y redistribuir las mejoras (Stallman, 2002, p. 43). El impacto de estas ideas fundacionales de todo un movimiento —y que pueden rastrearse hasta la llamada ética *hacker* de los *hackers* del MIT o incluso hasta los albores de los computadores electrónicos y los comienzos del *software*— ha sido enorme. El éxito de la metáfora de la colaboración en FLOSS que ha sido aplicada a varios campos, como el arte, la música, la educación, la medicina, la economía, la arquitectura, etc., ejemplifica las ventajas percibidas de colaborar utilizando un entorno de red (sea o no digital), tareas distribuidas, bienes culturales libremente accesibles y modificables, y procesos iterativos. El *software* está en todas partes no solo como una herramienta, sino también como una forma de pensar, lo cual refuerza posiciones de la filosofía de la tecnología recientes como la de «tecnicidad originaria» (Stiegler, 1998), donde la tecnología no se ve desde un ángulo instrumentalista, como un medio para un fin (es decir, una mera herramienta), sino como una demostración íntima de la condición humana.

Precisamente, el éxito de dicha metáfora requiere enfoques serios para comprender por qué es tan atractiva, haciendo de este tipo de *software* una especie de plantilla para varios procesos. Como concluye Eric S. Raymond, uno de los profesionales y principales defensores del FLOSS (especialmente del *software* abierto), cuando habla de Linux, su contribución no es solamente representar un sistema operativo libre, sino proporcionar todo un nuevo modelo de desarrollo de *software*. Ante este planteamiento surgen varias preguntas intuitivas: ¿cuáles son las características claves del FLOSS para fomentar la colaboración y construir esta metáfora que se está interpretando desde diferentes puntos de vista y cómo son los procesos necesarios para adaptar este modelo a campos tan diversos? ¿Se trata de una transición sin esfuerzo o, por el contrario, de un camino lleno de baches y opaco, como el mismo código de computador es percibido por algunos teóricos como, por ejemplo, Kittler? (Kittler, 1997). Un refinamiento de estas ideas permitió llegar a esta enunciación: ¿cómo evolucionaron las prácticas, metodologías y artefactos que permiten la colaboración en la cultura de *software* hasta que fueron plasmados en la actual metáfora de código abierto y cuáles son los problemas de traducir esta metáfora y su ética de trabajo a otros campos?

Como se discutió en la primera sección, los estudios de *software* fueron seleccionados como el marco teórico. Sin embargo, este sigue siendo un campo muy amplio y requiere herramientas más particulares para emprender un trabajo fructífero. Por esta razón, la pareja más precisa de ingeniería de *software* y sociología de la tecnología fue seleccionada. Sin embargo, lejos de ser consideradas como un problema, estas dificultades suscitan puntos de interés que pueden mejorar los resultados globales si son bien entendidos en sus limitaciones y oportunidades. De esta manera, se puede mantener una posición crítica para todo el proyecto. En particular, la ingeniería de *software* es en sí misma una metáfora procedente de otras prácticas como la ingeniería civil, introducida en el mundo del *software* desde 1968 en la conocida conferencia «*Software Engineering*» y convocada por la OTAN en Garmisch, Alemania (Gehl y Bell, 2012).

2 **Hay una ligera diferencia entre los dos conceptos. Mientras que el *software* libre hace hincapié en cuestiones políticas y éticas, el *código abierto* es más pragmático y destaca la conveniencia técnica de un código abierto legible/escribible. La mayoría de las veces esta distancia es ignorada y ambos enfoques considerados por la sigla más general que los aglutina: FOSS o FLOSS**

Hasta la conferencia de Garmish, la programación informática era considerada una especie de arte, alejada de la formalización y la estructuración. A medida que las aplicaciones informáticas se hicieron más grandes y complejas, se requirió un enfoque formal para su desarrollo. Sin embargo, para ver las tensiones que ello generó, considérese el modelo más renombrado de producción de *software*: el llamado modelo de cascada. Este modelo divide todo el modo de producción de *software* en etapas lineales y consecutivas agrupando las tareas en etapas denominadas *análisis*, *diseño*, *programación* y *puesta en marcha* de un producto de *software*. No es de extrañar que este modelo nunca haya cumplido las expectativas, lo que motivó a que varios modelos hayan aparecido como alternativas para la construcción de *software*. Como es común afirmar en este campo, el *software* está en una «crisis» constante, como de hecho fue planteado en la conferencia de Garmish. Aquí, el FLOSS juega un papel muy importante, mostrando otra lógica más «anárquica» (Taubert, 2006), donde conceptos como *remuneración monetaria*, *jerarquías* y *producto* son interpretados de forma completamente diferente. Como se expresa en otra metáfora, si el *software* comercial se desarrolla como una catedral, con maestros y artesanos y miles de tareas organizadas, FLOSS se asemeja a un bazar, donde el libre intercambio de ideas fluye y —aparentemente— no hay una jerarquía establecida. Y aunque dicha imagen dicotómica es problemática y simplista, reconoce el carácter revolucionario de las prácticas abiertas cuando se comparan con los modos clásicos de producción de programas (como el que representa el modelo cascada).

Para la otra parte de la máquina metodológica propuesta, los estudios de *software* frecuentemente acuden al trabajo de Bruno Latour y su teoría actor-red (ANT por sus siglas en inglés), el cual es, a su vez, uno de los componentes claves de los estudios que abordan las relaciones entre ciencia, tecnología y sociedad. Aunque este enfoque no está exento de críticas, ofrece una visión prometedora de la tecnología que puede ser aplicada al *software*. Para dar solo un par de ejemplos breves, el concepto de *red compuesta de actores*, que comprende elementos humanos y no humanos en la ANT, se asemeja a la diversidad de actores, artefactos conceptuales y digitales que se articulan en la colaboración llevada a cabo para producir una pieza de *software*. De manera similar, la ANT propone un método iterativo que puede ser comparado con modelos iterativos en el desarrollo de *software*. Como se ha señalado, la relación entre los códigos informáticos y Latour se describe en Gehl y Bell (2012) y Kitchin y Dodge (2011).

Finalmente, la mezcla metodológica donde la ingeniería de *software*, la filosofía de la tecnología y los desarrollos de Latour conviven bajo el aglutinante general de los estudios de *software* es complementada por la arqueología de medios. Este último componente permite buscar, analizar y organizar todo el rico corpus documental que la industria del *software* ha tenido desde sus albores, en los años cincuenta del siglo pasado, y que se encuentra en manuales, publicaciones técnicas, artículos científicos, etc. La arqueología de medios ha sido seleccionada porque en parte de su genealogía, como disciplina teórica, se encuentra la indagación por artefactos de *software* extintos (Zielinski, 2012, p. 4) y porque su método se enmarca dentro del «giro material», el cual es clave para entender la argumentación. De esta manera, la superestructura conceptual más abstracta de los estudios de *software* se ve sustentada con datos y evidencia obtenida de reflexiones materializadas en piezas de *software* y redes de computadores vistas bajo una óptica historiográfica.

Una vez que el *software* libre y abierto se ha establecido como una cultura sociotécnica con un arraigo histórico en la colaboración, se debe avanzar en otro aspecto para completar el propósito de la investigación de evaluar la metáfora de la colaboración en FLOSS y su impacto. En este sentido, la idea es realizar un mapeo entre los aspectos clave de FLOSS en otro campo, como una herramienta para evaluar las dificultades de esta transición y proporcionar una mayor comprensión no solo del propio FLOSS, sino también de este *open-sourcing* de varios discursos sobre los nuevos medios, dadas algunas aseveraciones teóricas según las cuales los nuevos medios son considerados como todo aquello que se ejecuta mediante *software* (Frabetti, 2015). Como se estableció, esto confiere al código de computador una importancia epistemológica que trasciende el instrumentalismo, más aún si es observable y modificable, como es el caso del código libre y abierto. Para este mapeo sería deseable tener un caso de estudio, con el fin de evaluar estas transiciones con el beneficio adicional de reducir el alcance a un solo ejemplo entre la rica oferta actual de los llamados *fenómenos de fuente abierta* o *libre*. Para ello, se ha seleccionado el movimiento *maker* (Walter-Hermann y Büching, 2013). Actualmente, este término —aunque de indiscutible origen comercial— se ha normalizado para abarcar las prácticas y herramientas —toda una subcultura sociotécnica— donde se emplea un modo de producción descentralizado, un enfoque colaborativo y un patrón de comunicación en red para crear objetos y donde se utilizan tecnologías como las impresoras 3D, cortadoras láser, fresadoras, entre otras herramientas de relativo bajo presupuesto.

Como manifestación del movimiento *maker*, ha surgido una serie de puntos de encuentro denominados *makerspaces* (un concepto emparentado con los *hackerspaces*) en donde temas como el *hardware* abierto, el diseño abierto y una ética del bricolaje, similar a la ética del *hacker* (presente en varias etapas de la saga del *software* libre o de código abierto, como se expondrá brevemente en la próxima sección), constituyen un ejemplo perfecto de literal «materialización» de la metáfora del

software, aunque teóricamente dicha materialización sea problemática. Como ejemplo de esta tensión, considérese la declaración del popular escritor de ciencia ficción Cory Doctorow en la que afirma que «compartir información es la norma, donde la ética y las prácticas del movimiento de *software* libre/de código abierto se han vuelto físicas» (Samtani, 19 de junio de 2013). El autor pone de manifiesto la dicotomía que sitúa al *software* como algo inmaterial e inasible en oposición a lo físico y tangible. De otro lado, la materialidad de la cultura *maker* y de los espacios de fabricación distribuida puede plantear varios retos a la hora de traducir la lógica del *software* libre o de código abierto en este campo, donde esta lógica no es una simple herramienta, sino un modelo. Con ello se despliega otra razón profunda y de carácter argumentativo: la dicotomía inmaterial/material que por años ha permeado la retórica del *software* (es decir, que el *software* es inmaterial) y que ha sido puesta en tela de juicio por el denominado «giro material», un movimiento epistemológico de corte posmarxista en el que se encuentran los *science and tecnology studies* (STS) y al cual los estudios de *software* se adhieren. Como manifestación de ello, desde la antología seminal sobre estudios de *software*, editada por Mathew Fuller (2008), se hace énfasis en la inconveniencia de dicha dicotomía y de su corolario del *software* como inmaterial, el cual aún persiste en imágenes populares como «computación en la nube» y «trabajo inmaterial», por mencionar solo algunas.

COLABORACIÓN EN LA HISTORIA DEL TÁNDEM SOFTWARE/REDES

Una vez establecido el marco general de los estudios de *software* y la arqueología de medios como estrategia general para la revisión documental, se procedió a rastrear la colaboración en la historia del *software*. Esto plantea un marco temporal que abarca desde los años cuarenta del siglo pasado hasta nuestros días, aunque por cuestiones de complejidad se ha dejado por fuera la ola de computación móvil, pues se parte de la premisa de que, aunque esta ha cambiado notablemente las plataformas y la manera de distribución del *software*, no representa un cambio abrupto en cómo el *software* es construido. Otro aspecto para destacar y que está implícito en el nombre de esta sección es la interdependencia entre *software* y redes de computadores. Si bien tienen un origen distinto, su desarrollo posterior a los años sesenta está íntimamente ligado, formando una dupla (o tándem) que desafía divisiones dicotómicas, al igual que otras ya comunes como *software-hardware*. En otras palabras, epistemológicamente, dicha división trae más problemas que respuestas, como atestigua el desarrollo simbiótico del *software* y las redes de computadores en la industria informática moderna, es decir, no pueden concebirse las redes sin el *software* y viceversa. Como detalle anexo que confirma esta relación, cabe anotar que tanto el *software* como las redes de computadores tienen parte de su origen en discusiones académicas y en el espíritu de colaboración que las alienta. Sin embargo, esto ha derivado en una tensión discursiva que llega al centro mismo de la concepción de qué es una tecnología. Por un lado, existe la constante tendencia a sistematizar la programación y la construcción de *software* en general desde un enfoque cientificista, como por ejemplo en las denominadas *ciencias de la computación*. De otro lado, existe una corriente que hace énfasis en el componente pragmático del desarrollo de *software*, donde se compara más con una disciplina artística que con la ciencia. Aunque esfuerzos como el de la ingeniería de *software* han procurado acercar esta práctica al primer enfoque, el comportamiento de esta cultura sociotécnica del *software* sigue mostrando la dificultad de dicho empeño. El objetivo de rastrear la historia de la colaboración en el *software* también parece confirmar dicha posición.

La lectura comercial moderna de la colaboración en la industria del *software* generalmente lleva a una definición idealista y lineal de cómo esta aparece. Se asume que hubo dos periodos consecutivos: uno en donde el *software* era comercial y otro posterior en donde aparece la definición moderna de *software* libre. Por supuesto, dicha postura ha sido cuestionada pero no hay un consenso en las alternativas propuestas. Por un lado, hay historiadores de la informática como Campbell Kelly que, en su estudio sobre la cultura corporativa de IBM, identifica varias prácticas de la década de 1950 como prácticas de *software* libre (más de treinta años antes de que tal concepto existiese) (Campbell-Kelly y García-Swartz, 2009). Por otro lado, hay estudiosos de la cultura FLOSS que señalan que dicha época ha sido retratada desde una visión romántica de la industria y que nunca existió una comunidad homogénea que girara alrededor de valores colaborativos en el desarrollo del *software*. Lo que sí ha sido claro, tras el análisis de los distintos ejemplos seleccionados, es que: a) la historia del desarrollo de *software* no es lineal y b) la colaboración en la industria del *software* se debate en una tensión que fue etiquetada por el padre del *software* libre, Richard Stallman, como un «idealismo pragmático», es decir, la colaboración se practica ya sea porque se considera la forma más eficiente de desarrollar *software* (parte pragmática) o porque hay un trasfondo ético y moral a su alrededor (parte idealista). Partiendo de estas premisas, se presentan los siguientes ejemplos que son relatados en forma cronológica para mayor sencillez.

En primer lugar, la invención del computador electrónico se dio cuando ni siquiera existía el *software* como tal o como concepto. Dicho artefacto surgió en un periodo donde la disciplina —aún en desarrollo— de la cibernética promulgaba el libre flujo de información (básico para la colaboración) basado en los principios

de circulación y retroalimentación (Brate, 2002, p. 28). Específicamente, el desarrollo de los primeros computadores digitales en los años cuarenta requirió de una comunidad dispuesta a compartir diseños para avanzar colectivamente, como puede atestiguarlo el diseño del EDVAC, un computador cuyo principio de programa almacenado (antecesor del *software*) ya estaba incluido y cuyo diseño fue distribuido para comentarios siguiendo la apertura de la tradición académica (Isaacson, 2014, p. 111). Estrictamente hablando de la construcción de *software*, dos esfuerzos tempranos exhiben rasgos colaborativos con propósitos pragmáticos. Antes de que la palabra *software* fuera acuñada, existió el *lenguage* PACT (*Project of the Advancements of Coding Techniques*) como proyecto de una pequeña comunidad para reducir costos de desarrollo de *software*, los cuales ya para la época (años cincuenta) eran costosos (Melahn, 1956, p. 266). Este proyecto condujo a la que sería la primera comunidad para compartir *software*, llamada apropiadamente SHARE y conformada por usuarios de IBM, en una época donde la falta de conocimiento en estos equipos obligaba a que se diera la cooperación entre sus propietarios (Armer, 1980, p. 123).

En segundo lugar, la década siguiente fue testigo de la consolidación de otra cultura sociotécnica frecuentemente asociada a la colaboración en el *software*: la cultura *hacker*. Esta se caracterizó por no haber surgido dentro de una cultura corporativa propiamente, sino dentro de las libertades provistas por un entorno universitario (Campbell-Kelly y Aspray, 1996, p. 225). Los *hackers* se caracterizaban por su búsqueda de la excelencia técnica y por el mejoramiento colectivo a través de compartir programas. Por tanto, colaborar era clave, tal como quedó establecido en el recuento periodístico desarrollado después en lo que se conoció como la «ética *hacker*» (Levy, 1994, pp. 28-34). Su libertad productiva, donde la artesanía y la curiosidad estimulaban una actitud *laissez-faire* hacía la productividad (Levy, 1994, p. 115), provocó una evolución del paradigma computacional imperante, cambiando de los sistemas de procesamiento por lotes a los sistemas interactivos. Como ejemplos podrían citarse piezas de *software* como el sistema ITS, el potencial del llamado «tiempo compartido» (en el que varios usuarios trabajaban en un solo computador) (Salus, 2008, p. 6) o incluso el más conocido sistema operativo UNIX, un sistema con su propia filosofía sociotécnica basada en la modularidad y eficiencia (Salus, 1994, p. 53) y que sería clave en el desarrollo de Internet y de muchas tecnologías actuales.

Finalmente, el giro en toda esta cultura de la colaboración vino con la comercialización del *software* como producto de creciente valor. Por ello, las prácticas colaborativas fueron puestas en tela de juicio, como quizá muestra la conocida carta de Bill Gates que inicia la comparación, ya común para la época, entre compartir *software* y la mal llamada piratería (Gates, 1976). Así, el giro de paradigmas dejó el terreno libre para que surgieran movimientos como el del *software* libre y el código abierto que, si bien defendían los valores históricos de la colaboración, ya se definían así mismos en contraposición a la comercialización excesiva del *software*, desplegando una conciencia propia inexistente en las prácticas anteriormente descritas.

El campo de las redes de computadores, que se define como un desarrollo entrelazado con la historia del *software*, también ha tenido una evolución donde la colaboración ha sido clave. Como adjetivo, en este trabajo, para denotar la enorme influencia de las redes en la vida diaria se ha usado la expresión «la sociedad en red», acuñada por el sociólogo Manuel Castells para designar la nueva morfología social moldeada por las redes (Castells, 1996, p. 469). Al igual que con el *software*, esta sociedad en red se empezó a tejer en gran medida en la academia; tiene una prehistoria que, aunque antecede a los computadores, sí presenta muestras del llamado pensamiento rizomático y distribuido que caracteriza esta época. Específicamente, el experimento mental denominado «memex» de Vannevar Bush (Bush, 1945) planteaba un dispositivo electromecánico y óptico que permitía llevar apuntes y relacionar información almacenada en microfilme de manera asociativa, usando algo llamado *sendero* o *camino* (*trails* en Inglés), el cual recuerda la funcionalidad de los enlaces en la web actual.

Propiamente hablando de la época donde el concepto de *software* ya existía, el esfuerzo de las redes de computadores surgió como una iniciativa para compartir recursos computacionales entre científicos, en una época en la que su acceso era escaso. Dicha posición era la argumentada por personas que la academia convocaba para ocupar cargos en la administración de investigación de los Estados Unidos, país donde los primeros prototipos de redes fueron exitosos. Destaca en particular la figura de J. C. R. Licklider, quien, en el rimbombantemente título *Memorando para los miembros de la red intergaláctica* (Licklider, 1963), ya planteaba el reto de la simbiosis entre el humano y la máquina, y su potencial para la cooperación colectiva anteriormente prevista en otros escritos (Licklider, 1960). De este furor intelectual y creativo vendría la que se conoce como la primera red de computadores: Arpanet, en 1969. Aunque las narrativas de la historia de la tecnología informática frecuentemente señalan a Arpanet como la red antecesora directa de Internet y en este ejercicio pierden la diversidad del relato al desconocer muchas otras redes surgidas con posterioridad, lo cierto es que Arpanet es el caso más paradigmático de una red surgida con propósitos colaborativos en la intersección también

colaborativa entre academia, industria y gobierno[3]. Desde su planeación, Arpanet tenía como objetivos compartir datos, la carga de procesamiento, *software* y otros recursos informáticos (Roberts, 1967, p. 1), lo cual podría considerarse una condición material necesaria para la colaboración. El componente que uniría todas estas partes es el de la comunicación, como lo ha mostrado el hecho de que una red de origen militar y académica mutara hasta convertirse en el metamedio de comunicación que es hoy en día. La comunicación es una actividad indispensable para la colaboración y su potencial dentro de las redes fue pronosticado casi desde su origen por personas como el mismo Licklider en 1968 (Licklider y Taylor, 1968).

Para establecer un contrapunto con el ejemplo previo, se puede mencionar brevemente a la red Usenet. Si en la década de 1960 Arpanet surgió como un esfuerzo colaborativo, aunque completamente jerárquico con todo el músculo financiero del Departamento de Defensa de los Estados Unidos, la década de 1970 —con el establecimiento de sistemas como UNIX y la aparición del computador personal— proveería el ambiente sociotécnico ideal para la aparición de Usenet, una red surgida de abajo hacia arriba, sin financiamiento y conformada por voluntarios que querían compartir información técnica sobre el sistema UNIX (Kehoe, 1993, p. 30). Debido a estas características, Usenet era llamada «*the poor's man Arpanet*» [el Arpanet del hombre pobre] (Daniel *et al*., 1980). Fue un esfuerzo colectivo construido por voluntarios que fue fundamental en la conformación de una cultura popular de la red. Adicionalmente, Usenet fue el espacio donde circularon mensajes que difundían por primera vez el concepto de *software libre* y se hizo el llamado a participar en el naciente sistema operativo Linux. De esta forma, puede decirse que Usenet era un espacio de comunicación compartido para colaborar (Wilbur, 1997, p. 13).

Finalmente, estos ejemplos hicieron parte en la conformación de la sociedad en red. Esta se estableció, en parte, cuando varias tecnologías de comunicación entre computadores fueron absorbidas por la omnisciente Internet a principios de 1990, ofreciendo un espacio global de comunicación que promovería la colaboración —como el surgimiento del movimiento de código abierto atestiguaría—, basándose en la continuación de una política de arquitectura abierta y estándares abiertos desarrollados colaborativamente (Leiner *et al*., 2009, p. 24; Abbate, 1999).

RESULTADOS PARCIALES Y TRABAJO FUTURO

Este análisis, basado en una remezcla metodológica que aquí se indica de manera sucinta por cuestiones de espacio, ha arrojado algunos preceptos conceptuales que pueden considerarse resultados y que servirán para emprender una parte restante, es decir, el caso estudio representado en la cultura *maker*. En particular, puede decirse que estos resultados, manifestados en definiciones, constituyen en parte un aporte a los estudios de *software*, a los estudios en ciencia y tecnología y, en general, a la consolidación de la idea del *software* como una cultura. En primer lugar, y como resultado de las hibridaciones entre *software* y redes, se ha pasado de la denominación *sistema sociotécnico* (propia de los estudios en ciencia y tecnología) para definir la industria y cultura del *software*, a la definición *red sociotécnica*. Este cambio, que parece ligero, obedece a principios históricos y estructurales del tema: si bien la palabra *sistema* sigue en uso y puede decirse que toda red es un sistema (mas no al contrario), esta apunta a una concepción cibernética de la tecnología como un sistema cerrado con entradas, procesamiento y salida. Por otro lado, la metáfora de la red es más actual y resuena con un proceso iterativo, colectivo, caótico y fuertemente enraizado en la comunicación, como es el desarrollo de *software* usando Internet y como sucede particularmente en el *software* libre.

En segundo lugar, y como consecuencia de lo anterior, se articulan dos ejes entrelazados. De un lado y siguiendo el lenguaje de la ya señalada teoría actor-red de Latour, la red sociotécnica de producción y distribución de *software* está conformada por nodos que son «actantes»; de manera simplificada, pueden ser personas o máquinas. Esto puede confirmarse al revisar todos los artefactos organizativos y tecnológicos que apoyan la producción de *software*, los cuales incluyen desde protocolos para la resolución de conflictos hasta piezas de *software* que integran y distribuyen automáticamente las contribuciones globales que componen una pieza de *software* desarrollada colaborativamente. Siguiendo la clasificación provista en Krieger y Belliger (2014), estos nodos-actantes estarían organizados en un «colectivo» que se subdividiría en la comunidad —el grupo humano de desarrolladores— y la infraestructura —la parte técnica— atravesados por protocolos de comunicación mixtos (humano-humano, humano-máquina, máquina-máquina) que complementarían el aspecto comunicativo aún débil en un constructo como la teoría actor-red, tal como lo señala Farías (2014).

Finalmente, esta conformación de la red sociotécnica y de la capacidad de maniobra de los nodos-actantes llevó a una definición de tres niveles sobre la agencia del *software* (código fuente más otros artefactos). Estos niveles también corresponden a una evolución de la historia de la

3 **Hubo otros ejemplos de redes de computadores ajenos a esta narrativa hegemónica, como OGAS (Unión Soviética), Cybersyn (Chile) y Minitel (Francia). Sin embargo, tales ejemplos están extintos y no hacen parte de los modos de producción de *software* modernos y distribuidos.**

computación y tienen un reflejo en avances tecnológicos y metodológicos en el desarrollo de *software*. En este orden, la agencia algorítmica (o en lotes) correspondería a la década de 1950 y al *software* como procesamiento de entradas en salidas sin ninguna interacción intermedia (algo relacionado con el ya mencionado concepto de sistemas). La agencia interactiva, que correspondería a las décadas de 1960 y 1970, abarca el desarrollo de tecnologías interactivas en tiempo real, forzando desarrollos como la programación estructurada y la programación orientada a objetos. Por último, estaría la etapa actual de agencia distribuida, caracterizada por la interacción global de actantes-nodos que confluyen en modelos iterativos de desarrollo de *software* en red. Es justamente esta agencia la que llevó a pensar en la denominada *fabricación distribuida* (cultura *maker*) como caso de estudio para cotejar la metáfora del *software* libre en otros entornos.

Como trabajo futuro queda contrastar las conceptualizaciones anteriormente dadas con lo que se ha denominado *cultura maker*, aunque dicho nombre no esté libre de cuestionamientos. Precisando lo referido en la sección anterior, por *cultura maker* se hace referencia a los procesos de fabricación distribuida que están basados en el diseño asistido por computador y su respectiva distribución y adaptación a través de Internet y que son efectuados por máquinas como impresoras 3D, cortadoras láser, fresadoras, entre otros. El término *maker* se ha conservado por la conveniencia y uso generalizado, pero proviene de la esfera comercial y fue acuñado por una revista, por lo cual no está libre de controversia. No obstante, tanto la industria como la academia han adoptado la palabra con entusiasmo para señalar una nueva revolución tecnológica en el ámbito de los objetos físicos. La cultura *maker* se ha seleccionado por los siguientes motivos: a) es una tecnología fuertemente enraizada en la cultura del *software*, b) cuenta con un modo de producción iterativo y colaborativo similar al del *software* libre, y c) permite poner en contexto la ya referida materialidad del *software*. Los dos primeros puntos son evidentes al considerar que los objetos que producen los dispositivos son controlados por *software* y que los diseños codificados digitalmente tienen rasgos colaborativos en cuanto a que son puestos a disposición de los otros para ser intervenidos, adaptados y distribuidos nuevamente, con una lógica exactamente igual a la del *software* libre. Sin embargo, el tercer punto es el que representa el punto central del debate y posiblemente una de las conclusiones finales del trabajo, en plena concordancia con el mencionado giro material. En particular, varios recuentos de la cultura *maker* —como la ya referida por el escritor Corey Doctorow— la ubican como un ejemplo de la materialización de las prácticas del *software* libre. Dicha afirmación lleva a una conclusión correcta, pero parte de una premisa errada. Específicamente, obedece a la visión tradicional del *software* como algo inmaterial, inasible y platónico, que a través de la fabricación de objetos se «materializa». Esta postura confunde conceptos como *físico* con *materialidad* y va en contra de la concepción moderna del *software* la cual le otorga un carácter material debido a aspectos lingüísticos e incluso electrónicos, como en el famoso ensayo de Kittler donde se argumenta que no hay *software* porque al final todo son impulsos eléctricos (Kittler, 1997).

La materialidad del *software* en realidad se ve confirmada al analizar históricamente las formas de organización del trabajo en la evolución de modos de producción del *software* y en especial del *software* libre. En este sentido, la colaboración se muestra no como una práctica reciente, sino como un ejercicio que siempre ha estado presente y que refuta la visión aristotélica de la tecnología como un saber secundario y relegado, o la visión cientificista donde la tecnología es simplemente un apéndice subordinado de la ciencia. Por el contrario, lo que la colaboración abierta en la industria del *software* muestra es que su modo de producción es un proceso difícil de caracterizar y sistematizar (como la llamada crisis del *software* y el surgimiento de la ingeniería de *software* muestran) y que el imperativo *hacker*, tan fuerte en el *software* libre y su ética de construir haciendo y el ensayo y error, la alejan de modelos de planificación de otras disciplinas y la acercan más a expresiones de la actividad humana como la artesanía y el trabajo manual. De esta forma, la colaboración en el *software* permite una nueva valoración de la tecnología en general, más acorde con las redes sociotécnicas donde humanos y máquinas confluyen en una labor productiva.

referencias

abbate, J. (1999). *Inventing the Internet*. Cambridge, ee.uu.: mit Press.

armer, P. (1980). Share - A Eulogy to Cooperative Effort. *Annals of the History of Computing*, 2, 122-129.

brate, A. (2002). *Technomanifestos*. México: Texere.

bush, V. (1945). As We May Think. *The Atlantic*. Recuperado de https://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/

campbell-kelly, M. y Aspray, W. (1996). *Computer: A History of the Information Machine*. Nueva York, ee. uu.: Basic Books.

campbell-kelly, M. y García-Swartz, D. D. (2009). Pragmatism, not Ideology: Historical Perspectives on ibm's Adoption of Open-Source *Software*. *Information*

Economics and Policy, 21(3): 229-244.

CASTELLS, M. (1996). *The Rise of the Network Society*. New Jersey, EE. UU.: Wiley-Blackwell.

DANIEL, S., Ellis, J. y Truscott, T. (1980). *Usenet - A General Access Unix Network*. Handout.

FARÍAS, I. (2014). Virtual Attractors, Actual Assemblages: How Luhmann's Theory of Communication Complements Actor-Network Theory. *European Journal of Social Theory*, 17(I), 24-41.

FRABETTI, F. (2015). *Software Theory. A Cultural and Philosofical Study*. Lanham, EE. UU.: Rowman & Littlefield.

FULLER, M. (2008). *Software Studies. A Lexicon*. Cambridge, EE. UU.: The MIT Press.

GATES, B. (1976). An Open Letter to Hobbyists. *Homebrew Computer Club Newsletter*, 2(1).

GEHL, W. y Bell, R. (2012). Heterogeneous *Software* Engineering: Garmisch 1968, Microsoft Vista, and a Methodology for *Software* Studies. *Computational Culture*.

ISAACSON, W. (2014). *The Innovators. How a Group of Hackers, Geniuses and Geeks Created the Digital Revolution*. Nueva York, EE. UU.: Simon & Schuster.

KEHOE, B. P. (1993). *Zen and the Art of the Internet: A Beginner's Guide*. New Jersey, EE. UU.: Prentice Hall.

KITCHIN, R. y Dodge, M. (2011). *Code/Space. Software in Everyday Life*. Cambridge, EE. UU.: MIT Press.

KITTLER, F. A. (1997). There is no *Software*. En J. Johnston (ed.). *Literature Media Information Systems: Essays* (pp. 147-155). Londres, Inglaterra: Routledge.

KRIEGER, D. J. y Belliger, A. (2014). *Interpreting Networks. Hermeneutics, Actor-Network Theory & New Media*. Bielefeld, Alemania: Transcript Verlag.

LEINER, B. M., Cerf, V. G., Clark, D. D., Kahn, R. E., Kleinrock, L., Lynch, D. C., Postel, J., Roberts, L. G., y Wolff, S. (2009). A Brief History of the Internet. *ACM SIGCOMM Computer Communication Review*, 39(5), 22-31.

LEVY, S. (1994). *Hackers: Heroes of the Computer Revolution*. Newton, Massachusetts, EE. UU.: O'Reilly Media.

LICKLIDER, J. C. R. (1960). Man-Computer Symbiosis. *IRE Transactions on Human Factors in Electronics*, (HFE-1) 4-11.

LICKLIDER, J. C. R. (1963). Memorandum for: Members and Affiliates of the Intergalactic Computer Network. Recuperado de: http://www.kurzweilai.net/memorandum-for-members-and-affiliates-of-the-intergalactic-computer-network

LICKLIDER, J. y Taylor, R. W. (1990). The Computer as a Communication Device. In *In Memoriam: J. C. R. Licklider 1915-1990*. Palo Alto: EE. UU.: Digital Systems Research Center. Recuperado de http://bitsavers.trailing-edge.com/pdf/dec/tech_reports/SRC-RR-61.pdf

MANOVICH, L. (2013). *Software Takes Command: Extending the Language of New Media*. Nueva York, EE. UU.: Bloomsburry.

MELAHN, W. S. (1956). A Description of a Cooperative Venture in the Production of an Automatic Coding System. *J. ACM*, 3(4), 266-271.

ROBERTS, L. G. (1967). Multiple Computer Networks and Intercomputer Communication. *Proceedings of the First ACM Symposium on Operating System Principles*, 3.1-3.6

SALUS, P. H. (1994). *A Quarter Century of UNIX*. Boston, EE. UU.: Addison-Wesley Publishing Company.

SALUS, P. H. (2008). *The Daemon, the GNU, and the Penguin. How Free and Open Software is Changing the World*. Reed Media Services.

SAMTANI, H. (19 de junio de 2013). Meet the Makers: Can a DIY movement revolutionize how we learn? *The Digital Shift*. Recuperado de http://www.thedigitalshift.com/2013/06/k-12/meet-the-makers-can-a-diy-movement-revolutionize-how-we-learn/

STALLMAN, R. M. (2002). *Free Software, Free Society: Selected Essays of Richard M. Stallman*. Washington, EE. UU.: GNU Press.

STIEGLER, B. (1998). *Technics and Time, 1. The Fault of Epimethus*. Stanford, EE. UU.: Stanford University Press.

TAUBERT, N. C. (2006). *Produktive Anarchie? Netzwerke Freier Softwareentwicklung*. Bielefeld, Alemania: Transcript-Verlag.

WALTER-HERMANN, J. y Büching C. (2013). *FabLab: Of Machines, Makers and Inventors*. Bielefeld, Alemania: Transcript-Verlag.

WILBUR, S. P. (1997). An Archaeology of Cyberspaces: Virtual, Communuty, Identity. En D. Porter (ed.). *Internet Cultures*, 5-22. Londres, Inglaterra: Routledge.

ZIELINSKI S. (2012). *Arqueología de medios. Hacia el tiempo profundo de la visión y la audición técnica*. Bogotá, Colombia: Universidad de los Andes.