saskab
http://www.ideaz-institute.com/
Cuaderno 19

# Un desarrollo de software latinoamericano, entre el software propietario y el software libre

Jaime Gutiérrez Alfaro
*Escuela de Ingeniería en Computación*
*Instituto Tecnológico de Costa Rica*
*Universidad Nacional*
Alajuela, Costa Rica
jgutierrez@itcr.ac.cr

## Resumen

La industria del software se aprovechó de tecnicismos y aspectos legales para privatizar el acceso al conocimiento que permite construir software. La filosofía del software libre plantea un posicionamiento a partir del análisis crítico de ese proceso de desarrollo tecnológico. En este ensayo se propone un diálogo entre la filosofía del Software Libre y el debate sobre una filosofía latinoamericana de Salazar Bondy y Leopoldo Zea, con el fin de explorar cuál debe ser el modelo técnico-legal para construir un desarrollo de software latinoamericano. Planteo tres formas en las cuales Latinoamérica participa en el proceso de construcción de tecnología computacional: (1) como personas usuarias de software propietario producido en países hegemónicos, (2) replicando la privatización del conocimiento y (3) produciendo software libre. Luego de analizar cada una de las posibilidades, concluyo que un desarrollo de software latinoamericano debe ser basado en el modelo de software libre.

**Palabras clave:** Filosofía latinoamericana, Software libre, Ética, Tecnología

*"Emancipate yourselves from mental slavery*
*None but ourselves can free our minds"*
*-- Bob Marley*

# Introducción

En la canción "*Redemption song*" Bob Marley hace un llamado a liberarse de la esclavitud mental. Les habla a sus pares, otras personas afrodescendientes del Caribe que habitan en territorios que fueron dominados por medio de sistemas coloniales entre el siglo XVI y el XIX. Marley no habla de una esclavitud corporal, física o material, sino que se refiere a la entidad abstracta donde viven las ideas, la mente. Desde donde cada persona elabora su visión de mundo y establece los límites de su imaginación.

La capacidad de imaginar nos permite generar representaciones abstractas a partir de la realidad, y nuevas representaciones abstractas a partir de las anteriores, hasta llegar a representaciones en las que luego es difícil dar trazo a la realidad inicialmente representada. El software es un ejemplo de estas representaciones de múltiples capas donde lo abstracto esconde la realidad.

Detallemos un momento las múltiples capas que es posible identificar en el software. Lo podemos hacer desde dos puntos de vista, desde los datos que se requieren para ejecutar una aplicación o desde las instrucciones que permiten ejecutar un programa. Dada la temática que quiero abordar en este trabajo me enfocaré en las capas abstractas de representación de la realidad desde las instrucciones. El software desde el punto de vista de la ejecución de programas informáticos se crea a partir de capas abstractas que tienen en un extremo una representación "comprensible" por una máquina electrónica digital y en el otro a las personas programadoras, entre estos extremos pueden existir varias otras capas.

A manera de ejemplo, tomemos un programa que imprime en la pantalla de la computadora el mensaje "hola mundo". Una representación de este programa es `print("hola mundo")`. Coincidiremos que es fácilmente comprensible por humanos aún sin entrenamiento técnico en programación. Ciertamente para un hablante del Inglés será tan sencillo como leer y podrá comprender el funcionamiento del programa. Para personas hablantes de otras lenguas, una

vez explicada la traducción de la palabra "*print*" será sencillo comprender qué hace el programa. Otra representación del mismo programa es 11011010 01000110 10011001 11001100 11110000 11111111 10010100 10111110 11001100 10011001 11011010 01000110 10011001 11001100 11110000 01010101 10101011 11001101. Esta representación no es comprensible de manera sencilla por una persona, aún con entrenamiento técnico. Aunque se aproxima mucho a instrucciones que puede comprender una computadora, hace falta que esos números se transformen en impulsos eléctricos digitales para que puedan llegar a ser ejecutados por un procesador digital.

Este tecnicismo del software plantea la posibilidad de separar el conocimiento del producto. Las representaciones más abstractas y cercanas a las personas son conocimiento, mientras que la representación más cercana a los impulsos eléctricos son productos. No pretendo hacer una comparación entre una computadora y un humano, pero esbozaré un ejemplo trasladando la situación a los humanos para tratar de explicar esta separación que planteo. Imaginemos por una parte un libro escrito en un lenguaje comprensible por humanos y por otra el estímulo mental que supone haber asimilado el texto por medio de una lectura. En el hipotético caso que se pueda transferir el conocimiento del libro a la mente/cuerpo sin necesidad del libro, estaríamos separando el producto del conocimiento representado en el libro. De esas dimensiones es la separación entre las representaciones del software.

En la siguiente sección del ensayo, daré detalles sobre el contexto en el surge el movimiento del software libre y de la privatización del conocimiento. En la tercera pondré en diálogo la filosofía del software libre con el debate sobre una filosofía latinoamericana de Salazar Bondy y Leopoldo Zea. Cierra el trabajo una sección de conclusiones.

## La filosof*ía del Software Libre*

Los desarrollos técnicos, los inventos y el conocimiento aplicado, estimulan las posibilidades imaginativas de cada sociedad. En estas

posibilidades surge un amplio rango de futuros posibles, desde sociedades dominadas por humanoides electrónicos que nos esclavizan hasta una sociedad donde los artefactos electrónicos automatizan las tareas de las personas aumentado el bienestar de la sociedad. Sin embargo, la posibilidad de acercarnos a una u otra posibilidad requiere la capacidad para moldear la tecnología según los intereses de las comunidades de personas que las utilizan.

La sociedad China del siglo XIV tuvo acceso a los descubrimientos técnicos y a los inventos requeridos para llevar adelante un proceso de industrialización, pero no ocurrió. Cuatro siglos después la sociedad occidental Europea, llevó adelante la revolución industrial (siglo XVIII y XIX) abrazados de una estructura económica y social capitalista feroz, que se aprovechó de los sistemas coloniales impuestos en regiones como Latinoamérica. En occidente, hacia finales del siglo XIX se inventan productos tecnológicos en los cuales se pone en práctica el conocimiento eléctrico. Ya en el siglo XX aparece la industria computacional con características propias, primero gracias al financiamiento de los mercados militares y luego impulsada por la cultura de libertad, innovación y emprendedurismo de las universidades del oeste de los Estados Unidos (Castells, 2005, p. 35-38).

En los años ochenta del siglo anterior ocurre lo que Stallman llamó "El colapso de la comunidad", refiriéndose de manera muy directa a la comunidad de personas desarrolladoras de software, pero con implicaciones que van más allá de este grupo (Stallma, 2004). Previo al colapso, el costo económico asociado a la producción del hardware era muy alto si lo comparamos con el software. Este último se vendía como parte de la máquina pues constituía un elemento indispensable para poder operarla. El software era, tanto la representación de instrucciones comprensibles por un humano (el cual se llama código fuente), como la representación de los impulsos eléctricos (conocido como código máquina), era conocimiento y producto. No se concebía la posibilidad de comprar una computadora sin software y mucho menos sin acceso al código fuente, ni tenía valor comercial el software sin el hardware. Intercambiar software con otras personas programadoras (o no) era una práctica normal, equivalente a intercambiar un libro, una receta de cocina o una práctica agrícola.

El abaratamiento del costo del hardware abrió la posibilidad de convertir

a la computadora en un electrodoméstico más. Uno distinto pues no planteaba una solución a una necesidad doméstica, no al menos en la misma forma que una lavadora, refrigeradora o cocina. La computadora como electrodoméstico fue creada por los "*personajes profesionalmente adictos a la solución industrial de problemas creados por una industria"* usando palabras de Ivan Illich (1973). Un detalle que no hay que dejar desapercibido es la aceleración con la cual se da este proceso de electro domesticación de la computadora. Como referencia, en solo 3 años (entre 1959 y 1962), los semiconductores, un componente básico para la fabricación de tecnología computacional bajó de precio un 85%, esa misma bajada de precio tomó 70 años en la tela de algodón, el precio de un circuito integrado pasó de 50 dólares a 1, en nueve años (1962 a 1971) (Castells, 2005: p70-71). La obsesión por acelerar el proceso técnico supuso un desinterés por profundizar en la capacidad de reflexión y análisis crítico sobre los efectos sociales de la producción tecnológica computacional.

En estas circunstancias el software adquirió un valor comercial por sí solo. La industria del software sacó provecho de los tecnicismos y la legislación de propiedad intelectual para generar lucro. Desde la perspectiva técnica, una persona usuaria de un software sólo requiere acceso al código máquina, pues finalmente es ese código el que se ejecutará en la computadora. Por lo tanto, para la industria el negocio se convirtió en vender código máquina separado del código fuente, es decir vender un producto del cual no se tiene acceso al conocimiento. Las leyes de propiedad intelectual permitieron a la industria establecer más restricciones, prohibiendo que una persona pueda compartir copias del software que había comprado.

Sin acceso al código fuente, ni posibilidades de compartir el software, inicia la privatización del conocimiento técnico computacional. Se crea una separación entre quienes tienen los medios productivos para generar nuevo conocimiento computacional y quienes estarían condenados a ser usuarios/consumidores de un producto. La industria privatizadora, se instala como la única dueña de la capacidad de crear nuevos productos y de reutilizar el conocimiento existente (el cual es propiedad privada de la misma industria). Cualquier intento de crear nuevo software implica por lo tanto un esfuerzo considerable, pues al no existir código fuente que legal o técnicamente se pueda

reutilizar, cada construcción de conocimiento iniciará desde las capas abstractas más cercanas a la máquina. Ante las barreras, el software privativo ya convertido en un conocimiento-mercancía, impone una única visión de mundo homogeneizadora que les permite a las industrias maximizar ganancias, con lo cual se evidencia además que no existe reparo en invisibilizar otras visiones de mundo que no son compatibles con la perspectiva que se está codificando y que genera ganancias comerciales.

Como respuesta surge el Software Libre, una propuesta planteada por Richard Stallman (2014), un programador sin pretensiones filosóficas. El análisis crítico de las implicaciones de aplicar una lógica capitalista post revolución industrial al desarrollo de software, es el punto de partida del *idealismo pragmático* de la filosofía del software libre. El idealismo se manifiesta en la visión de mundo de *sociedad libre* en la que el esfuerzo se pone en función de la comunidad, sin relaciones de dependencia de propietarios de conocimiento, una sociedad más igualitaria, equitativa y horizontal. El pragmatismo es dado por la concreción de mecanismos legales y técnicos para contribuir al ideal.

El software libre se valió del instrumento legal que privatiza el conocimiento, las leyes de propiedad intelectual, para resguardar el conocimiento transformado en código fuente. Lo hizo atendiendo a la esencia de la ley de derecho de autor, la cual es que las personas autoras tienen el derecho a decidir de qué forma se comercia y distribuye su obra. Dado que las personas programadoras son las autoras del software, se creó una *licencia* (mecanismo legal) para que puedan manifestar su deseo de garantizar que las personas puedan ejecutar, copiar, distribuir, modificar y mejorar el software sin restricciones. Se *jaqueó* el derecho para utilizarlo en favor de la comunidad. Esto quiere decir que el software libre siempre debe ser código fuente y código máquina, sin restricciones legales que limiten a las personas.

La filosofía del software libre se plantea ante un cambio de paradigma latente, es la respuesta crítica al reconocimiento de que pueden darse cambios sustanciales al funcionamiento de nuestra sociedad, provocados por aspectos técnicos y legales que fácilmente se pueden ampliar a otros aspectos de la vida. La cultura, el conocimiento generado de forma colaborativa e intercambiado entre generaciones, está en riesgo de ser privatizada. No se niega que

anteriormente hayan existido grupos sociales dominantes sobre otros, tampoco se discute si se ha salido de esas sociedades jerárquicas, lo que se plantea es que el software libre tiene la oportunidad de abolir las jerarquías en la producción de software, de establecer una sociedad de pares.

La comunidad de personas desarrolladoras de software libre es la que *colapsa* y el Software Libre aparece como un modelo de licenciamiento que ofrece una solución pragmática para preservar la práctica de compartir el conocimiento computacional. Sin embargo, el riesgo que representa la privatización del software y de manera generalizada la privatización del conocimiento es una amenaza a las *comunidades* que tienen prácticas similares a esa comunidad colapsada.

## Debate sobre la tecnología computacional latinoamericana

En la sección anterior presenté detalles sobre el proceso de producción de tecnología computacional, deteniéndome en el surgimiento de la filosofía del software libre y las posibilidades que ofrece esa propuesta como respuesta al modelo de privatización del conocimiento plasmado en el software propietario. Sin embargo, hasta ahora el recorrido histórico y el debate presentado está circunscrito al norte global, principalmente los Estados Unidos. En esta sección haré el ejercicio de contextualizar el debate sobre las distintas formas de producción de tecnología computacional al entorno latinoamericano, generando un diálogo entre las ideas presentadas en la sección anterior, con las ideas de Augusto Salazar Bondy, a partir del texto ¿Existe una filosofía de nuestra América? (1968) y de Leopoldo Zea, con el libro La filosofía americana como filosofía sin más (1969).

Considerando los procesos de privatización del conocimiento tecnificado y la filosofía del Software libre, ¿Cuáles son nuestras posibilidades reales de participar en el desarrollo de software? Y ¿Cuáles son las implicaciones de participar de una u otra forma? Estas preguntas las abordaré revisando tres posibilidades de participación en el proceso de construcción de tecnología computacional (1) como personas usuarias de software propietario producido en países hegemónicos, (2) replicando la privatización del conocimiento y (3)

produciendo software libre. En las siguientes tres subsecciones revisaré cada posibilidad y las implicaciones de una u otra.

### *Personas usuarias de software propietario*

Asumir un rol de personas usuarias de software propietario creado en el contexto del norte global implica que en nuestra región nunca tendremos acceso al conocimiento que permite producir software. Que el software utilizado será producido en países hegemónicos, en la actualidad principalmente empresas localizadas en los Estados Unidos, para satisfacer las necesidades, problemáticas e interés de esa cultura. Es aceptar una dominación digital, asumiendo el rol de consumidores del producto que nos sea dado, con las condiciones que impongan las personas que produzcan el desarrollo tecnológico. La única posibilidad en la cual nuestra cultura podría ser tomada en consideración para el diseño de software, será en la medida que tenga potencial de negocio desde la perspectiva de las corporaciones.

En la línea de pensamiento de Salazar Bondy esta posibilidad implica condenar nuestras posibilidades de generación de software a la dependencia y sujeción, asignándole a los países hegemónicos el rol de producción teórica y a nosotros la aplicación o el mero uso de las ideas que desarrollen y codifiquen desde perspectivas que no son nuestras (1968, p. 114-116). Participar como usuarios de software también implica que no tendremos acceso al idioma del proceso de producción del software, el código fuente, pues es precisamente el que se niega cuando se privatiza el software. Es perpetuar nuestra no vigencia en el proceso mundial del poder. Participar como usuarios en el proceso de producción de tecnología guarda relación con la forma con que Salazar Bondy caracteriza la producción de filosofía, cultura y de conocimiento en América Latina.

Desde esta posibilidad de participación en el proceso de construcción de tecnología computacional, no existirá tecnología de software propia, en los términos de Salazar Bondy (1968, p. 88-89). Nuestra tecnología no sería auténtica ni original. Es una condición donde se repite la perspectiva jerárquica, en la que el software auténtico es creado fuera de la región y sólo podemos hacer

uso de ese conocimiento sin asimilación. La peculiaridad, a lo mucho, estaría dada por la manera en que usamos o aprendemos a usar el software. Sin embargo, en el conocimiento codificado como tal no será posible incorporar nunca miradas propias.

Desde la perspectiva de Zea hay coincidencia en que este tipo de acercamiento al software supone una imposición por parte de los países hegemónicos, una nueva forma de subordinación. En este caso ya no porque se considere al latinoamericano inferior en términos de humanidad, sino que es una imposición y dominación guiada por una perspectiva capitalista post industrial. Por lo tanto, cuando el productor de software limita el acceso al código fuente al no-occidental, lo está enajenado del acceso al conocimiento para desarrollar sus propios intereses, para el productor de software es una ventaja comercial.

Un concepto que plantea Zea es el de asimilación, hacer propio lo extraño (1969, 26). Se podría argumentar que siendo usuarios es posible asimilar el software hegemónico y que ocurre cuando por ejemplo cambiamos el fondo de pantalla de nuestro teléfono celular o computadora, o cuando cambiamos el idioma en el que se presentan las aplicaciones. Sin embargo, esta posibilidad es únicamente viable porque el software ofrece la opción. Ser un usuario experto que domina todos los detalles de cómo funciona una aplicación de software no quiere decir que tengamos la posibilidad de modificarla y adaptarla a nuestras necesidades de manera libre, pues únicamente será posible en el tanto el software permita la adaptación. La asimilación, como usuarios, será ilusoria pues siempre es dependiente de que el software la permita. Cuando un software no permita cambiar el fondo de pantalla no existirá tal posibilidad, por lo tanto, participar como usuarios de software no permitirá hacer propio lo extraño en un sentido profundo, sino superficial. Tampoco podemos hablar de que al usar el software tengamos la posibilidad de hacer copias con algo de nuestro modo de copia (1969, 45). Debido a los tecnicismos del software, la copia será exactamente igual y no hay opciones de modificación. Por lo tanto, no podemos decir, siguiendo a Zea que la copia o la calca llevaría sin pretenderlo algo de nuestro modo.

***Replicar el modelo de construcción de software propietario***

La segunda posibilidad es replicar el modelo de privatización del conocimiento, es decir crear software latinoamericano que opera en monopolios de conocimiento localizados en nuestra región. Esto supone que el software latinoamericano atendería a las particularidades de la región, pero limitando el acceso al conocimiento más allá de pequeños grupos conformarían nuevas élites científicas, tecnológicas e intelectuales. Esta es una posición que conduce a mantener una relación de subordinación, que se traslada del pensamiento hegemónico del norte global a un pensamiento hegemónico local, manteniendo la jerarquía entre usuarios y productores presentada en la sección anterior. El conocimiento codificado, es decir el acceso al código fuente, queda limitado a un pequeño grupo y el resto de la población debe asumirse como consumidores del producto que construyan.

Esta posibilidad tiene elementos en la dirección de producir conocimiento o software original, usando los términos de Salazar Bondy (1968, 88:89). Replicar el modelo de privatización implica que tendríamos acceso al idioma del desarrollo de software, al profesionalismo, seriedad y rigurosidad con el cual se construye, pero no de manera generalizada porque conlleva a que el acceso al conocimiento se concentre en la élite que pueda apropiarse de ese conocimiento, con lo cual se produce conocimiento original pero que conduce a una nueva enajenación. Por lo tanto, si bien se da una mejora en la forma de acceso al conocimiento, creamos una figura que concentraría el conocimiento construyendo una nueva dependencia, un nuevo punto en el tránsito de dominación hegemónica que parte de España llegando a Estados Unidos y ahora añade una industria de software latinoamericana bajo el modelo de privatización del conocimiento. Salazar Bondy, haría un llamado de atención sobre las implicaciones de esta posibilidad, como se puede extraer de la siguiente cita: *"...como seguirá tomando de fuera, quizá por mucho tiempo, conceptos y valores, deberá ser vigilante y desconfiada en extremo, a fin de evitar —por la crítica y la consulta de la realidad— la recaída en los modos alienantes de reflexión"* (Salazar, 1968: 118).

En la línea de pensamiento de Zea replicar el modelo de construcción de software propio tendría dos posiciones de análisis. Por una parte, implica que se

puede construir tecnología propia que atienda a nuestra propias realidades y problemas, lo cual supone una ruptura con la perspectiva Europea más allá de las valoraciones que hagan países hegemónicos de nuestra producción de software. Sin embargo, por otra parte, Zea encontraría en este modelo una manera de mantener la dominación en la forma de colonialismo interno. Desde Zea se puede ir más allá y cuestionar qué sentido tiene replicar un modelo de producción de software que no se cuestiona *para qué se hace*, qué sentido tiene saber *cómo* producir software si se crea una condición de dominación entre productores y usuarios, ambos latinoamericanos.

La posición de Zea sobre aplicar la filosofía Europea en nuestro contexto con la expectativa de acercarnos al desarrollo, sería igual a replicar el modelo de software propietario en nuestro contexto con la expectativa de que esto nos aleje del subdesarrollo, en palabras de Zea: "*asimilemos la nueva lógica que ello nos permitirá alcanzar el desarrollo científico y técnico de los Estados Unidos, el mundo occidental y la URSS*" (p.48). Y la misma crítica de Zea aplica para la producción de software, "*nada hará la nueva lógica para que termine nuestra situación de subdesarrollo, sino enajenarnos, una vez más pensando que la palabra, sin la acción que la acompañe, bastará para cambiar nuestra situación*" (p.49.). Replicar el modelo de software propietario ni nos acerca ni los aleja a nuestro desarrollo, el tipo de problemas que se atiendan con la producción de software marcará nuestro camino hacia una situación distinta o no. Sin embargo, el modelo de software propietario por su condición de ocultar el conocimiento codificado inevitablemente conduce a mantener nuestra situación como está, solo cambia el dominador, pero el dominio se mantiene.

El colonialismo interno que resalta Zea en la actitud de replicar el pensamiento Europeo, es evidente en este modelo de producción de software (1969, pp. 89:90). El productor de software latinoamericano que reproduzca código propietario se posiciona como el nuevo dominador, una especie de *tecno-criollo*, que impone su visión de mundo a los otros dominados de latinoamérica, personas usuarias, *cosas que compran*, un software que se crea en la región, que atiende nuestros problemas pero que nos limita en el acceso y la posibilidad de asumirlo como nuestro. Un colonialismo interno que fácilmente decanta en otras formas de explotación, al reducir al usuario a un objeto que compra un

software, formas que buscan la homogeneización de las personas para satisfacer la mayor cantidad de necesidades con el menor esfuerzo, siguiendo la lógica capitalista que está intrínseca en el modelo de software propietario.

Crear software propietario, aunque latinoamericano, supone que cualquier nuevo desarrollo de software se debe hacer desde cero. Es decir, cada producto de software es a su vez una isla de conocimiento porque quedará en manos de las empresas productoras del software. Crear un desarrollo tecnológico que atienda a una nueva necesidad, que no sea de interés de algún productor de software propietario existente, implica empezar un desarrollo desde cero (o casi desde cero). Por lo tanto replicar el modelo de software propietario nos obligaría a entrar en la misma moda de los fabricantes de software hegemónicos.

### *Construir software libre*

La tercera posibilidad es crear nuestro software, ya sea desde cero o apropiándonos de software inicialmente construido en otros contextos. Crear, de forma continua software, implica que debe ser en el marco de la propuesta del Software Libre, caso contrario caería en las otras dos posibilidades, ser usuarios o replicar un modelo de privatización del conocimiento. Esta posibilidad implica desaparecer del debate las restricciones de acceso al conocimiento, con lo cual el debate sobre *nuestro software* adquiere mucha similitud con el debate sobre nuestra filosofía.

El modelo del software libre presenta coincidencias con el camino revolucionario para construir nuestra propia filosofía, que propone Salazar Bondy (1968, 110:113). La propuesta del Software Libre ofrece posibilidades de apropiación/asimilación real, no solo porque nos da acceso al idioma del conocimiento, sino porque nos permite romper el modelo de dominación que presentan las dos formas de aproximarnos al software recién exploradas.

Construir nuestro software no implica necesariamente hacerlo de cero, es decir se puede sacar provecho a software propietario para construir software libre. Sin embargo, esto plantea un dilema para el movimiento de software libre, que Stallman zanja de forma pragmática al indicar que se debe ser estratégico y sacar provecho del software no libre para mejorar el software libre (2004, 33).

En esta línea piensa Salazar Bondy sobre la filosofía latinoamericana, cuando dice que para construir nuestra filosofía podemos hacer uso provisional de la filosofía hegemónica, pero siempre con la intensión final de construir nuestro propio conocimiento o cultura (1969, pp. 115-116).

Zea por su parte no tiene reparo en tomar de otro conocimiento para resolver lo propio, sin que esto implique tampoco perder originalidad. Reusar el software hegemónico y asimilarlo de forma auténtica, hacerlo útil para nuestros fines. Utilizar el conocimiento para producir software con el fin de romper la dominación entre humanos, ese camino sería el que nos acercaría a una producción de software legítima, adoptando el conocimiento existente y adaptándolo para este fin. Acceder al código fuente, como lo ofrece el software libre, es esencial para concretar esta apropiación tecnológica, o asimilación usando un término de Zea que se aplica de manera similar a este caso.

La posibilidad de construir software libre es la más concordante con la mirada de Zea sobre el problema de una filosofía latinoamericana. El software libre es una filosofía comprometida, que reflexiona sobre los problemas de la realidad y actúa para transformarlos, una actitud como la que Zea ve en varios filósofos latinoamericanos. Otro aspecto en el cual hay una coincidencia es en la propuesta de poner el conocimiento a disposición no solo de los latinoamericanos, sino de la humanidad. Construir nuestra tecnología usando software libre es aportar con conocimiento codificado que puede ser adaptado por otras sociedades, en otros contextos y que puede ayudar a corregir problemas usando tecnología, no como un instrumento de dominación de personas usuarias, sino como conocimiento y producto que pueden ser tomados para ponerlos al servicio de las personas y de sus intereses.

Las reflexiones de Zea dejan ver que él tendría una posición coincidente con la filosofía del software libre, una defensa de la apropiación del conocimiento, pero siempre pensando antes en la sociedad que en la técnica. Haciendo un análisis crítico de la realidad, comprender *para qué* hacer software y en qué condiciones esta tecnología genera reducción de brechas sociales, económicas y técnicas. Producir software libre significa que se tienen la capacidad técnica, se domina el *cómo*, pero sin repetir el mismo error de producir una sociedad dominada por la técnica.

El software libre construido en Latinoamérica será original en la medida que responda a nuestras necesidades, las cuales al fin y al cabo son humanas. Poner la tecnología computacional a funcionar para mejorar la condición de vida de las todas las personas. Si bien, construir tecnología para este fin no es algo que únicamente se puede lograr con software libre, se cae en una dominación cuando el software no es libre, pues no hay posibilidad de apropiación de la solución tecnológica e inevitablemente el camino de dependencia es enajenante. El software libre, da garantía de que el conocimiento puede ser compartido, con lo cual siempre está la posibilidad de asumir el proceso tecnológico y mantener acorde con las necesidades de las personas.

Siguiendo a Zea, hacer puro y simple software, lo latinoamericano será por añadidura. Pero hacer software implica tener acceso al código fuente y al idioma de la programación, por lo tanto, hacer software únicamente es posible replicando el modelo de software propietario o construyendo software libre. Hacer software propietario latinoamericano, llevará añadida nuestra visión de mundo, pero replicará la dominación del software hegemónico. Hacer software libre elimina la dominación y mantiene la posibilidad de añadirle lo latinoamericano.

## Conclusión

Hemos revisado el contexto en el cual surge la tecnología computacional, se explicaron las particularidades técnicas y legales del desarrollo de software, y se prestó atención especial a las diferencias entre el modelo de producción basado en la privatización del conocimiento y el modelo del software libre. Seguidamente se hizo el ejercicio de poner en diálogo el proceso de construcción de software con el debate sobre la filosofía latinoamericana de Salazar Bondy y Zea. Enfoqué este diálogo en explorar tres posibilidades en las cuáles participa Latinoamérica en los procesos de producción de software: (1) como personas usuarias de software propietario producido en países hegemónicos, (2) replicando la privatización del conocimiento y (3) produciendo software libre.

Si bien las tres alternativas exploradas no son excluyentes una de la otra y más bien están ocurriendo al mismo tiempo en nuestra región, se ha ofrecido una valoración de las opciones en términos del debate sobre nuestro conocimiento, más allá de si es codificado o filosófico. La posición de los filósofos Salazar y Zea nos ayuda a dimensionar la relación entre dominación y apropiación tecnológica. A partir de estas posiciones el ejercicio de valorar cuál es la propuesta tecnológica más concordante con la perspectiva latinoamericana pasa a distintos planos, el individual, el colectivo (a lo interno de instituciones u organizaciones) y el plano de las políticas públicas que se plantean desde gobiernos a distintos niveles como pueden ser los municipales, nacionales o regionales.

Seguirá abierta la pregunta sobre qué tipo de desarrollo tecnológico queremos en Latinoamérica, pero las posiciones de los filósofos Salazar Bondy y Zea nos dan pistas de qué un desarrollo de software propio debe ser basado en un modelo de software libre. Una de las pistas se manifiesta en el ejercicio de pensar el software libre desde Latinoamérica. En el contexto del surgimiento de la filosofía del software libre, la dominación tecnológica que se busca revertir es la que imponen las empresas estadounidenses que promueven la privatización del conocimiento, la liberación que ofrece la filosofía del software libre esta reducida a ese momento puntual, aproximadamente en los años setenta del siglo XX. El debate sobre una filosofía latinoamericana visto desde el software libre (manteniendo la mirada latinoamericana) aporta una dimensión de análisis sociohistórica sobre la significación de la relación intelectual, filosófica y tecnológica entre las regiones hegemónicas (en este caso de especial atención la estadounidense) y la subordinación de nuestra región. Para las y los latinoamericanos que participen de las discusiones entorno a nuestro desarrollo de software, la revisión de las posturas de Salazar y Zea da soporte y justificación sobre la dimensión ética y política que conlleva decantarse por una propuesta de licenciamiento tecnológico sobre otra. En este sentido tanto las preguntas sobre la existencia de una filosofía latinoamericana como la posibilidad de un software libre latinoamericano se pertenecen.

La privatización del software es la consolidación de una era en la cual se rompe la posibilidad de acceso al conocimiento, ya no por el subdesarrollo

mental sino por la imposibilidad. Ya no por una falta de rigurosidad, calidad o profesionalismo, sino por una falta de acceso a conocimiento privado. Sin el código fuente es imposible asimilar el software y hacerlo parte de nuestro contexto histórico, social y cultural. Esta privatización se manifiesta cuando el software producido es propietario, no importa si se construye en un país hegemónico o internamente en Latinoamérica, cualquier de las dos vías establece la imposibilidad de apropiarse del conocimiento. La industria de privatización del conocimiento instaura una dominación epistemológica que requiere formas radicales de asumir el proceso de construcción de conocimiento. El software libre latinoamericano puede ser un camino de resistencia a la dominación que imponen las formas de privatización del conocimiento.

**Referencias**

Castells, Manuel. La era de la información. Sexta Edición. México: Siglo Veintiuno Editores. Vol. I, 2005.

Illich, Ivan. Energía y equidad. Disponible en línea http://habitat.aq.upm.es/boletin/n28/aiill.html, 1973.

Salazar Bondy, Augusto. ¿Existe una filosofía de nuestra América? México, D.F: Siglo XXI editores, 1968.

Stallman, Richard M. Software Libre para una sociedad Libre. Madrid: Traficantes de sueños, 2004.

Zea, Leopoldo. La filosofía americana como filosofía sin más. México, D.F: Siglo XXI editores, 1969.