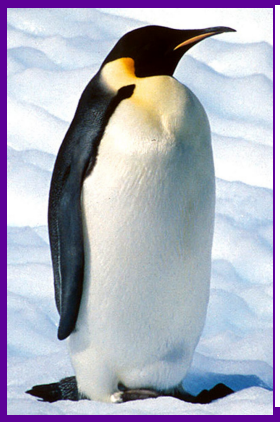
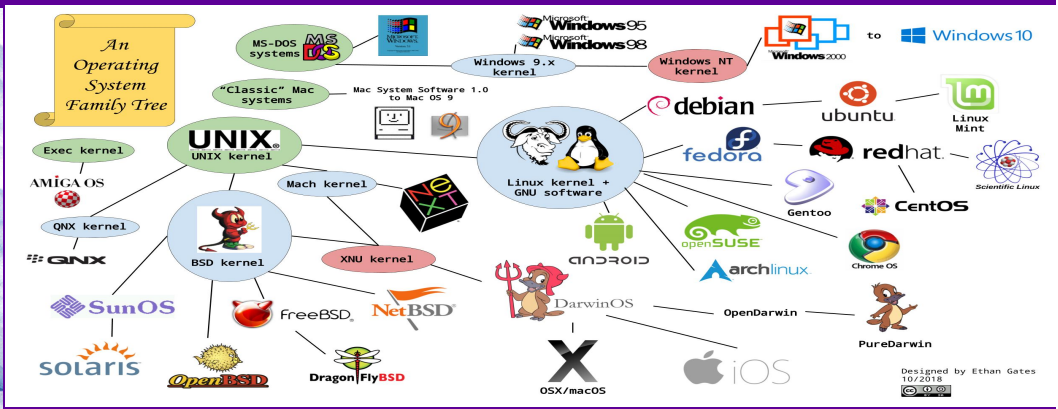
La victoria de Linux

Media vida con los computadores (1989-2024)





Lorenzo Peña

La victoria de Linux

Lorenzo Peña

...

2025-04-29

Versión 4.29

---

[1]. En los términos de la *Licencia Laurentina* (http://lorenzopena.es/hispano/copyrigh.htm) viene ofrecido gratuitamente a los lectores este trabajo, propiedad intelectual de su autor, Lorenzo Peña y Gonzalo.

Implicando la propiedad de un bien el exclusivo derecho de su dueño a usarlo y a beneficiarse de él, nada en la ley vigente permite a persona alguna diversa del autor de este texto tenerlo o usarlo, savo únicamente la concesión del autor, quien, por el presente documento, tan sólo la otorga condicionalmente.

En efecto, la Ley de propiedad intelectual para nada limita los derechos del autor, sino exclusivamente los del adquirente de una copia de la obra de que se trate.

Ahora bien, ninguna copia de la presente obra vendrá adquirida por nadie que no sea el propio autor, ni en propiedad ni en lícita posesión. La posesión del soporte material de una copia no conlleva, en absoluto, la lícita posesión de dicha copia. Todas las copias del ensayo permanecen constantemente en la exclusiva posesión de su dueño y autor, Lorenzo Peña, quien ni las cede ni las transfiere a nadie, ni siquiera las presta, aun consintiendo su uso —y eso condicionalmente.

En virtud del mero hecho de acceder a una copia de este ensayo —comoquiera que haya llegado a hacerlo—, queda ligado con el autor, por un contrato sinalágmatico, quien efectúe tal acceso.

A quienes no respeten íntegramente todas las condiciones —e.d. todas las cláusulas de la Licencia Laurentina— prohíbeles expresamente el autor descargar el ensayo, tenerlo guardado en sus dispositivos, leerlo o aprovecharse de su contenido, sean cuales fueren el modo y el fin de tal aprovechamiento.

En el mismo momento en que se viole alguna de las cláusulas de la Licencia, vendrá *ipso facto* revocada la autorización para tener o mirar el texto, quedando así obligado el poseedor del soporte material, no sólo a

Resumen

Escudriñando el itinerario del autor como usuario de recursos electrónicos, empréndese una indagación sobre el sistema operativo que adoptó a poco de haber empezado su recorrido digital, Linux, al cual ha permanecido fiel a lo largo de 26 años. Viene zurrado su principal rival, Windows de Micro$oft, como un hinchado y ostentatorio bodrio impuesto al consumidor embaucándolo y constriñéndolo, cuando su rendimiento resulta ramplón pese a los oropeles. Alábase a Linux por ser libre en el doble sentido de gratuito y respetuoso de las iniciativas del usuario, si bien la ideología del «*software* libre» resulta deficiente.

• • •

Palabras clave

Linux, Unix, Windows, *software* libre, sistema operativo, DOS, Linus Torvalds, Gary Kildall, Bill Gates, Richard Stallman, GUI, línea de órdenes, *systemd*, WordPerfect, Ubuntu, Slackware

---

Abstract

Scanning the author's itinerary as a user of electronic devices gives rise to the study of the operative system he went in for early in his digital journey and which has remained his chosen companion for 26 years, viz. Linux, Its main rival, Micro$oft's Windows, is trashed as an obtrusive, gaudy and bloated gimmick forced upon users through hoax and coercion while its performance turns out to be tawdry and shoddy. Linux is moreover praised for being free in two senses: it is both respectful of the user's choices and also free of cost, while the ideology of «free *software*» is found fault with.

• • •

Key-words

Linux, Unix, Windows, free *software*, operative system, DOS, Linus Torvalds, Gary Kildall, Bill Gates, Richard Stallman, GUI, command line, Ubuntu, Slackware

Sumario

Capítulo 0

Introducción

¿Quién está autorizado a pronunciarse sobre una materia? ¡Todos y cada uno!

Cada uno, si por «estar autorizado» entendemos que le es lícito. La libertad de opinión permite a cada quien expresar francamente su parecer. Cualquiera puede opinar sobre lo que sabe y sobre lo que no sabe. Sobre lo que conoce un poquito y sobre lo que conoce a fondo. Sobre materias en las cuales es un ducho investigador y sobre otras en las cuales son superficiales sus conocimientos.

Mas hay otro sentido en el cual «estar autorizado» significa estar capacitado para hablar o escribir con autoridad. Para opinar con autoridad sobre una materia hay que vivir como un serio estudioso de la misma, haber trabajado con ahínco y dedicación para dominarla, descubriendo sus complicaciones, superando sus dificultades, adentrándose en sus meandros, hallando el hilo de Ariadna que nos vaya conduciendo a la salida del laberinto y estando así habilitados a ofrecer al público una clave para descifrar los enigmas de la disciplina. Es un trabajo de años.

¿Está, en ese sentido, autorizado el autor de estas páginas a escribir sobre temas informáticos? La respuesta es dialéctica: sí y no.

Lo está en alguna medida. ¿En qué medida? ¿Grande o pequeña? Eso es mejor que lo adjudique el lector. O que lo decidan los especialistas.

No soy ningún técnico informático. No tengo estudios de ingeniería ni de ciencia de la computación. No ha sido ésa mi carrera académica. Soy un filósofo, que ha consagrado su vida académica a las disciplinas de la metafísica, la lógica matemática, la filosofía del lenguaje y —desde hace casi seis lustros— a la lógica y la filosofía jurídicas.

¿Es, entonces, pecar de pretencioso o excesivamente osado escribir un libro sobre temas de informática? ¡Acaso lo sea! De nuevo, dejo al lector juzgar si es así o no.

Dudo que los investigadores especialistas en una disciplina posean un exclusivo privilegio para hablar de ella. Prueba de que no es así la constituyen los esfuerzos que suelen desplegar muchos de ellos (al menos algunos de ellos)

para hacérnosla comprender a quienes carecemos de tal saber especializado. Con éxito o sin él, muchos físicos se empeñan, a través de charlas y escritos divulgativos, en transmitirnos algún conocimiento sobre la física, sobre la mecánica cuántica, la relatividad, la teoría de cuerdas, el multiverso, el principio de incertidumbre o la concepción cíclica de Roger Penrose (galardonado con el Premio Nobel).

¿Conseguimos quienes tanto ignoramos captar algo de todo eso, siquiera atisbando, columbrando —o haciéndonos la ilusión de haber al menos vislumbrado de qué se trata? No lo sé.

Aquello de lo que estoy seguro es que, no sólo por la libertad de opinar, sino también por el beneficio del enriquecimiento cultural, en aras una sociedad con mayor vitalidad intelectual, resulta conveniente que no detenten el monopolio de la palabra los duchos en la materia, quienes tienen un saber pericial (los «expertos» como ahora hay que decir), sino que también, participando en un diálogo, reflejen aquello que hayan entendido los destinatarios de sus (mejor o peor logradas) exposiciones.

Eso es así mucho más aún en temas que tienen una relevancia directa en la vida de la gente —o en facetas de esa vida—, en temas atinentes a las técnicas de las cuales nos servimos a diario, de las que somos directos usuarios.

No somos usuarios de la física cuántica ni de la astronomía. Ni siquiera, en rigor, de la ingeniería naval (salvo los pocos cuya profesión está involucrada en la construcción de barcos o de puertos o en las técnicas de navegación u otras actividades afines).

Mas hay ámbitos de la tecnología en los cuales somos usuarios todos, o casi todos —o al menos muchísimos. Los unos, usuarios avanzados, experimentados, aventajados podríamos decir. Los otros —quizá más pragmáticos—, usuarios sin otra pretensión que la de servirse de los instrumentos sin sacrificar tiempo ni esfuerzo a su comprensión y sin anhelar virtuosismo alguno en su manejo.

Pocas áreas de la tecnología nos resultan hoy más familiares que la informática, en particular en lo que atañe al uso de computadores personales (PCs). No construimos puentes ni diques ni generadores de electricidad, pero a diario manejamos dispositivos electrónicos; más específicamente somos muchos quienes usamos computadores (de sobremesa o portátiles) para una u otra actividad cotidiana —sea de trabajo o de productividad, sea de entretenimiento, sea lúdica.

¿En qué nivel se sitúa el autor de este ensayo en la escala que iría de los desconocedores de la materia sobre la cual escribe a los genuinos peritos y sabios? De nuevo ¡quede la pregunta sin respuesta, estándole vetado ofrecerla al propio autor!

Lo que espero vaya quedando claro con la lectura de los capítulos que

integran este libro es que el autor, sin ser experto en estos temas, no ha dejado de trabajar en ellos, indagando, esforzándose por entender, acopiando datos y explicaciones de los verdaderos técnicos.

Hízolo, inicialmente, por puro interés, dado que, una vez que hubo iniciado el manejo de computadores, le convenía manejarlos con la máxima competencia que estuviera a su alcance.

Siendo un investigador científico en las áreas de la lógica matemática y la filosofía del lenguaje —entre otras— así como un estudioso de la filosofía griega, manejando un computador de sobremesa para escribir sus trabajos académicos, húboselas en seguida con dificultades a las cuales no se enfrentan quienes escriben sobre otros temas, en particular quienes no están constreñidos por su disciplina a usar notaciones simbólicas ni grafías diversas de la latina. Fue ése un acicate, ya desde los orígenes, para comenzar a adquirir, sobre cuestiones informáticas, un dominio que fuera, como mínimo, aquel que serviría para poder resolver las aludidas dificultades.

Sólo que en ese estudio, proseguido y ahondado a lo largo de más de cinco lustros, el autor de este libro fue aprendiendo a captar problemas filosóficos relacionados con la informática. Y los hay. No sólo fui captándolos, sino también esforzándome por elaborarlos teoréticamente, ya que, si la informática en sí caía fuera de mis campos de investigación vocacional, entraba perfectamente en el espacio de la filosofía aplicada lo que podríamos llamar «filosofía de la informática»; y, claro está, la filosofía aplicada es filosofía.

En efecto —según lo irá comprobando el lector a medida que vaya recorriendo los siguientes capítulos—, la informática nos presenta desafíos que nos interrogan filosóficamente, constituyendo genuinas cuestiones filosóficas.

Un programa de computación es una obra intelectual que se expresa a través de un lenguaje de programación, en un acto de escritura o redacción que cae bajo el genérico concepto de «acto de habla» y que, además, se rige por unas reglas lógicas, en íntimo contacto con sistemas de lógica matemática. En todas esas facetas quedan interpelados los filósofos: el filósofo del lenguaje y el lógico, en primer lugar. ¿Nada tendrán que decir sobre el entrecruzamiento de su propia temática profesional con los problemas que surgen al analizar esa peculiar obra intelectual que es un programa de computación?

Yendo más lejos, surge la cuestión de cuál sea el estatuto jurídico de esa obra intelectual. ¿Admitiremos la propiedad intelectual? ¿A quién pertenecerá el programa? ¿Qué deberes o derechos recaerán sobre su dueño y cuáles sobre los intermediarios de su transmisión y sobre los usuarios finales? ¿Qué reclamaciones podrán legítimamente elevar los unos frente a los otros? ¿Qué principios han de regir esa maraña de dificultades jurídicas? ¿Qué filosofía ha de orientarnos en ese laberinto?

Tirando del hilo, sale el ovillo. Aquí estamos ya empantanados en honduras filosóficas: la relación entre el individuo y la colectividad, los derechos fundamentales del hombre y su nexo con las libertades del usuario de *software*

así como con los legítimos intereses de los autores de dicho *software*, las peculiaridades de la propiedad intelectual frente a la material y otra serie de cuestiones similares.

* * *

En los tiempos que corren no goza la filosofía de mucha aceptación social. Enséñase poco, teniendo que defender a brazo partido su ya escasa presencia (tendencialmente residual) frente al expansionismo de las «ciencias sociales y políticas», la «educación cívica» y otros quehaceres intelectuales que alardean hoy de mayor crédito, al ser presuntamente útiles, mientras que la filosofía no serviría para nada ni resolvería nada, al venir enfrascados los filósofos —durante siglos o aun milenios— en sus sempiternas e insolubles controversias.

Frente a tales acosos, hay tres reacciones posibles de los cultivadores profesionales de la filosofía (enseñantes e investigadores).

La primera es la de plantar cara, batallar, salir a la palestra por los fueros de la filosofía pura y dura, a pecho descubierto, afrontando gallardamente el desafío.

La segunda es la de enrocarse o (mejor metáfora) arriar velas, guareciéndose en sus camarotes y esperando que, pasado el temporal, vuelvan tiempos más benignos para la especulación.

La tercera es la posiblemente menos honrosa de las tres, pero la más pragmática y eficaz: sacrificar lo más auténticamente filosófico de la filosofía, ladearlo, arrinconarlo, dedicándose a filosofar sobre temas de actualidad; quien dice «filosofar» quiere decir reflexionar, teorizar con una dosis de vocabulario filosófico o un estilo que, de cerca o de lejos, recuerde el de los filósofos.

Siempre ha optado por la primera de esas tres estrategias el autor de este libro; en la medida en la que le haya sido imposible, por la segunda. Mas tampoco ha desdeñado del todo la tercera. Por el contrario, apostar única y totalmente por la tercera opción ha sido lo que ha decidido la abrumadora mayoría de aquellos colegas que lo han rodeado, al menos de los profesionalmente cercanos.

De ahí ha surgido la proliferación de las «filosofías de …». Filosofía de la geografía, filosofía de la intersexualidad, filosofía de la inteligencia artificial, filosofía del cambio climático, filosofía feminista (¿o también masculista?) y así sucesivamente.

¿Por qué no una filosofía de la computación? Este libro aspira a ofrecer justamente eso: una filosofía de la computación.

No se engañe el lector con el estilo, en buena medida, narrativo de este ensayo ni con el hecho de que, en parte, la autobiografía informática sea —al menos en sus primeros capítulos— un hilo conductor, puesto que, por detrás y más allá de todo eso, irá descubriendo un abordaje de genuinos problemas

filosóficos atinentes al uso de los computadores, a la programación y al estatuto jurídico de las entidades dedicadas a ofrecernos *software* así como al de ese mismo *software* ofrecido.

Problemas que nos llevarán lejos, lejísimos, hasta darnos de bruces con cuestiones de genuina filosofía, como la existencia de un derecho natural (no positivo, no promulgado) así como su relación con la ética y con los derechos fundamentales del hombre.

* * *

Justifica —a mi parecer— sobradamente que un filósofo se adentre en el estudio de los temas abordados en este libro el hecho de que, haciéndolo, desembocamos en cuestiones cuya autenticidad filosófica no resulta polémica.

Muchos son, y muy heterogéneos, los problemas aquí estudiados, sin que salte forzosamente a la vista el nexo entre ellos. Que tal vínculo quede un poco agazapado o escondido no quiere decir que no exista. Seguramente resultará posible descifrarlo al haber concluido la lectura de la obra, a la luz de las consideraciones filosóficas que afloran más en los últimos capítulos.

Capítulo I

Mi primer período como usuario de computadores (1989-1998): DOS y OS/2

En mi ensayo de 2016 «Nuevas consideraciones sobre el Linux»[2] tracé mi recorrido como usuario de productos electrónicos y como aficionado a la informática.[3]

Ha sido mi situación la de un usuario terminal que, no resignándose a serlo a ciegas, mal que bien, con esforzado empeño y marcada tozudez, va aprendiendo briznas de conocimientos informáticos y hasta alguna noticia sobre los instrumentos electrónicos, todo ello a distancia de la mayoría de los usuarios, que no se ven animados por tales ansias de hacer las cosas a su propio modo; afanes cuyo mayor precio no es el tiempo sacrificado, sino el constante riesgo de que todo se derrumbe en cualquier momento, a causa del atrevimiento juntado a la impericia.

Eso sí, gracias a ese denuedo resulta factible no sólo ir paulatinamente alcanzando mayores rendimientos de los artefactos usados, sino —lo que es mucho más— hacer frente a retos intelectuales que le permiten a uno perfeccionarse, hacerse más dueño de su propio obrar, más autor de su pensar, más competente en sus tareas.

Aun tratándose inicialmente de un mero utensilio, la electrónica ha llegado para quedarse, ocupando una parte de nuestras vidas.

Al recordar mis primeros nueve años de usuario informático —antes de adoptar el Linux en abril de 1998— viénenme a la mente mis afanes, mi empecinamiento, mi afán de mejoras.

---

2. V. http://lorenzopena.es/linux/nuevas.htm y https://lorenzopena.es/linux/nuevas_consideraciones.pdf. En ese mismo directorio tengo desplegados varios documentos más de mi autoría sobre el Linux, escritos anteriormente. V. http://lorenzopena.es/linux/index.htm.

3. Aun sin haber superado jamás la modesta condición de un mero diletante, por falta de unos saberes técnicos que difícilmente pueden conseguirse sin haberse consagrado uno a intensa capacitación discente, aprendiendo de los maestros —no autodidácticamente—; es más, ni siquiera como autodidacta puedo preciarme de haber sido, en estos temas, un aventajado estudiante, habiendo renunciado al estudio de lenguajes de programación y, todavía más, a la adquisición de conocimientos electrónicos, o sea al dominio intelectual del *hardware*.

Sin haber sido nunca un bricoleador, sin saber nada de mecánica, sin apenas haber manejado las acostumbradas herramientas domésticas, híceme un experto en destripar los computadores, estando cada poco desatornillando las caracasas para insertar, en las ranuras de expansión, ora nuevos y mejores módulos de memoria RAM, ora un disco duro adicional, ora una tarjeta de CDRom o una de sonido, ora tarjetas ethernet, ora una nueva y mejor tarjeta de vídeo, o una SCSI;[4] y así sucesivamente.

Conocía yo al dedillo, no sólo los diversos tipos de ranuras y tarjetas de expansión, sino también otros muchos datos sobre el *hardware*, además de ser siempre aficionadísimo a los dispositivos de almacenamiento externo, empezando por aquellos ZIP y JAZ de IOMEGA, después los discos duros USB externos y, claro está, algo después los CDROM y DVDROM regrabables, que durante años usé en mis cotidianos respaldos; a lo cual hay que agregar el cableado necesario para esas múltiples tareas, que iba evolucionando.

Más tarde, en los primeros años de la *web* —y, más que nada, con la llegada de la banda ancha en 2002—, organicé en mi casa una red cableada con tres computadores (uno de ellos viejo y desechado en mi centro de trabajo, el Instituto de Filosofía del CSIC); ese computador avejentado y desfasado actuaba de cortafuegos. (Todavía entonces no teníamos wifi.) Cada computador desempeñaba una función, disponiendo yo de un conmutador KVM que alternaba el acceso del teclado, el ratón y la pantalla entre esos computadores.

Todo eso quedaba en poquísimo comparándolo con mi ahínco por aprender y manejar el *software*. Aquellos CDs de Walnut Creek de los años noventa eran minas de plata,[5] de donde iba yo extrayendo decenas de aplicaciones para el sistema operativo DOS, muchas de ellas en régimen de

---

4. Fue el SCSI una técnica que experimentó un fulminante auge y apogeo en los últimos años del pasado siglo y los primeros del actual, viniendo pronto reemplazada por el USB. Otra víctima del raudo progreso fue el PDA, asistente personal digital —con varias marcas, siendo la más conocida el PalmPilot, cuyo sistema operativo era una adaptación del DR-DOS. Esas agendas eran caras para prestaciones bastante limitadas. (Fue el PDA uno de los pocos *gadgets* que yo no llegué a comprar.) En seguida los *smartphones* las harán obsoletas. Igualmente las primeras lectoras electrónicas resultaron costosas, difíciles de manejar y, en suma, de escaso servicio, por lo cual —en el segundo decenio de este siglo— serán sustituidas por los *smartphones* y las tabletas.

5. Walnut Creek CDROM Inc. era una empresa proveedora de *freeware*, *shareware* y *free software* fundada en la villa californiana de Walnut Creek en 1991. Pagando una suscripción se recibían sus periódicos CDRoms, atiborrados de todo ese *software*, mucho de ello destinado a correrse en el sistema operativo DOS. Después dicha empresa abrió el sitio ftp.cdrom.com, del cual podíamos bajarnos contenido sin necesidad ya del soporte material de los CDRoms. Pudo ayudar a que me pasara yo al Linux/Slackware el hecho de que Walnut Creek en seguida se convirtiera al el propagador de esa (casi) primera distribución de Linux. (Vagamente mi memoria tiende a decirme que compré un CDRom con alguna de las primeras versiones de Slackware, posiblemente en 1996.)

En el año 2000 esa compañía se fusionó con otra, dejando en la estacada a Slackware, que quedó huérfano, a la espera de fundar su propia firma comercial con su propio sitio web, del cual me bajé yo, en adelante, las posteriores versiones de esa mi distribución.

freeware o *shareware*.[6]

Desde 1990 hasta algunos años después estuve suscrito a varias revistas sobre los computadores personales. Recuerdo *PC-Magazine* y *PC World*; puede que alguna más. Leía con avidez sus páginas. También estudié a fondo los manuales de varios programas, como las sucesivas versiones del WordPerfect, el DataBase (que, aun habiéndolo usado sólo unos meses, me había enseñado a crear acervos de datos que posteriormente migré al WordPerfect 5.1), las propias versiones del sistema operativo (fui pasando del MS-DOS al PC-DOS de IBM, el DR-DOS y finalmente el OS/2, en sus tres versiones sucesivas: OS/2 a secas, OS/2 Warp y OS/2 Warp Connect).

A lo cual agrégase el aprendizaje de uno de los primeros programas de OCR (Reconocimiento Óptico de Caracteres), gracias al cual (con aquella rudimentaria y lentísima escrutadora comprada en el otoño de 1989 [un *flat scanner* cuya marca no recuerdo]) empecé la titánica tarea de digitalizar (y —en la medida de lo posible— convertir en texto editable) una porción no desdeñable de mi propia producción intelectual anterior, mas también de textos de otros autores que posteriormente fui poniendo a disposición del público desde los comienzos de la *web*, hacia 1996.).

No ya trabajoso sino, peor, voraz consumidor de tiempo fue el estudio de la impresión. Tuve que empollarme largos y áridos tratados sobre los controladores de impresión, aprendiendo, desde zero, un montón de cosas sobre las técnicas, las fuentes, los tipos de impresoras, las dificultades, las soluciones (si bien éstas, en buena medida, tuve que ir inventándolas yo mismo). Fui afanosamente aplicando tales conocimientos a una tediosa y extenuante labor de edición de los controladores de impresora del WordPerfect.

Gracias a lo cual pronto mis presentaciones por escrito tuvieron esa hermosa vistosidad y elegancia que contrastaba con los pedestres textos producidos por el M$Word, ramplones, manidos, vulgares, clónicos, impersonales, inatractivos, cortados todos por el mismo patrón y cuya lectura se le hacía a uno cuesta arriba sólo de verlos.

Acaso no sea del todo exacto decir que estuvo enteramente privada de las artes de la programación mi afición a la informática en los años del DOS (1989-98). Verdad es que no aprendí ningún lenguaje de programación propiamente dicho (aunque había acariciado el propósito de adquirir rudimentos de lenguajes sencillos, como el BASIC y el PROLOG; lamentablemente no saqué nunca tiempo para cumplir esa intención). Ahora bien, aprendí a fondo el mecanismo de las macros del WordPerfect 5.1 —que no deja de constituir un genuino lenguaje de programación; lo aproveché escribiendo y aplicando en mi trabajo cientos de tales macros.

---

6. Unas cuantas de ellas las pagué a sus autores, como humilde contribución personal a remunerar su trabajo. Es obvio, no obstante, que, en la mayoría de los casos, no me conduje con tanta gratitud —ni mis finanzas me lo hubieran permitido.

Igualmente escribí y utilicé a fondo no pocos *scripts* del sistema operativo DOS, habiendo estudiado los manuales sobre sus órdenes, la concatenación de las mismas, las funciones, el empleo de las variables, los bucles, las iteraciones. Más aún cuando pasé del intérprete de órdenes por defecto (el *command.com*) a otro más poderoso y elaborado, el *4dos.com* (al cual se sumará, en mi período con el OS/2, el correspondiente intérprete de órdenes *4os2.com*). (Los *scripts* para sendos intérpretes avanzados no ostentaban la usual desinencia «.bat», sino «.btm».)

Algunos de mis *scripts* (tanto .bat cuanto .btm) eran capaces de realizar tareas de cierta complejidad, que bien me hubiera gustado hallar en aplicaciones de ajena autoría ya listas para usar, habiéndome ahorrado así el trabajo. No voy a enorgullecerme de tan modestísima hazaña —que a cualquier programador profesional le resultará pueril. Lo cual no quita para que, en comparación con el usuario medio, fuera verdad que yo manejaba el utillaje informático con un grado de proficiencia.[7]

No voy a pasar por alto el aprendizaje (igualmente en la segunda mitad de los años noventa) del lenguaje HTML (en las versiones entonces disponibles); gracias a ese estudio (que prosiguió y se ahondó en años siguientes) elaboré

---

[7]. Una de mis astucias fue ponerme a salvo de los virus, que, cual pavorosa epidemia, en seguida inundaron los computadores que corrían bajo DOS. Hícelo editando —con una aplicación de edición de ficheros binarios— el kernel del sistema operativo, o sea los ficheros de arranque (io.sys y msdos.sys para el MS-DOS; ibmbio.com y ibmdos.com para el PC-DOS —y, si mal no recuerdo, también para el DR-DOS).

Cambié las extensiones adjudiucadas a los binarios ejecutables, que eran «.com» y «.exe» así como la extensión «.bat» de los *sripts* —programas también ejecutables escritos por el usuario. En lugar de esas tres extensiones, inventé otras, arbitrarias (no las recuerdo, pero una de ellas podría ser, v.g., «y7v» o cualquier otra ristra de tres caracteres, aleatoriamente escogidos, de entre los 38 admisibles según la gramática del DOS [los 26 alfanuméricos del alfabeto latino internacional, sin distinción entre mayúscula y minúscula, más los diez guarismos y dos signos: el guión y el subrayado]). Así, el procesador de órdenes, el COMMAND.COM, vendría llamado de otro modo, v.g. «SWP49BA.Y7V». También ese COMMAND.COM, redenominado, lo editaba del mismo modo. (Huelga decir que asimismo modifiqué el config.sys, evidentemente.) A partir de ahí, a todos los binarios ejecutables les fui cambiando el nombre, uno por uno, adaptándolos al nuevo canon que yo mismo había fijado; asimismo los ficheros «.bat» llevarían otra extensión por mí inventada.

Sabía yo que cualquier virus operaría infiltrándonos algún binario ejecutable con una de las extensiones estándar; ese ejecutable infiltrado buscaría los demás binarios ejecutables para infectarlos de suerte que, en lugar de su cometido, realizaran fechorías. Sólo que, con mis alteraciones, nada de eso podría llevarse a cabo. Lo primero que, para surtir su maléfico efecto, hubiera tenido que hacer el virus sería buscar en el kernel cómo se llamaban en mi máquinas los ficheros ejecutables; mas, para hacerlo, necesitaba haber ejecutado algo con un binario ejecutable; binario que, en mi máquina, no lo sería mientras el infectante no hubiera adivinado mis peculiares extensiones. (De vez en cuando podía yo modificarlas.)

Quedaba, no obstante, la posibilidad de lo que, en criptología, se llama un «desciframiento de viva fuerza», cual sería ir ensayando, una por una, todas las posibilidades. Quien recuerde el cálculo de permutaciones sabrá que, de entre 38 elementos, tomados de tres tres en tres —con posibilidad de repetición—, existen 38^3 cúmulos, o sea 54.872. Dadas las circunstancias, esa vía estaba bloqueada.

El hecho es que —de entre mis conocidos— fui uno de los pocos que nos libramos de venir contagiados por virus.

nuevas macros avanzadas del WordPerfect que me permitieron convertir documentos del WordPerfect 5.1 en textos escritos en HTML (incluso con toda la gama de signos de lógica, de matemáticas y de griego; ahí paró mi empeño —sin pretender abarcar otras grafías, que no eran pertinentes para mi trabajo como editor de la revista electrónica de filosofía *SORITES*).

Continúo hoy escribiendo mis textos .html con los conocimientos entonces adquiridos (si bien los he ampliado un poquito más recientemente), rehusando casi todas las innovaciones del HTML5 (particularmente los estilos CSS) y no titubeando en usar elementos y atributos *deprecated*. Hay quien se admira de lo anticuado de ese lenguaje HTML de mis textos actuales. Pienso yo que un adecuado criterio para juzgarlos es el de su visibilidad, su eficacia y su sencillez, no el de si la acuñación del código es reciente o antigua.

En efecto, con ese lenguaje presento textos mucho más perspicuos y elegantes que tantos documentos hinchados, abultadísimos, cuya fuente resulta ilegible por lo sobrecargadísima que está de decenas y cientos de redundantes códigos; son ésos los resultados de haber escrito el texto en un programa de *bloatware* como el ínclito M$-Office (o su equivalente de Mac) para luego someterlo a la exportación al HTML que ofrecen tales programas.

Transfórmase así fácilmente un texto de poquitas páginas en un fichero de más de un megabyte. Resulta penosísimo leer su fuente para corregirla, inspeccionarla, inspirarse en ella o para cualquier otro fin.

Pregúntome cuántos hoy conocen el lenguaje HTML, en ninguna de sus sucesivas versiones. (El ámbito de mi desconfianza ni siquiera exime a todos los expertos informáticos.) Conocer ese lenguaje significa ser capaz de escribir con él sobre el papel o con un teclado (sin chuletas), en lugar de confiar la tarea a la función exportadora o conversora del programa de tratamiento de texto (sea el Libre Office o cualquier otro).

Capítulo II

La adopción del Slackware-Linux en 1998

No será mi principal acicate, para abrazar el Unix/Linux en 1998, comulgar ideológicamente con el liberismo del GNU y del movimiento del «*software* libre» (cuyos principios expondré y discutiré en el capítulo XXII). Serán, antes bien, otros cuatro factores.

1º.— Mi relativa insatisfacción con el OS/2, que, no colmando mis expectativas, no se adaptaba del todo a mis necesidades.

2º.— La esperanza (pronto cumplida, aunque nunca completamente) de que con el Linux me fuera mejor —a sabiendas de que, siendo *unixoide*, era un sistema operativo de calidad y de potencia superiores; un sistema más profesional, usado, no sólo por los ingenieros electrónicos e informáticos, sino también por los matemáticos —digamos por la gente (en ese tipo de asuntos) más competente, más seria y más estimada por mí.[8] Quizá no me faltara un tantico de esnobismo en esa pretensión de acercarme al modo de obrar de un ámbito vocacional por el cual sentía —y siento— fortísima admiración.

3º.— Mi anhelo por hacer las cosas a mi modo, buscando nuevas vías, aprendiendo, no contentándome jamás con lo que, empaquetado, me

---

[8]. Sobre todo en aquellos años, durante los cuales, frecuentando los simposios internacionales de lógicas multivalentes, estuve en contacto con ese sector profesional.

Un contacto ligeramente estrechado por mi estancia semestral en Canberra, 1992-93, como investigador visitante en la Universidad Nacional Australiana. Si bien el centro que me invitaba era el Departamento de Filosofía, mi despacho se ubicó en la unidad de electrónica e informática (un barracón prefabricado, justo enfrente), donde —en palabras del P. Mariana— pasé con un informático polaco varios meses «entretenidos ambos en conversaciones instructivas y amistosas, en lo que encontrábamos no poco placer y esparcimiento»; uno de nuestros temas de plática era el Unix, cuyas virtudes él alababa, mostrando escaso aprecio por el DOS.

En esa unidad era yo el único en usar un PC; servíanse los demás de terminales vinculadas a un *mainframe* central bajo Unix. Inconveniente, para mí, de tal sistema era la imposibilidad de tener un autor lo que él mismo había escrito, ya que los terminales no admitían disquetes ni otros dispositivos de almacenamiento externo —para evitar así los virus. El único almacenamiento estaba en la máquina *mainframe* y la única posibilidad de poseer una copia suya el autor era imprimiéndolo.

En general, sin embargo —y bien a mi pesar— escasa vecindad he solido tener, a lo largo de mi vida, con matemáticos e ingenieros.

vendían las multinacionales del *software* (ya fuera Micro$oft con el Windows o la IBM con el OS/2; menos aún el Mac de Apple, que jamás se ha cruzado siquiera en mi camino).

Entrelázanse en este tercer factor dos hebras. La una es mi empeño, desde mi temprana mocedad, en pensar a mi modo y —en la medida de lo lícito y factible— obrar a mi modo según esos pensamientos propios, desagradándome seguir la corriente. (No se trata de ningún prurito de distinción ni afán de originalidad, sino más bien de no influenciabilidad por «lo que se lleva».) La segunda hebra contenía una desconfianza, no exenta de hostilidad, para con las potencias del gran capital.

4º.— Una preocupación, en parte por el precio, mas también en parte por la licitud del uso que venía haciendo hasta entonces del *software*.

A lo largo de aquellos años noventa y del siguiente decenio cambié varias veces de computador. Cada vez que lo hice tuve que pagar la licencia del correspondiente sistema de Micro$oft (ya fuera el M$-DOS al comienzo, o de uno u otro de los sucesivos y malhadados Windows, como el Windows95, cuyo CD de reinstalación todavía guardo). Resultaba dificilísimo (o más bien, en la práctica, imposible) comprar un computador IBM-compatible que no llevara incorporado, como OEM, el sistema operativo que hubiera impuesto a la sazón Micro$oft. Cuánto encarecía eso la venta lo ignoro.

Inicialmente el Windows98 fue lanzado al precio de 392 dólares; dólares de entonces; una suma que viene a equivaler, aproximadamente, a un millar y medio de dólares actuales. (La mayoría de los lectores desconoce, sin duda, lo carísimo que era entonces todo el material electrónico.) Cierto que la propia firma Micro$oft, rebajando su avidez de lucro, saldó después ese mismo sistema operativo a poco más de cien dólares; una verdadera ganga, gracias a la cual esa multinacional vendió tantísimas copias que, habiéndose ya convertido Bill Gates en el hombre más rico del mundo, ascendió a una altura de sideral opulencia.

Como comprador de un utillaje electrónico necesario para mi trabajo, sentíame yo agraviado por ese expolio, que me resultaba un abuso de derecho. Conque no poco me satisfacía relegar tales sistemas operativos, impuestos y costosísimos, reemplazándolos por otro libremente elegido y, además, gratuito.

Mas ese cuarto factor tiene otro componente. Sucedíanse a ritmo veloz las versiones del DOS y del Windows. Ya en el propio DOS, había ido reemplazando yo —según lo he relatado más arriba— la distribución de Micro$oft por la de IBM, luego la de Compaq y finalmente la de Digital Research (el DR-DOS). Mas fue, sobre todo, abrumadora la secuencia de los efímeros Windows sucesivos: el 3.0, el 3.1, el 95, el 98, el ME, el Millennium, el Vista y el XP. (Seguirían otros muchos, como el Windows 7, el 8, el 10 y el 11, etc.) Además, ningún nuevo Windows aseguraba la compatibilidad hacia atrás. (Al adquirir un nuevo dispositivo periférico solía venir con un *driver* para Windows;

sólo que raras veces era utilizable ese *driver*, pues, cuando no estaba hecho para una versión de Windows pretérita, lo estaba para otra que aún no tenía uno instalada en su máquina.)

Podía yo ir consiguiendo copias gratuitas de esas nuevas distribuciones del Windows, pues gentilmente me las iba procurando el gerente de mi centro de trabajo. ¿Era legal ese traspaso? Desconozco si mi institución había realizado una compra corporativa o no; de haberlo hecho, podría lícitamente usarse en todos los computadores internos. Mas ¿también en los que tuvieran en sus domicilios los miembros de la institución, por mucho que se utilizaran para su trabajo institucional?

Ni apenas conocía yo tales problemas jurídicos ni, desde luego, me quitaban el sueño. Solía decirse que estábamos usando *software* «pirateado». Dejo de lado lo abusivo e impropio de ese uso del vocablo. Sea como fuere, prefería yo recurrir lo menos posible a un real o presunto pirateo. Con el Linux no había pirateo ninguno. La copia y la difusión eran perfectamente legales.

¿Intervino un quinto factor ideológico? Sí, mas no exactamente una adhesión a los ideales liberistas del GNU de Richard Stallman (que —con deleite y entusiástica adhesión— viniéronme referidos por los profesores de aquel cursillo sobre el Linux al cual tuve la fortuna de acudir en la Universidad Politécnica de Madrid en abril de 1998 —constituyendo el empuje decisivo para pasarme a ese sistema operativo).

Según lo veía yo, el Linux (o el GNU/Linux, como se decía al comienzo) lo producían ingenieros electrónicos e informáticos, generalmente investigadores universitarios (v.g. del MIT o de las Universidades de Stanford y Berkeley), por más que algunos de ellos trabajaran en relación con grandes empresas electrónicas. Estando entonces todavía en sus comienzos las agencias de notación académica, no dependía de su productividad la remuneración de los investigadores. Gozando de la confianza de las autoridades académicas, podían producir por amor a su vocación, a la ciencia y al servicio público, sin afán de enriquecerse con ello, lo cual permitía difundir gratis sus programas, causando indirectamente la gratuidad del sistema operativo Linux.

Hoy sé que incluso entonces las cosas eran más complicadas. De hecho la distribución del Linux a la cual más adicto he estado yo a lo largo de mi itinerario electrónico, Slackware, la produce una microfirma comercial (ligada entonces a la casa de Walnut Creek que vendía los CDs con aplicaciones de *freeware* o *shareware* para el DOS; según lo he recordado, yo estaba suscrito a esa producción serial de CDs).

No duró la adscripción académica de los adalides del GNU/Linux, entre ellos dos apóstoles (y, en cierto modo, émulos), Linus Torvalds y Richard Stallman, quienes hubieron de buscarse la vida de otro modo.

En seguida la distribución más reputada de Linux será la Red Hat (convertida unos años después en RHEL, *Red Hat Enterprise Linux*), producida por una importante firma mercantil, hoy absorbida por la IBM.

Producidas asimismo por compañías con fin de lucro lo son otras distribuciones, entre ellas Ubuntu —obra de la firma británica Canonical, cuyo patrón es el empresario surafricano Mark Shuttleworth; una distribución que, junto con sus derivadas ramificaciones, ha sido —y, en mi modesta opinión, sigue siendo— la preferida de la mayoría de los usuarios del Linux.

Son también comerciales Suse y, al menos, otras dos: Manjaro y ElementaryOS.[9]

Que una distribución del Linux sea producida por una empresa con ánimo de lucro no significa, empero, que la empresa se lucre con tal distribución. Canonical no gana ni una libra esterlina distribuyendo gratuitamente Ubuntu; el beneficio procede de los servicios que presta, ésos sí de pago.

Algunas entidades, declaradamente sin afán de lucro, productoras de distribuciones de Linux (de nuevo gratuitas) puede que, en realidad, sean cooperativas u otras sociedades no totalmente altruistas.

No resulta fácil el distingo jurídico entre entes colectivos con y sin ánimo de lucro. Sí lo poseen aquellos que reparten dividendos. Y no los que tienen declarado un fin benéfico. Mas que se abstengan de repartir dividendos las cooperativas y los fondos de inversión no obsta a que sus socios se lucren gracias a su respectiva participación societaria. Por otro lado, gracias a espléndidos donativos y subvenciones, pueden procurarse ganancias de su actividad aun las asociaciones caritativas, en las cuales la membría no se vende; en virtud de lo cual pueden remunerar pingüemente a sus directivos y a otro personal.[10]

Mientras vivamos en una sociedad de propiedad privada y en una economía de mercado, es quimérico esperar que el *software* escape al poder del capital y a los circuitos del lucro o que sea «libre». Lo cual no suprime la enorme diferencia entre, por un lado, la insaciable y avasalladora codicia de los monopolios u oligopolios como Micro$oft y, por otro lado, pretensiones muchísimo más modestas y asumibles de otros colectivos que, ciertamente, aspiran a vivir de su trabajo, mas igualmente saben mostrarse mesurados y aun generosos.

Para los adeptos de «todo o nada» será irrelevante que al principal

---

9. Ésta última ha menguado, convirtiéndose en una microempresa individual.

10. Asimismo viven —al menos en parte— de las dádivas y compras simbólicas de sus seguidores los influenciadores y animadores de canales linuxeros; lejos de que esos modestísimos ingresos los lleven a la opulencia, es un hecho, no obstante, que su actividad no resulta meramente altruista; no hay en mis palabras la más mínima acritud u hostilidad, ya que a menudo aprecio sus opiniones e informaciones —infinitamente mejor fundadas, desde luego, que las mías—, tomándolas en cuenta para mis decisiones, sin forzosamente atenerme a ellas.

artífice del Linux, el informático fino-estadounidense Linus Torvalds,[11] se le atribuya una fortuna de sólo 150 millones de dólares; o sea un dólar por cada millón de dólares integrado en el patrimonio de Elon Musk, 151 billones. El número uno es Jeff Bezos con 177 billones. El tercero es Bill Gates, con 124 billones; el cuarto, Mark Zuckerberg con 97 billones. Larry Page, de Google, no llega a los 92 billones. Quédase el preboste de Apple, Tim Cook, en la modesta cifra de 1'3 billones de dólares.

El otro gran líder y fundador del Linux (del GNU/Linux, según quiere él que se diga) es el programador Richard Stallman, cuya fortuna se desconoce, existiendo señales de que vive en la pobreza (pareciendo extraídos sus únicos recursos de honorarios por las conferencias que pronuncia).

Conque, si el flujo de dinero no es ajeno al mundo del Linux, hay una sideral distancia entre el más ilustre y acomodado de los linuxitas y cualquiera de los oligarcas MATAFA (Micro$oft, Apple, Tesla, Amazon, Facebook y Alphabet [la empresa dueña de Google y Youtube]).[12]

* * *

¡Cerremos esa digresión! Pregúntome ahora cómo obtienen beneficios las empresas productoras de distribuciones del Linux. Imagino que es típico el caso de Canonical, quien ha desarrollado una línea de negocio, consistente en la suscripción «Ubuntu Pro», que da derecho, junto con un soporte (servicio posventa no precedido de venta alguna), a frecuentes parcheos de seguridad (taponar en el descargado *software* cualesquiera boquetes por donde pudiera colarse un *malware* o pudiera tener lugar una intromisión).

Según cuál sea el equipo asegurado (un PC personal o un servidor que atienda a un mayor o menor número de máquinas terminales) y según el nivel de servicio contratado, las suscripciones van de $25 al año a $3400 anuales.

Que Canonical preste tal servicio es una de las causas del recelo y la malquerencia hacia esa firma (y, de rebote, hacia su distribución) por parte de un cierto sector de usuarios de Unix/Linux, quienes sospechan que Canonical empieza por lo gratuito para granjearse así clientes a los cuales, una vez encandilados, ofrece un mejor servicio, éste de pago.

Notemos que algo similar sucede en otros campos. También es gratuito el sistema operativo Android, mas en seguida invítanos Google a visitar la tienda Google Play para comprar allí aplicaciones adicionales (sin ninguna duda con una ganancia económica para Alphabet Inc.) En el ámbito académico, el

---

[11]. Más que finés (según él orgullosamente se proclama), Torvalds es un sueco de Finlandia; digamos: un sueco con ciudadanía finlandesa. Sólo que esa pertenencia nórdina es residual y, en rigor, pretérita. A todos los efectos Torvalds es un norteamericano *true blue*.

[12]. Podríamos, alternativamente, aplicarle la denominación «MAXAFA», toda vez que Elon Musk es patrón de Tesla y de la presunta red social X.

portal Academia.Edu, al ofrecer una serie de servicios gratuitos a la comunidad universitaria mundial, exhorta a sus usuarios a hacerse clientes, comprando servicios adicionales.[13]

Personalmente no tengo objeción. Soy yo partidario de una sociedad sin economía de mercado y sin propiedad privada **de los bienes materiales**. Mas, en habiéndola, dentro de sus reglas, ¡que cada cual se busque la vida! Al revés, agradézcole yo a Canonical lo que nos da gratis, sin sentir inquietud alguna que me empuje a contratar ninguna de esas suscripciones, al fin y al cabo voluntarias.

¿Cuán provechosos son los negocios de Canonical? Sus ingresos en 2023 han sido de 251 millones de dólares (habiendo experimentado un aumento de 44 millones respecto al ejercicio precedente), con un beneficio bruto del 80%, si bien la ganancia neta es de sólo 47 millones, yéndose el resto en gastos administrativos, de venta, de *marketing* y de I+D. La ganancia neta ha bajado en 15 millones respecto al año 2022).[14]

El 80% del beneficio proviene de servicios de suscripción, como el *Ubuntu Pro*, con un 20% generado por otros servicios profesionales junto con el *crowd-founding*. Paradójicamente, de esa cifra de negocios, el 80% procede de USA, mientras que Europa apenas contribuye con un escuálido 8%.

Canonical tiene (en 2023) 514 empleados, con 21 patentes concedidas.[15]

Modestísimas son esas cifras, no ya en comparación con la de oligopolios o monopolios como Micro$oft, sino incluso con la más poderosa firma desarrolladora de Linux, RedHat, con casi 4.000 millones de ganancia anual.[16]

Lejos de condenar a Canonical por afanarse en su propia rentabilidad (remunerando a sus accionistas), congratúlome de ello. Yo saco ventaja, puesto que, gracias a prosperar financieramente, puede Canonical seguir ofreciéndonos su maravilloso producto **gratis**, Ubuntu.

Dudo que el único fin de los dirigentes de Canonical sea enriquecerse más, pues, de ser así, ¿no les valdría más buscar otra línea de negocio —o,

---

[13]. Bien sé que contra Academia.Edu se ha lanzado, con saña y arrebatado furor, un atronador griterío. No contentándose con denigrar los servicios que presta esa plataforma, conminan a cuantos universitarios desplieguen en ella sus documentos a cancelar tales cuentas, abochornándolos por seguir usándola.

Sin tener yo nada que ver con los dueños de la plataforma —ni prejuzgar cómo actuarán en el futuro, al carecer, lamentablemente, de dotes proféticas—, acaso, en otro momento, me decida a refutar tales invectivas.

[14]. Tomo esos datos de la página linuxiac.com/canonical-reports-revenue-growth-in-2023/.

[15]. Información facilitada por la página pitchbook.com/profiles/company/82903-06.

[16]. RedHat ha sido comprada por la IBM, mas subsiste, al parecer, con su propia personalidad jurídica.

como mínimo, ser menos generosos? Su ánimo de lucro en nada nos afecta a la abrumadora mayoría de los usuarios. Supongo que han suscrito contratos de soporte muchas empresas, públicas o privadas, que corren Ubuntu-Linux en sus computadores, especialmente en sus grandes servidores. (Sin embargo, al parecer, hay también muchos servidores con Ubuntu que no se abonan a tales servicios, a diferencia de lo usual con RedHat, de cultura más empresarial y ganancia.)

Capítulo III

La gratuidad del Linux, genuina libertad

Jamás habría conquistado el Unix/Linux su gran popularidad sin la Fundación Linux, entidad exenta de ánimo de lucro, que no por ello deja de ser un consorcio empresarial, financiado y auspiciado por oligopolios del sector (incluso por Micro$oft, que ha cambiado de táctica para con el Unix/Linux, cejando en su primera reacción de desdeñoso hostigamiento para adoptar otra de satelización; volveré sobre ello en el capítulo XXI).

Desconozco cómo pueda saberse qué cantidades aportan a la *Linux Foundation* las grandes firmas que integran su patronato.

Abriendo el portal de esa fundación (en 2024-06-07) entérome de que ese Patronato abarca trece miembros de platino (o sea aquellos que se han comprometido a entregar, regularmente, más dólares a las arcas de la fundación).[17] Siguen a continuación doce miembros de oro.[18]así como Figuran más abajo los 1243 miembros de plata.[19] A la cola están 441 miembros asociados. (¿Serían acaso miembros de cobre o de hierro?)[20]

Pregúntase uno qué finalidad persiguen esos patrones. En algunos casos, está clara. Gracias al kernel de Linux son posibles los sistemas operativos Android y MagicOS, lo cual redunda en enorme ganancia de Google,

---

17. Trátase de: Ericsson, Fujitsu, Hitachi, Huawei, Intel, Meta, Micro$oft, NEC, Oracle, Qualcomm, IBM/RedHat, Samsung y Tencent.

18. De entre los cuales cito a Baidu, Google, Dell, Cisco, Honda, Panasonic, Sony, Toshiba y *web*ank.

19. Entre ellos: Comcast, Adobe, Alibaba, AWS, American Express, Anaconda, Bell, BMW-Group, BNP-Parisbas, Bosch, Bull, Deloitte, Ebay, Fujifilm, GitHub, GoDaddy, GoldmanSachs, Hewlett-Pachard, HSBC, Ikea, Leica, Lenovo, LG-Electronics, Linkedin, Mastercard, Michelin, Mitsubishi, MorganStanley, Motorola, Netflix, Nissan, Nokia, Orange, Simba, Société Générale, Swisscom, WaltDisney-Studios, Suse, Suzuki, Uber, Swisssbank, UBS, Verizon, Vodafone, WesternDigital y China Telecom (junto con varias otras empresas chinas, públicas o privadas).

¡Inesperadamente figura en esa lista Nvidia! Lo resalto por la obstinada resistencia de esa firma a colaborar con Linus Torvalds en el suministro de los *drivers* necesarios para que el kernel de Linux pueda soportar los dispositivos de dicha marca, lo cual resulta imprescindible, sobre todo en lo atinente a tarjetas gráficas.

20. De ese medio millar, citaré a las Universidades de Stanford, Princeton y Monash (junto con otras más modestas, entre las cuales al menos tres españolas), la Banque de France-Eurosystème, la BBC y Blockchain.

Samsung, Honor[21] y otros fabricantes de *hardware* y *software* para dispositivos móviles (teléfonos y tabletas). En los demás casos, no estoy en condiciones de conjeturar el motivo.

Sea como fuere, lo único que todo eso quiere decir es que el Linux no es un empeño de puro altruismo alejado de cualquier afán de lucro, al no estar al margen del mundo de las finanzas y los negocios.

Pero no nos venden su producto ni la Fundación ni sus patronos ni las agrupaciones o empresas que publican las distribuciones del sistema operativo. Nos lo dan gratis. Al menos todas aquellas a las que he accedido yo por la *web* tienen colgadas sendas distribuciones libremente descargables: «libres» en todos los sentidos de la palabra. Es posible que haya excepciones, mas yo no las conozco. Ni siquiera aquellas distribuciones que vienen calificadas de «comerciales» exigen un pago previo a la descarga, por mucho que exhorten a los donativos, o sea limosnas.

Lo cual no es óbice para que algunas o muchas de las distribuciones vendan CDs o DVDs con su contenido. Es como si un autor/editor cuelga su libro libre y gratuitamente descargable en la *web*, a la vez que también lo vende en papel. (Con la diferencia de que hay muchos amantes del libro-papel, mas no tantos del CD/DVD de *software*.)

De entre las distribuciones a las cuales he accedido yo, me he topado con una que parece reclamar un pago: ElementaryOS.[22] El precio, sin embargo, es la voluntad, a partir de 10 dólares, o sea irrisorio. (Y acaso sea descargable sin abonar ni siquiera esos diez dólares.) (También es de pago la distribución Astra Linux, rusa, en su modalidad avanzada, ofreciendo otra gratuita, la común.)

Benefícianos tal gratuidad, no sólo a los millones de usuarios de una u otra distribución del sistema operativo Unix/Linux, sino también a los mucho más numerosos compradores de *smartphones* y tabletas Android y Honor (sistemas operativos cuyo núcleo o kernel es el del Linux, sin que eso implique la mismidad de los sistemas operativos en sí —sobre lo cual volveré más abajo).

---

[21]. Honor era, en realidad, una filial de Huawei, que —al venir ésta última empresa china golpeada por las sanciones estadounidneses— optó —con la anuencia de su casa matriz— por separarse para escapar a las sanciones. Su sistema operativo, MagicOS, también se denominó «Honor», siendo, inicialmente, un derivado de Android. Y sigue siéndolo.

Actualmente Huawei ya no comercializa dispositivos con ningún derivado de Android sino que en sus productos de *hardware* instala su nuevo sistema operativo, HarmonyOS, de fuente abierta mas, al parecer, totalmente desvinculado de Linux —e incluso guiado por un principio opuesto, ya que, mientras el kernel de Linux es un núcleo monolítico, Harmony es un micro-kernel. Huelga explicar aquí tal diferencia (tarea que le costaría esfuerzo al autor, quien difícilmente aportaría al lector una dilucidación satisfactoria.)

[22]. Trátase de una distribución que se ufana de ser ética, orientada a atraerse usuarios de Windows y, principalmente, de Mac.

Quien adquiere un dispositivo móvil de Samsung, de Xiaomi o de Honor paga únicamente el *hardware* (más los impuestos aplicables), pero no el *software*, que se nos regala sin costo alguno.

Cierto que en las tiendas de Huawei y de Google podemos hallar muchas aplicaciones de pago; mas aun sin ellas funcionan bien los sistemas operativos basados en el kernel de Linux, respectivamente Honor y Android. Quien se conforme con lo básico nada pagará por el *software*.

A buen seguro que pocos de los patronos de la *Linux Foundation* actúan desinteresadamente. (Acaso lo hagan las Universidades integradas en el pelotón de cola.) Habría de averiguarse cómo venga a redundar en ganancia suya tal generosidad (siendo verosímilmente variable según los casos). Es seguro, empero, que no se trata de la relación mercantil directa. Ninguno de ellos nos reclama ni un euro a cambio.

Otorgo yo enorme importancia a ese factor crematístico porque (según se verá mejor en el capítulo XXII) discrepo de la ideología liberista de Richard Stallman y de la *Free software Foundation*, para quienes el *software* ha de ser libre mas no tiene por qué ser gratuito.

De varias de las libertades preconizadas por Stallman y sus seguidores prescindo yo gustoso, apreciando, en cambio, la libertad de usar un producto sin sufrir, por ello, el castigo de verme privado de un dinero que prefiero dedicar, ora a la satisfacción de necesidades propias y de mi familia, ora a una donación filantrópica.

Siendo un producto que le da sopas con ondas al obeso, torpe y desaliñado Windows, el Unix/Linux, de venir producido para ganar dinero, encarecería lo que sus usuarios gastamos para poder manejar nuestros computadores. (O, más verosímilmente, caería en bancarrota, impotente para competir en ese terreno con el gigante Micro$oft.)

De tener que pagar por las distribuciones del Linux, nadie osaría hacer entre ellas *zapping* o *hopping* (zigzagueo), probando primero ésta, luego la otra y así unas cuantas más —para, a la postre, quedarse con aquella que a uno más le plazca (o incluso abandonarlas todas, regresando a la servidumbre de Micro$oft o a la de Apple.)

Que ciertos magnates contribuyan a la financiación de la *Linux Foundation* en su propio beneficio no es óbice para que entre la Fundación y nosotros, los usuarios, la relación no sea mercantil ni lucrativa. Recibimos su producto como una donación. El dinero acopiado por la Fundación ignoro cómo se emplee (aunque en su portal *web* pregona sus benéficas actuaciones).[23]

---

[23]. Si he de creer lo que he leído en una página *web*, del presupuesto de la *Linux Foundation* sólo un insignificante 2% se destina al desarrollo del kernel Linux, yéndose el grueso de la inversión en programas de IA y de mejora de las redes de nube. No habiendo podido ni confirmnar ni falsar esa información, no estoy en condiciones de apreciar ni su veracidad ni su veridicidad . Mas, de ser así, resultaría engañosa la

Sé, cierto, que vienen profesionalmente retribuidos por su trabajo algunos miembros del equipo que, capitaneado por Linus Torvalds, entrega al público las sucesivas versiones del kernel del Linux —esa clave de bóveda sin la cual no existiría el sistema operativo (por mucho que los entusiastas del GNU insistan —no sin razón— en que tampoco habría sistema operativo si únicamente se tuviera el kernel).

Mas he leído que no pocos contribuidores regulares del equipo lo hacen a fuer de «voluntarios», o sea de benévolos, sin recibir ni un dólar por su colaboración.

A pesar de ese benevolado, está claro que no habría Unix/Linux sin programadores del máximo nivel bien remunerados, fuera del ámbito universitario. Siendo, pues, ilusoria la imagen de un *software* producido por investigadores desinteresados y ajenos al mundo del capital, eso —insisto y repito— no suprime la diferencia.

En la elaboración y difusión del Unix/Linux están involucradas grandes firmas con ánimo de lucro. No lo hacen por benevolencia. Así y todo, lo que sale de esas oficinas es un *software* no sólo gratuito, sino, además, de fuente abierta, lícitamente reutilizable, copiable, distribuible y modificable por los adquirentes —bajo ciertas condiciones nada onerosas para el común de los usuarios—, al paso que, no sólo es de pago el de Micro$oft —por más que hoy ya nadie se arruine para comprarlo—, sino que, además, resulta ilícito —y, si no infactible, sí arduo— copiarlo, compartirlo con familiares o amigos, replicarlo o manejarlo de alguna manera que no entre en los límites del uso expresamente autorizado por el dueño de tal sistema operativo.

Eso sin contar otras tribulaciones y hasta vejaciones que inflige Micro$oft a sus clientes y de las cuales luego hablaré, como la de tener que registrarse y solicitar la venia del monopolio para usar, en la propia máquina, en el propio hogar, el *software* que uno ha comprado legalmente con su dinero; un uso que hoy resulta imposible sin la expresa anuencia de Micro$oft, quien, a cambio de otorgárnosla, nos ficha y nos controla, convirtiendo así la venta en un mero arriendo bajo condiciones leoninas.

Según lo argumentaré luego con detalle, el Unix/Linux es infinitamente superior, más potente, más ágil, más fácil y mejor adaptado a las necesidades de muchos usuarios (entre ellos el autor de estas líneas); también es muchísimo más bello.

Pero, sobre todo, nos hace libres. No es que el sistema operativo en sí sea libre. Sólo puede ser libre quien posee entendimiento y voluntad, o sea capacidad de obrar (eso a lo que algunos filósofos se refieren con el feo

---

autodenominación de esa Fundación. Antes bien daría la impresión de que se se estaría cubriendo la Fundación con el prestigioso rótulo de «Linux», tan atractivo para millones de usuarios en todo el mundo, para financiar otras actividades que, no teniendo nada que ver con nuestro amado sistema operativo, son provechosas para los detentadores de esas billonarias fortunas.

anglicismo «agencia»); carece de tal capacidad un sistema operativo. Tampoco es libre un cuaderno ni un anaquel ni una silla, ni un poema. Es libre el usuario, es libre quien se sienta, consulta el cuaderno, toma un diccionario del anaquel o recita el poema.

Tener que pagar un precio es una limitación de nuestra libertad de usar el producto. Limitación frecuentemente asumida de grado por nosotros, los compradores, siempre que, siendo razonable el importe, percibamos una adecuación entre lo que perdemos pagándolo y aquello que ganamos usando el producto así comprado. Mas no cabe duda de que más libres somos con relación al agua de la fuente pública con la cual llenamos nuestro botijo, a la asistencia médica gratuitamente dispensada, al alumbrado de las calles, al mar en que nos bañamos, al aire que respiramos, a los podcasts que escuchamos sin haber tenido que pagar ni un euro por descargarlos.

La gratuidad es una libertad, al estribar en la posibilidad de actuar según nuestra voluntad sin someternos a la onerosa (y a veces aflictiva) condición de pagar por ello. (Un pago siempre equivale, en cierto modo, a una multa.)

Sabemos que no habría sociedad si fuéramos absolutamente libres en todo. En aras del bien común ha de venir cercenada nuestra ansia de libertad. De grado o por fuerza aceptamos muchas limitaciones a la libertad, conscientes de que, haciendo falta bastantes de ellas para el bien común —aun siendo arbitrarias otras de esas restricciones—, la sociedad sólo puede existir cuando incumbe a alguien la potestad de fijar tales limitaciones, mientras que la ausencia de autoridad, la anarquía, únicamente nos libraría de las limitaciones arbitrarias a cambio de suprimir los deberes de cuya vigencia nos beneficiamos (el pacto social).

En ese dilema, nos gusta la libertad. *Cæteris paribus* es mejor ser libre. Cuanto más libre sea uno, más disfruta de la vida. El ansia de libertad no es vicio alguno. Todos los animales aspiramos a ser libres, libérrimos, a poder ejercer nuestra voluntad sin trabas, sin condiciones, sin estar amenazados con consecuencias molestas por nuestra decisión.

La gratuidad es, por consiguiente, una libertad. Poder leer un libro (electrónico o en papel) sin tener que pagar una multa es leer libremente; significa, en cambio, no ser —en eso— libres tener que pagar la multa.[24]

Reitero que no estoy pidiendo que todo sea gratis. Ni en la sociedad actual neocapitalista ni en otra alternativa sin propiedad privada y con economía planificada. Lo único que digo es que, en la medida en que hayamos de pagar por el uso o disfrute de un bien o de un servicio, en eso no somos libres; en eso cercénase nuestra libertad.

Si nos impusieran una tasa por acceder a la playa o por cruzar la calle

---

[24]. O un «canon» que vaya a engrosar las arcas de algún conglomerado, como en España la SGAE.

o por pasearnos en el jardín público, o por contemplar los fuegos artificiales (en suma por beneficiarnos de lo que los economistas llaman «bienes públicos»), veríamoslo, no sólo como una merma de nuestro bienestar, sino también de nuestra libertad. Cuando, por el contrario, ábrese libremente al público un parque anteriormente de pago, lo sentimos como un ensanchamiento de nuestra libertad.

Pues bien, con Unix/Linux somos libres principalmente porque no nos cuesta ni un euro. Siendo libres de ensayar y usar una distribución del Linux, igualmente lo somos de ensayar y usar cualesquiera otras, sin venir por ello castigados a pagar a sendos distribuidores.

Sospecho que, además, la libertad o gratuidad es una de las causas de la alta calidad y utilidad del producto. De algo han de vivir quienes consagran, total o parcialmente, su existencia a crear, no sólo el núcleo del Linux, sino también el *software* necesario para que, integrado con el kernel, resulte un genuino sistema operativo.

A todos ellos les deseo lo mejor. Mas agradezco que me dejen ser libre, usando su producto sin por ello venir castigado a pagar multa, léase precio. ¡Ojalá consigan un adecuado medio de vida sin arrancarnos un pago a los usuarios! En ese clima de libertad, florecen mil flores, habiendo emulación entre las distribuciones que, gracias a múltiples colaboraciones, atienden a gustos sumamente variados.

Muy distinto es lo que sucede con el producto de un establecimiento con afán de lucro, sobre todo cuando las condiciones oligopólicas del mercado desaniman de toda competencia creíble. Al productor monopolístico u oligopolístico únicamente le interesa ganar dinero, cuanto más mejor, imponiendo a los cautivos compradores el precio que le da la gana, sin aliciente alguno para esmerarse por la calidad y utilidad del producto —y todavía menos por satisfacer gustos y necesidades muy variables de unos a otros. Si acaso, contarán el boato y la fastuosidad.

En suma, apreciamos la libertad del Unix/Linux posiblemente, sobre todo, porque es *free Linux* igual que en las recepciones se ofrece a los invitados *free lemonade*. La mayoría de los linuxitas no modificamos las fuentes programativas del *software* ni producimos *software* derivado; podemos compartir mas no aspiramos a distribuir.

No se me oculta que de esa libertad únicamente disfrutamos gracias a que otros han sido libres de alterar y distribuir el sistema operativo (dos de las libertades preconizadas por Richard Stallman), brindándonos esa prodigiosa y riquísima gama de distribuciones. Por ello disto de menospreciar el valor de las libertades reclamadas por Stallman o de ignorar que es gracias a libertades de esa índole (u otras parecidas) como ha resultado posible el maravilloso mundo del Unix/Linux. Mas sin la gratuidad de poco servirían esas otras libertades. (Volveré sobre esto.)

Capítulo IV

Mi luenga adhesión al Slackware

Retorno, tras esta digresión, a 1998, aquella primavera en la cual yo, desechando finalmente el OS/2, abracé el Unix/Linux, al cual ya jamás he renunciado.

Coincidió, fortuitamente, ese viraje en mi vida con otro atinente a mi orientación profesional y académica, al iniciar los estudios jurídicos que me llevarán a emigrar, de mi precedente vocación (la de un metafísico y filósofo del lenguaje), a otra nueva, la de un filósofo del Derecho (o, más exactamente, un académico consagrado a la lógica y la filosofía jurídicas).

Tal vez el tiempo y el denuedo que, a lo largo de años sucesivos, he consagrado al Unix/Linux hubiera hecho mejor en dedicarlos exclusivamente al estudio jurídico y jusfilosófico, contentándome, en lo que atañe a la electrónica, con un uso mínimo y convencional.

¿Es así? Los desvelos y sinsabores que me ha deparado mi empeño ahondando en el manejo del utillaje electrónico e informático han venido compensados por la fruición de ir venciendo obstáculos, resolviendo satisfactoriamente dificultades a costa de intensos y dolorosos afanes, con ahínco y tesón. (Es como en la práctica de la bicicleta. Sufre el ciclista escalando las rampas del puerto, mas sólo así siente después la hilaridad del descenso.)

¿Puede catalogárseme como un *geek* o como un *nerd*? No creo merecer tales honores. Sin embargo, entiendo que, comparada con la actitud de la mayoría de los usuarios, mi relación guarda un cierto parecido con la de esos *nerds* o *geeks* —quienes, si bien gozan de la admiración o la envidia de algunos, tienden a provocar burla o condescendencia de los más.

Entre abril de 1998 y enero de 2024 he sido un fiel y constante usuario de Slackware-Linux. (Había fracasado mi breve intento, en los primeros momentos, de instalar la ya entonces mucho más reputada distribución Debian. Frecuentemente lo presuntamente difícil se me hace a mí fácil y viceversa.)

Fui pasando, una tras otra, por las sucesivas versiones de Slackware. Durante los primeros años únicamente la usé en modo texto, rehuyendo el

entorno gráfico (el X-Windows, reemplazado después por X11 o Xorg), que por entonces resultaba difícil de configurar. (El propio usuario había de decirle al sistema operativo cuáles eran las características de la pantalla, como su resolución y su tasa de refresco, teniendo, para ello, que haber estudiado el manual del monitor; aun tomando las máximas precauciones, era probable y frecuente fracasar con ese entorno gráfico.)

Además, confieso que los entornos gráficos no me atraían nada, pero nada. Justamente mi disidencia con relación a lo que comúnmente se usaba había empezado con mi rechazo del Windows 3.0 ó 3.1, en los primeros años noventa, por mi apego al texto.

El entorno texto no era ni es únicamente el del CLI (*command line interpreter*), llamado también «terminal», «consola», «prompt» o «inductor», «línea de mandos» (o «de órdenes» —o «de comandos» para los amantes del barbarismo). También estaban los TUI (interfaces de usuario en modo texto), de las cuales voy a hablar en seguida.

Yo me sentía muy a gusto ante el CLI. Pasé en vela la noche que siguió a la llegada a mi casa de mi primer computador, «Ofelío», el martes 7 de febrero de 1989, consagrando esas horas de silencio a estudiar el manual del sistema operativo MS-DOS —en la versión 3.3, si mal no recuerdo, que es la que venía, a fuer de OEM (*original enterprise management*), vendida junto al *hardware* por el fabricante del mismo (en mi caso, Compaq).

Desde ese día fui, modestamente, un usuario experto o avanzado. Jamás me quedé sin saber qué hacer ante el *prompt*. Había aprendido las órdenes y pronto comencé a escribir mis propios *scripts* o *batches*.

En seguida salieron otros entornos en modo texto, los ya mencionados TUI (*Text User Interface*). De esas interfaces textuales puedo citar las utilidades Norton —que yo no tuve ocasión de conocer ni de usar— y el PC-Tools, que manejé en varias versiones sucesivas (la última de ellas, el *PC-Tools DeLuxe*). Eran entornos con menúes, donde se marcaban las teclas de atajo con colores vivos y resaltados. (Mi primer monitor había sido monocromo; adquirir uno de color, un año después, constituirá un enorme paso adelante.)

El propio programa de tratamiento de texto, el WordPerfect, ofrecía un TUI (un gestor de ficheros integrado). Resultábame fácil memorizar las pulsaciones de teclas de atajo.

No me apartaron del entorno texto ni del CLI mis años de OS/2 (de 1993 a 1998), pues justamente una de las ventajas del sistema de la IBM respecto al naciente Windows era posibilitar la multitarea evitando tener que usar el entorno gráfico. (El OS/2 ofrecía también un entorno gráfico-ratonero, mas no lo imponía; resultaba factible abrir simultáneamente varias consolas o terminales, corriendo en ellas sendas tareas a la vez, sin tener que atravesar el entorno gráfico, prescindiendo totalmente de él.)

Resultábanme a mí repelentes los entornos gráficos por dos motivos. El

primero era el obligado uso del ratón, un utensilio sumamente incómodo, antojadizo, impredecible, errático, indócil, fácilmente desviado y de pésima puntería;[25] además, mientras tengo ambas manos sobre el teclado, pertúrbame desplazar la derecha para —tras luengo forcejeo— atinar con los botonzuelos del ratón.

El segundo motivo es la sensible disminución de visibilidad: los entornos gráficos abandonan la vistosidad del color (hasta tal punto que a quienes están habituados a esos entornos parécenles agresivos o chillones los programas con menúes en modo texto, TUIs [*text user interface*]). El entorno gráfico constriñe al usuario a habituarse a unos signajos o garabatejos dizque «intuitivos», con unos menúes en gris sobre gris (o celeste sobre azul u otra combinación similar), con letruja menudica y un fondo de pantalla cada vez más deslumbrante (como si el sistema operativo estuviera concebido para que el usuario se pasara, embelesado, las horas muertas contemplando ese cuadro, sin realizar tarea alguna.)

Nunca me había gustado eso; sigue sin gustarme. Hoy lo he tenido que aceptar, a regañadientes, para algunas aplicaciones; aquellas que puedo, en cambio, sigo usándolas en modo texto; en general paso ante el computador el 90% del tiempo con programas en modo texto; y, además, para aquellos programas que irremediablemente tengo que correr en modo gráfico acudo a la pantalla completa, sin ventanas.

Salvo el aprendizaje del vocabulario y de la sintaxis del Unix/Linux, en los primeros años de mi adopción del Slackware-Linux, a partir de 1998, mi uso del computador siguió siendo parecido al que había tenido con el OS/2 y con el DOS. Lo más importante e imprescindible para mí fue instalar y hacer funcionar correctamente el Dosemu.

---

[25]. Ningún ratón parece satisfecho con su alfombrilla, exigiendo otra mayor. Atribuyéndole un imaginario ánimo, se complacería, retozonamente, en jugar con nosotros al escondite, ocultando su puntero en el rincón más inesperado de la pantalla, de donde, para sacarlo, hay que ladear el teclado, desplazando el ratón (y su alfombrilla) por toda nuestra mesa de trabajo. Tan indócil y perezoso es el punterillo del roedor que hemos de apretar, repitiendo varias veces los esfuerzos para llevarlo donde nos hace falta. ¡No digamos cuando lo necesitamos en esas aspas menudicas, donde, para cerrar la ventana o la aplicación, hay que picar habiendo atinado con absoluta exactitud!

Y no hablo ya de la frustración que sentimos cuando un movimiento del ratón nos saca —inadvertidamente y contra nuestra voluntad— del escritorio activado, llevándonos a otro donde, de momento, nada queríamos hacer.

Pasan los años y los decenios, mas los ratones siguen siendo caprichosos, torpes y díscolos. ¡Qué estorbo! ¡Qué trasto! (Además, son todos igual de indómitos: los cableados y los inalámbricos.)

¡Cuán fáciles y rápidas resultarían esas operaciones prescindiendo del ratón, si fuera posible, mediante la pulsación de teclas!

Capítulo V

La significación para mi trabajo del Dosemu y del MC

Pese a su nombre, el Dosemu no es —según los expertos— exactamente un emulador del sistema operativo DOS, sino de una capa de *software* que permite correr ejecutables del DOS, sin pasar por lo que técnicamente se califique de emulador. Careciendo de formación técnica, no puedo ni siquiera entender esa sutileza; menos aún expresar sobre tal asunto opinión alguna.

El Dosemu era y es indispensable para mí,[26] pues desde 1989 hasta hoy el principal uso mío del computador es teclear y elaborar documentos de texto, todo lo cual siempre lo he hecho —y sigo haciéndolo— con el WordPerfect. (De febrero a abril de 1989, el WordPerfect 4.2; de abril de 1989 a diciembre del año siguiente, el 5.0; desde diciembre de 1990 hasta hoy, el WordPerfect 5.1.)

Además, durante años seguí usando varias utilidades del DOS a las cuales estaba habituado (entre otras el Back-and-Forth), dentro de las sesiones emulatorias del Dosemu. No figurando éste entre el *software* incorporado a la distribución Slackware, tuve que hallarlo y configurarlo denodadamente. (Entonces, estando en pañales el internet, escasa información suministraban los motores de busca, como Altavista, hoy fenecida.)

Durante la mayor parte del tiempo transcurrido desde 1998, mi navegador usual ha sido el Lynx, en modo texto. (Hoy ha dejado de serlo, pues las páginas *web* están tan orientadas al gráfico que poco legible resulta su información si uno se empeña en acceder con el lynx u otro navegador en modo texto).

Mi programa de correo electrónico bajo el Unix/Linux siempre ha sido y sigue siendo el Mutt, asimismo en modo texto.

Desde el comienzo de mi uso del Unix/Linux estuve empleando (y sigo haciéndolo hoy) un gestor de ficheros en modo texto, el MC (*Midnight Commander*) —el cual, a lo que parece, es una adaptación de las utilidades Norton para DOS. Con el MC realizo las nueve siguientes tareas:

1ª) Visualizar ficheros de texto (y hoy también ficheros PDF, gráficos, .html,

---

26. V. http://lorenzopena.es/dos/dosemu.htm

.doc e incluso binarios).

2ª) Conectarme, como cliente FTP, con nodos de mi red hogareña o con servidores foráneos de FTP.

3ª) Editar ficheros de texto (.txt o .html, junto con scripts del *shell* propio del inductor en el nuevo sistema operativo).

4ª) Copiar o mover ficheros y directorios del panel de la izquierda al de la derecha o viceversa.

5ª) Borrar o renombrar ficheros y directorios.

6ª) Comprimir ficheros y descomprimir archivos —o bien examinar sus contenidos—, siendo un archivo un embalaje comprimido, v.g.un .zip o un .tar.gz.

7ª) Comparar los contenidos del directorio del panel izquierdo con los del panel derecho.

8ª) Realizar respaldos y restauraciones (siendo también posible conectarse en el MC con otros nodos o dispositivos mediante los tres protocolos alternativos Bluetooth, Samba y NFS).

9ª) Llevar a cabo muchas otras tareas avanzadas cuya enumeración omito para no sobrecargar esta exposición.

Es una pena que sea juzgada hoy *deprecated* tan utilísima y cómoda aplicación (fácil de configurar —o «costumbrizar»— según las preferencias de cada usuario), no figurando ya en el *software* instalado directamente por casi ninguna distribución del sistema operativo Unix/Linux. Cierto, podemos buscarla e instalarla (yo lo hice desde el primer día en que tuve que acudir a nuevas distribuciones del sistema operativo —al no poder ya seguir con mi viejo compañero, el Slackware). Desazóname imaginar que algún día los repositorios de esas distribuciones dejen de albergar al MC, por ser viejo, caduco, pasado de moda; un TUI, no un GUI, que es lo que se lleva ahora. De hecho en algunas distribuciones instalar el MC ya resulta arduo.

Capítulo VI

Llega el entorno gráfico

No había venido, pues, sustancialmente alterado mi manejo del computador por haber saltado en 1998 del OS/2 al Unix/Linux. Pasé de la más sencilla sintaxis del DOS-OS/2 a otra más compleja, la del Unix, que memoricé e internalicé bien, igual que había aprendido la precedente.

Años después tuve que resignarme a instalar también el gestor de entorno gráfico X-Windows.

Paréceme oportuna aquí una digresión. Adelantándome al sucinto historial que trazaré más abajo sobre la génesis y el devenir de los computadores domésticos, mencionaré que en julio de 1970, en pleno auge de los minicomputadores (*vide infra*, capítulos XII y XIII), la compañía Xerox —enriquecida con su cuasi-monopolio en las fotocopiadoras— creó una filial dotada de amplia autonomía, la PARC (Palo Alto Research Center), encomendándole la tarea de desarrollar un *software* apropiado para que Xerox pudiera también hacer negocio en ese campo de la computación, al cual hasta entonces era ajena. (Al igual que el equipo Unix —al cual me referiré después—, el de Palo Alto estaba formado por jóvenes ansiosos de emanciparse de la tutela de los mayores, ideando, no sólo nuevos modos de hacer las cosas, sino nuevas cosas por hacer.)

Abundaron los inventos del PARC: el ethernet, la impresora láser, el lector electrónico, el sistema operativo Pilot y, principalmente, el entorno gráfico-ventanero, el célebre GUI, junto con su instrumento, el ratón.

Iban tales novedades en contra de cuanto se hacía entonces, durante la década de los 70, cuando se desarrollan los microcomputadores y posteriormente sus derivados los computadores hogareños o domésticos (de todo lo cual hablaré en el capítulo XIII).

Ni el Unix ni los nuevos sistemas operativos diseñados para los microcomputadores tenían prevista interacción gráfica alguna del usuario con la máquina; antes bien, heredaban del Multics la interacción exclusivamente en modo texto, desarrollando el CLI (*command line interface*, o sea el inductor o línea de instrucciones), al cual se agregaron los primeros TUIs (*text user interface*) —con menúes, ciertamente, pero en modo texto, siendo el teclado el único instrumento de transmisión de tales instrucciones a la máquina. (No

siendo del todo exacto este último aserto podemos, para mi propósito, prescindir de complicaciones.)

Desarrolló el PARC un minicomputador original, el XeroxAlto, con su propio sistema operativo, el primero en explotar el entorno gráfico y en ofrecer un tratamiento de texto en WYSIWYG. A pesar de la crisis económica de los 70, el equipo de Palo Alto produjo dos millares de esos computadores, interconectados por ethernet. Sólo que la firma Xerox, que no estaba por la labor, fue dando largas a su comercialización. No corrían buenos tiempos para el consumidor, costando cada uno de esos minicomputadores un mínimo de 32.000 dólares (o sea cerca de 140.000 de 2024.) Finalmente en 1981 —ya en pleno auge de los computadores domésticos— lanzará Xerox al mercado los minicomputadores STAR junto con su impresora láser, costando cada equipo 100.000 dólares

Gustosamente recibió ese grupo de Palo Alto visitas de dos jóvenes programadores ajenos a la firma, Steve Jobs y Bill Gates, quienes —sin pagar regalía alguna a Xerox— copiarán ese diseño gráfico. Mejor zahorí, adelantóse Steve Jobs en Apple, a comienzos de la siguiente década (los 80); hízolo Gates después, ya en los 90, con el Windows.

No resultaba fácil que se adaptara el Unix a esos entornos gráficos, poco acordes con su propio talante y con sus propósitos. Mas los tiempos cambian. A los Unixes no les quedaba ya más remedio que incorporar un entorno gráfico, con el ratón y la edición de texto en WYSIWYG. El producto se llamó «X-Windows», inventado en el MIT bajo patrocinio de la IBM, Hewlett-Packard, Sun y AT&T. Fue desarrollándose hasta llegar a la oncena versión, creada en 1987. Desde entonces se denominó «el X11».

Para ser útil ese sistema de ventanas había de articularse. Lo hará con dos implementaciones sucesivas: el XFree86 (creado en 1992) y después (en 2004) el X.org. (El número 86 parece referirse a los computadores domésticos entonces más avanzados, los 386.)

Desde muy pronto incorporaron ese entorno gráfico, el XFree86, varias distribuciones del Linux; mas, de preferirlo, podía el usuario ejecutar la distribución permaneciendo en modo texto, suficiente para la multitarea, al resultar factible abrir simultáneamente varias consolas (o terminales). Lo peor era, además, que —según ya lo he recordado— resultaba arduo y engorroso configurar el entorno gráfico. El usuario había de facilitar todos los datos atinentes al monitor, al teclado, al ratón y a las impresoras; datos que solía desconocer.

* * *

Cierro aquí esta digresión histórica para retomar mi personal relación con el Unix/Linux desde finales del pasado siglo.

Por las dos razones ya evocadas, repugnábanme los entornos gráficos; tardé años en activar el X11. Cuando lo hice, procuré siempre acudir a los

interfaces más simples, aquellos que menos se alejaran del modo texto. Es más, mientras pude mi preferido interfaz para el entorno gráfico fue el *Ratpoison*, una aplicación que permitía (con limitaciones) manejar los programas de modo gráfico sin usar para nada el ratón. (Llegó empero el momento en el cual había dejado eso de resultarme posible, no quedándome más remedio que usar el molesto roedor, con el cual nunca me he familiarizado).

La principal diferencia para mí entre antes y después de abril de 1998 no fue la instalación de nuevas versiones del sistema operativo. Ya lo había hecho antes, habiendo adoptado primero el MS-DOS 4, luego el 5, luego el PCDOS de la IBM, luego el DR-DOS, finalmente las sucesivas versiones del OS/2. Sólo que ahora pasar de cada versión del Slackware a la siguiente implicaba, por mi parte, un esfuerzo y una iniciativa mayores. Empecé, además, a compilar yo mismo ejecutables y librerías e incluso el kernel y hasta el propio compilador. Sin ser los únicos, sí eran SourceForge y FreshMeat los repositorios que más frecuentaba (casi a diario, salvo algunos períodos de impedimento o de abstinencia).

Todo eso fue paulatinamente dejando de resultar factible después de 2007/08. La última vez que compilé con éxito un kernel fue en 2014. ¿Qué sucedía?

Resúltame imposible saber en qué medida los cambios que voy a rememorar ahora hayan venido determinados por la vertiginosa evolución del *hardware*, cargada de serias consecuencias. Dudo hasta qué punto lo que tuvo lugar fue la adopción de una nueva cultura informática por el equipo encabezado por Linus Torvalds, quizá en parte como adaptación a esa evolución del *hardware*.

De resultas de ello, mutó el propio Slackware. Es posible que resultara cada vez más fácil y rápido de instalar, pero también, una vez instalado, más difícil de alterar, remodelar, reconfigurar o retorcer al gusto del usuario. Había que conformarse con el kernel originario que venía con la distribución. (Hasta entonces yo me había librado de esos inflados núcleos «huge», reconfigurando siempre un kernel mucho más esbelto, quitando soporte a recursos que yo no tenía o no necesitaba y, en la medida de lo posible, extrayendo del propio kernel el soporte a otros recursos como módulos; de ese modo el arranque de la máquina era más veloz y con menor riesgo de abortar).

A partir de 2010 o así, fui dejando de acudir a FreshMeat y SourceForge (igual que a otro sitio asociado a ambos, SlashDot, que era una especie de red social, preferentemente para aficionados al Linux o a otros sistemas alternativos).

Fue en octubre de 2014 cuando, por última vez, instalé con éxito una nueva versión del sistema operativo Slackware-Linux, la 14.1. Sin embargo, no me bastaba en absoluto disponer del *software* incorporado a esa distribución, por lo cual hube de acudir a unas cuantas fuentes ajenas para procurarme lo que más necesitaba: el GIMP (programa de diseño gráfico); el LibreOffice (que

es lo que su propio nombre indica, un completísimo *Office*); el Dosemu; el Glabels (programa de cartelería anexo al gestor de entorno gráfico Gnome, descartado por Slackware a favor de su rival KDE); el Cxoffice (un emulador de Windows que yo había comprado y que mejoraba sensiblemente al Wine, el cual, de todos modos, tampoco figuraba en el elenco de aplicaciones integradas en la distribución de Slackware); el programa de correo Mutt; el navegador Chrome de Google; el Adobe-Reader; y muchísimas otras utilidades principalmente orientadas a la producción y manipulación de documentos de varios formatos, sobre todo PDF.

Al comprar un nuevo computador —el viernes 15 de enero de 2016—, me esforcé por instalar una versión más moderna del Slackware. No habiéndolo conseguido, ensayé un montón de distribuciones alternativas. No tuve éxito, retornando al Slackware 14.1. Tampoco fueron felices mis varios intentos posteriores de actualizar la misma distribución o de reemplazarla por otras. ¿Me faltó paciencia? Yo quería una distribución donde pudiera pronto horadar la corteza (o el caparazón) de la interfaz gráfica (además de preferir aquella que fuera menos sobrecargada y más liviana), buscando mis programas de siempre: Mutt, MC, Dosemu, Lynx, Glabels, un emulador de Windows que funcionara (como el Cxoffice, que ya resultaba incompatible con las nuevas distribuciones de 64 bits) y el VirtualBox de Oracle, que se me fue haciendo imprescindible).

Capítulo VII

2024: Adopción del Ubuntu

Así he continuado durante ocho años, hasta que, finalmente, mis necesidades me han llevado en 2024 a reemplazar el computador comprado en 2016 por tres nano-PCs: el uno como NAS doméstico (nube casera) y los otros dos para ubicarlos en sendos lugares alternativos de trabajo.

El precio conjunto de los tres ha sido de 684 €. Como comparación: mi ahora desechado —o arrinconado— computador de enero de 2016 me había costado 379'50 €; pude entonces adquirirlo sin sistema operativo (gracias a haberlo comprado en una tienda de barrio que había accedido a ello); de haber aceptado el Windows preinstalado (Windows 10 Home 64 bits), OEM, habríame costado cien euros más, regalados a Micro$oft sin provecho alguno para mí (que habría perdido, sumando el IVA, 121 €). (Por otro lado, el único computador portátil que he tenido —comprado en mayo de 2014 y que regalé a un conocido en diciembre de 2017— me había costado 699 €; de ellos supongo que unos 150 se los debió de llevar la firma Micro$oft por un sistema operativo, el Windows 8, que a mí jamás me sirvió absolutamente para nada —causándome tan sólo un suplicio hasta que lo borré a los nueve días de la compra.)

Sé que hay dónde adquirir computadores sin sistema operativo, pero encontrarlos me resulta difícil o imposible, pues, por circunstancias de salud, he de conformarme hoy con aquello que puedo comprar a través de internet.

Mi calamitosa y exasperante experiencia con los tres preinstalados Windows-11 la narraré más abajo. Sólo me dejó hastío, frustración, hartura e irritación. Sensaciones que no duraron, pues en seguida borré del todo ese sistema operativo preinstalado para dedicar íntegramente al Unix/Linux el almacenamiento interno de los tres nanocomputadores.

¿Cómo así dije adiós al Slackware? Por una razón. Aunque la instalación de la última versión estable de Slackware se realizaba bien, sin embargo fracasaba a la hora de arrancar desde el almacenamiento interno.

Hoy ya sé por qué. El gestor de arranque de Slackware siempre ha sido el Lilo (*Linux Loader*), adaptado a los discos configurados con una tabla de particiones DOS y cuyas particiones tenían un MBR (*master boot record*), en el cual de colocaba ese gestor para que desde él buteara el kernel.

En el moderno *hardware* ha cambiado el *firmware* de los computadores. Ya no es BIOS (*Basic Input Output System*), sino EFI (*Extended Firmware Interface*) —o, más recientemente, UEFI (*Unified Extended Firmware Interface*). A esa modificación le es correlativa otra (no podría yo decir hasta qué punto sean inseparables; no dejan de estar asociadas): los discos ya no tienen el tipo DOS de tabla de particiones, sino la GPT (que significa «*Guid Partition table*», al paso que «GUID» significa «*Globally universal identifier*»). Las nuevas particiones (las de discos GPT) ya no tienen MBR. El disco tiene, en cambio, una pequeña partición EFI (luego hablaré de eso con un poco más de detalle).

Para esos discos no sirve el Lilo, sino otro gestor de arranque, el GRUB (ahora sustituido por el Grub2, que sigue llamándose «Grub»). Slackware nunca adoptó el Grub, el gestor de arranque que se ha ido generalizando entre la mayoría de las distribuciones. Yo mismo confieso sentir desgana al Grub, por varias razones más abajo evocadas. Al no poder aferrarse al Lilo (para mí de tan gratísimos recuerdos, que funcionaba tan bien, siendo elegante, amigable y facilísimo de configurar), Slackware lo ha reemplazado por ELilo, (EFI Lilo), presuntamente apropiado para los discos GPT en computadores con UEFI.

Sólo que, en su instalación, no crea Slackware la partición EFI. Por lo cual, si bien, al concluir la instalación, aconseje instalar el Elilo en vez del Lilo, no se produce tal instalación; por lo tanto, el sistema, al rebutearse, no arranca. (De todos modos, el programa Elilo dejó de mantenerse hace ya dos lustros —siendo, pues, de temer que, aun en el supuesto de que se use bien, no funcione correctamente o según lo esperado.)

En el propio portal de Slackware aconséjase instalar el GRUB, reconociendo empero que el usuario ha de hacerlo por su iniciativa y sus medios:

> Set up Grub as boot loader on UEFI *hardware*.— Installation and running of Slackware on computers that use Unified Extensible Firmware Interface (UEFI) rather than traditional BIOS poses problems. It may be impossible to boot the official install media via UEFI directly and the installation discs do not include a UEFI bootloader.[27]

Lamentablemente, además de ser confusas las indicaciones suministradas en esa página, nada de todo eso se avisa al usuario cuando está instalando una nueva versión de Slackware (o cuando está instalando esa —por lo demás magnífica— distribución en una nueva máquina)

Tras ese fracaso con Slackware, dadas las peculiaridades de los discos GPT y el comportamiento un tanto tiquis-miquis del entorno UEFI, comprendo cómo actúa la instalación del Lubuntu, intolerante con que el usuario haya determinado por sí mismo qué particiones quiere hacer en su disco duro, fijando la capacidad de cada una. No se dejó instalar así el Lubuntu, abortando

---

[27]. Viene descrita la instalación de Slackware en los sistemas UEFI en la pág[a] http://docs.slackware.com/howtos:slackware_admin:installing_on_uefi_hardware.

el proceso, el cual únicamente se completó con éxito cuando le hube dejado al programa de instalación ocupar todo el disco duro, decidiendo qué particiones haría. (E hizo esa partición EFI para después colocar en ella el gestor de arranque, GRUB.)

No es eso todo. Los fabricantes de computadores personales, para comercializar sus máquinas con Windows, están constreñidos a haber recibido previamente una licencia de Micro$oft que implica la activación de un dispositivo de «secure booting». Un computador con ese dispositivo activado no podrá instalar ningún otro sistema operativo —ni siquiera en una partición aparte—, a menos que el desarrollador de tal sistema haya pagado a Micro$oft un fielato que lo habilite para la instalación en esa máquina.

RedHat, Ubuntu y Fedora (entre otros), al parecer, han pagado tal fielato, mas no Slackware (una distribución económicamente casi en la ruina, cuya supervivencia resulta milagrosa; además dudo que Patrick Volkerding, su desarrollador, se preste a esa infame extorsión mafiosa; todavía queda dignidad en el mundo.)

Es desactivable el dispositivo de «secure booting» (yo ahora lo tengo desactivado); para desactivarlo el usuario ha de saber qué sea, porque, cuando abre el UEFI (o sea el viejo BIOS, para entendernos), asoman muchas opciones de configuración de significado desconocido y enigmático; conque no se decide uno a modificar opciones por defecto hasta haber averiguado, con un grado de certeza, de qué se trata —no sea que, por pasarse de atrevido, la máquina deje de funcionar.

Tras mi fracasada instalación de Slackware, ¡mis cálidos besos de adiós a esa para mí entrañable distribución, con la gratitud de mi corazón por 26 años de íntima convivencia!

De los tres computadores recién comprados que acabo de mencionar, uno de ellos lo he convertido en un NAS, una nube doméstica, habiendo instalado en él una de las variantes del Ubuntu, el Kubuntu.

Elegí Kubuntu a sabiendas de que el Ubuntu a secas usa como entorno gráfico (*desktop environment*) el Gnome, que yo ya había conocido por mis tres años de uso de la distribución FEDORA en aquel pequeño portátil o ultrabook de marca Lenovo, que anduve usando —auxiliarmente— entre 2014 y 2017, una máquina en la cual no había sido capaz de hacer funcionar el Slackware.

Nunca fue muy satisfactoria mi experiencia con el FEDORA ni con el Gnome. Con ellos siempre me sentí de visita. Hube de lamentar, entre otras cosas, no poder (o no saber), en ese entorno, instalar el Dosemu, habiendo de conformarme con un sucedáneo, el DOSBOX, que, además de impedirme ver mis documentos del WordPerfect en pantalla completa —habiendo de resignarme a una ventanuca—, bloqueaba, no sólo la impresión, sino también la ejecución de grandes macros. (Cabe la posibilidad de que causa de tales restricciones fuera, ora la reducida capacidad del propio ultrabook, ora mi insuficiente aprendizaje de las peculiaridades de esa distribución, a la cual ni

por lo más remoto consagré el empeño que había dedicado al Slackware.)

Gustábame poco el entorno gráfico Gnome; conque opté por Kubuntu, casi igual al Ubuntu, sólo que usando, en vez del engorroso y tumefacto entorno Gnome, otro entorno gráfico, el KDE, al cual estaba yo ya acostumbrado por haberlo utilizado el Slackware (si bien en Slackware había acabado reemplazándolo por otro mucho más sencillo, el XFCE).

Destinando a la función de servidor ese aparato, intenté al principio (tras haber estrepitosamente fracasado en mi intento de dejarlo con el Windows-11, a ver si así conseguía que hiciera la función de servidor), instalar en él alguna distribución de las especialmente diseñadas para la función de servidor: Ubuntu-server, Linux-Alpine y Proxmox. (Carecen de entorno gráfico.) No me fe muy bien con ninguna de las tres. El Alpine y el Proxmox me resultaban alejados de todo aquello a lo que estoy familiarizado.[28]

Por otro lado, todas las versiones actuales de cualquier distribución del Linux —y de otros sistemas operativos, como el Windows—, en cuanto detectan la resolución máxima del monitor 2560×1600, a ella se aferran, con lo cual, por lo diminuto de la letra, hácense ilegibles todos los mensajes de texto. En el entorno gráfico es fácil corregir y reprimir esa malsana tendencia, imponiendo una resolución menor, 1920×1200. Mas sólo posteriormente he hallado un truco para agrandar la letra en el CLI (consola, terminal en modo texto) —truco que, sin embargo, no funciona en todas las distribuciones.

Pese a mi preferencia por el CLI, o sea por el entorno de texto, resulta difícilmente sustituible el entorno gráfico en las modernas distribuciones, así sea para abrir en él terminales o consolas, que emulan el CLI, en las cuales realizo unos cuatro quintos de mi trabajo.

He optado por el Lubuntu en los dos computadores que he destinado a ser mis nuevos instrumentos de trabajo. ¿Por qué Lubuntu —y no Ubuntu, Kubuntu, Xubuntu, Wubuntu o cualquier otra de las muchas variantes de Ubuntu? Porque la distribución Lubuntu viene con la interfaz de escritorio (*desktop environment*) LXQT, la más estilizada y esbelta de todas, exenta de las horteradas que inflan a otras. A ella viene asociado el gestor de ventanas OpenBox, que me resulta facilísimo de implementar a mi gusto y de conformidad con mis necesidades.

* * *

¿En qué consiste ese trabajo mío? Mis tareas cotidianas son éstas:

— Teclear (principalmente en el WordPerfect 5.1 a través del Dosemu y recurriendo a mis cientos de macros).

— Revisar e incrementar mis acervos de datos.

---

[28]. Según veremos más abajo, pese a su nombre, Alpine Linux no es Linux. ¿Lo es Proxmox? No lo sé. No lo parece mucho, ¡la verdad!

— Editar documentos.

— Elaborar mis páginas *web* y colgarlas en mis portales[29] (incluyendo entre ellas documentos en audio y multimedia,[30] que, sin embargo, no produzco con el computador sino con una tableta Android).[31]

— Imprimir.

— Convertir textos de un formato a otro (para lo cual suelo utilizar el LibreOffice).[32]

— Conectarme (ya sea por Bluetooth, ya sea por alguno de los tres protocolos telnet, ftp o samba) con mis tabletas android y con otros dispositivos de mi red doméstica (entre ellos con el NAS hogareño), para transferir ficheros en uno u otro sentido.

— Visualizar, leer y modificar PDFs (para lo cual dispongo de un abanico de programas).

— Producir, con el Glabels, carteles, láminas y carátulas (para mis propias publicaciones electrónicas o para anunciar determinados eventos).

— Visualizar gráficos (y también ocasionalmente convertirlos de un formato a otro así como, a veces, corregirlos con el GIMP).

— Acceder al internet —para hojear la prensa digital y ciertas páginas *web* seleccionadas, echando algún vistazo a FB u otras redes sociales así como, esporádicamente, introducir en ellas alguna aportación.

— Consultar el correo electrónico, recibiendo, leyendo y enviando mensajes.

— Comunicarme por telegram con varios interlocutores que también han escogido ese vehículo.

— Gestionar ficheros, sincronizando directorios en varios recursos de almacenamiento.

— Realizar frecuentes respaldos de varias índoles.

— Mantener una colección actualizada de distribuciones debidamente grabadas en llaves USB para su eventual ensayo. (Últimamente me inclino más

---

29. Que son dos: http://jurid.net y http://eroj.org.

30. Y también en mi canal de YouTube (@LorenzoPena) —así como, de tarde en tarde, también en otro que tengo abierto en la red audiovisual Odysee.

31. Es heteróclita e irregular esta tarea (no ajustándose a periodicidad alguna), abarcando labores como el mantenimiento (muy variable según los tiempos) de mis tres bitácoras («El pueblo español», «Bitácora JuriLog» y «Sic et non») y de mi propio podcast («El bien público»).

32. Ciertas conversiones, no pudiendo hacerse con el LibreOffice, las realizo con el programa —que he comprado— WordPerfect 2021, el cual corre en una máquina virtual Windows10 bajo el VirtualBox.

por amontonar esos ficheros *.iso en una sola llave USB con el Ventoy, lo cual es un alivio.)

— Activar nuevas aplicaciones, actualizando las ya instaladas.

— Almacenar, ordenar y clasificar (en directorios apropiados) podcasts —que normalmente transfiero, desde una tableta, al computador (sea por ftp o por samba)— para irlos paulatinamente grabando en llaves USB, a fin de, posteriormente, irlos escuchando (a través de una minicadena de sonido) mientras estoy realizando otras actividades.[33]

— Correr, en el VirtualBox, máquinas virtuales con otros sistemas operativos; tres de tales máquinas virtuales son sendas versiones de Windows (XP32, XP64 y 10), únicamente utilizadas para ejecutar programas nunca portados al Linux —ya sean antiguos o nuevamente comprados como el WordPerfect-2021—. Mencionaré entre los antiguos el OCR Textbridge más el M$-Word-97 —que exclusivamente me sirve para convertir algún documento *.doc al formato WordPerfect 5.1. (La conversión inversa la hago con el LibreOffice.)[34]

---

[33]. Hoy se han aliviado tareas pasadas relativas a la manipulación y la conversión de ficheros de audio y de vídeo; tareas que eran particularmente tesiosas cuando los ficheros audio no eran descargables (ni en formato .mp3 ni en ningún otro), sino escuchables en *streaming* (habiendo sido producidos en formato *real audio*, .rm); además entonces no existían aún los aparatos que hoy podemos utilizar para la reproducción de audios (sino únicamente reproductores de CDs portátiles).

Habiéndose aligerado tales tareas, no han desaparecido del todo, puesto que hay audios únicamente descargables en formatos no reproducibles en mis dispositivos externos, como, v.g., .m4a y .wma. Gracias a mis programas —bajo Linux, como el *ffmpeg*— los convierto al formato .mp3.

[34]. Una de mis máquinas virtuales, que corren bajo el programa de Oracle VirtualBox, contiene un sistema operativo Windows10 nunca «activado», haciendo caso omiso al aviso de Micro$oft que me impone la obligación de activarlo. Me lo he bajado del internet; siéntome sobradamente autorizado a usarlo, sin cometer pirateo alguno, ya que he sido —contra mi voluntad— constreñido a pagar tres veces el Windows11 en los últimos 300 días, sin poder servirme de él para nada, al resultarme inutilizable salvo, precisamente (y aun eso para fines limitadísimos —tendiendo a ser casi excepcionales o, como mínimo, ocasionales—) en el marco de una máquina virtual que corra bajo Linux.

Para escapar al acoso de Micro$oft —que me hostigaba con sus amenazas de preceptiva actualización, so pena de no poder seguir corriendo su sistema operativo—, no sólo he desconectado del internet esa máquina virtual, sino que, además, he conseguido cancelar la función de actualización automática. (Para mayor seguridad, en la configuración correspondiente he logrado pausar las actualizaciones hasta el viernes 31 de diciembre del año 9999.)

Nótese la diferencia entre, de un lado, la soberbia y avasalladora Micro$oft, que despóticamente impone a sus clientes (o, más bien, siervos) sus obligatorias actualizaciones (atosigándolos para que obedezcan), y, del otro lado, las empresas productoras de distribuciones de Linux —como, en mi caso, Canonical con el Ubuntu—, las cuales ofrecen sus actualizaciones, quedando en la libre voluntad del linuxita aceptarlas o rechazarlas.

— Visualizar algunos vídeos (generalmente de YouTube).[35]

— Grabar vídeos con mi *webcam* Nuroum V15-AF, mediante el programa Kamoso.[36]

En mi trabajo lo abrumadoramente mayoritario es el texto, la palabra escrita.

Dedúcese de esa lista (seguramente no exhaustiva) que yo no uso mis computadores de sobremesa para nada lúdico.[37] Mis PCs son instrumentos de trabajo, no de esparcimiento.

La manera de organizar mi actividad con el computador es la siguiente. Tras haber ocultado la imagen de fondo del escritorio LXQT y desactivado los iconos (dejando así un escritorio oscuro y desnudo), configuro el OpenBox de manera que queden permanentemente abiertos doce escritorios (*desktops*), a cada uno de los cuales accedo con una tecla caliente (*hot-key*, un atajo): al escritorio nº 1 accédese con la pulsación <Meta>+F1; al nº 2, pulsando <Meta>+F2. Y así sucesivamente hasta el duodécimo.

Ábrense también trece aplicaciones, las más usuales, meramente pulsando sendos atajos, *hotkeys*: <Meta>+k para abrir la consola (*konsole*);[38] <Meta>+l para el LibreOffice; <Meta>+a para el Atril (un visualizador de PDFs); <Meta>+v para el navegador Vivaldi; <Meta>+b para el VirtualBox; <Meta>+g para el Glabels; <Meta>+p para el Gimp; <Meta>+w para el visualizador de gráficos Gwenview; <Meta>+o para el Okular; <Meta>+h para el navegador Chrome. (En el futuro —y al compás de mis necesidades— podré agregar otras *hotkeys* adicionales.)

Del WordPerfect 5.1 tengo tres clones, que se abren, respectivamente,

---

35. Para poder escuchar su audio he instalado —a la salida de auriculares de mi computador— un minialtavoz provisto de clavija jack.

Gracias a tal accesorio puedo también escuchar ficheros audio (de música u otros); confieso, empero, que tales audiciones me resulta hoy más cómodo y agradable hacerlas con mi tableta android —reclinado por la tarde en el diván; incluso cuando estoy trabajando con el computador, la música de fondo la prefiero proveniente de mi tableta.

36. Posiblemente a causa de mi desconocimiento —agravado por un insuficiente estudio de cómo solventar tales dificultades— han fracasado mis intentos de grabación audiovisual con otros programas más generalmente utilizados y de mayor reputación. (Puede que también sea de baja calidad mi *webcam*.) Sea por lo que fuere, me resulta más fácil y cómodo hacer las grabaciones de vídeos con la tableta en vez de con el computador. (No he probado esas u otras aplicaciones en máquinas virtuales, donde podría haber ejecutado versiones más actualizadas de Lubuntu —y, por lo tanto, también del *software* de grabación. Acaso lo haga en el futuro.)

37. Confieso no haber jamás practicado juego electrónico alguno ni sentido por ese pasatiempo el menor atractivo.

38. En Lubuntu existen varios (emuladores de) terminales. Para mí, por su sobriedad y configurabilidad, el mejor es *konsole* —que no viene en la distribución por defecto—. Emula perfectamente el terminal de toda la vida, como si no estuviéramos en modo gráfico.

con <Meta>+1, <Meta>+2 y <Meta>+3.

Existe una comunicación entre el entorno gráfico y la consola (lo cual engloba al gestor de ficheros MC [*Midnight Commander*], ejecutado en la consola, lo mismo que el cliente de email, *mutt*). Un bloque de texto seleccionado (y copiado al *clipboard*) en un programa de entorno gráfico (como, v.g., uno de los navegadores Vivaldi y Chrome o el LibreOffice) puede recuperarse en un TUI —como el MC o el *mutt*—, pulsando la combinación de teclas <Ctl>+<Shift>+v.

No vale tal procedimiento para el WordPerfect, puesto que éste corre bajo un emulador del DOS (o sea en una máquina virtual); no obstante, si bien el WordPerfect corre, normalmente, en pantalla completa, pulsamos, para conmutar la pantalla a otro escritorio, la combinación <Ctl>+<Alt>+f. Intercambiamos bloques de texto entre dentro y fuera del WordPerfect por otros métodos.

Mi manera de trabajar es ésta. Tras haber encendido la máquina y buteado la partición usual de trabajo, aparece el escritorio número 1. De ahí paso a otro escritorio pulsando la tecla correspondiente; en él pulso el atajo de una aplicación (v.g. <Meta>+v para lanzar el navegador Vivaldi) o bien abro una consola (donde puedo ejecutar cualesquiera órdenes del CLI o abrir un programa de TUI, como el navegador *lynx*, el cliente de email *mutt* o el gestor de ficheros MC). Voy así ocupando unos cuantos escritorios (rara vez ocupo los doce a la vez).

Para el resto, cuento con el gestor de tareas que se abre en el escritorio principal, asimismo con una *hot-key*, la tecla <Menu>. Del aborrecido ratón sírvome únicamente cuando la aplicación no me deja otra alternativa; en pudiendo, prescindo totalmente de él.

Hoy mi uso del CLI es más modesto de lo que fue hará unos 15 años, porque se han simplificado varias tareas, requiriendo menor iniciativa mía.[39]

Para afrontar esas tareas había ido yo ideando y ejecutando complicados *scripts* que se encargaban de esas ejecuciones.

Muchísimas otras tareas asimismo requerían afán e inventiva, al paso que hoy resultan fáciles, viniéndonos las aplicaciones correspondientes ya configuradas y listas para ejecutarse.

La disminución del uso ha menguado mi familiarización con el CLI, porque la función hace al órgano; un órgano se va anquilosando y atrofiando a medida que va dejando de ser funcional.

Aun así, para no pocas tareas (en general sencillas), continúo acudiendo

---

[39]. No cesaron de golpe las operaciones más arriba referidas atinentes a la manipulación, conversión y almacenamiento de ficheros audio (inicialmente en *real audio*, *.rm) al llegar, hacia 2005, los primeros escuchadores portátiles de MP3 (todavía entonces de escasa capacidad y bastante caros).

al CLI; disto, pues, de haber olvidado todas las órdenes que forman parte del vocabulario del Unix.[40]

* * *

¿Ha sido un camino de rosas mi aprendizaje del Ubuntu? Contestaría afirmativamente, sin olvidar cuán erizados de espinas están los rosales. Me fue preciso acostumbrarme a un entorno que, si bien —en lo principal— es, en buena medida, similar a mi viejo compañero de tantos años, el Slackware, no deja de presentar sus peculiaridades, varias de ellas debidas a que hoy las cosas se hacen de otro modo, habiendo cambiado incluso el propio *hardware*.

Provocáronme varios desastres mi *hybris* y mi hábito de obrar a mi modo; descalabros que me han forzado, varias veces, a la desinstalación y subsiguiente reinstalación total. (Felizmente, había sido siempre lo suficientemente precavido, habiendo depositado a salvo cuanto después hube menester para —no sin trabajo— reconstruirlo todo, mejor que antes.)

Tras esos tropiezos, ahora ya funciona todo a las mil maravillas, mejor de como había funcionado en mis años de Slackware (salvo que hoy soy menos libre, viéndome sujeto a mayor disciplina).

Superado ese aprendizaje (que, lejos de resultar doloroso, ha sido, si no agradable, al menos estimulante), difícilmente podría ser más espléndida y satisfactoria mi experiencia con el Ubuntu, concretamente con el Lubuntu. Disfruto con él más incluso de lo que disfruté con el Slackware —habiendo, claro, pagado el precio de tener que habituarme a hacer unas cuantas cosas de otro modo, renunciando a mis arraigados hábitos y gozando de menor libertad.

¿Qué estimación tengo del Lubuntu —y en general de todo el Ubuntu? La de que es fantástico, hermoso, práctico, funcional, amigable, productivo, seguro, confiable, económico en recursos, rapidísimo y fácil de manejar (salvo cuando uno —como testarudamente tiende a hacerlo un servidor—, con su propio riesgo, se mete por torcidas veredas, empujado por su propio afán de hacer otras cosas o de hacerlas de otro modo).

---

[40]. Propóngome, no obstante, reactualizar mis conocimientos ampliándolos, al haberme percatado de que con ello se perfeccionaría mi manejo del CLI y, por ende, del computador.

Capítulo VIII

Distribuciones de Linux

Antes de probar el Lubuntu (y otros dos sabores del Ubuntu —existen varios más), había ensayado otras distribuciones.[41]

Si, a la postre, las he acabado desechando, ello ha podido deberse a motivos de gusto o a un ocasional tropiezo en el intento de instalación —o, más probablemente, a no haber hallado, de entrada, el *software* cotidianamente usado por mí, v.g. el Midnight Commander o el Dosemu (que después tampoco he encontrado en el Lubuntu, habiendo tenido yo que buscarlos e instalarlos). De haber perseverado, supongo que, en la mayoría de los casos, habría acabado venciendo esos obstáculos.

Mi opción por el Ubuntu no está, pues, racionalmente motivada, al no haber emanado de un cuidadoso examen comparativo de los pros y los contras de cada distribución. Mi elección ha sido un tanto casual.

¿Puedo o quiero afirmar que sea Lubuntu la mejor? Tan perentorio aserto ni por lo más remoto estoy epistémicamente autorizado a proferirlo, habiendo sido momentáneo e inconstante mi uso de otras distribuciones (las más veces apenas incipiente o meramente esbozado).

Únicamente he sido usuario asiduo de Slackware (más, auxiliarmente —durante pocos años—, de Fedora, según ya lo he recordado).

Desconozco si hay distribuciones mejores que Lubuntu. Sólo sé que ésta a mí me resulta utilísima y satisfactoria (aunque, desde luego, imperfecta), mientras que con otras he tropezado; o, en todo caso, me han resultado menos atractivas.

En el fondo todas son Unix/Linux. Las diferencias son, en su mayor parte, de presentación estilística. Las hay también de método implementativo, que para el usuario terminal suelen resultar poco relevantes (tendiendo a estar reservado su interés a estudiosos de la informática). Asimismo varía el elenco

---

41. He aquí una incompleta lista algunas de las que he usado o intentado usar. A salvo de verosímiles olvidos, son éstas: Slackware, Knoppix, Fedora, Debian, Mandrake, Mandriva, Manjaro, Mageia, Linux Mint, Mx Linux, Arch linux, EndeavourLinux, Suse, Salix, Slax, ProxMox, Kali, Ubuntu Server y Alpine (que, en realidad, no es Linux). De ahí paso, sucesivamente, a Kubuntu, Xubuntu y Lubuntu. Si cuento bien, son 22 al menos una vez intentadas.

de aplicaciones incorporadas, mas eso suele ser libremente modificable por el usuario, según sus preferencias.

Pienso que todas esas discrepancias juntas suman, digamos, un 15 o, a lo sumo, el 20% del uso y del contenido del sistema operativo. En su 80 u 85% todas las distribuciones son iguales. Linux.

Y ¿por qué, de entre los sabores de Ubuntu, opté por Lubuntu? No conociendo, hasta hacía poco, ni el Xubuntu ni el Lubuntu, había escogido, inicialmente, el Kubuntu; tras varios ensayos, me quedo con el entorno más sencillo, el Lubuntu.

Dicen que el Lubuntu es un Ubuntu de pobres, adecuado para computadores viejos o de escasa potencia. Posiblemente mis máquinas sean aparatos poco poderosos —aunque aquel que más habitualmente uso tiene 16 gigas de RAM. ¿Se desbordarían mis computadores por la hinchazón de los entornos gráficos más ostentatorios y espectaculares, como el Gnome y el KDE? No lo creo; los he podido ejecutar perfectamente (tanto Kubuntu cuanto Fedora-Gnome).

Lo que es verdad, en cambio, es que, a mi juicio, resulta irracional dedicar los recursos electrónicos a esas pompas y exuberancias ornamentales. Un lavaplatos sirve para lavar la vajilla. Un computador para trabajar.

Mi razón para elegir el Lubuntu, en vez de sus hermanos mayores, es, sencillamente, que para nada necesito esos lujosos entornos gráficos. Casi todo mi trabajo lo hago en modo texto (el email con el Mutt, el tecleo con el WordPerfect, la gestión de ficheros con el MC y no pocas tareas administrativas desde el propio CLI).

Es más, según ya lo he dicho, corro yo en pantalla completa —lo más parecido al CLI— incluso aquellos programas que han de ejecutarse forzosamente en un entorno gráfico (el Okular para visualizar PDFs —entre otras cosas—, el Glabels para producir carteles y láminas, el Vivaldi como navegador, el VirtualBox para abrir sesiones virtuales de otros sistemas operativos, etc).

No requiriendo esa invasión apabullante del gráfico, reconozco que, además, me causa desagrado. Hasta tal punto que incluso el Lubuntu, según viene, modifícolo yo, simplificándolo todavía más, al reemplazar su imagen de fondo de escritorio por una pantalla de color oscuro (marrón o azul marino), en la cual los iconos quedan totalmente invisibilizados. Nunca uso, pues, iconos para nada; el escritorio de Lubuntu ofrece una alternativa en modo texto: el menú desplegable pulsando Alt+F1.

¿Cómo así he dado con el Lubuntu de entre la multitud de distribuciones? El número 280 de 2024 de la revista digital *Linux-magazine* contiene un artículo de Bruce Byfield, por el cual me entero de los siguientes datos, basados en material recopilado por el portal *distrowatch.com*. En los primeros días de 2024, *DistroWatch* enumeraba 958 distribuciones de Linux,

de las cuales permanecían activas 274. Siendo muchas distribuciones derivativas de otras —al paso que algunas constituyen meras ocupaciones de un puñado de desarrolladores—, resulta normal que bastantes se extingan al poco, lo cual no es óbice a que, por otro lado, constantemente nazcan otras nuevas.

Aunque 47 distribuciones vienen calificadas de durmientes (acaso en hibernación, pudiéndose dar que alguna de ellas reviva), 637 están oficialmente clausuradas. (En 2014, estaban activas 285; en 2011, 323; la tendencia es, pues, ligeramente menguante.)

En este ámbito de las distribuciones del Linux existen hoy siete nuevas tendencias:

1ª. Las encapsulaciones (de las que más abajo hablaré).

2ª. La aparición de nuevos compiladores y nuevas utilidades, que nos desconciertan a quienes estábamos acostumbrados al Linux de un período anterior (un cambio cuyo ulterior ahondamiento me pregunto si correría el riesgo de que un día el Linux ya no fuera Unix).

3ª. El creciente uso —desplazando a nuestro habitual EXT4— de BTRFS y XFS, dos nuevos sistemas-de-ficheros (*file systems*), o sea modalidades de formateo de los dispositivos de almacenamiento (sean discos duros u otros).

4ª. El cambio que ya se ha producido del viejo sistema de arranque, heredado del Unix, el SysVinit, por uno nuevo nada amigable, dizque más eficaz y «moderno», el *systemd* (también me referiré, en el capítulo XXI, a ese cambio —que yo vigorosamente desapruebo).

5ª. El reemplazo del viejo entorno gráfico, el X11, por otro nuevo que va en auge, el Wayland (aún no generalizado pero que verosímilmente lo estará dentro de unos años).

6ª. La sustitución del viejo método de arranque o buteo, el LILO (modificado posteriormente como ELILO) por otro, a mi juicio frágil, abstruso, inamigable y dificilísimo de configurar, el GRUB —si bien, afortunadamente (según es normal en Linux) hay varias alternativas, una de las cuales, el rEFInd, la he adoptado yo, prescindiendo así del GRUB.

7ª. El desarrollo de versiones empotradas del Linux, o sea de las que se destinan, no a gobernar el funcionamiento de una máquina, sino a activarse, con alguna autonomía, modularmente, dentro de un computador que, en su conjunto, venga regido por otro sistema operativo (siendo ése el caso, v.g., del WSL, «*Windows subsystem for Linux*», que magnánimemente otorga Micro$oft, dentro de su sistema, para que los aficionados jueguen al Linux, siempre que la máquina continúe bajo el mando de la propia Micro$oft a través de su producto Windows).

Dista de gustarnos a todos esa intensa renovación por la cual atraviesa el Unix/Linux. Tiene el ser humano una innata tendencia a aferrarse a los hábitos, adaptándose frecuentemente mal, o a rastras, a los nuevos cambios. Alinéome yo entre quienes, prefiriendo la estabilidad, muéstranse a menudo reacios a las novedades.

Además de ser instintiva, fúndase en tres razones mi actitud conservadora.

— La primera es el temor a que lo que esté detrás de algunas de esas novedades sea cuestión más de moda que de adaptación a genuinos requerimientos.

— La segunda es que yo uso el computador para mi trabajo, mientras que las mutaciones frecuentemente no conllevan mejoras ni ventajas para ese trabajo, sino sólo un mayor aprovechamiento de los recursos electrónicos para fines que a mí no me conciernen ni afectan —v.g. los relacionados con los juegos, con lo audiovisual o con la seguridad.

— La tercera consiste precisamente en que, desde mi propio uso de los computadores y mi perspectiva, es infundada esa angustia hoy generalizada por la seguridad, puesto que las circunstancias de mi labor y de mi uso de la informática determinan un bajo nivel de peligro, ya sea de intrusiones ajenas o de contaminación de *software* maligno; de hecho, a lo largo de siete lustros no he sufrido nunca ni lo uno ni lo otro —del mismo modo que no ha sido jaqueada jamás ninguna de mis páginas *web*. (Es más, pienso que la obsesión por la seguridad puede causar más mal que bien; luego explicitaré esta idea.)

* * *

Les resultará extraña —y hasta incomodante— esa proliferación de distribuciones del Linux a quienes estén familiarizados con Micro$oft o con Apple. Pienso que hay en nuestro subconsciente una tendencia maltusiana que nos lleva, instintivamente, a preferir los desiertos a las selvas, lo estéril a lo frondoso. Muchos parecen estremecerse ante el hecho de que ya somos ocho miliardos de humanos en el Planeta; querrían que sólo fuéramos la mitad, o acaso la cuarta parte (aunque pocos sean quienes piensen que ellos mismos están entre los sobrantes). Como ya Pascal se mostraba contrariado por los millones de estrellas en el firmamento. ¡Demasiadas! ¿No sostuvo Fray Guillermo de Occam, OFM, que *entia non sunt multiplicanda præter necessitatem*?

Frente a esas ontologías de la miseria y la escasez, tenemos las más atractivas metafísicas de la abundancia, como la de Leibniz, que suscribe el autor de estas páginas; para Leibniz la realidad se rige por el principio de que todo posible tiende a existir, resultando de tal *conatus ad exsistendum*, como realidad, aquella que comporte el máximo composible de ser.

Felicítome yo de esa frondosa fecundidad del Unix/Linux, con la

resultante diversidad de distribuciones, que es una enorme riqueza. Mas hemos de ser conscientes de nuestras propias limitaciones. Igual que nadie puede ir probando qué tal se le daría la amistad con todos —para, a la postre, quedarse sólo con los mejores o con aquellos con quienes congenie más—, tampoco podemos querer probar todas las distribuciones del Linux, ni siquiera todas aquellas de las que más se habla en los círculos linuxitas.

La gran mayoría de las distribuciones de Linux yo ni siquiera las he ensayado.[42] Ni tampoco he probado el sistema operativo ChromeOS de Google (que usa el kernel de Linux, sin por ello ser Linux) ni otros UNIXes, como Solaris y los tres derivados gratuitos de BSD.[43]

Percátome de que contra el Unix/Linux objetarán los lectores acostumbrados a Windows que, al ignorar cuál distribución sea la mejor, motivo suficiente para no adoptar tal sistema operativo lo constituye esa misma pluralidad de distribuciones.

¿Qué valor conceder a ese argumento? ¡Ninguno! Argumentando así jamás compraríamos un par de zapatos, ya que desconocemos cuál sea la mejor zapatería o la mejor marca y no tenemos tiempo para irlas visitando todas, una por una. Ni adquiriríamos una modesta llave USB, al toparnos con una profusión de marcas sin tener garantía alguna de que nuestra opción vaya a ser la óptima. Ni adquiriríamos una vivienda ni haríamos nada en la vida.

¿Quiere Ud, amigo lector, probar el Unix/Linux? Marque ese nombre en un motor de busca.[44] ¡Eche a los dados los primeros seis que le salgan al paso!

Abriendo esa busca en Petal, en seguida asoman Linux Mint, Ubuntu, Debian, Red-Hat, Arch Linux y Fedora. Siendo seis, puede Ud hacer corresponder cada uno de ellos a una cara del cubo. ¡Que salga el que quiera la Fortuna!

Si me pidiera Ud mi consejo, sería el de optar por Ubuntu —ya sea el Ubuntu puro, ya sea, preferiblemente, alguno de sus otros sabores, como el Kubuntu. Primero, porque es una buena distribución —no exenta de defectos—.

---

42. Figuran entre las muchísimas distribuciones que yo nunca he ensayado las 57 siguientes: Absolute Linux, AlmaLinux, Alt-Linux, Antergos, AntiX, ArchBang, Artix, Asahi, Astra, BlackBox, CENTOS, CHAOS, Clear Linux, Corel Linux, CRUX, DamnSmall Linux, Devuan, Dragora GNU/Linux-Libre, Elementary OS, Feather Linux, Finnix, Foresight Linux, Fuduntu, Garuda, GeckoLinux, Gentoo, GNU Guix, GoboLinux, HandyLinux, Hyperbola GNU/Linux-libre, KaOS, KateOS, Korora, Mepis, Miracle Linux, Modicia OS, Netrunner, Nitrux, NixOS, Oracle Linux, Parabola GNU/Linux-libre, Platypux, Pop!Os, Porteus, Puppy Linux, PureOS, Q4OS, Qubs OS, RedHat, Rocky Linux, Solus, SolydXK, SparkyLinux, SteamOS, Trisquel GNU/Linux, VoidLinux y Zorin.

43. Ni, por cierto, se ha cruzado jamás en mi camino nada de Apple —ni en computadores ni en dispositivos móviles.

44. Duckduckgo, Yahoo, Ecosia, Qwant, Yandex, Lycos o cualquier otro.

Y segundo porque muchos usuarios de Linux lo son de Ubuntu o de alguna de sus variantes, con lo cual resultan altas las probabilidades de hallar al respecto una valiosa ayuda e información en foros, bitácoras y canales de Youtube o de Odysee.

Además, habiéndose derivado de Debian el propio Ubuntu (junto con todos sus sabores y sus propios derivados), también suelen ser pertinentes, para aclarar y solucionar problemas, los foros de usuarios del Debian (menos concurridos que los de Ubuntu, pero también a veces con opiniones más solventes).[45]

No es propósito de este ensayo pregonar ni catequizar a nadie. No necesita Linux prosélitos. Además, creo poco (o, mejor dicho, no creo nada) en la eficacia de las homilías y de los sermones.

Nadie va a pasarse de Windows al Unix/Linux por haber leído este trabajo. Mi consejo, según figura en los párrafos precedentes, no es el de adoptar el Linux sino el de cómo solventar las dudas en el supuesto de que Ud, amigo lector, haya concebido ya, independientemente, el propósito de hacerlo mas esté perplejo o titubeante ante la dificultad de la pluralidad de distribuciones.

Sólo que, siendo —como lo es— un consejo, al fin y al cabo, permítome agregar otro —de nuevo, en ese mismo supuesto—: cuando Ud adopte Linux, no mantenga en su disco duro el *dual booting* Windows/Linux.

A mí, que lo intenté, el *dual booting* no me funcionó, porque Windows se había adueñado de la partición de arranque NVME0N1p1, rehusando cederla o compartirla, de suerte que me resultaba imposible arrancar desde el almacenamiento interno del computador sin que lo que se abriera fuera el Windows. (Lo narro más abajo, en el capítulo XI.)

Verosímilmente podría vencerse esa resistencia con mayor pericia o acierto. Me temo, así y todo, que los gestores de arranque modernos son poco tolerantes para con la división del mismo dispositivo entre varias particiones de arranque. (Dirígese aquí también el reproche a Linux, cuyo actual gestor de buteo más generalmente usado, el GRUB, es susceptible de fallar en situaciones de compartición; por eso he acabado yo por prescindir de él, adoptando el rEFInd.)

Para compartir una misma máquina dos sistemas operativos es preferible añadir un dispositivo adicional de almacenamiento interno (NVME, hda, sda u otro), de suerte que se abra un sistema operativo distinto en cada uno de los dos dispositivos (entre los cuales se puede optar en el EFI [ex-BIOS]

---

45. Dicho lo cual, tampoco juzgaría yo desacertado optar por Zorin o por Linux Mint o por Deepin o por Pop!OS o por ElementaryOS —cinco distribuciones, al parecer, especialmente orientadas a quienes inmigran de Windows, resultando, en cambio, seguramente menos atractivas para quienes, como yo, no habiendo sido nunca usuarios del Windows, tendemos a rechazar los entornos gráfico-ratoneros.

o *firmware* de la máquina; de haber adoptado rEFInd como gestor de buteo, ese excelente programa nos servirá para hacer arrancar uno u otro de nuestros dos dispositivos internos).

A falta de ese adicional dispositivo de almacenamiento interno, juzgo prudente hacer una copia de salvaguardia de la partición Windows en un dispositivo externo (un disco o una llave USB) para luego crear, ya dentro de Linux, con el VirtualBox, una máquina virtual Windows donde se restablezca el contenido previamente salvaguardado. (Eso es complicado, no cabe duda, exigiendo una cierta destreza; mas se puede buscar ayuda por el internet, pareciéndome verosímil hallarla.)

Otra posibilidad es la siguiente. Aun habiendo eliminado Windows (10, 11 o el que sea) del disco duro de nuestro computador (u otro dispositivo de almacenamiento interno, como un NVME o eMMC), podemos descargarnos una imagen *.iso de tal sistema operativo del portal de Micro$oft.[46] Buscamos la imagen .iso que deseemos, de Windows10 o de Windows11, pudiendo escoger el idioma preferido. (Siempre que puedo, escojo yo, personalmente, para cualquier *software* el inglés, ya que las traducciones al español me resultan, todas ellas, erróneas y confundentes —empezando por esa equivocación de traducir «file» como «archivo».)

Una vez descargado ese fichero *.iso, podemos, con él, instalar, dentro del VirtualBox, una máquina virtual Windows. (Los detalles de cómo hacerlo los encontrará el lector en la *web*.) El VirtualBox, de la casa Oracle, está disponible para varios sistemas operativos, incluido Windows —de suerte que también algunos, o muchos, clientes de Micro$oft lo conocen y usan. (En Linux vengo usándolo yo desde hace muchos años, considerándolo un programa necesario.)[47]

Al instalarse esa máquina virtual, el Windows demandará una clave. Tal vez haya compradores afortunados a quienes, al venderles la máquina, les haya entregado el vendedor un DVD en cuyo sobre esté impresa su clave —o ésta quede manifestada de otro modo. Mis recientes compras de sendos computadores no vinieron acompañadas de tales dispositivos ni recuerdo haber visto clave alguna (aunque confieso que puede haberme causado inatención a tales detalles mi total falta de familiaridad con cuanto sea de Micro$oft).

A esa exigencia de clave puede el instalante replicar que activará el

---

46. microsoft.com/software-downlad/

47. Es posible que no basten las sucintas indicaciones del párrafo anterior; podrá hallar el lector mucho más expertas y detalladas explicaciones en varios vídeos de YouTube, bastando buscarlos introduciendo la ristra «windows 10 in virtual box»; entre otros, uno excelente es youtu.be/OWmD8obq4eQ?si=UrAP_wlJx744KuFz; otro —que yo no he mirado mas parece bien hecho— está en español y es de Andrés Manzano: youtu.be/jH1Fz0yXSuE?si=FWSkT5celpdhCpYz. Para instalarse Windows 11 en Virtual Box es preciso recurrir a ciertas astucias a fin de superar las zancadillas de Micro$oft; hállanse las explicaciones en estos dos vídeos: youtu.be/5TAGzPeqww8?si=D_Jc2WxC_7jSnjju y youtu.be/ifUJt1tqP_Q?si=5lOxkbuoC48yq1sr.

producto más adelante, con lo cual, de momento, se completa la instalación.

Creada la máquina virtual Windows, es posible instalar, dentro de ella, las aplicaciones que uno tenga por convenientes —concretamente todos esos programas que, por más que quisiéramos los linuxitas tener disponibles en nuestro propio sistema operativo, rehúsan producir los desarrolladores salvo para Windows y Mac.

De no tener clave, la instalación virtual de Windows va a incordiarnos posteriormente para que la activemos, acosándonos a la aceptación de las actualizaciones preceptivamente impuestas por Micro$oft. Existen varios modos de desactivar tal actualización automática.[48]

Según cuáles sean las aplicaciones que deseamos correr en esa máquina virtual Windows, puede resultar razonable —y aun recomendable— desconectarla del internet; en efecto, Micro$oft nos amenazará —al menos en caso de no haber desactivado la actualización automática— con los virus y las contaminaciones malignas; desconectando del internet nuestra máquina virtual estamos a salvo, no sólo de intrusiones y acosos de Micro$oft, sino también de maléficas infiltraciones (nunca sabe uno provenientes de dónde —no faltando quienes sospechen que pueden venir, a veces, de los productores de antivirus).[49]

La máquina virtual podrá comunicarse con nuestro sistema operativo Linux sin necesidad de internet, por otras dos vías (dispositivos externos USB y directorios compartidos), a fin de pasar contenidos de dentro afuera de la máquina virtual o viceversa.

Por último, hay que señalar que correr en Linux alguna aplicación diseñada para correrse en el sistema operativo de Micro$oft no siempre requiere acudir a una máquina virtual con Windows. Algunas de tales aplicaciones (mas no las recientes) pueden correrse con el programa Wine, fácilmente instalable en cualquier distribución de Linux; otros dos programas —más potentes que Wine y de mayor alcance— son PlayOnLinux y CrossOver. (De éste último —que es de pago— tuve yo una versión ya antigua, que me resultó utilísima durante años, si bien hoy, ya desfasada, la he desinstalado.).[50]

* * *

Asombra a los usuarios de otros sistemas operativos (en particular a los

---

48. V. alguno de estos tres sitios: www.wikihow.com/Turn-Off-Automatic-Updates-in-Windows-10, www.tomsguide.com/how-to/how-to-turn-off-automatic-updates-in-windows-10 y www.windowscentral.com/how-stop-updates-installing-automatically-windows-10.

49. V. *supra*, nota 34.

50. Sobre cómo instalar —y hacer funcionar— el PlayOnLinux (concretamente para correr en él alguna versión del Micro$oft-Office), v. el vídeo youtu.be/naGZpnHMXY8?si=quidPZCRWqp7lbLA1. También vale la pena abrir la pág^{a} *web* es www.playonlinux.com. En cuanto a CrossOver, su pág^{a} es www.codeweabers.com.

del Windows) cuán conflictiva y batalladora se muestra la comunidad linuxita,[51] a la cual han calificado de «tóxica».

Lo que sucede es que, si bien para muchos linuxitas su sistema operativo es un mero utensilio, para bastantes es algo más, mucho más. Es una pasión. Fúndese esa pasión por el Unix/Linux con aquella que sienten hacia su particular distribución. Arden los duelos entre los adeptos de las diferentes distribuciones (al par de los que enfrentan a hinchas de equipos deportivos o de partidos políticos rivales).

Felizmente, echando agua al fuego, tienden a suavizar esas contiendas los influenciadores (v.g. los titulares de canales de YouTube con temática linuxita) —si bien tal vez, a veces, involuntariamente alimente la polémica alguno de ellos con sus evaluaciones, bien intencionadas mas, inevitablemente, debatibles. Está claro que, al ofrecérnoslas, cumplen la función de brindar una guía pericial en la frondosa selva del Unix/Linux.

Ya he dicho por qué me abstendré yo de emitir opinión alguna contraria a otras distribuciones, limitándome a alabar aquella por la cual he optado. El lector interesado en esas controversias hallará acaloradas disputas en bitácoras y foros de discusión linuxitas.

En este mundo del Unix/Linux no sólo oponen las discusiones —frecuentemente subidas de tono— a adeptos de sendas distribuciones, sino también a quienes apoyan determinadas innovaciones (frecuentemente generalizadas a varias distribuciones) contra quienes (así sea por inercia) preferimos aferrarnos a los modos ya consagrados de hacer las cosas. A alguna de estas disputas me referiré después.

De momento limítome a reseñar un punto del forcejeo entre distribuciones diversas: el atinente al número de usuarios de una o de otra. En general en Linux es sumamente cuestionable cualquier estadística, porque ni el sistema operativo se compra ni el usuario se registra, con lo cual están siempre basados en frágiles inferencias los cálculos sobre cuántos somos y, de ellos, quiénes usan una distribución y quiénes otra.

Esgrímese el real o presunto mayor número de usuarios como un argumento a favor de la propia distribución (un entimema en el que se presupone la premisa mayor: que los más no se equivocan).[52]

---

[51]. Si es que puede hablarse de comunidad, vocablo inapropiado para una suma tan vasta, desperdigada y heteróclita de individuos a quienes no une nada salvo ser, total o parcialmente, usuarios de un mismo sistema operativo, bajo dispares distribuciones.

[52]. El sitio Distrowatch nos ofrece una escala de preferencia: Linux Mint, Zorin, Pop!_OS, KDE Neon, Elementary OS, VanillaOS, LinuxLite, TUXEDO_OS, Kubuntu, RhinoLinux, VoyagerLive, Lubuntu, BodhiLinux, Xubuntu, Linuxfx, BrOS, Makulu Linux, Ubuntu MATE, Ubuntu Studio, LXLE, Linux Kodachi, Nitrux, Ultimate Edition, Ubuntu Unity, Armbian, Ubuntu Budgie, Ubuntu Cinnamon —y así hasta un total de 159. No otorgo yo valor alguno a esa ordenación, que me resulta arbitraria.

Dos dificultades rodean a cualquier jerarquización. La primera es la base estadística. *Distrowatch* parece fundarse en el número de consultas en su espacio *web* (o acaso en el de los *downloads*), lo cual nada significa, pues la abrumadora mayoría de los usuarios de alguno de los muchos Ubuntus no se lo bajan de *Distrowatch* —y, a lo mejor, ni siquiera abren nunca esa página *web*. Imagínome que lo propio sucede con otras distribuciones.

La segunda dificultad es la de cómo individuar las unidades de cuenta. ¿Tomaremos a LinuxMint o a Debian como unidades? Los desarrolladores de cada uno de esos rótulos entregan, bajo sendas etiquetas, una pluralidad de distribuciones alternativas. No hay un solo LinuxMint, sino varios —uno de ellos, por cierto, mera variante de Ubuntu.

En cambio, bajo el rótulo «Ubuntu» sólo figura uno de los sabores de Ubuntu (podríamos denominarlo «Gubuntu» o «Ubuntu Gnome»), al cual hay que agregar otros siete avalados por la misma empresa productora, Canonical: Kubuntu, Lubuntu, Ubuntu Budgie, Ubuntu Kylin, Ubuntu MATE, Ubuntu Studio y Xubuntu, junto con 29 derivadas.[53]

No sé cuán arriesgada sea mi apreciación, a ojo de buen o mal cubero. Opino que la mitad de los linuxitas usan una u otra de las 37 distribuciones que, en buena medida, vienen a ser variantes del Ubuntu (abarcando la lista de esas 37 distribuciones tanto a los sabores de Ubuntu cuanto a las distribuciones derivadas). De esa mitad, quienes usan uno u otro de los ocho sabores aprobados por Canonical serían, verosímilmente, al menos tan numerosos como cuantos usan alguna de las 29 derivadas. (Y hasta me pregunto si no habría que añadir, además, a los usuarios de la versión Ubuntu de Linux Mint, con lo cual serían 30 las distribuciones derivadas.)[54]

Un acercamiento sería posible convocando, periódicamente, congresos o simposios internacionales Unix/Linux, siempre que en su código normativo se prohibieran los reproches a otras distribuciones. Sería lícito criticar tales o cuales aspectos concretos del *software*, de manera que no se tomara nunca como blanco a distribución alguna. Centraríase cada ponente en argumentar a favor de tales o cuales realizaciones o propuestas, siempre bajo reglas de buen tono, cortesía y mutuo respeto.

A toda costa habría de evitarse que se convirtiera en un centro de poder —en un polo de encumbramiento— la oficina o secretaría permanente entre los

---

[53]. Son derivadas de Ubuntu aquellas distribuciones que, sin gozar de dicho aval, no dejan de ser variantes o cuasi-variantes suyas. La lista es larga: Wubuntu, BackBox, BackSlash Linux, BlankOn, Bodhi Linux, Elementary OS, Enso OS, Feren OS, Freespire, GalliumOS, Guadalinex, Greenie, KDE neon, Linspire, Linux Lite, LliureX, LXLE, Nova, OSGeoLive, Peppermint OS, Pop!_OS, Regolith, SharkLinux, SuperX, Trisquel GNU/Linux, UBports, Voyager Ubuntu, WattOS y Zorin.

[54]. Nada de ello prueba que sean peores Slackware, MXLinux, NIXOS, Alt-Linux, CENTOS, LinuxMint, Debian, Devuan, Red-Hat, Fedora, Manjaro, Mageia, Macrium, Acronis, Arch, EndeavourOS, Linspire, Nova Linux, Manjaro, Open Euler, ROSA, AcademiX, AstraLinux, Porteus o Garuda.

congresos. Debería, antes bien, ser anónima, permaneciendo en la penumbra, sin celebridad. Podría convenirse la existencia de una plataforma *web*, donde, en armonía y concordia, convivieran defensores de varias distribuciones, principalmente de aquellas que gozaran de mayor relieve social o que merecieran un puesto de honor por su veteranía, como Slackware.

Sería muy provechoso que en esos simposios hubiera participación de otros sistemas operativos de la misma familia Unix, principalmente los tres derivados del venerable BSD. Todos somos Unix, siendo eso lo que nos une. Hemos de luchar contra quienes pretenden separarnos.[55]

---

[55]. No se me oculta que ya ha habido una pluralidad de congresos y conferencias en torno al Linux, empezando por los «Linux Kongressen» —celebrados, casi todos, en Alemania—, de 1994 a 2010, los «Linux Symposia», en el Canadá de 1999 a 2014, los LinuxCon de 2009 a 2016 y los ulteriores *Open Source Summits* de 2017 a 2023. Casi todos esos encuentros han sido regionales; poquísimos de ámbito mundial. Ha ido decantándose, cada vez más, su orientación hacia las ideas de *fuente abierta* —ladeando al Unix/Linux y a sus distribuciones. Pocas de esas distribuciones han participado en tales reuniones —quizá ninguna.

Lo que yo estoy proponiendo es un encuentro mundial periódico sobre el Unix/Linux —incluyendo, prominentemente, el GNU—, en el cual cobren protagonismo, junto con los desarrolladores del kernel, y de las utilidades GNU, las principales distribuciones.

Capítulo IX

Defensa del Ubuntu

Se han esgrimido contra Ubuntu seis objeciones.

— La primera es que viene producido por una empresa, siendo preferibles las distribuciones que emanan de su respectiva comunidad.

Personalmente desconfío de ese vocablo de «comunidad», no sólo con relación al cúmulo de usuarios del Linux en general, sino también al de Debian o cualquier otra distribución. Hay una suma dispersa de usuarios, la mayoría de los cuales usan la distribución en silencio. La presunta comunidad suele estar formada por los opinantes en redes sociales, bitácoras y foros. Cada distribución viene producida por un colectivo, sea una firma comercial, sea un club. Los productores pueden estar atentos a las críticas que reciben, mas responsables de la distribución son exclusivamente quienes la desarrollan y ponen en circulación.

— La segunda objeción es que Canonical toma sus decisiones sin atender a las expectativas o las preferencias de sus usuarios.

Esa acusación es calumniosa. Canonical ha cambiado a menudo sus decisiones justamente ante las reacciones expresadas en esos medios, a pesar de que no sean forzosamente representativas de lo que piensen la mayoría de los usuarios, ya que la mayoría es silenciosa.

— La tercera objeción es el recurso de Ubuntu a las encapsulaciones *snap* —un tema que se abordará más adelante.

— La cuarta objeción es que no siempre ha sido acertada su opción por el gestor de entorno gráfico, o GUI (yendo y viniendo del Gnome a ciertas alternativas, como Unity, que no han gustado a todos).

Respondo diciendo que Canonical avala varios sabores con sendos entornos gráficos, aunque otorgue su preferencia a uno de ellos. Para aquellos a quienes no nos gusta Gnome está la posibilidad de optar por uno de los otros sabores —como hago yo quedándome con el Lubuntu.

— La quinta objeción es que en una de sus sucesivas versiones hubo ofrecido

un enlace que conducía a Amazon, lo cual fue visto como una forma de publicidad.

Desconozco si así fue. En todo caso resulta dudoso que nadie necesite ese enlace para abrir el sitio de Amazon. Sea como fuere, ¿es peor ofrecer un enlace directo a Amazon que uno a YouTube o Google? ¿Por qué?

— La sexta y última objeción es que Canonical espacia demasiado las sucesivas versiones de su sistema operativo. En abril de cada año par emite una versión LTS (*long term support*); entre medias, cada semestre difunde una versión cuya estabilidad no está igualmente garantizada. Un bienio se les hace a algunos comentaristas una eternidad; e incluso un semestre les resulta larguísimo. Además, quéjanse de que, si bien —en ese compás de espera— pueden ir actualizándose componentes sueltos, el resultado será un híbrido.

Tal objeción será pertinente para quienes anhelen estar constantemente con lo último o para los obsesionados por la seguridad (que ansíen incesantemente nuevas medidas de blindaje), pero la abrumadora mayoría de los usuarios no compartimos ni ese anhelo ni esa ansiedad. (Para mí sucede al revés: no sólo estoy corriendo ahora la versión de Lubuntu 22.04 de abril de 2022, sino que pienso seguir haciéndolo mientras no me haya visto constreñido a cambiar por alguna imprevista circunstancia de fuerza mayor.)

Cerraré este capítulo con una reflexión más genérica. Mi impresión es que la mayoría de las discusiones entre adeptos y adversarios de distintas distribuciones giran en torno al GUI, o interfaz gráfico (Gnome, KDE, XFCE, Cinnamon, etc) o al gestor de ventanas (como el OpenBox) —interfaz y gestor suelen ir acoplados entre sí hasta presentarse como dos caras de lo mismo.

Mas eso sólo atañe a la corteza, a la cáscara. Cierto, muchos usuarios se quedarán en esa superficie. No pocos de ellos, viniendo del Windows, están habituados a esa relación: abrir el escritorio (*desktop*), pinchar en un icono, no salir nunca del entorno gráfico.[56]

Para un linuxita fetén, hay que escaparse de esa jaula gráfica, accediendo al CLI. Una vez en la línea de órdenes, las diversas distribuciones son bastante semejantes entre sí, contando más sus similitudes que sus disparidades.

Por otro lado, son deseables los mestizajes. Posee sus propias cualidades y ventajas cada familia de distribuciones. Algunas de sus ventajas son inseparables de rasgos propios —que, desde otras opciones u otros puntos de vista, puede que no resulten atractivos o satisfactorios.

---

[56]. Por cierto, ¿cuán acertado es traducir «desktop» como «escritorio» cuando en él ningún usuario escribe nada? ¿No hubiera sido más adecuado traducirlo, literalmente —y según el diccionario—, como «pupitre» o «buró»?

No obstante, son posibles —y frecuentemente benefeciosas— las hibridaciones; los de la familia Ubuntu podemos aprender de los de Fedora, de los de Slackware, de los de Arch —y seguir aprendiendo de los de Debian, rama de la que procede todo nuestro linaje.[57]

Quizá, rebajando el orgullo, podrían los de Fedora y Arch aprender algo de Ubuntu, en vez de mirarlo por encima del hombro.

En suma, nada bueno dicen de nosotros la hostilidad y la acrimonia que tantas veces enfrentan a los *tifosos* de diferentes distribuciones (si se me permite el italianismo), siendo secundario lo que nos divide y teniendo todos tanto que ganar de una mejor comprensión entre sensibilidades o preferencias diversas, frecuentemente de puro estilo.

* * *

Para acercar a las varias distribuciones del Linux —facilitando, a la vez, la transición al mismo de usuarios descontentos del Windows— ha propuesto Jack Wallen la creación de un Linux oficial, inspirado en Debian pero que tomaría algunas ideas de otras distribuciones. A mi modo de ver, resulta quimérica tal propuesta. Desde luego es posible que alguien cree una nueva distribución y la llame «Linux oficial», lo cual simplemente agregaría un morador más al territorio de las distribuciones, con el inconveniente de que esa autodenominación provocaría el furor de muchos linuxitas, tal vez la mayoría.

Hay que valorar la ciberdiversidad del Unix/Linux por su riqueza, por la libertad de opción, por el incentivo que implica para la creatividad y hasta por el placer que causa el ensayo sucesivo o alternativo de varias distribuciones, cada una de las cuales ofrece ciertas ventajas, procurando al usuario una experiencia singular.

---

57. No se me oculta que también ha habido malos contagios, como el del malhadado *systemd*, al cual me referiré en el capítulo XXI.

Capítulo X

Mi *software*

En torno a un núcleo o kernel (encargado de comunicarse con el **firmware** —un *software* que viene con la máquina, sin el cual ésta sería un trozo de metal), un sistema operativo es una, mayor o menor, colección de *software*, suficiente para que el usuario pueda —sin ningún *software* sobreañadido— manejar productivamente el computador, realizando algunas tareas de entre aquellas que se espera ejecutar con tales máquinas, ya sean de estudio, de trabajo, de diseño o de recreación.

Sin embargo, sólo existe un sistema operativo cuando es extensible, e.d. cuando a ese *software* preincorporado puédesele agregar otro adicional que permita a muchos usuarios llevar a cabo tareas irrealizables con sólo el *software* aportado por la instalación del sistema operativo. ¿Convierte eso a dicho *software* adicional en una parte del sistema operativo?

Uno de los rasgos individuantes de un sistema operativo es qué *software* adicional soporta. La facilidad de su incorporación es otro factor determinante: más podrá decirse —aunque sea ya en un sentido lato— que un sistema operativo engloba un cierto *software* cuando éste haya venido preparado y ofrecido a los usuarios, libre y gratuitamente, en repositorios administrados por los desarrolladores del propio sistema operativo. Menos, cuando, por mucho que el sistema operativo soporte ese *software* (o sea, resulte compatible con él), no sólo el *software* mismo no está preparado en ningún repositorio ofrecido por los desarrolladores del sistema operativo, sino que hallarlo, descargarlo e instalarlo son tareas que incumben exclusivamente al usuario, por su cuenta y riesgo.

Todavía menos cuando —además de asumir, él solo, esa tarea de busca, cercioramiento, descarga e instalación (para nada auxiliada ni auspiciada ni guiada por los productores del sistema operativo)—, ha de pagar el usuario ese *software* sobreañadido.

Suele decirse que la disponibilidad exclusiva de una amplia gama de *software* constituye uno de los motivos por los cuales mantienen su adhesión (su sumisión, diría yo) los adeptos y usuarios del Windows.

¡Bien! Sólo que tal *software* para nada forma parte del propio sistema operativo Windows. Es un producto de otras casas y firmas, que únicamente

venden versiones adaptadas al Windows (¿a qué Windows? —porque, a menudo, cuando un programa corre en el Windows-X, no corre en el Windows-Y, ya sea X<Y o X>Y).

Ha de buscarse la vida el usuario, indagando cuáles sean esos otros programas, hallando dónde procurárselos, pagando el precio (muchas veces elevado) e instalando el *software*, sin que, en caso de problemas, quepa en absoluto invocar la ayuda de los desarrolladores del sistema operativo, que se llamarán andana.

No es virtud alguna del Windows que sólo vendan *software* susceptible de ejecutarse bajo ese sistema operativo firmas como Adobe, Alludo (ex-Corel), los desarrolladores de OCRs (y, en general, de controladores de escrutadoras) igual que la abrumadora mayoría de los productores de juegos. Esas compañías piensan en su ganancia, calculando que les trae cuenta no producir otras versiones de su *software* que aquella que tenga más probabilidades de venderse.

De hecho no muchas son las aplicaciones venales para Unix/Linux; haberlas, haylas; yo mismo compré, hace ya no pocos años, una versión del Corel-WordPerfect para Linux —que pronto dejé de usar por varias razones; otra aplicación que también compré (lo he recordado unos párrafos más atrás) fue CrossOver, gracias a la cual pude, durante bastantes años, correr en Unix/Linux programas para Windows, como el M$Word, el WordPerfect de Corel, una versión del Reader de Adobe y varios más; el entorno que lo facilitaba era el emulador Wine del Unix/Linux; sólo que éste resultaba inmaduro e inamigable, facilitándose considerablemente su uso gracias a la licencia pagada a Crosssover. Más recientemente he comprado el programa Ermine, del cual hablaré más abajo. Vagamente recuerdo haber comprado algunas otras aplicaciones, hoy olvidadas. Eso sí, han sido excepcionales tales compras.

En y con Unix/Linux casi todo es gratis. No sólo el propio sistema operativo, sino prácticamente todo el *software* soportado, ya venga ofrecido en los repositorios oficiales (u oficiosos) de los desarrolladores de alguna distribución del Linux, ya sea puesto a disposición del público por terceros, como el CDRTools, creado por el difunto Jörg Schilling (sobresaliente, fecundo y polémico programador alemán, muerto en octubre de 2021 de cáncer de riñón; uno de esos hombres eminentes, de personalidad recia y singular, a veces peleona, que descuellan por su labor y por la originalidad de sus tesis). De encominable valor era su *software*, gracias al cual pudimos, durante años, no sólo grabar CDs y DVDs, sino también volcar en ficheros audio los contenidos de CDs musicales —todo ello cuando no existía ningún programa alternativo, o los que había eran muy inferiores en calidad y fiabilidad. (Obra suya fueron algunas aplicaciones que yo sigo utilizando hoy y que, con ventaja, reemplazan a las que figuran por defecto en las distribuciones de Linux.)

Proliferan hoy las aplicaciones para el Linux gratuita y libremente ofrecidas al público en sitios de *software* no auspiciados ni reconocidos por los

productores de ninguna distribución del Linux. Al parecer casi todas corren igualmente bajo otros sistemas operativos de la familia Unix, como Solaris y BSD. (Entre esos sitios citaré Sourceforge y Codeberg; en julio de 2024 éste último alberga 143.827 proyectos.)

¿De qué viven tales sitios, cuyo acceso es gratis? ¿Sólo de las donaciones? ¿Acaso de la publicidad —tal vez subliminar?

Lo ignoro. Su mera existencia prueba que el dinero no lo es todo, a diferencia de cómo piensan los multimillonarios del sexteto MATAFA.

Repasando la lista de mi *software* (e.d. de aquel que yo ejecuto efectivamente de manera más o menos habitual), percátome de lo poco que venía incorporado en mi distribución actual del Linux (el Lubuntu 22.04.4). La mayoría me las he buscado e instalado yo.

Éste es el elenco de aplicaciones que uso, unas de ellas todos los días, otras a menudo —o al menos de vez en cuando—:

— rEFInd como administrador de arranque (*boot manager*), que da paso al kernel.

— Ifconfig (de la colección net-tools).

— Libre-Office (poco, y ese poco únicamente para conversiones).

— Okular (visualización y manejo de ficheros gráficos, especialmente .ps y .pdf).

— Dosemu-debian (del cual hablaré después).

— Virtual-box (para correr máquinas virtuales con otros sistemas operativos).

— Vivaldi browser.

— Lynx (un navegador en modo texto —que hoy, desgraciadamente, ya no resulta de mucha utilidad, porque tanto abusan del modo gráfico las páginas *web* que lo ocultan todo frente a navegadores en modo texto —o incluso rotundamente le dan con la puerta en las narices, no dejándoles ni siquiera atisbar).

— MC (Midnight Commander, gestor de ficheros).

— Gwenview (otro visualizador de imágenes).

— Gimp (dibujo y manipulación de gráficos).

— Glabels (carteles, etiquetas y carátulas).

— ps2pdf, pdftk, pdfcrop, pdfjam y otras aplicaciones de manejo de pdfs.

— Back-in-Time y mirdir para respaldos (sirviendo mirdir, en general, para igualar un directorio de destino a otro dado como fuente).

— Mutt (cliente de email).

— ftpd, telnetd y samba —para conexiones de intrarred doméstica.

— lookat (un visualizador de texto) más el antiword (con el cual uno ve como texto un documento del .doc).

— lshw, survey, lowercase —utilidades varias.

— ren y otras utilidades de redenominación.

— unix2dos y recode (utilidades de conversión de ficheros de un mapa de caracteres a otro).

— varias utilidades de formateo y configuración de dispositivos de almacenamiento periféricos.

— las utilidades necesarias para la impresión y otras similares.

Otras utilidades que uso son: dd (para volcar una imagen de disco sobre una unidad externa o viceversa); Stacer (para tareas de limpieza), TexInfo, Atril, Evince, el Chrome de Google, Telegram.

Puntualmente he usado los manejadores de paquetes Debian: Gdebi, Muon y Synaptic, mas he dejado de usarlos; del mismo modo quedan sin usar muchas otras aplicaciones que tengo instaladas, si bien no descarto correr alguna de ellas cuando se presente la ocasión y tenga algún motivo para hacerlo.

Alguna más podría mencionar, como lftp, wget y losetup. De pascuas a ramos, MSWord bajo Wine, para convertir documentos del .doc al WordPerfect nada más. En total, serán unas cuarentaitantas aplicaciones.

Varias de las aplicaciones que yo he incorporado a mi acervo de *software* ya estaban empaquetadas y ofrecidas en los propios repositorios de Ubuntu, preparadas para descargarse e instalarse; lo cual las convierte, digamos, en cuasicomponentes del sistema operativo, o de la distribución Ubuntu. (No en componentes del todo, en sentido estricto, puesto que, según se había instalado en el computador la distribución, el usuario no disponía de tales aplicaciones sin realizar, por su propia iniciativa, operaciones adicionales —aunque, a cambio, sí dispusiera de muchas otras, que yo no uso; varias de ellas, como el navegador Firefox, las he borrado.)

Aquí he de hacer una importantísima aclaración. Durante la mayor parte de mi experiencia como usuario de Slackware, no tuve conocimiento alguno de qué fuera un paquete ni, por consiguiente, de en qué consistiera un gestor de paquetes. Tardíamente me enteré de que otras distribuciones los tenían. En realidad, Slackware también tenía paquetes, pues cada una de las aplicaciones que contenía venía, junto con las librerías y otros elementos auxiliares, embalada en un archivo de desinencia «tar.tx», comprimido; instalarlo requería descomprimirlo, ponerlo en el directorio raíz y dar la orden «install».

Lo mismo podía hacer el usuario, si le daba la gana, con *software* de cualquier otra proveniencia; yo lo hice muchísimo, a menudo compilando las aplicaciones, no limitándome a instalarlas en su forma binaria. (No es que me

bajara cualquier aplicación tentadora; atendía a señales de credibilidad y fiabilidad del sitio; de hecho jamás me bajé *malware* de ningún tipo; más bien lo que me ha causado efectos negativos —y en una ocasión un descalabro— ha sido fiarme de los consejos de algunos opinadores foreros que se expresan con aplomo, envueltos en un halo de enterados —o hasta de expertos.)

Tan insolente desenvoltura en la relación entre Slackware y sus usuarios podía provocar, y provocaba, conflictos de librerías (más abajo volveré sobre esta importante cuestión). El usuario de Slackware era libre de poner y de quitar lo que le diera la gana, sin que nadie le estuviera asestando ni directivas ni advertencias (menos todavía estorbos o barreras) sobre qué podía o debía hacer en su propia máquina. A él incumbía también pagar las consecuencias eventuales de sus actos.

Muy tardíamente, hacia el final de mi itinerario con el Slackware, supe de un gestor de paquetes, al parecer inicialmente desarrollado por iniciativa de terceros, mas creo que a la postre incorporado a la distribución misma, el *slackpkg*. Yo lo usé en la última instalación de Slackware, aquella que llevé a cabo en la máquina que compré en enero de 2016, puesto que tuve que sobreinstalar, adicionalmente al Slackware 14.1, varias capas como el Multilib y otras, desarrolladas por voluntarios, gracias a las cuales pude correr en Slackware una aplicación para el Gnome (el Glabels) junto con programas de 32 bits, como el Google-Chrome y el Adobe-Reader; sin el slackpkg dudo que hubiera podido realizar, por mí mismo, operación tan compleja.

Desde hacía decenios, en cambio, venían facilitando las cosas a sus respectivos usuarios otras distribuciones, mediante el empaquetamiento, que daba resuelto el problema de los conflictos de librerías (o, si se prefiere la más exquisita traducción, «bibliotecas»). El usuario de Red Hat o el de Fedora solían bajarse únicamente *software* empaquetado en un repositorio de su propia distribución, para lo cual bastaba ejecutar, en el CLI, la orden «sudo dnf» (después reemplazada por la eufónica «sudo yum») para ver qué paquetes eran instalables.

Igual sucedía con las numerosas distribuciones emanadas de Debian, sólo que aquí la orden es «sudo apt». Esa pauta la heredaron aquellas distribuciones como Ubuntu (con sus varios sabores), derivadas de Debian.

Incluso diría más: concibiéndose Ubuntu como un Linux para legos (un Linux para gentes cualesquiera sin la más mínima pretensión de ser «nerds» o «geeks» ni nada remotamente parecido), a fin de disputarle el terreno a Windows, reclutó un público —en general proveniente de Micro$oft— sin inclinación alguna a fuñar en el *software* (ni menos en el *hardware*), contentándose con lo que le brindaran los desarrolladores del sistema operativo, cumpliendo disciplinadamente sus instrucciones de actualización de

software y no buscando ni modificando ni quitando nada por iniciativa propia.[58]

Los paquetes de Red-Hat y Fedora llevan la desinencia «.rpm», al paso que los de Debian y sus bifurcaciones llevan la de «.deb». La diferencia no es nominal, tratándose de dos tipos diversos de empaquetamiento, sin que siempre funcione (o aun siquiera resulte instalable) el programa Alien, que promete convertir los paquetes de uno de esos tipos en paquetes del otro. (Existen otros empaquetamientos, como los «.pkg.tar» de la distribución Arch Linux y sus derivadas.)

A diferencia de la indómita y salvaje libertad de la cual, durante decenios, altivamente disfrutábamos los usuarios de Slackware, los de otras distribuciones fueron siempre más disciplinados y prudentes, dejándose guiar por sus expertos o gurúes, quienes iban indicándoles, no sólo qué podían instalar y qué no, sino también de dónde bajárselo.

Al haberme visto constreñido a abandonar el Slackware, pasándome al Ubuntu en 2024, he tenido que renunciar a tan omnímoda libertad. Propúseme —sin hondo convencimiento— resignarme a no usar más que aquel *software* que me autorizaran mis superiores, el que figurase en los repositorios de Ubuntu (o en otros que, ya más arriesgadamente, aprendí a incorporar a la lista de repositorios legítimos —si bien posteriormente los he borrado, a causa de una mala experiencia).

Sólo que me han podido mi indisciplina y mi ansia de libertad, llevándome frecuentemente a saltarme tales restricciones, buscando *software* de mi conveniencia por otros sitios (aunque siempre reputados).

Si no me he atenido a las restricciones emanadas de arriba es que, por esa vía, resultábame imposible instalar algunas de mis más caras aplicaciones, como el Dosemu y los navegadores Chrome y Vivaldi. Osadamente, estos programas (junto con algunos otros) los he bajado e instalado al margen de los cauces prefijados —habiéndome, eso sí, previamente enterado por internet de cómo hacerlo y cerciorado de que eran genuinos y seguros los portales donde estaban disponibles; lejos de quedar defraudadas mis expectativas, me resulta plenamente satisfactorio ese *software* que, por mi cuenta y riesgo, me he bajado e instalado (burlando a los pilotos que, en la distribución de Ubuntu —y hoy casi en cualquier otra—, vigilan, residentes en memoria, que el usuario no se desmande).

En concreto —junto con el gestor de ficheros MC— dos programas, el Dosemu y el navegador Vivaldi, siguen siendo mis valiosos compañeros diarios y mis mejores instrumentos, sin que ninguno de ambos se encuentre en los

---

[58]. Una de las causas de la presuntamente decreciente popularidad de Ubuntu estos últimos años puede ser que, de entre esos neófitos, un número de ellos hayan avanzado en el conocimiento y manejo de sus máquinas, decidiendo pasarse a otras distribuciones —como Arch y sus derivadas— en las cuales requiérese mayor espontaneidad del usuario, a quien no se le marca la pauta de cómo ha de actuar.

repositorios oficiales u oficiosos de Ubuntu. (Tengo entendido que existen problemas de licencia; por gratuitos que sean, esos programas se distribuyen bajo licencias no totalmente compatibles con las oficiales del GNU, de las cuales hablaré en el capítulo XXII; por ello, los puristas juzgan que, por mucho que sean *freeware*, no son *free software*, en la acepción vigente en ese movimiento del «free *software*».)

Otro programa, el Ermine (una aplicación de estatificación de ejecutables, o sea de encapsulación), he tenido que buscármelo por más pedregoso e insólito sendero —comprando una licencia de un mes de duración únicamente—. Mi razón para hacerlo fue convertir en inmutables ciertas aplicaciones, como el MC, mas, principalísimamente, el Dosemu. (Volveré sobre ello en el capítulo XV.)

Capítulo XI

El calvario de Windows

En un ensayo precedente narré mis nueve días de tribulaciones con el Windows 8 instalado de fábrica en el ultrabook (pequeño portátil) que, comprado en 2014, estuve usando algo más de tres años.

Baldíos resultaron mis intentos por trabajar con ese horrible sistema operativo, el cual parecía complacerse en ponerme zancadillas, acorralándome para que comprara antivirus e impidiendo cualquier uso sereno y productivo del portátil. Días amargos, de frustraciones, que inicialmente quise atribuir a mi falta de costumbre con un sistema operativo nunca antes usado (ni él ni sus predecesores).

Cierto que ocasional, esporádicamente he abierto sesiones de uno u otro Windows (sea el 95, el 98, el ME, el XP); ha sucedido de manera puntual, para ejecutar, momentánea y excepcionalmente, tareas específicas (ejecutar programas que únicamente corrían en el entorno Windows, como el OCR Textbridge y el M$-Word —que yo únicamente he utilizado para convertir textos en formato .doc a documentos .wp del WordPerfect 5.1; hace años, acceder a páginas *web* oficiales que tan sólo toleraban el malhadado Internet Explorer).

Sin embargo, comparando el Windows8 con sus predecesores me percaté del deterioro. Los viejos Windows siempre me habían resultado cargantes, poco manejables, torpes, plúmbeos, inamigables, inflados, más ornamentales que productivos. Pero, dentro de su pesadez, renqueando, hacían su labor. Al menos entonces accedíase fácilmente al panel de control, detectando *hardware* añadido y configurando el escritorio (mal que bien). El sistema no me acosaba ni me importunaba incesantemente. Esas pálidas ventajas ya se habían desvanecido con el Windows8.

Borrado ese Windows8 tras la novena penitencial, poco he abierto en estos últimos años alguna máquina virtual Windows (Windows-XP), de tarde en tarde, con el VirtualBox, justamente para las tareas que acabo de indicar. Es probable que se hayan espaciado tanto esas sesiones que puedan contarse cuatro o cinco al año.

Al comprar en 2024 los tres pequeños computadores a los que me he referido más arriba —que son los que utilizo para mi trabajo—, tuve que pagar el Windows-11 por cada uno de ellos. Tres desembolsos, por sendos Windows.

En cada caso, Micro$oft me forzó a registrarme con todas mis generales de ley y varios de mis secretos más un buzón de email que fuera, para cada uno de los tres casos, distinto de cualquier otro usado antes en mis tratos con Micro$oft. Micro$oft prohíbe que un mismo usuario (o un mismo buzón de correo) tenga a su nombre varias máquinas donde venga instalado su sistema operativo. Agoté así la lista de mis buzones.

Tras ficharme y tenerme así múltiplemente registrado (exigiéndome, en cada caso, inventar una contraseña a su gusto), Micro$oft graciosamente me permitió, ¡por fin!, acceder a mis máquinas, las máquinas que yo había comprado con mi dinero.

Fue mi intención inicial la de preservar en las tres ese sistema operativo, aunque particionando el almacenamiento interno para que coexistieran Windows y Unix/Linux. Es más, destinado a la función de NAS —o sea nube doméstica— uno de los tres computadores, pensé incluso en dejarlo con Windows y sin Linux, esperando que, por malo que fuera ese sistema operativo, para eso al menos serviría. Me equivocaba.

Ha resultado imposible conservar el Windows. Trabajar con él, más imposible todavía. ¡Un tormento!

Comprendo que muchos usuarios están satisfechos de su Windows; deséoles que lo disfruten. Para mí ha resultado deleznable e inútil; un estorbo, que sólo me ha hecho perder mucho tiempo para, al final, tener que venir borrado.

Alaban muchos al Windows por su belleza. Yo lo he visto más feo que un trueno. Resulta ilegible sin lupa la letra de los menúes y mensajes. Llevóse bastante tiempo hallar el Panel de control; una vez encontrado, resistióse, como gato panza arriba, a venir configurado según mis preferencias. La única posibilidad que quedaba era optar por el alto contraste. Sólo que, ¡amigo!, con ese alto contraste la pantalla se vuelve lúgubre, tétrica, macabra.

No sólo eso, sino que, sobre todo, en cuanto el usuario modifica la configuración del color, no pocos menúes porfían en no darse por enterados de ese cambio del color (aceptándolo únicamente para el fondo o únicamente para el primer plano), exhibiéndose así en combinaciones como la de letra amarilla sobre campo blanco o la de gris oscuro sobre campo negro; o sea, resultan totalmente ilegibles.[59]

¡Cuidado con agrandar la letra porque entonces los menúes, desplegándose en cola vertical, desbordan la pantalla, resultando inaccesibles! ¡Nada, hay que resignarse a mirar la pantalla con lupa!

Siendo el protocolo SMB (samba) privativo de Micro$oft, esperaba yo que

---

[59]. Resulta para mí difícil entender que sean tan ineptos y descuidados los informáticos de Micro$oft que no hayan caído ni siquiera en una consecuencia tan palmaria; es como si no hubieran revisado ni aun superficialmente su cacareado producto.

mis nuevas máquinas detectaran mi red local, accesible por ese protocolo. (Hácelo el Linux-Ubuntu a la primera, sin el menor tropiezo.). ¡No! Por más vueltas que le di, ¡nada! El Windows-11 no reconoce el protocolo de Micro$oft SMB ni se comunica, pues, con mi red local.

Tras horas intentando hacer algo con el sistema ventanero, descubrí la tienda de Micro$oft. Pareciéronme de escasísimo interés las aplicaciones que en ella encontré. Tal vez valía la pena alguna que otra de pago (que ni siquiera ofrecía una versión modo de *shareware*); no quise gastar nada para un sistema que sólo estaba usando a prueba, tentativamente, y por el cual ya había forzosamente pagado al comprar el *hardware*. Resultóme de poca utilidad alguna aplicación gratuita que me bajé; todas engorrosas, torpes, inamigables.

Repasando el elenco de las aplicaciones incorporadas al sistema operativo, fui eliminando las de carácter lúdico o recreativo. Salváronse de la poda: el engorrosísimo outlook para correo electrónico (un programa que, resultándome hinchado e inamigable, había ya eliminado en mi tableta android); el Edge (en lo cual reconozco que Micro$oft ha mejorado mucho con relación a aquel viejo e incómodo Internet Explorer); y el Office365.

Abrí el Office365. Mis objeciones al WYSIWYG ya las expuse en un artículo de hace muchos años, «Why I am not a Wordist».[60] Para mi trabajo no me siento capaz de usar ni ese Office ni ningún otro —ni siquiera el LibreOffice que tengo gratuitamente en Unix/Linux, del cual únicamente me valgo para conversiones (para convertir documentos del WordPerfect 5.1 a algún formato de moda, como .odt o .docx).

Ahora bien, por mucho que fuera mi desagrado frente a otros programas gráficos de tratamiento de texto WYSIWYG, no estaba aún curado de espanto para lo que sufrí al darme de bruces con el Office365.

Una buena parte de la pantalla quedaba ocupada por varias hileras grisáceas de garabatillos o iconzuelos —todos, más o menos, del mismo color y muy semejantes entre sí (para distinguirlos, memorizando cuál es la función de cada uno, serán menester un ojo muy adiestrado y bastante mayor agudeza visual que la mía). Una amplísima columna de la izquierda de la pantalla presentaba no sé qué contenidos (estilos o algo así), mientras que toda la mitad derecha se la comía una exhibición de plantillas. Del escaso espacio restante (el dedicado a introducir el texto, una cuarta parte de la pantalla aproximadamente), sólo estaba a disposición del usuario la columna central, entre anchos márgenes sin contenido a derecha e izquierda (consecuencia del WYSIWYG).

Siendo yo tan empecinado, no me conformé. Estuve bregando durante horas para devolver la pantalla al uso esperado de ella en un programa de tratamiento de texto, o sea el de escribir ese texto y verlo escrito. En vano.

---

60. V. http://lorenzopena/dos/word.pdf, http://lorenzopena/dos/word.wp y http://lorenzopena/dos/word.htm.

Comprendí que se trataba de una inmodificable configuración de fábrica; eran lentejas, las tomas o las dejas.

Todavía no resuelto a dejarlas, quise, al menos, guardar un minúsculo escrito para salvarlo en el formato WordPerfect —pues, al fin y al cabo, es ése el único uso que hago yo del M$-Word. Díjome entonces el programa que —sin consultarme ni siquiera advertirme— ya había salvado mi documento (con el nombre original de «mi documento.docx»), en la nube, en SU nube, la nube de Micro$oft. Podía, sí, bajármelo. En cuanto a una opción para guardarlo en otro formato, ¡nanáin! Sólo estaba accesible el formato .docx.

Es, a lo mejor, una versión capada del Office365 la que viene preinstalada en el computador como parte del sistema operativo Windows-11 OEM. No lo sé. No percibí advertencia alguna en ese sentido indicando que, mediante pago, pudiera comprarse otra versión más completa.

Estaba agotada mi paciencia.

Aun así tuve intención de preservar en cada una de las tres máquinas su partición Windows, achicándola para un buteo dual Linux/Windows. De hacer una copia de seguridad del sistema operativo para su restauración no hallé nada.

Contrariamente a la propaganda que había leído en alguna página *web* de Micro$oft, resultó imposible achicar desde dentro las particiones. (Micro$oft ocupa todo el espacio de disco sin dejar hueco para ningún otro sistema operativo.) Desde fuera (con el programa de Unix/Linux GPARTED) conseguí esa contracción, instalando el Linux en otra partición.

Sin embargo, el Windows había tomado posesión de la partición de arranque (la /boot/EFI), de modo que, tras la instalación del Linux, éste no podía arrancar. Arrancando desde el almacenamiento interno del computador, siempre se adueñaba de la máquina el Windows-11.

Comprendí que la única manera de conseguir una coexistencia de los dos sistemas operativos sería instalar, dentro del computador, otro disco duro, agregando una configuración alternativa del EFI. Resultábame, por el momento, una tarea engorrosa, no estando dispuesto a posponer la utilizabilidad de la máquina recién adquirida.

Conque, finalmente, ha tenido que hacerse, en las tres máquinas, la instalación del Ubuntu (Kubuntu o Lubuntu) borrando completamente el malhadado Windows-11. (Mi único lamento es el de no haber procedido a la eliminación del sistema de Micro$oft según entraba la máquina en mi domicilio.)

Mala era mi opinión del Windows, pero esta horrenda experiencia me ha dejado escozor y despecho. Felizmente, según pasan las semanas, se van disipando las iras.

Mi conclusión es que, si no siempre es fácil la adecuada instalación y

configuración del Linux (en la distribución que cada cual escoja), en todo caso para mí el Windows resulta un sistema, no sólo totalmente inamigable y hostil, sino, peor que eso, inutilizable. Ni le he hallado atractivo alguno ni he visto para qué pueda servir.

Entiendo que el juego es uno de los usos más frecuentes de los computadores portátiles o de sobremesa. Parece que, entre los jugadores, un 98% usan Windows. Ese ámbito del juego parece ser aquel en el cual peor parado queda el Linux en comparación con Windows, ya que únicamente en esta última plataforma están disponibles la mayoría de los juegos.

Si mis dispersas observaciones son de fiar, la mayoría de los linuxitas mantienen en sus computadores una partición Windows que utilizan —conjeturo— esencialmente para jugar. En mi opinión sería más acertado —acaso también ligeramente más arduo— usar, para ello, una máquina virtual Windows cargable con el VirtualBox.

Para mi propia actividad, resulta absolutamente irrelevante todo eso. No practico el juego en ninguno de mis dispositivos —y, francamente, jamás he sentido ni la más mínima afición lúdica. (No estoy con ello emitiendo juicio negativo alguno con relación a los ludófilos, tan libres de sus pasatiempos como lo soy yo de los míos.)

Tampoco me interesan las aplicaciones de Windows orientadas a la música, a la fotografía o a lo audiovisual.[61]

En resumen, haciéndome cargo de que puede resultarles satisfactoria a muchos usuarios su experiencia con Windows (ya sea por afición a los juegos, ya por razones profesionales, principalmente las atinentes a uso de aplicaciones de Adobe como el Photoshop), constato que la mía ha sido calamitosa y desesperante.

A nadie estoy predicando que se pase al Unix/Linux. Sólo estoy describiendo mi propia vivencia.

No por ello he renunciado a tener, dentro del Ubuntu —gracias al VirtualBox—, máquinas virtuales Windows (de uno u otro Windows). Cíñese, exclusivamente, su infrecuentísima utilización a los programas que ya he mencionado —más una versión gráfica del WordPerfect, el WordPerfect 2021, que he comprado (usándolo exclusivamente para conversiones).

* * *

---

[61]. Aunque tengo ahora instaladas en mi Lubuntu 22.04 aplicaciones audiovisuales, tal instalación ha sido reciente, ya que, en general, para tales menesteres resúltame hoy más cómodo y práctico usar mi tableta android —según ya lo he indicado *supra*, en la nota 35. (Años atrás, con Slackware, sí tuve y usé a menudo aplicaciones audiovisuales de varias índoles, con las cuales —entre otras tareas— no sólo convertí a formato digital el contenido musical de mis discos compactos y de vinilo sino que pude visionar no pocos vídeos y escuchar un número de ficheros audio en diversos formatos.)

¿Cómo me explico que haya tantísimos usuarios que, aun sin sentir acaso amor alguno por el Windows, se aferran a él? Por cuatro motivos.

El primero, es que les viene dado en las máquinas que han comprado, a lo cual se atienen; ni se interesan por posibles alternativas ni, aun conociéndolas, sienten apetencia alguna por cambiar.

El segundo motivo es la costumbre. Bastantes de los usuarios que instalan, desde un CD —físico o virtual—, un sistema operativo en su computador o que ensayan otro sistema operativo, como el Linux (del cual habrán oído hablar), regresan al Windows, porque están habituados.

Esa tendencia viene reforzada entre los no abundantes usuarios avanzados de Windows, o sea aquellos que no se han conformado con el manejo superficial de, meramente, picar en uno u otro icono del escritorio para ejecutar tal programa de un elenco limitado, el ofrecido por Micro$oft. En alguna medida son *nerds* del Windows. Al pasar al Linux, siéntense forasteros. Si únicamente quisieran picar en un icono para abrir el Office, el navegador de internet, el lector de correo electrónico, el reproductor de vídeos o de música, entonces resultaríales bastante similar al Windows —que es aquello a lo cual están habituados— cualquier distribución moderna del Linux (puesto que casi todas se han denodado por semejarse al Windows lo más posible, facilitando su adopción por el usuario normal de Micro$oft).

Mas justamente no sucede así a quien se maneja con mucho mayor soltura en Windows (sabiendo acceder a su CLI e impartir las órdenes adecuadas para una gama de resultados y, en general, siendo diestro en configurar, a su propio modo, el sistema operativo —aun en la escasísima medida en que Micro$oft se lo permite). Él, experto en Windows, se da de bruces con un sistema operativo muy diferente y que no le gusta, por lo cual casi siempre retorna a Windows, borrando el Linux de su máquina.

El tercer motivo es que únicamente para Windows está accesible una amplia gama de programas, lúdicos y serios.

Entre esos programas sólo accesibles bajo Windows figuran incluso algunos que yo mismo utilizo (así sea esporádicamente), como el WordPerfect gráfico. (Para ello válgome —ya lo he dicho— de una máquina virtual bajo el VirtualBox.) Ahora bien, en mi caso ese uso es tan marginal y accesorio que resulta superfluo (útil, mas innecesario).

Para muchos usuarios no existe en Linux alternativa a los programas que ellos usan en Windows. Así, p.ej., he leído y escuchado que en el escritorio de Windows puede uno (previa compra) instalar un programa idóneo para registrar y editar vídeos con una alta calidad profesional (colijo que se trata del Adobe Premiere Pro, estando también disponible otro más barato de la casa Corel).

Desconozco esos programas. Mis propios vídeos no sé cómo puntuarán en cuanto a calidad o profesionalidad. No obstante lo dicho en el capítulo VII,

no suelo valerme del computador para grabarlos, prefiriendo acudir a aplicaciones disponibles bajo Android, con mi tableta. (La superioridad óptica de los vídeos ajenos producidos en Windows no la niego, mas, a ojo, no he sido yo capaz de apreciarla —como no sea que en mis vídeos es más limitada la perspectiva de fondo, lo cual dudo que importe tantísimo tratándose de la exposición de comentarios, opiniones o argumentaciones, o sea de productos cuyo contenido es la palabra, no el panorama.)

De todos modos, existe varias aplicaciones de producción y edición de vídeos en Unix/Linux.[62]

Lo propio sucede con el Photoshop, también de Adobe, que usan, v.g., fotógrafos profesionales. Dicen que el GIMP de Linux, además de tener menores capacidades, resulta poco amigable.

Sólo que, insisto, esos programas —a los cuales no pueden renunciar sus respectivos usuarios— no forman, en absoluto, parte del sistema operativo Windows. Trátase de aplicaciones desarrolladas y vendidas por terceros (sobre todo por Adobe), no por Micro$oft. Como sistema operativo, el Windows es una birria.

El cuarto y último motivo es que hay *hardware* únicamente compatible con Windows. Es posible. En mi personal experiencia, únicamente me he enfrentado con incompatibilidades cuando he querido vincular una escrutadora (*scanner*) al computador. Hasta el punto de que tuve que devolver la última escrutadora que compré (cuyo anuncio falazmente afirmaba su soporte para Fedora y Ubuntu). Me resigné a otra, autónoma, sólo que de menor calidad, la cual produce PDFs sin necesidad de conectarse a ningún computador.

Me ha sorprendido que alegue un ventanero que con Linux no ha podido imprimir. Sin ningún problema —salvo el de hallar la configuración adecuada— he usado yo mis sucesivos Linux con impresoras de diversas marcas: Brother,

---

62. Entre ellas, Blender, DaVinci-Resolve, OpenShot, Shotcut, Flowblade, Lightworks y Kdenlive. (Parece darse un consenso de que esas aplicaciones son de menor calidad que el Adobe Premiere, indisponible en Unix/Linux.)

Algún peldaño por debajo en reputación figuran Cheese, OBS, Motion, Guvcview, Webcamoid y Camorama.

Por Hache o por Be, yo me he visto incapaz de utilizar todas esas aplicaciones. Las unas, por dificultades de instalación (quizá superables con mayor empeño). Las más, por ser programas que imitan a los propios del Windows, muy ventaneros, atiborrados de iconzuelos y complicadísimos menúes gráficos, resultando así de nula amigabilidad —al menos para mí, o sea para alguien acostumbrado al modo texto; colocado ante sus múltiples hileras y columnas de menúes icónicos, al usuario no habituado a esos entornos conviértesele en una carrera de obstáculos el más mínimo paso.

Al final me he quedado con Kamoso, que, a falta de otras cualidades, posee las de la sencillez y una cierta amigabilidad. Sin embargo, sigo prefiriendo grabar mis vídeos en mi tableta, con aplicaciones de Android con las cuales estoy familiarizado. (Vide *supra*, nota 36.)

HP, Xerox, Ricoh y varias más.[63]

* * *

Terminaré este capítulo confesando cómo me siento con relación a esa penosa experiencia del Windows 11. De haberla tenido gratis, hubiera sido, meramente, una de tantas actividades que emprende uno tentativamente, mas a la cual renuncia al no resultarle valedera (o, peor, al provocarle malestar ante unos elementos hostiles de escasa o nula utilidad que causan enojo).

Ahora bien, a mí me han forzado a pagar por cada una de esas tres aflictivas experiencias (y por la cuarta de 2014, ésta del Windows8). A pagar un dinero que me arrancaron contra mi voluntad. ¿Cuánto? Lo desconozco.

Consultada la página *web* de Micro$oft en 2024-07-07, veo que 259€ (más IVA, evidentemente) es el precio de la licencia para el Windows 11 Pro (el que venía preinstalado en cada uno de los tres nanocomputadores que he comprado en el último semestre).

Entiendo que, cuando Micro$oft contrata la preinstalación en un computador de su horrible sistema operativo —como OEM—, otorga al fabricante un descuento, que se repercute en el precio que paga el comprador final. Resulta que dos de los nano-pcs que he comprado me han costado menos que esa licencia.

¿Cuánto habría ahorrado si me hubiera sido posible adquirirlos sin pagar a Micro$oft licencia alguna por un producto que yo no quería? De los 199€ que me ha costado el Beelink MiniPC S12 Pro, de los 144 que me ha costado el MinisForum GK41 y de los 340 € que me costó el Mele Quieter3C N505, ¿podría haberme ahorrado la mitad —o al menos un tercio— sin el asalto de Micro$oft, cuyo nefasto producto no fue ni deseado ni bienvenido ni utilizado? ¡Quizá!

Me han esquilmado. ¿Que se trata de una suma de poco monto? ¡Cierto! Sólo que (a expensas de adquirentes muchos de ellos más pobres que yo, para quienes tal suma no es baladí) multiplícase ese despojo por millones de millares. Así se construye la fortuna de Bill Gates y su cuadrilla.

Cuán falaz es la legislación dizque antimonopolística en ambas orillas de la Mar Océana pruébalo el hecho de que —por un producto tan cochambroso e inseguro— siga sojuzgando a sus clientes un monopolio que tiene atrapados

---

[63]. Es tan larguísima la lista de las impresoras soportadas por CUPS —el programa de impresión que viene incorporado a mi actual distribución, Lubuntu 22.04—, que se extendería por muchas páginas su reproducción; sólo diré que, entre las muchas marcas de esa lista, hállanse: Zobra, UTAX, Toshiba, Sharp, Samsung, QMS, Panasonic, Olivetti, NEC, Mitsubishi, Minolta, Lexmark, Kyocera, Kodak, Apple, Canon, Compaq, DEC, Dell, Epson, Fujitsu, Fujifilm, Gestetner, Hitachi, Honeywll, HP, IBM, Xerox, etc; de cada una de esas casas, un montón de modelos.

Resúltame, por consiguiente, difícil de creer que haya alguna impresora de uso corriente que no esté soportada en Linux.

a más del 70% de los usuarios de computadores.

Recuerdo cómo, años atrás, había regateado con Micro$oft la comisión de Bruselas para que no privilegiara, frente a navegadores de la competencia, su propio programa *Internet Explorer*. Lo cual es fútil en comparación con el abuso, infinitamente mayor, de imponer el propio sistema operativo, no sólo frente al Linux, sino frente a otras alternativas disponibles (como Solaris, BSD y Chrome).

Comprendo que los fabricantes y mercaderes de *hardware* quieran proporcionar a los compradores un producto que funcione desde la primera hora de estar en el domicilio del adquirente. Sólo que habría otras soluciones. P.ej. que se brindara la opción de comprar sin sistema operativo; o la de adquirir el computador con otro sistema operativo (¡gratis!); o, en última instancia, la de que la mercancía viniera con Windows preinstalado, pero en versión gratuita, que cesaría de ser operativa al cabo, v.g., de dos semanas —a menos que, entre tanto, el comprador registrase y pagara su *software*.

En lo que quieren, los reguladores de Bruselas bien se las arreglan para vetar prácticas real o supuestamente obstaculizadoras de la competencia que perjudiquen a otros potentados del gran capital.

Sólo que del Linux no es valedor ninguno de esos potentados. (Aunque la IBM haya absorbido a RedHat, que produce una de las más célebres distribuciones de Linux, ese negocio está basado en el soporte, no en la venta —siendo, por lo demás, marginal dentro de ese conglomerado que es la IBM; además RedHat es una distribución orientada a servidores, no a compradores domésticos.)

Antes de concluir, he de mencionar que, en el momento de redactar el presente ensayo, no desconozco que en Inglaterra el comprador de un computador portátil Dell Inspiron, Mr. Dave Mitchell, de Sheffield, consiguió (en 2006) que el fabricante le devolviera 55 £ por un Windows XP-Home, preinstalado, que el adquirente no deseaba. En 2014 el Tribunal Supremo italiano (*Corte di Cassazione*) declaró ilegal la práctica de forzar a los compradores de un PC a pagar por un Windows preinstalado. Mas ¿cuándo se había interpuesto la demanda para obtener el reembolso? ¡En 2005! Al demandante, Marco Pieraccioli, la Corte ordenó devolverle la suma que comportaba la licencia de Micro$oft. Consideró el Tribunal que se trataba de una práctica monopolística.[64]

---

[64]. Ya en 1999, en un celebérrimo juicio en los Estados Unidos, el juez federal Thomas Penfield había condenado a Micro$oft por sus prácticas monopolísticas, transgresivas de la legislación *anti-trust*. Los entusiastas linuxitas creyeron erróneamente hallar en tal fallo una prohibición de la vinculación OEM, o sea de la venta ligada del sistema Windows con los computadores —de sobremesa o portátiles—, excluyendo a otros sistemas operativos.

Era leer mal la sentencia. Cierto es que en ésta no faltaba una alusión a la práctica de los OEM, pero el fallo no la declaraba ilícita:

> The firm charges different OEMs (original equipment manufacturers) different prices for Windows, depending on the degree to which the individual OEMs comply with

Por lo que he leído, resultan disuasorias, de lo onerosas que son, las condiciones para lograr la devolución de lo indebidamente pagado (suponiendo, además, que la reclamación pudiera prosperar en España, de lo cual no conozco precedente alguno). Además, en mi caso, los fabricantes están lejos (en la ciudad de Shenzhen) y la compra se realizó a través de Amazon.

Microsoft's wishes.

Ese fundamento fáctico no se traducía en ningún pronunciamiento normativo que cerrara el paso a la práctica de los OEMs (venta ligada). El único reproche era por discriminación, no por el procedimiento en sí del OEM, que efectivamente se ha perpetuado.

Capítulo XII

Unix, Linux y GNU

El UNIX es el gran sistema operativo que, desde comienzos de los años setenta del pasado siglo, viene usado para el manejo de grandes computadores. De ese uso inicial (su empleo como el sistema operativo de computadores de instituciones públicas y privadas y de grandes empresas) quiso pasarse a su uso para el manejo de los microcomputadores; escaso éxito tuvo ese tránsito, ya que pronto surgieron alternativas, aparentemente más sencillas, destinadas a usos mucho más modestos en máquinas de escasa potencia y limitadísimas prestaciones, por lo cual —sobre todo a medida que muchos microcomputadores se concibieron como «home computers»— adoptáronse para ellos sistemas operativos simplificados, cuya morfología y sintaxis divergieron de las del Unix.

Del principal de ellos, el CP/M, emanará el DOS (sistema operativo de disco), del cual se adueñará Micro$oft. (En capítulo XIII voy a hablar de ese asunto.)

Más abajo tendré ocasión de señalar algunas de las diferencias esenciales entre la morfología y la sintaxis del Unix (incluido el Linux) y la del DOS (en parte heredada por el actual Windows, si bien —tengo entendido— con hondas modificaciones; no hablaré de la sintaxis del Windows porque la desconozco).

Surge el Unix como fruto de una revolución informática en 1968. Hasta ese momento estaba el mundo de la electrónica dominado por las macrocomputadoras; éstas habían evolucionado con enorme rapidez. Si las de los años 40 y primeros 50 pesaban unas cuantas toneladas, ocupando enormes estancias, a comienzos de los 60 habíanse reemplazado esos mastodontes por computadores muchísimo menos voluminosos; fue la época de florecimiento de los *mainframes*, grandes armarios, conectados por cable, formando una intranet, con puestos de trabajo o terminales (*work stations*).

El máximo ingeniero electrónico de aquel período fue el hispano-estadounidense profesor del MIT y doctor en física Don Fernando José Corbató y Jensen (1926-2019) —oriundo del Levante español—, animador del grupo de trabajo e investigación MAC (*Mathematics and Computation*). La computación no era aún una disciplina aparte, estando considerada una rama de la

matemática y dominada por matemáticos.[65]

Donosamente ataviado, con su terno oscuro y su distinguido cuello de pajarita, el Prof. Corbató fue el padre del sistema operativo Multics, que solventaba la adaptación a la nueva fase, en la cual surgían dos problemas:

1º Cómo evitar los embotellamientos en la interfaz hombre/máquina —e.d. cómo transmitirle a la máquina instrucciones de modo que no se produjera hiato en su funcionamiento (cada hora de interacción costaba 600 dólares)—.

2º Cómo posibilitar que compartieran varios usuarios el acceso a la máquina.

Una primera solución la había proporcionado el recurso a cintas magnetográficas en las cuales habíanse grabado las sucesivas instrucciones emanantes de varios usuarios. El recorrido ininterrumpido de la cinta prevenía los embotellamientos, pero no resultaba idóneo para facilitar el acceso de varios usuarios.

De la genial mente de Corbató había brotado la idea del *time-sharing*, compartición del tiempo (o tiempo compartido). Fue también él quien había diseñado los terminales, programando en lenguaje de ensamblado (un lenguaje de bajo nivel relativamente cercano al lenguaje binario de la máquina). Sólo que aquellos primeros terminales eran máquinas de escribir. El usuario tecleaba sus instrucciones (inductos) en la máquina y ésta emitía sus eductos también tecleando automáticamente. El usuario veía ese educto leyendo el papel que iba desenrollando el carro de la máquina de escribir. (Acudíase también, claro, a la impresión, pero eso vino algo después.)

Inventáronse más tarde las pantallas, inicialmente pequeños visores de una o varias líneas, que ocupaban una parte del frontal de los terminales.

Surge así —especialmente diseñado para el computador GE-645— el sistema operativo Multics (*Multiplex Information and Computing Service*). De la mano de Corbató fue convirtiéndose en una enorme y complicadísima estructura, un sistema operativo de general aplicabilidad que serviría de «todo para todos» (*all for everybody*); quería solventar muchos problemas empleando con profusión amplios recursos, sin escatimarlos.

Pregúntome si, en parte al menos, la causa de que no alcanzara tan pleno éxito como lo habían esperado sus promotores no fueron las limitaciones del *hardware* entonces disponible, que no había avanzado con la misma rapidez del *software*.

---

[65]. V. «Fernando J. Corbató 1926-2019» por Jerome H. Saltzer. De esa fuente tomo los siguientes datos. Hijo del castellonense, emigrado a California, Hermenegildo Corbató (quien llegó a ser profesor de literatura española de la UCLA), Don Fernando José ha fallecido a los 93 años de edad el 12 de julio de 2019 en Massachusetts. Se había doctorado en física en 1956 sustentando su tesis «A calculation of the energy bands of the graphite crystal by means of the tight-binding method», bajo la supervisión de John Slater.

Hízose palmario que Multics podía conseguir sus objetivos, resultando, eso sí, un sistema enorme y no muy amigable.[66] Resultaban difíciles de leer por los no matemáticos sus programas —escritos en lenguaje de ensamblador, no en los lenguajes de programación de alto nivel, que por entonces empezaban su existencia.

El Multics lo estaba pilotando Corbató desde el MIT, mas esa institución académica no disponía para tal empresa de fondos suficientes. Hubo de asociarse con los laboratorios Bell y con General Electric (posteriormente Honeywell). Sin embargo, esas firmas juzgaron que estaban gastando en el proyecto demasiado dinero, por lo cual lo abandonaron.

El sistema Multics hubo aportado novedades que han permanecido hasta hoy, incluso en sistemas como DOS y Windows, alejadísimos de su espíritu —entre ellas la noción misma de fichero (*file*) como unidad mínima de almacenamiento y el árbol jerárquico de directorios (en lugar de que exista un único directorio que abarque todos los ficheros).

Además, el equipo de Corbató había contribuido al progreso de la electrónica con otros inventos, como el tratamiento de texto, el *cloud computing* y el correo electrónico. Varias de esas aportaciones, sin embargo, sólo alcanzarán su plena utilidad mucho después, cuando lo facilite la implementación de nuevo *hardware*, especialmente el PC y el internet. Corbató era un pionero por delante de su tiempo.

Según lo expone Jerome H. Saltzer (en el portal de la *National Academy of Engineering*) con el Multics Corbató se había adelantado a ideas posteriormente apropiadas por el sucesivo equipo Unics, como la de introducir las revisiones una por una, no simultáneamente. Si bien inicialmente Corbató había programamado en lenguaje de ensamblado, también le corresponde el honor de haber creado e impulsado lenguajes de alto nivel, desbrozando el camino para el uso del PL/I (*Programming Language One*).

De otro lado, aquella década de los 60 fue la del nacimiento de los minicomputadores. No se trataba ya de grandes máquinas del tamaño de un armario, sino de otras como consolas o aparadores (de hecho, ahí surgirá el empleo, en este campo, del vocablo «consola»). Incluso eran adosables a una mesa de trabajo. Resultaba ya posible reemplazar los terminales conectados a una máquina central por consolas independientes.

Desde los últimos años 60 surgieron nuevos sistemas operativos. Hasta qué punto eran derivaciones del Multics no lo sé. Hoy generalmente se han olvidado, porque, en su mayoría, estaban diseñados para un determinado *hardware*; *hardware* que pronto quedará obsoleto, llevándose al museo su

---

66. No obstante, las críticas al Multics puede que hayan sido forjadas (o desmesuradamente abultadas) por una leyenda negra, lanzada desde las firmas mercantiles que se habían desenganchado del proyecto, no juzgando rentable su inversión en él —con la anuencia de programadores que, legítimamente, estaban buscando nuevas vías, más sencillas y elegantes, o acaso inicialmente menos ambiciosas.

respectivo sistema operativo.

De ese defecto va a librarse, en cambio, el Unix, obra de hombres como Ken Thompson y Dennis Ritchie (inventor del lenguaje de programación C). Trabajaban para la Bell, habiendo quedado ociosos cuando esa firma hubo dejado en la estacada el proyecto Multics.

Eran informáticos de nuevo cuño, jóvenes, emancipados de la tutela de los matemáticos. Hasta en su atuendo y apariencia marcaron la diferencia, a tono con el espíritu de aquellos años: barbas y melenas desgreñadas, pantalones vaqueros, posturas no acordes con los viejos cánones de corrección.

Por antífrasis frente a «Multics», llamaron «Unics» a su nuevo sistema operativo, desarrollado en los laboratorios de Bell, no en un departamento universitario. (Al parecer fue Brian Kernighan el inventor del nombre.)

Inicialmente diseñaron el nuevo sistema para consolas independientes, como la PDP-11, con sólo 24 Kb de memoria, de los cuales únicamente 8 Kb quedaban disponibles para el usuario. Ningún fichero podía exceder los 64 Kbytes. Su disco era de medio Megabyte. Dadas esas limitaciones, apenas servía ese computador para nada más que el procesamiento de textos.

Trabajando con recursos tan sumamente mediocres (incluso para su tiempo) —siendo eso lo que les acordaba su empleador, que no pecaba de prodigalidad—, entiéndese que fuera inicialmente intención de Ritchie y Thompson abandonar cuanto les parecía excesivo y sobrecargado del Multics, incluso la multitarea, el acceso multiusuario y hasta la compartición del tiempo.

Cuatro fueron los principios inspirativos del nuevo sistema operativo:

1º, Portabilidad: producir un sistema operativo que pudiera correr en máquinas de diversa arquitectura —gracias a lo cual el Unix no ha fenecido como los demás sistemas operativos que fueron sus coetáneos.

2º, Claridad, sobriedad y simplicidad de los programas.

3º, Modularidad: en cada caso hacer una sola tarea mas hacerla bien.

4º, Complementariedad: alcanzar resultados de mayor envergadura, no acudiendo a grandes recursos, sino mediante la cooperación o la concatenación de recursos menores. (Uno de sus inventos fue el *pipe* o tubo, representado gráficamente por la barra vertical, «|», que sirve justamente para concatenar dos ejecutables haciendo que el educto del primero se convierta en inducto del segundo.)

En mayo de 1975 produjo y emitió la empresa Bell Labs, filial de la AT&T, la versión VI de Unix, facilitando a las Universidades (bajo demanda) la fuente (todavía no se tomaba la precaución de tratar esas fuentes como secretos comerciales).

Al año siguiente el instructor de ciencia de la computación de la New

South Wales University, el australiano John Lions (doctor en ingeniería de control por la Universidad de Cambridge), escribe un poligrafiado para sus alumnos, titulado *A Commentary on the Sixth Edition UnIX Operating System*. En 1977 lo publica a imprenta la Universidad, como libro de texto.

Sólo que en junio de 1979 la AT&T anuncia su Versión VIII del Unix, revocando el previo consentimiento para que en la Universidad se lea o estudie el código fuente de la versión precedente —ilegalizando así el libro de John Lions.

Enorme malestar y desasosiego va a causar esa revocación, creando un caldo de cultivo en el cual, poco después, surgirá (muy verosímilmente en parte como respuesta a esos hechos) el movimiento del *software* libre encabezado por Richard Stallman, al cual voy a referirme más abajo.

Tras esa prohibición, de efecto retroactivo, seguirá leyéndose y estudiándose clandestinamente el libro de Lions, circulando en forma fotocopiada y en seminarios nocturnos. Tan *best-seller* será que, cuando, finalmente, se autorice cuatro lustros después, se editará (ya con un ISBN) en una tirada de 15.000 ejemplares. Atravesaba el Unix su período de esplendor y apogeo.

En 1980 concede la Bell a sus dos empleados Thompson y Ritchie un año sabático en la Universidad de California en Berkeley, uno de los grandes polos del avance electrónico. Hallan en Berkeley un excelente foco de incubación y mejora de su sistema operativo, al encontrarse con su anfitrión Robert Samuel Fabry (profesor de ciencia de la computación), quien va a recibir una encomienda de la DARPA (agencia de investigación del departamento de defensa) con el fin de aplicar el UNIX a estructuras de computación distribuida.

Ciertamente había suscrito la Universidad un contrato de licencia con la AT&T-Bell, mas su propio Unix no era copia del de esa empresa, sino que —salvo algún detalle— tratábase de un producto intelectual novedoso y original: el sistema BSD, versión 4.1.

Y es que, cuando se discute si un sistema operativo (p.ej. el Linux) es, o no es Unix, conviene tener presente cuán esponjoso, flexible y fluido es el concepto del Unix.[67]

No es Unix lo que coincida con un códice depositado en algún archivo o lo que se ajuste a una determinada redacción canónica de cierto programa informático, sino todo aquello que siga unas pautas, compartiendo varios rasgos comunes, inspirándose en los mismos principios y ofreciéndole al usuario una experiencia similar, por el uso de idénticas o parecidas morfología y sintaxis en el lenguaje de interacción sistema/usuario.

Con entusiasmo y energía abrazaron pronto el nuevo sistema operativo

---

67. No ignoro que «Unix» es una marca registrada; prescindo por ahora de ese aspecto jurídico, al cual me referiré un poco más abajo.

Unix muchos departamentos universitarios.

Felizmente, fueron descartándose varias de las ideas de la génesis del Unix (al principio escrito «Unics»), retomándose —aunque no se confesara— las aportaciones del proyecto Multics. Al mirar hoy todo aquello con perspectiva histórica, es de justicia reconocer que mucho de lo conseguido en Unix no fue sino una reelaboración —ciertamente más elegante y mejor pergeñada— del Multics, del cual se reveló, a la postre, heredero y continuador. La ruptura había sido transitoria y superficial.[68]

Pese a la popularización del Unix en los años 70, las grandes empresas todavía siguieron usando otros sistemas operativos que juzgaban más sólidos y fiables. En la década siguiente alcanzaría el Unix la hegemonía de la cual sigue gozando a ocho lustros de distancia (hoy bajo la modalidad Linux, principalmente).

No estoy menospreciando las contribuciones de Ritchie, Thompson, Kernighan y sus muchos colaboradores. Conservan hoy su plena validez y vigencia los cuatro principios fundacionales del Unix. Contrariamente al nihilismo de algunos (los post-unixarios), Unix no está muerto. Vive, no sólo en Linux, sino también en las tres variantes gratuitas de BSD, en AIX de IBM, en Mach de NeXT y en varios sistemas de *freeware* o *free software*, como Solaris.

* * *

En los primeros tiempos de la electrónica, había existido escasa preocupación por la propiedad intelectual de los programas informáticos o de los binarios ejecutables resultantes de su compilación. Lo costoso era el *hardware*. No solían ser celosos de su obra los programadores, compartiéndola fácilmente con sus colegas. Entre el mundo académico y el industrial había cooperación.

Sin embargo, a medida que disminuía el precio del *hardware*, incrementóse relativamente el del *software* (insisto: relativamente al costo total); agravóse la competencia entre los fabricantes, menudeando los pleitos, con acusaciones de plagio. Pasa entonces a venir celosamente guardado ese *software* que, hasta bien entrados los años 70, había ido y venido sin que nadie estuviera obsesionado por atribuírselo como propiedad intelectual.

La jurisprudencia aclaró algo que estaba en duda, a saber: que los programas eran obras del espíritu, protegidas por la ley del *copyright*. Siéndolo, también lo eran los resultados de compilarlos, traduciéndolos a lenguaje-máquina, binario, con sólo 0 y 1. Sólo que los informáticos, hasta entonces, no se habían considerado escritores o autores.

Del Unix, en pleno auge e irradiación, quiso apropiarse la empresa AT&T, que incluso más tarde llegará a litigar (en 1992) contra la Universidad

---

68. Diríamos que una crisis de adolescencia, la «edad del pavo».

de California en Berkeley. Pretendía la AT&T que la Universidad únicamente había sido su comisionada o mandante, habiéndose excedido, según ella, al difundir su propia versión de Unix, BSD, lo cual conculcaría la licencia obtenida de la AT&T. La Universidad ganará el juicio (sacrificando únicamente una insignificante porción de su propia implementación del Unix).

La AT&T comercializó su propio UNIX System III, convertido posteriormente en System V, siendo el conflicto con Berkeley tan sólo una de la varias «guerras del Unix», batallas jurídicas donde todos peleaban contra todos: AT&T contra Sun (litigio saldado por una transacción en la cual Sun pagó 200 millones de dólares); SCO contra la IBM y, separadamente, contra otras compañías, entre ellas Novell y Red Hat (la demanda de SCO vino al final desestimada, aunque había sido admitida). Sin llegar a los tribunales, otras rivalidades opusieron a Sun con la IBM, quedando envueltas en esos enfrentamientos HP, DEC y otras.

En ese ambiente los programadores empiezan a ser desconfiados, cerrando con llave los cajones de sus mesas de trabajo al finalizar su jornada, generalmente ya de madrugada.

* * *

Frente a esos nuevos e incómodos aires álzase un puñado de jóvenes programadores, esforzándose por salvaguardar el viejo espíritu cooperativo. Encabézalos Richard Stallman, un neoyorquino nacido en marzo de 1953, interesado por la computación desde los 17 años de edad. Tras pasar por Harvard, había recalado en el MIT donde en 1983 funda el proyecto GNU, cuya denominación es un acrónimo recursivo, «**G**nu is **N**ot **U**nix».

No se crea lo que afirma tal denominación, la cual, en rigor, es falsa. El GNU es Unix. Sólo que no podía decir que lo era porque la AT&T reclamaba la propiedad de Unix. Conque hicieron un Unix que decía no serlo.

Hoy la marca «Unix» es propiedad del Open Goup, un consorcio cuyos miembros de platino son DXC, Fujitsu, Huawei, IBM, Intel, Open Text y la Shell. Ese consorcio hubo adquirido la titularidad de dicha marca registrada al transferírsela Novell, que se la había comprado a la AT&T. El Open Group, aun careciendo de ánimo de lucro, otorga el certificado Unix, que autoriza el uso de la marca, mediante una tasa y en función del criterio de adecuación al *Single Unix Specification*, consistente en un cúmulo de reglas sobre el interfaz de programación, el *shell* de la línea de órdenes y los mandos a disposición del usuario.

Corrobora totalmente ese criterio mi propia tesis sobre qué es y qué no es Unix. En ningún momento se dice que sea Unix lo que reproduzca un determinado código o comparta un núcleo fijado en el programa fuente.

No obstante, dado el monto de la tasa, han rehusado someter sus producciones a la certificación Unix del Open Group los desarrolladores de BSD

y de Linux.[69] Lo cual no obsta para que venga generalmente reconocido que BSD es Unix. Con idéntico fundamento, lo es Linux.

¿Era Unix el proyecto puesto en marcha por Stallman en 1983? A mi juicio, sí lo era. Siempre lo fue. No lo hubiera sido si por «Unix» hubiera que entender un determinado programa escrito en lenguaje C (o en cualquier otro), un texto canónico. Mas ¿qué regla semántica nos impone una definición tan estrechísima del sustantivo «Unix»?

En lugar de copiar el programa escrito (en C) por los originarios desarrolladores del Unix, el equipo GNU podía acudir a *reverse engineering*, una tarea computacional que se vale de la abducción (operación inferencial lógica inversa de la deducción): tratábase de conjeturar un programa cuya compilación produjera el mismo resultado que aquel sometido a tal ingeniería al revés. (Viene a ser una operación similar al desciframiento de un texto encriptado o de la grafía antigua de un idioma desconocido, como el Lineal A cretense o el etrusco.)

Sólo que, en este caso concreto, ni siquiera era menester acudir a la ingeniería inversa para averiguar la fuente, toda vez que en 1975 la propia AT&T-Bell había facilitado (bajo demanda, ciertamente, y únicamente para el mundo académico) la fuente de la versión VI del Unix.

Que pueden llamarse lícitamente «Unix» dos programas, escritos separada e independientemente el uno del otro, demuéstralo el propio fallo judicial (*US District Court for the District of New Jersey*, Sept. 8, 1993) que —en medio de tumultuosas controversias— zanjó el juicio entre la AT&T y la Universidad de Berkeley.

El GNU no ha sido copiado, como tampoco lo fue el BSD. Si llamamos, apropiada y correctamente, «Unix» a la sintaxis y morfología accesibles al usuario y a la estructura del sistema (su árbol de directorios) —junto con el funcionamiento general del sistema operativo según todas las pautas del Unix, incluyendo su acervo esencial de mandos en el CLI—, GNU es Unix; tan Unix como cualquier otro Unix. Linux no es, por consiguiente, un sistema operativo unixoide, sino que es Unix, igual que lo son BSD (con sus variantes y derivadas), Mach, SunOs/Solaris, HP-UX o AIX.

Aquello que individúa a un sistema operativo es su funcionamiento, su relación funcional con los aparatos y con los usuarios. No lo subyacente a ese funcionamiento. Determinante de la identidad de un sistema operativo es la descripción definida del mismo según resulte accesible al usuario y a los observadores. El mismo sistema operativo puede ser superveniente sobre varias programaciones generativas —igual que los procesos mentales pueden ser supervenientes sobre una pluralidad de hechos somáticos.

A mi modo de ver, quienes reservan la denominación «Unix» al originario

---

69. Con una excepción, la distribución de Linux EulerOS —si bien no renovó su certificación en 2022.

de Thompson & Ritchie, no sólo olvidan que éste se había bifurcado en ramificadas variantes —pronto enzarzadas en reyerta, disputándose la marca—, sino que, además, es radicalmente errónea la semántica bautismal de Putnam & Kripke, según la cual habría un algo subyacente, referible por un acto radical de arbitraria denominación, que, de una vez para siempre, fijaría la denotación del nombre en cuestión, independientemente de cualesquiera cualidades observables o relaciones funcionales entre el ente así denominado y los demás.

Resulta ilusoria e inservible esa teoría pura de la denotación. Cualquier denotación útil para la comunicación lingüística ha de basarse en algo que los hablantes puedan compartir, no en un mágico remitirse a unos míticos bautizadores originarios y a una ficticia cadena de transmisión denominativa intergeneracional. Todo eso es fábula. Lo único que pueden compartir los hablantes de un idioma es una descripción funcional.

Con la precisión de que, más que de una particular descripción definida, trátase de un racimo difuso de descripciones funcionales. No existe un cúmulo preciso, fijado para siempre, de condiciones necesarias y suficientes para que un sistema operativo sea Unix. Hay un elenco de rasgos que es difuso, fluido y oscilante. Además, hay grados. Un sistema operativo puede ser más Unix que otro.[70]

* * *

Para que tengamos un sistema operativo es menester una suma de dos elementos: un núcleo o kernel más un cúmulo de utilidades. El equipo GNU estaba produciendo ese cúmulo, pero va a empantanarse con el desarrollo de su planeado kernel, el HURD, por disputas y devaneos acerca de los términos legales; y es que, en ese lapso, Stallman había desarrollado la ideología liberista, a la que me referiré después, abrazándola con intransigencia, igual que lo ha seguido haciendo desde entonces.

Causará efectos perversos tal inflexibilidad, siendo uno de ellos no haber podido disponer a tiempo de un kernel con el cual la obra de GNU pudiera fructificar en una auténtico sistema operativo.

Ahora bien, en agosto de 1991 el estudiante sueco-finlandés Linus Torvalds había producido la primera versión del kernel Linux. Púsola en seguida bajo la licencia GPL de Richard Stallman, teniendo así lugar una confluencia entre: por un lado, ese núcleo o kernel (desde entonces siempre desarrollado bajo la —directa o indirecta— supervisión de Linus Torvalds); y, por otro lado, la suma de *software* del GNU.

Stallman siempre ha rechazado llamar al sistema resultante «Linux», denominación que estaría ninguneando al GNU. (Ha insistido en llamarlo «GNU/Linux» o «GNU+Linux» o «Lignux».)

---

70. Mucho me temo que la deriva actual de no pocas distribuciones del Linux lleva a nuestro amado sistema a ser menos Unix que antes; refiérome en particular al desgraciado *systemd*, del cual hablaré en el capítulo XXI.

Sólo que la lengua la hace el uso. Etimológicamente es incorrecto llamar a Francia «Francia», país de los francos, cuando hubiera sido más apropiado conservar la denominación de «Galia» —aun sin coincidencia precisa de las fronteras, igual que las de la Italia moderna no son las de la Italia de Julio César, que no se extendía al norte del Rubicón.

El Linux se llama «Linux»: 1º, porque así lo llaman los hablantes, en particular sus desarrolladores y sus usuarios; 2º, por ser poco felices las denominaciones alternativas (concretamente las propuestas por Stallman y su equipo), careciendo de sonoridad y de percutancia; y 3º, porque el bisílabo «Linux» suena como una transliteración de «Unix», sólo que con la «L» delante.

He de reconocer, no obstante, que a mí no acaba de gustarme la palabra «Linux», reprochándole su consonante inicial, que sobra. Debería llamarse «Inux», pronunciado «ínux». (En español, además, «el Linux» suena igual que «el Ínux».) Denominarlo «Inux» sería el mejor modo de manifestar que es Unix.

Es la «L» un tributo personal que no tiene su razón de ser. Sin menospreciar la individual aportación de Linus Torvalds, el Linux no es creación suya. Como tampoco es creación de Richard Stallman. Es el resultado de miles de contribuciones y confluencias. Sobre todo, por ser un clon de Unix —adaptado y vuelto a generar—, es, indirectamente, creación de cuantos trabajaron por construir, primero, el Multics y, después, su criatura, el Unix.

¿Cree alguien que, cuando se jubilen Torvalds y Stallman, se acabará el Linux? Anda equivocado quien piense así.

Por loables que sean los motivos de Richard Stallman, resulta molestísima la denominación «GNU», precisamente por el implícito aserto de que **NO** es Unix.

Sin necesidad de que demos el brazo a torcer ante Stallman renunciando a nuestro bonito y eufónico vocablo «Linux» (ya que sé de sobra que proponer «Inux» me llevaría a luchar contra los molinos de viento), no por ello cabe desconocer que, sin el GNU, no habría ni hay Linux.[71]

---

[71]. Inmensa ha sido la obra del equipo GNU. De tal multitud de programas, cabe destacar, como más sobresalientes, estos 21: Bison (generador de analizadores sintácticos); Bash (intérprete de órdenes); Bbf (archivos de librerías); Binutils (ensamblador y enlazador GNU más utilllajes afines); Classpath (librerías para Java); Emacs (editor de texto); GCC (compilador); GDB (depurador de aplicaciones); GNU Ghostscript (cúmulo de aplicaciones para PostScript y PDF); GIMP (edición de imágenes); Glibc (librería para el lenguaje C); GMP (librería para cálculos); GNOME (entorno de escritorio gráfico); GNUnet (software para redes); GNUstep (cúmulo de librerías OpenStep); GNU MDK (utillaje para programar en MIX); GSL (librería científica); Guix (gestor de paquetes); Gzip (herramienta de compresión de ficheros); Octave (programa de computación numérica); Texinfo (programa de documentación).

Es mucho más vasto el elenco de librerías y utilidades desarrollado por el equipo GNU. Sin pretensión alguna de exhaustividad, mencionaré unas cuantas: GNU 3DLDF, a2ps, Acct, Acm, Adns, Alive, Anastasis, Anubis, APL, Archimedes, Aris, Artanis, Aspell, AUCTeX, Autoconf, Autoconf-archive, AutoGen, Automake, libavl, Barcode, Bash, Bayonne, Bazaar, Bc, Behistun, BFD, Bool, BPEL2oWFN, C-graph, ccAudio, Ccd2cue, Ccide, ccRTP,

No sé cuánto de todo lo muchísimo producido por GNU figure en nuestras distribuciones de Linux. ¿Menos de la mitad? ¡Da igual! Con el kernel de Linux a palo seco no hay sistema operativo.

Claro que es posible combinar ese kernel con otro *software* que no sea el de GNU. Linus Torvalds afirma que es Linux cualquier sistema cuyo núcleo sea el del Linux; mas eso es un serio error.[72]

El Chrome de Google y el Android no son Linux, en absoluto. La experiencia del usuario de Android no se parece en nada a la del usuario del Unix, incluido el Linux. Siendo tantas y tan visibles las diferencias entre el Android y cualquier distribución del Linux, ni siquiera merece la pena explayarse en ellas. Quienes (para dispositivos diversos) usamos ambos sistemas operativos, Android y Linux, bien sabemos hasta qué punto resultan divergentes e incompatibles, siendo menester acudir a métodos de reelaboración y conversión cuando tenemos que pasar contenidos del uno al otro.[73]

---

ccScript, Cflow, Cgicc, Cim, CLISP, Combine, CommonC++, Complexity, Config, Coreutils, Cpio, Cppi, CSSC, Cursynth, Dap, Datamash, DDD, ddrescue, DejaGnu, Denemo, Dia, Dico, Diction, Diffutils, Direvent, DotGNU, Dr.Geo, Easejs, Ed, EDMA, Electric, EMMS, Enscript, Fdisk, Ferret, Findutils, FisicaLab, Foliot, Fontopia, Fontutils, FreeFont, FreeIPMI, Freetalk, FriBiDi, G-Golf, Gama, garpd, Gawk, Gcal, Gcide, GCL (GNUCommonLisp), GCompris, Gdbm, Gengen, Gengetopt, Gettext, Gforth, Ggradebook, GIFT, Glean, Global, GLPK, Gmediaserver, Gnash, GNATS, GNU-pw-mgr, Gnuastro, GNUbatch, GNUbiff, GNUBoot, GNUcap, GnuCash, GnuCOBOL, GnuComm, GnuDOS, GNUInteractiveTools, GNUlib, GNUmed, GNUmeric, GNUmp3d, GnuPG, GNUprologjava, GNURadio, GNUschool, GNUsound, GNUspool, GnuTLs, GNUzilla, Goptical, Gorm, Gpaint, Gperf, Gprofng-gui, Gprolog, Greg, Grep, Gretl, Groff, GRUB, Gsasl, Gsegrafix, GNU Scientific Library, GNUSlip, GNUSource Release Collection, Generic Security Service, GTick, Gtypist, Guile, Guile-cv, Guile-dbi, Guile-ncurses, Guile-opengl, Guile-SDL, Gurgle, Gv, GVPE, GWL, Gxmessage, HaliFAX, Health, Help2man, Hp2xx, Httptunnel, Hyperbole, Icecat, Idutils, Ignuit, Indent, Inetutils, Inklingreader, InklingReader, Intlfonts, JACAL, Jami, Java-getopt, Jel, Jitter, Jwhois, Kawa, Less, GNUCLibrary, Libcdio, Libdbh, LibertyEiffel, libextractor, Libgcrypt, Libiconv, Libidn, Libidn2, Libmatheval, Libmicrohttpd, LibreJS, Libsigsegv, Libtasn1, Libtool, Libunistring, Libxmi, Lightning, Lsh, M4, MACChanger, Make, MARST, MARST, Maverik, Midnight Commander, Mcron, MCSim, Mediagoblin, MemPool, Mes, Metahtml, Mifluz, MIT/GNUScheme, Moe, MPC, Mpfr, Mpria, Mtools, Nana, Nano, Ncurses, Nettle, Ocrad, Oleo, oSIP, Panorama, Parallel, Parted, Patch, Paxutils, PCB, Pexec, Pies, Plotutils, poke, Proxyknife, PSPP, Psychosynth, Pth, R, Radius, RCS, Readline, Recutils, Reftex, Remotecontrol, Rottlog, Sather, SCM, Screen, Sed, Serveez, Sharutils, Shepherd, Shishi, Shmm, Shtool, SLIB, Smalltalk, SpaceChart, Spell, Src-highlite, Stow, Stump, Superopt, Swbis, Tar, Termcap, Termutils, Teseq, TeXmacs, Time, TRAMP, Trueprint, Unifont, Units, UnRTF, Userv, UUCP, Vc-dwim, VCDImager, WB, Wdiff, *web*socket4j, Wget, Which, Xlogmaster, Xnee, Xorriso.

[72]. Error garrafal que desgraciadamente pone de relieve uno de los lados no encomiables de la idiosincrasia del gran programador nórdico-norteamericano: un creciente egocentrismo, que desemboca en confundir el kernel con el sistema operativo, lo cual lleva a menospreciar la aportación del GNU (y de otros miles de contribuidores a las múltiples distribuciones). Un kernel mondo y lirondo no es un sistema operativo.

[73]. A menudo un fichero que nos hemos bajado del internet con una tableta Android ni siquiera podemos copiarlo a un computador con Linux sin haberle previamente cambiado la denominación, al resultar incompatibles los criterios denominativos de ambos sistemas operativos. Cuando, tras varias manipulaciones, hemos logrado realizar la copia, el fichero resultante tiene —igual que aquellos que proceden de Windows— el atributo de ejecutable, aunque sea un libro, un documento PDF o un audio.

Una distribución, Alpine Linux, no contiene las librerías del GNU. Habiendo apenas probado esa distribución, no estuve con ella más de diez minutos, al haberme causado tal perplejidad que en seguida desistí de usarla, pasando a otra cosa. Al fin y al cabo, llámese como se llame, no es Linux. Insisto: sin GNU, no existe el Linux. Lo que carezca totalmente del *software* GNU no será Linux; como tampoco lo será un sistema con ese *software* mas con otro núcleo o kernel —como el HURD, un proyecto que lleva decenios gestándose, sin haberse abandonado.

De prosperar finalmente el núcleo o kernel HURD, al cabo de decenios de frustración, constituiría el GNU su propio sistema operativo, con lo cual tendríamos dos sistemas operativos diversos que compartirían, no sólo su estructura (la del Unix), sino también mucho de su *software* esencial, sólo que con diferentes núcleos —igual que hoy ya tenemos cuatro sistemas operativos que comparten el mismo núcleo (Linux, Chrome, Android y Honor).

El usuario de un aparato (no *rooteado*) con Android no tiene acceso a varios directorios esenciales que penden del directorio raíz ni le es posible enseñorearse de su máquina (salvo por el arriesgadísimo y casi rocambolesco procedimiento del *rooteo*).

Va a empeorar la relación entre el Android y sus usuarios la llegada de la versión decimoquinta de ese sistema operativo, la cual ni siquiera va a permitir correr aplicaciones no expresamente autorizadas por el productor del sistema, Alphabet (Google), o no bajadas de la tienda Google Play.

El Android es un buen sistema operativo, pero su manejo se parece al de Windows, donde al usuario se le permite escaso margen de libertad. Nada de CLI, nada de compilar, nada de instalar lo que uno quiera. Y todo en modo gráfico —aunque el ratón venga aquí reemplazado por los dedos.

Capítulo XIII

Los computadores domésticos y el surgimiento del DOS

En los capítulos VI y XII hemos tenido ocasión de hablar de la evolución de los computadores, desde aquellos gigantes de los años 40 a los minicomputadores de los 60/70. Ahora toca referirnos a un paso adelante, que fue la creación de los microcomputadores, aparatos mucho más pequeños (y bastante menos caros), transportables (*luggable*), colocables sobre una mesa (de ahí su denominación de «computadores de sobremesa», que ha pervivido).

Para ubicar en su contexto ese nuevo invento de los años 70, hemos de relacionarlo con las funciones o tareas asignadas a los computadores.

Si la función crea al órgano, también sucede a menudo lo inverso. No siempre se sabe de antemano qué funciones podrá desenvolver un nuevo aparato ni, por lo tanto, para qué servirá; o no del todo. (Si al diseñarlo no se tuviera la menor idea de su posible utilidad, a nadie se le ocurriría concebirlo ni fabricarlo; sin embargo, es frecuente que, pensado inicialmente para una función particular, el invento la sobrepase en seguida, revelándose capaz de servir para otras tareas.)

Nadie ignora que la primera función de los macrocomputadores del primer período (los años 40) era exclusivamente militar, en varias facetas: desde el espionaje (y, más especialmente, la criptografía) hasta los cálculos atinentes a la producción y al uso de nuevas armas, entre ellas las bombas nucleares, pasando por otros cálculos de trayectoria, resistencia de materiales y, en suma, implementación de la ingeniería militar, sobre todo la naval. (La Armada estadounidense siempre fue una poderosa y eficaz impulsora de la investigación computacional.)

En el siguiente decenio, los computadores van a convertirse en auxiliares de las grandes obras públicas y de los censos, o sea servidores de la ingeniería en grandísima escala y gestores de esos enormes acervos de datos que son los censos. Usábanlos también grandes departamentos universitarios de matemáticas (entre ellos el MIT) como instrumento para solventar complejos problemas que hasta entonces habían permanecido irresueltos.

Entre finales de los 50 y mediados de los 60 van a agregarse nuevas funciones, sobre todo con la creación de los minicomputadores. Vendrán adoptados por los grandes bancos (sumergidos por sus acervos de datos, que

ya resultaban inmanejables) y por algunas compañías de aviación. Es dudoso que la navegación aérea hubiera podido experimentar el vertiginoso crecimiento de aquellos años si no hubiera recibido el apoyo de los computadores.

En 1970 había salido a las pantallas cinematográficas la película de Elio Petro (que fue galardonada con varios premios) «Indagine su un cittadino al di sopra de ogni sospetto». Manifestó al público cómo las policías (al menos aquellas de los países ricos que habían sabido invertir en la explotación de las nuevas técnicas) poseían, gracias a los computadores, enormes bases de datos en las cuales todos estábamos fichados y que, sobre todo, por mecanismos de cruce de datos, permitían averiguar muchísimas cosas. La vida privada había dejado de serlo. Al *panópticon* de los computadores policiales pocas cosas se le escapaban.

Mas el manejo de acervos de datos interesaba también a muchos otros establecimientos. En la segunda mitad de ese decenio (los años 70) empieza la utilización de computadores en los ámbitos biblioteconómico y universitario —para la elaboración de concordancias y, en general, para el manejo y hallazgo de bibliografías pertinentes—, todo lo cual resultaba utilísimo para la actividad académica, desde la redacción de tesis y monografías hasta el más avanzado trabajo de investigación. Sin la computación no habría sido posible el desarrollo de la labor científica de los últimos decenios, cuando, en determinadas disciplinas, puede duplicarse en un año la suma de conocimientos.

* * *

Retornando a aquel período de los años 70, cabe precisar que, por entonces, apenas estaba en sus inicios esa adopción de los computadores en el ámbito académico.

Con los microcomputadores van a extenderse las funciones: una gama mucho más amplia de cálculos (y, por consiguiente, de aplicaciones a la ingeniería y a la técnica en general), procesamiento de acervos de datos (ya incluso los de un tamaño modesto), los primeros programas de tratamiento de texto e impresión (al comienzo destinados a facilitarles a los programadores informáticos la ardua tarea de escribir sus programas, justamente cuando estaban floreciendo los primeros lenguajes de alto nivel). Y también los primeros juegos (al comienzo tratábase de ejercicios matemáticos con carácter lúdico).

Percatáronse los fabricantes de microcomputadores del potencial público que tendrían en la vida profesional y empresarial, recomendando su compra y utilización por firmas comerciales, establecimientos públicos y privados, bufetes de abogados, clínicas, todos los cuales estaban desbordados por sus bases de datos, ya difíciles de manejar con los viejos ficheros. No obstante, el elevado precio de tales aparatos disuadía a muchos posibles compradores.

El incremento del tamaño del mercado y simultáneamente los avances técnicos fueron permitiendo una lenta bajada de los precios.

* * *

Húboseles ocurrido a esos fabricantes que, si el microcomputador había accedido al despacho de abogados y de notarios, a la consulta de médicos, a las oficinas de gestoría y otras, podía también franquear el umbral doméstico, entrando en los hogares. Fue vista esa perspectiva con gran escepticismo. Los magnates de la computación, como Xerox e IBM, concebían su línea de negocio como algo serio, dudando que el computador tuviera futuro alguno en la vida de las familias. Un mandamás de la IBM había afirmado que a nadie se le ocurriría tener un computador en su casa.

Otros fabricantes fueron más optimistas en sus previsiones (que pudieron ser corazonadas, porque nadie era capaz de demostrar nada al respecto). El hecho es que proliferaron los computadores vendidos como «home computers» y destinados a las familias.

Uno de los muchos fabricantes fue Digital Research Inc., una empresa al comienzo modestísima montada por Gary Kildall y su esposa. Gary Kildall (profesor asociado de ciencia de la computación en la escuela naval de posgrado de Monterey, California) había creado, en 1973, su propio sistema operativo, el CP/M (*Control Program for Microcomputers*). Lo va a licenciar a muchos otros fabricantes de microcomputadores, entre los cuales cabe citar: Xerox, Gnat Computers, Osborne, Kaypro, MSX, Commodore, TRs, Amstrad, Altair e IMSAI.

No había brotado el CP/M de la nada en el cerebro de Kildall, quien se había inspirado en varias fuentes, entre ellas el sistema operativo TOPS-10 que corría en el computador DECsystem-10, un *mainframe*, así como el sistema CP/CMS de la IBM. Kildall escribió en su propio lenguaje de programación, el PL/M (creado en virtud de un contrato con Intel).

Nótese que la locución entonces empleada no era la de «personal computer», sino «home computer». Sin embargo, no todos los fabricantes compraron la licencia de CP/M; entre los recalcitrantes estaba la tejana Tandy, que, habiendo adquirido la cadena comercial RadioShack, jugaba con ventaja. Su propio microcomputador de 1977, el TRS-80, se denomina «personal computer». Si bien, privilegiadamente, Tandy, Apple y Commodore van a recibir una gran propulsión publicitaria, en el *software* para microcomputadores la hegemonía siguió, indiscutiblemente, correspondiendo a CP/M y a Digital Research.

Aquellos primeros computadores hogareños (o familiares) sufrían todavía serios defectos. Uno de ellos era el teclado. Alguno de aquellos teclados únicamente escribía en mayúsculas; además no eran muy fiables. Otro defecto era la pantalla. Aun habiéndose revelado insatisfactoria la solución de ofrecer una micropantalla de pocas líneas en el mismo frontal de la máquina, resultaba demasiado caro agregar un monitor, por lo cual algunos de aquellos aparatos se vendieron para conectarse al televisor familiar mediante un adaptador de radiofrecuencia.

Otro problema era la grabación. Durante muchos años el medio de grabación de ficheros había sido la cinta magnetográfica; en los

microcomputadores se adopta la *cassette*, tan popular en aquellos años para audición de música y grabación de emisiones radiofónicas.

* * *

La IBM había desdeñado toda esa tendencia pensando que, tratárase o no de una moda pasajera, para su propia línea de negocio poco interés presentaba fabricar computadores de hogar. No por ello menospreciaba el futuro de los microcomputadores, mas aquellos que había sacado al mercado iban orientados a las oficinas. Tales fueron el Datamaster, con un microprocesador 8085, al precio aproximado de 10.000 dólares y previamente el IBM 6580 Displaywriter, con un programa incorporado de procesamiento de texto, el Textpack (y que corría bajo el sistema operativo CP/M de Kildall); la IBM lo alquilaba por 270 dólares mensuales.

No resultaba serio anticipar que los consumidores particulares quisieran hacer un desembolso —al principio nada desdeñable— para una utilidad que no se percibía bien; ni, fuera de ello lo que fuere, se antojaba tal línea muy acorde con la profesionalidad de una empresa de la solera y la reputación de la IBM, caracterizada por sus máquinas de coste y prestigio elevados.

Fue, dentro de la IBM, un grupo de originales (William Lowe y Philip Estridge) el que había porfiado con los mandamases para que les permitieran entrar en esa nueva línea de negocio, lo cual se les acabará consintiendo a cambio de imponerles el brevísimo plazo de un año para coronar su misión. Va a trabajar con ahínco ese equipo en Boca Ratón, en la Florida, para cumplir a tiempo el cometido.

Acudirán a la adquisición de un sistema operativo no elaborado por los propios informáticos de IBM. Son tres los motivos para esta decisión. Primero, la propia compañía toleraba ese ramal sin brindarle ni su aliento ni su apoyo —con lo cual no mucho podía esperarse de los empleados de la firma. Segundo motivo: de hecho, al estar a punto de agotarse su plazo, el equipo de Boca Ratón no tenía aún un sistema operativo apropiado para la nueva máquina. Y tercer motivo: a la sazón la IBM estaba envuelta en pleitos por plagio; adquiriendo un *software* producido por otra entidad, a ésta incumbiría hacer frente a futuros litigios sobre propiedad intelectual.

En septiembre de 1981 Digital Research había vendido —directamente— más de 260.000 licencias de su sistema operativo CP/M, cifra a la cual habrían de agregarse las sublicencias. Muchas compañías fabricaban microcomputadores vendidos con ese sistema operativo CP/M. Habíase impuesto tal sistema como el estándar (aunque ya sabemos que ni Tandy ni Apple se habían subido al carro; cada una de ellas usaba su propio sistema operativo, en lo cual iban contra la corriente).

Conque a Digital Research (y personalmente a Gary Kildall) dirigióse, precipitadamente, el equipo «personal computer» de la IBM (que se apropiará de esa locución como si fuera su propia marca, desbancando la anterior denominación «home computer»). Al no haberse llegado a un acuerdo (en una

célebre entrevista, envuelta en la leyenda), gracias a un enchufe familiar (concretamente materno), el joven Bill Gates, de la pequeña firma Micro$oft, recibe el encargo de licenciar a la IBM un sistema operativo para el nuevo PC.

Sólo que Gates no tenía ninguno. Contacta a su conocido Tim Paterson, de una modesta firma local de Seattle (su ciudad natal en el Estado de Washington), Seattle Computer Products, SCP. Para los aparatos que vendía en ese estrecho mercado, usaba la SCP un sistema operativo, el QDOS, también llamado «86-DOS», escrito por el propio Paterson.

Parece hoy demostrado (en cualquier caso ése ha sido el fallo judicial y *res judicata pro ueritate habetur*) que ese QDOS era, en gran medida, un clon del CP/M.

Con una jugada rayana en la estafa, Gates compra el QDOS, viniendo recompensado Paterson con un empleo en Micro$oft, en cuyas oficinas se va a pulir el QDOS, redenominándolo «MS-DOS». Poco antes de sonar las campanadas, Gates licencia a la IBM ese sistema clonado, mas con la expresa condición de que la licencia no sea exclusiva.

Cumplióse el plazo. El 12 de agosto de 1981 tuvo lugar, en el Hotel Waldorf Astoria de Manhattan, el anuncio del primer IBM-PC, al precio de 1.565 dólares por un equipo que comprendía un monitor fabricado en el Japón y un teclado, mas sin ningún dispositivo de almacenamiento.

No había disco duro, mas, adicionalmente, podía adquirirse un complemento por 3.000 dólares; comprendía un par de disqueteras (para grabar en *floppies*) —junto con una tarjeta controladora insertada en una ranura de expansión— más una impresora de agujas (*dot-matrix*) Epson. (También se podía conectar el aparato con una grabadora externa de *cassette*.)

Es más, el IBM-PC se vendía sin sistema operativo a menos que el comprador sumara 40 dólares más por la licencia del MS-DOS. (Temiendo una demanda judicial de Kildall —que de todos modos se produjo—, la IBM accede a ofrecer como alternativa el CP/M, mas al precio de 240 dólares.)

Salvo por su mejor teclado, el primitivo IBM-PC era peor y más caro que varios aparatos de la competencia. (Es un mito el cuento de que la IBM inventó el computador personal; ni siquiera le corresponde la paternidad de la denominación.)

* * *

Desafiando las expectativas, la IBM en seguida estaba vendiendo, cada mes, mil centenares de sus PCs, obteniendo con ello un beneficio de un miliardo de dólares en un año. (Parece enorme, mas hay que relativizar la significación de esa cifra: en 1981 la IBM había ganado 29 miliardos. El PC no iba a ser lo que enriqueciera a la compañía, que no lo necesitaba.)

Tres años después, en 1984, la IBM sacará al mercado un nuevo modelo, mejoradísimo, el AT, con disquetera incorporada de 1'2MB, con o sin

disco duro de 20 MB (opcional), con 256 KB de memoria y ocho ranuras de expansión.

Pese a ese éxito comercial, pronto las cosas se le van a torcer a la IBM, al salir a la venta los clónicos de otras firmas, empezando por Compaq, HP, Epson y Commodore, sin contar las marcas blancas. Si se habían quedado en la cuneta varios de los anteriores vendedores de «home computers», van a prosperar, en cambio, esos nuevos rivales, los clónicos. La IBM no se había preparado bien para hacer frente a esa nueva competencia ni mediante la propiedad industrial (patentes y marcas) ni insertando cláusulas prohibitivas de la ingeniería inversa en su licencia de uso.

¿Habíanse equivocado los agoreros al presagiar un fracaso comercial del computador hogareño? En cierto modo habían acertado. El IBM-PC y sus clónicos se vendieron por millones, sólo que principalmente a oficinas. Hasta los noventa el IBM-PC apenas entrará en los domicilios privados. Tardaron no pocos años en conquistar los hogares, donde, con preferencia, seguían vendiéndose microcomputadores orientados a los juegos, como Atari y Commodore.

No obstante, produjose un factor que —no me cabe duda— ayudó a que un número de familias adquiriese un PC: en aquellos años 80, no sólo se produce una explosión numérica de la juventud universitaria, sino que también se generaliza evaluar a los alumnos y estudiantes mediante la asignación de *coursework*, —las monografías o ensayos—, incluso en la enseñanza media. Pareciendo hoy consustancial a la enseñanza, constituía entonces una novedad ese sistema de evaluación, conduciendo a que muchas familias adquiriesen máquinas de escribir; puestos a ello, podían dar el paso adelante de optar por un PC y una impresora —sólo que, claro, hasta entrados los años 90 resultaba tan cara esa compra que únicamente se decidían a ella las familias económicamente holgadas.

El costo era demasiado alto. Un dólar de 1981 equivale a 3'46 dólares de hoy, con el resultado de que el PC de 1981 —sin disquetera ni ningún otro dispositivo de almacenamiento— valdría 5.415 dólares, cerca de un millón de pesetas.

Y además ¿para qué usos? Podía comprarse el procesador de texto WordStar (ya previamente disponible para el sistema CP/M), pero, según lo he señalado, ni la mayoría de las familias usaban máquina de escribir ni, además, pudieron, así a la ligera, pasar de la máquina de escribir portátil al computador.

Otro uso era un juego de ficción interactiva, Microsoft Adventure. Entre el *software* adquirible (a pagar adicionalmente) figuraban el programa de cálculo VisiCalc, el Lotus 1-2-3, el Flight Simulator y el dBASE (también previamente soportado por CP/M).

Por otro lado, el PC permitía, gracias a su puerto serial, conectarse a una red, mediante el protocolo TCP/IP, a través de un modem (aparato externo

modulador/desmodulador) adosado a una línea telefónica, formándose así, en seguida, el incipiente internet. (Púsose a la venta en 1982 por 700 dólares el Hayes Smartmodem 1200.)

No hay que confundir el internet con la *web*; ésta sólo surgirá con el protocolo http, pero desde hacía años existía el internet con sus protocolos telnet (creado en 1973), ftp (creado en 1971 y adaptado al protocolo TCP/IP en 1980), SMTP (para el email, también creado en 1981) y kermit (creado en la Columbia University en ese mismo año de 1981).

Ahora bien, en aquellos años 80/90 eran muy minoritarios los usuarios hogareños del internet a través del modem. Sólo después, con la creación de la WWW (*world wide web*) y la introducción de la versión 3 del lenguaje HTML en marzo de 1995, empezarán a generalizarse las páginas *web* escritas en ese lenguaje. Hasta 1995/96 pocos conocían esa incipiente telaraña o tenían acceso a ella.[74]

El nuevo PC —junto con esas aplicaciones que he reseñado— resultaba valioso para oficinas, públicas o privadas, para bibliotecas, museos, instituciones educativas, académicas y culturales, para investigadores y doctorandos, para escritores, para el trabajo académico, para los negocios modestos que no podían comprar minicomputadores; y desde luego hubo algunas familias que también adquirieron un PC, pero no muchas hasta finales de la década de los 90, cuando el IBM-PC empezará a popularizarse gracias al internet-*web*.

---

[74]. De todos modos resultaba engorroso el uso del Modem; serán la banda ancha, el cable coaxial, el ADSL y, principalmente, la fibra óptica —posteriormente también el 3G— los que inciten a la gente a acceder al internet.

Capítulo XIV

Las diferencias entre Unix y DOS

No ha pasado de ser una inevitable digresión cuanto, en el anterior capítulo, he dicho sobre la evolución del *hardware*. Céntrase en el *software* mi interés. ¿Qué sucedía con el sistema operativo?

Ya he dicho que la IBM había ofrecido su PC con el MS-DOS preinstalado (ella lo había comercializado como PC-DOS). Era mendaz la oferta alternativa del CP/M (dada la diferencia de precio, sin que, al parecer, se hubiera advertido de ella a Kildall); impúsose *de facto* el monopolio del PC-DOS=MSDOS.

Los fabricantes de clónicos se las arreglarán con Micro$oft para obtener licencias, ya que el contrato entre M$ y la IBM era no-exclusivo. (Mas para ellos la licencia sí será un contrato exclusivo, comprometiéndose a que todas sus máquinas se vendieran con el sistema operativo de Micro$oft.)

¿Cuáles eran la sintaxis y la morfología del DOS? Sustancialmente, las del CP/M.

Al adaptarse a máquinas de escasísima potencia y de poquísima capacidad de almacenamiento —las cuales, por añadidura, estaban aquejadas por deficiencias del teclado y de la impresión—, introdujo Kildall la simplificación de descartar la letra minúscula. Sólo se usarían las mayúsculas. Aunque, en su evolución posterior, nuevas versiones del DOS y de sus descendientes hayan permitido la minúscula, son, todos ellos, sistemas *case-insensitive*, lo cual significa que se consideran idénticos cualesquiera dos ficheros cuyos nombres únicamente difieran por el distingo entre mayúsculas y minúsculas. (Si existe un fichero «Fichero.xyz», no puede haber otro que se llame «fichero.xyz». Notemos que esa indiferenciación también afecta al Android —que es una de las muchas razones por las cuales Android no es, en absoluto, Linux.)

Otra innovación de Kildall (heredada en el DOS) fue la limitación de los nombres de fichero al patrón 8+3: ocho caracteres alfanuméricos para el nombre propiamente dicho más tres para la desinencia. De hecho la noción misma de desinencia parece haberla inventado Kildall.

La significación de las desinencias (o sufijos) de tres caracteres estriba

en que sirven para clasificar los ficheros.

En Unix lo que determina la clasificación de un fichero es el cúmulo de sus atributos. Con el atributo «x» un fichero es ejecutable. En el CP/M-DOS, esa determinación viene dada por la desinencia: son ejecutables los ficheros con una de las desinencias «com», «exe» o «bat». Mas otras desinencias también resultan pertinentes para que los programas tengan acceso a los ficheros. P.ej. los ficheros con desinencias «ovl» y «dll» son módulos que cargan en memoria, complementariamente, elementos accesoria o ulteriormente agregados a un determinado fichero ejecutable (*.exe), sin estar directamente incorporados al mismo para evitar que, en el momento de comenzar su ejecución, rebase la memoria disponible. (Son similares a las librerías del Unix, sólo que éstas no se reservan a un ejecutable, siendo compartidas.)

Consiéntame el lector una digresión. La noción de desinencia, no sólo ha pervivido en Windows, sino que incluso ha contagiado a Linux. Tiende a resultarme hoy absurda, al haberse implementado (en el sistema de ficheros FAT32 o vfat) los nombres de fichero largos —y hasta con espacios intercalados. Es como si para denominar a las calles, en lugar de decir algo como «calle del General Francisco Salazar, nº 11», dijérase «General Francisco Salazar, nº 11.cal». Hubiera sido más lógico cambiar la sintaxis, v.g. acudiendo a llaves y prefijando la clasificación fuera de su alcance (v.g.. «Documento{informe sobre los acontecimientos}», en lugar de «informe sobre los acontecimientos. doc», lo cual suena ridículo).

Por otro lado, es dudosamente acertado introducir espacios en blanco o tabuladores dentro de los nombres de fichero. Puede usarse para eso el guión bajo,«_». De manera general, considero más aconsejable usar en tales nombres únicamente signos ASCII estándar, para así soslayar los constantes problemas de convertibilidad (o, peor, de inconvertibilidad) entre nombres de fichero: 1º, escritos en ASCII, con la página IBM, 437; 2º, escritos en ANSI; 3º, escritos en mapa de caracteres de Windows CP850; y 4º, finalmente —según se hace ahora— en UTF8 (Unicode).

Varias veces me ha sucedido que, al copiar un fichero de un dispositivo a otro, o a otra máquina, lo copiado resulta, no ya totalmente ilegible, sino también imborrable. (En Unix/Linux es posible usar, en un nombre de fichero, cualquier signo que no sea NULL ni «/», incluso un salto de página o un retorno de carro. ¿Para qué podría servir el uso de denominaciones tan extrañas? Pues, v.g., para ocultar un fichero, dificultando su acceso.)

De manera general la noción misma de desinencia únicamente se justificaba para aquellas maquinitas de los años ochenta, incapaces de desbordar la limitación 8+3. En sistemas de ficheros que permiten nombres de fichero comenzados por punto, ¿tomaremos como desinencia todo el nombre sin ese punto? (¿Y si empiezan por un punto seguido de otro? ¿Cuál será la desinencia del fichero «..verdades»?) ¿Cómo tratar las desinencias de desinencias, p.ej. con un fichero «abcdexyzuvw.efgh.jkl.mnst»? Por más que se haya implementado ese concepto de desinencia en el Bash de Linux —sirviendo

para reemplazos automáticos— y que se use en el *mailcap* para invocar el programa idóneo a cada caso, llévame a conjeturar que sería preferible otro procedimiento clasificatorio lo problemático —a menudo, cuestionable— de ese concepto de desinencia.

Como ex-usuario del OS/2, pienso que fue una valiosa idea de ese sistema operativo la de los atributos extendidos, que asociaban a cada fichero con unos metadatos. Sobre el papel el mismo método existe en Unix/Linux, el de los atributos extendidos. Sólo que apenas se usa, estando poco y mal implementado, carente de regulación (al no venir apenas pautado por el estándar POSIX —que hoy, además, osan poner en tela de juicio los adalides del malhadado *systemd*); y, peor que eso, al resultar escasísimamente conocido y utilizado por los usuarios.

Cierro la digresión, pasando a considerar la tercera novedad de Kildall (así como, en pos de él, del CP/M-DOS): el uso de las letras A, B, C, D, … seguidas del signo de dos-puntos, «:», para designar a los conductos, o sea dispositivos de almacenamiento. El «A:» y el «B:» están reservados a los disquetes o *floppies*; el «C:» al disco duro —o a la primera partición del mismo. Cuando, más adelante, se configuren los discos duros dividiéndolos, para formatearlos, en varias particiones, vendrán éstas designadas como «D:», «E:», etc.

Posteriormente, habrá que reservar nuevas letras al agregarse los dispositivos externos —los de CDRom y los de SCSI (como los .zip y los .jaz), así como, desde la versión 1.1 del protocolo USB en agosto de 1998, los dispositivos USB. (No sé cuáles versiones del DOS les den soporte; posiblemente algunas del DR-DOS de Caldera. No así la última versión del MS-DOS [sin Windows, o sea sin entorno gráfico], la 6.22, emitida en 1994. Hoy el USB nos da la impresión de habernos acompañado toda la vida, mas sólo se ha generalizado hace veintimuypocos años.)

Marca una ruptura radical con el Unix ese uso de una letra (A, B, C, …) seguida del signo «:» para denominar a un conducto (a un dispositivo, interno o externo, de almacenamiento); en Unix todo es un fichero.

Para el Unix los directorios son ficheros, unidades mínimas de almacenamiento: los ficheros anidados en un directorio no son subunidades del mismo, ni partes del mismo, sino unidades independientes de almacenamiento, unidas al directorio por una relación externa, la de ubicación.

También los dispositivos son ficheros, Todos los recursos son ficheros, sean dispositivos enteros, sean particiones; sean internos o sean externos. (Incluso una impresora o un escrutador vienen considerados como ficheros.)

Digamos que el CP/M-DOS es pluricategorial mientras que el Unix/Linux es unicategorial, lo cual quiere decir que en CP/M-DOS no tiene sentido asignar a un conducto propiedades o atributos de un fichero o directorio, mientras que en Unix/Linux sí lo tiene. (Otra cosa es que tal asignación sea verídica o no lo sea.)

Cuarta novedad, el separador entre directorios, subdirectorios y ficheros es la barra inclinada inversa, «\», mientras que en UNIX es la directa, «/». En CP/M-DOS úsase ésta última para los parámetros de una instrucción dada en el CLI, mientras que tales parámetros introdúcense en Unix con el guión «-».

He de precisar, no obstante, que Kildall inicialmente había renunciado al árbol de directorios del Unix (a la jerarquía de ficheros), dadas las limitaciones físicas de los microcomputadores; en el CP/M y en el MS-DOS versión 1, todos los ficheros estaban al mismo nivel, en un solo y único directorio, sin jerarquía.[75]

Quinta novedad: el intérprete de órdenes, al cual Kildall había llamado «CCP» (*Console Command Processor*) y Paterson/Gates llamará «COMMAND.COM». En el Unix emítense las instrucciones en el CLI a través de una concha (*shell*), existiendo varias conchas, alternativas entre sí, siendo la más difundida el Bash (u otra anterior más simple, el mero SH).

Su papel no es similar al del COMMAND.COM en el DOS, que es un interfaz obligado entre los programas ejecutados por el usuario y aquellos que corren automáticamente al lanzarse el núcleo del sistema operativo, mientras que ninguna concha del Unix es imprescindible.

Sexta novedad de Kildall: el abandono del sistema de directorios del Unix. No podía ser de otra manera cuando el CP/M y el MS-DOS 1 renunciaron a la jerarquía o el anidamiento de directorios y ficheros, amontonando todos los ficheros en un único directorio.

Aunque desde el MS-DOS versión 2 reintrodújose la jerarquía de directorios, no fue ya para nada la del Unix/Linux, en la cual el árbol no es arbitrario, sino ajustado a una pauta de coherencia, sirviendo para clasificar los ficheros según su función. En el DOS no hay regla alguna ni de agrupamiento ni de denominación.

La séptima y última novedad de Kildall (y, en pos de él, del MS-DOS) estriba en la morfología y la sintaxis de las instrucciones emitidas en el CLI.

Hay algunas instrucciones que se han tomado del Unix (aunque mayusculizándolas, evidentemente) —v.g., «at» para posponer una tarea—. Muchas otras se han modificado o alterado. P.ej., en lugar de «ls» —que pide al CLI decirnos cuáles son los ficheros en el directorio activado—, en CP/M-DOS la orden es «DIR»; posiblemente porque inicialmente habría un solo y único directorio.

Por un afán de claridad, en vez de «cp» de Unix, el DOS usa «copy» y «xcopy»; en vez de «cmp», «comp»; en vez de «mv», «move»; en vez de «rm», «del»; en vez de «cat», «type»; en vez de «grep», «find» (mientras que la orden «find» desempeña en Unix/Linux una función totalmente diversa). Al revés el «clear»

---

[75]. Kildall, empero, restableció, en una versión posterior de su sistema, la función multiusuario, que de algún modo podía servir para hacer las veces de una pluralidad de directorios, mas no jerarquizados, no anidados.

del Unix se ha reemplazado por «cls» en el DOS; el «rmdir» por «rd» (aunque también se usa en el DOS «rmdir»); «mkdir», por «md».

Algunas instrucciones han cambiado de significado. En Unix «date» sirve para que el sistema diga en qué día se encuentra —si bien, con parámetros adicionales, puede modificar esa datación, que es la principal función que esa orden tiene en el DOS. (Sin embargo, con parámetros diversos, ambas órdenes pueden cumplir la misma función, a salvo de algunas diferencias secundarias; con la precisión de que en el DOS se espera que el reloj de la máquina sea el de la hora legal, no el del huso horario.) De otro lado, «ver» pregunta a la máquina la versión de su sistema operativo, una función similar a la del mando «uname -v» del Unix.

La orden «ren/rename» del DOS no existe en Unix, donde se usa «mv»; sin embargo, sí existen en Linux utilidades de esa denominación, que, sobre el mando *mv*, ofrecen la ventaja de que no se va a borrar el fichero afectado por un error de escritura o un despiste, sino sólo a renombrar.

Algunas órdenes inspiradas en las del Unix juegan un papel diverso en CP/M-DOS, v.g. «attrib», dada la índole radicalmente diferente de los atributos de fichero entre el DOS y el Unix

Hay órdenes del Unix que, naturalmente, se han eliminado, como «ln», porque en CP/M-DOS no hay *links* (que en Unix permiten que un mismo fichero figure con varios nombres e incluso ubicado en varios directorios, o sea accesible por varias vías).

En cambio, se han agregado otros mandos como «append», «assign», «subst», «chcp», «chkdsk», «diskcomp», «diskcopy», «driverquery», «mode», «sys» y «format». (En el unix existe otra orden para formatear un dispositivo, «mkfs», con variantes según el sistema de ficheros escogido.)

En esta enumeración he omitido aquellos mandos que únicamente se emplean en el interior de *scripts* batch (correspondientes a los «sh» del Unix), sobre bucles, variables, condicionales, iteraciones, etc. Igualmente he pasado por alto las órdenes cuya incorporación fue posterior, al reemplazarse el DOS por el Windows, incluyendo aquellas que invocan funciones de red o de conexión a internet, ya que justamente todo eso estaba faltando en el DOS.

En resumen claramente se había inspirado Kildall (como indirectamente lo hará, en pos de él, su clónico de Micro$oft, el MSDOS) en la sintaxis y la morfología del Unix, tratando de simplificarlas y adaptarlas a máquinas menos potentes, más limitadas y rudimentarias, y a usuarios de menores conocimientos o menor aplicación.

Con todo, para el usuario del CLI en Unix/Linux, trabajar en el CLI del DOS no resulta nada exótico, sino una tarea familiar, muy parecida a la suya, lo mismo que escribir ficheros de instrucciones *batch* (o *btm*) en vez de sus acostumbrados «**.sh*»

La experiencia del viejo unixita, acostumbrado a la interfaz de texto

—sin entorno gráfico ni ratón—, es similar a la del usuario del DOS de los años 80/90 (a salvo, claro, de las diferencias).

Para mí, en concreto, no resultó perturbador pasar de la multitarea según la venía practicando yo con el OS/2, en modo texto, a la multitarea del Unix/Linux —la del comienzo, antes de verme forzado al entorno gráfico—, abriendo simultáneamente doce consolas y saltando de la una a la otra mediante una *hotkey* (Alt+Ctl+1,2,…,12).

Antes del OS/2, en el DOS, lo único que había conseguido era la conmutación de tareas, que no es lo mismo; en la conmutación, sólo se ejecuta una única tarea, congelándose cualesquiera otras hasta que vuelvan al primer plano. (Constituía eso ya un gran avance con relación a las primeras versiones del DOS, cuando, para abrir el WordPerfect, había que haber cerrado cualesquiera otros programas, como el Database, el PCTools o cualquier otro.)

Capítulo XV

La insuperable permanencia del DOS mediante el DOSEMU

¿Por qué me ocupo tanto del DOS? ¿No pertenece al pasado? Para mí, no, en absoluto. ¡Todo lo contrario! El DOS no está periclitado ni rivaliza con el Unix/Linux, siendo su hermano menor y su indispensable ayudante.

Redúcese mi hodierno uso del DOS al del Dosemu. ¿Forma parte el Dosemu del sistema operativo Linux? Si no me equivoco, el Dosemu puede ejecutarse en otros sistemas de la familia Unix, como Solaris y los varios derivados del BSD (Berkeley *software* Distribution).

Durante años el Dosemu formaba parte del *software* directamente incorporado a varias distribuciones de Linux, p.ej. Debian.[76] Hoy ya no.

Me ha resultado difícil instalarlo en mi presente distribución de Lubuntu (la 22.04.4). Y es que, desgraciadamente, al Dosemu —denigrado como obsoleto— lo han desechado los repositorios de Ubuntu —y, a menos que esté yo equivocado, los de casi todas las distribuciones, aunque siga presente y actualizado en la distribución rusa Alt-Linux.

No deja, empero, de ser el Dosemu una aplicación pensada y diseñada para el Linux, que, hasta hace pocos años, hallábase en los repositorios creados por los productores y desarrolladores del Linux, ofrecida gratis a los usuarios. Sin ser, pues —del todo y en sentido estricto—, una parte del propio sistema operativo, tampoco es ajena al mismo, teniendo más bien un estatuto intermedio entre estar dentro y estar fuera.

Un equipo de la distribución Alt-Linux, integrado por los dos programadores Grigory Batalov y Michael Shigorin, ha venido actualizando el Dosemu por lo menos hasta el 8 de julio de 2019.[77] Peca de un defecto, no obstante, su meritoria labor: la versión actualizada que amablemente nos

---

[76]. En cambio, Slackware nunca lo incluyó, ya fuera porque no todo el contenido del Dosemu caía bajo la licencia oficial del Linux, la GPL de GNU, ya fuera porque Slackware era muy del Unix —sin concesiones a emulantes de otros sistemas operativos, tratárase del DOS, tratárase del Windows. (Por eso tampoco incluyó jamás el Wine.)

[77]. Versión 050405-alt4, tarea #234104. V. https://git.altlinux.org/srpms/d/dosemu-freedos.git.

ofrecen únicamente permite abrir el Dosemu —que, por lo demás, funciona perfectamente— en una ventana, no en pantalla completa —lo cual resulta particularmente molesto e irritante cuando se trabaja con un monitor portable.[78]

No son los rusos los únicos que han seguido trabajando en el Dosemu. Entre 2012 y 2020 también lo han hecho otros siete ingenieros informáticos: Evangelos Foutras, Felix Yan, Maxim Baz, Levente Polyakz, Sergej Pupykin, Balló György y Laurent Carlier.

Mi actual versión de Dosemu es la 3.6.24. Me la bajé, como paquete Debian (no específicamente para Ubuntu), del sitio *sourceforge.net*. También descargué la fuente, para compilarla estáticamente; fracasó tal compilación, faltándome las librerías idóneas.

* * *

Ha llegado el momento hacer una digresión sobre las librerías. En Unix/Linux, cuando el usuario imparte una orden a la máquina, ejecuta ésta una tarea, corriendo un fichero binario —un ejecutable. Al cargarse en memoria, si es muy voluminoso, existe un riesgo de que desborde esa memoria en el momento de iniciarse la ejecución. Para evitarlo, acúdese a las librerías, o sea a módulos coejecutables, pautadamente invocables por aquellas aplicaciones que están corriendo, según las circunstancias. Una misma librería es así invocable por diversos ejecutables.

Notemos que en el DOS existe un procedimiento parecido. En ese sistema el problema es mucho mayor, dadas las serias limitaciones de memoria del DOS (entre otras el límite de 640 Kb de memoria base). Acúdese entonces a esos módulos invocables, los ficheros OVL (*overlay*) y DLL (*dynamic link library*); sólo que en el DOS cada ejecutable tiene sus propios DLLs y OVLs, sin compartirlos.[79]

La desventaja de actuar así —invocando, según los requerimientos de la tarea, módulos externos coejecutables, o sea librerías— es que, a consecuencia de esa estructura, una actualización del sistema operativo que conlleve una renovación de librerías podrá acarrear —y de hecho frecuentísimamente acarrea— que dejen de funcionar ciertos ejecutables, habiendo también de venir renovados por versiones actualizadas. De no

---

78. Uno, en mi caso, de 16 pulgadas de diagonal, con una razón de 10×16.

Quiero aclarar, no obstante, que no puedo excluir la posibilidad de —con pericia— hacer funcionar esa versión del Dosemu en pantalla completa; sólo que yo no lo he conseguido.

79. Posee su propio módulo externo mi programa favorito del DOS, el WordPerfect 5.1. Siendo el binario ejecutable el WP.EXE, su librería aneja es WP.FIL. Gracias a ella podía el WordPerfect lanzarse sin desbordar la memoria disponible, invocando posteriormente el auxilio de esa librería únicamente a medida que lo fuera necesitando. Y así sigue funcionando, si bien hoy programa y librería resultan de un tamaño exiguo, dadas las actuales disponibilidades tanto de memoria cuanto de espacio en disco.

haberlas, tendrán que reemplazarse por otros programas con la misma función. No siempre los hay.

Acudióse, para hacer frente a esa dificultad, a compilar ciertos ejecutables estáticamente. Compilar un programa es someter el programa propiamente dicho —salido de la pluma de quien lo ha escrito (en un lenguaje de programación de alto nivel)— a un proceso de ensamblado y compilación en sentido estricto, cuyo resultado es un fichero en lenguaje binario (lenguaje-máquina), de 1s y 0s, que es aquel que, al venir instruida, ejecuta la máquina.

Pues bien, puede hacerse tal compilación estática o dinámicamente. La compilación dinámica no embute las librerías en el ejecutable, sino que habilita a éste a invocarlas según los requerimientos de su tarea, mientras esté corriendo. La compilación estática, en cambio, produce unos ejecutables mucho más abultados, a veces gigantescos, en cuyo seno contienen réplicas o reproducciones de las necesarias librerías. Un binario estáticamente compilado es (o eso se espera) ejecutable en nuevas versiones del sistema operativo (y en diversas distribuciones de Linux).[80]

En aquel período, hoy ya lejano, en el cual solía yo compilar no pocos de los ejecutables que usaba en Slackware-Linux —entre el 1998 y 2014—, rara vez tomé la precaución de compilar estáticamente; con el paso de los años y de las actualizaciones me fui haciendo consciente de esa necesidad, justamente al toparme con que, tras una actualización, un determinado programa ya no podía correr, por incompatibilidad de librerías.

Me bajé entonces una aplicación, *statifier*, que precisamente servía para convertir en estáticamente compilados ejecutables compilados dinámicamente. Sólo que pocas veces funcionaba. (Aun así conservo algunos ejecutables que por entonces estatifiqué con esa aplicación.)

También supe de otra aplicación más eficaz, sólo que de pago, el Ermine. No la adquirí, pues no estaba públicamente anunciado su precio, habiendo que indagarlo por un mensaje de email.[81]

Al dar ahora, en 2024, el considerable salto de Slackware 14.1 a Lubuntu 22.04.4, temblaba por la utilizabilidad o no de mis más caros programas, particularmente el Dosemu.

Confirmando mis temores, no resultó compatible con las nuevas librerías el Dosemu que venía usando yo desde hacía bastantes años en Slackware 14.1. Sí, en cambio, la versión, ya mencionada —la 3.6.24—, que me he bajado (compilada como paquete Debian) de *sourceforge.net*.

No sólo está el Dosemu ausente de todos los repositorios de Ubuntu,

---

80. Posiblemente también en otros Unixes, como los tres derivados «libres» de BSD.

81. En realidad ahora sé que, caso por caso, viene convenido el precio mediante negociación. (Negociar, ya se sabe, implica regatear.)

sino también de los PPA —*personal package archives*, repositorios no oficiales, surgidos por iniciativas particulares, pero que gozan de un cierto reconocimiento en la «comunidad» de usuarios de Ubuntu.[82]

Tras haberme instalado, con éxito, dentro de Lubuntu 22.04, el Dosemu 3.6.24, me ha asaltado esta inquietud: si me viera un día forzado a actualizar el sistema operativo Lubuntu, ese programa, mi Dosemu, dejaría posiblemente de correr, por incompatibilidad de librerías.

Para afrontar tal eventualidad, he adquirido, pagando, una licencia del citado Ermine (válida durante un mes exactamente). Habiéndome ajustado a sus instrucciones, tengo ahora un Dosemu estáticamente compilado (o eso me creí). Es un ejecutable muy voluminoso, lo cual hoy no es problemático (con una maquinita que tiene 16 Gb de memoria). (Puede que, en vez de abrirse en un segundo, se abra en dos o tres segundos.)

Lamentablemente, no resulta esa compilación tan estática como esperaba yo. Habiéndola ensayado en una reciente versión de Lubuntu, la 24.04 —que, a título de prueba, lancé en una máquina virtual con VirtualBox—, he constatado que mi Dosemu «erminiado», dizque estático, no lo es del todo. No puede correr en Lubuntu 24.04.

Siendo mi versión de Ubuntu la 22.04, de abril de 2022, ¿por qué no instalé la 24.04, que acababa de lanzarse cuando configuré mi nueva máquina, precisamente el pasado mes de abril? El motivo es que, al emitirse la 24.04, miré unos vídeos donde se alertaba acerca de cierta inmadurez o precipitación de la versión recién salida del horno, aconsejándose esperar a una próxima revisión.[83] Pensé que la 22.04 sería más segura.

Percátome de cuán acertada fue mi opción, ya que con la versión más reciente resultaría imposible seguir corriendo el Dosemu, lo cual sería una tragedia sin paliativos para mi uso del computador —centrado en mi trabajo con el WordPerfect 5.1.

Yo no he menester de lo más novedoso, lo más al día. Yéndome bien con la 22.04, como efectivamente me va, no quiero ascender a ninguna versión posterior —salvo que algún día sobreviniera un imprevisible percance. En tal eventualidad (que para mí sería un siniestro), habría de estrujar nuevamente mi cerebro, poniendo a prueba mi tesón, a fin de encontrar alguna solución al

---

82. Su lugar ha venido suplantado por un pretendiente cuyos méritos desconozco, el autodenominado «Dosemu2», para mí deletéreo. Han conducido siempre a tener la máquina colgada —con mis aplicaciones brutalmente matadas— mis pruebas con él —y eso habiendo seguido atentísimamente, con el máximo esmero, todas sus instrucciones, sin omitir ni una sola. Tras una obstinada reiteración de los intentos (cada vez con mayor cuidado, si cabe), siempre ha resultado un calamitoso fracaso. Naturalmente ese pseudo-Dosemu2 lo he borrado; no tengo la menor intención de volver a probarlo, pareciéndome un *software*, no digo maligno, pero sí pésimamente diseñado y, desde luego, vitando.

83. Tal revisión ya la ha ofrecido Canonical; es la 24.4.1, difundida a fines de agosto de este año, 2024.

problema de la incompatibilidad de librerías. (De momento la única opción que se me ocurre sería ejecutar algunos de mis programas en máquinas virtuales a través del emulador VirtualBox de Oracle.)[84]

* * *

Gracias al Dosemu podemos en Linux correr el WordPerfect 5.1. Lo cual entiendo que para muchos resultará irrelevante. Para el autor del presente ensayo, es, en cambio, indispensable e insustituible, al estar su uso del computador únicamente orientado a su trabajo; ese trabajo, a la escritura; esa escritura, a un programa con estas diecisiete capacidades:

01) Utilizar para el texto toda la pantalla, del borde izquierdo al derecho, sin márgenes ni interposiciones de ninguna clase (sin notificaciones, sin hileras o «barras», ni arriba, ni abajo ni a ningún lado).

02) Disfrutar del máximo contraste y del mayor aprovechamiento del color (no del dibujo), con cromática plena y claramente diferenciada para los diversos atributos de letra (romana, itálica, negrilla, grande, subescrita, subrayada, versalita, manuscrita, bastardilla, etc), con colorido que resalte, vivo (si se quiere, chillón).

03) Manejar el texto en formato borrador —sin gráficos— (no en el malhadado e imperspicuo WYSIWYG).[85]

04) Tener habilitada la función de revelar códigos, al alcance de una pulsación de tecla.[86]

---

[84]. Dejo esos ensayos como tarea pendiente para cuando me encuentre más desahogado.

[85]. De mis objeciones al WYSIWYG (explicadas en mi artículo «Why I am not a wordist», *op.cit.*), la principal es que con el WYSIWYG esfúmase la diferencia entre escribir un manuscrito/borrador y pasarlo a limpio.

Al escribir yo un texto tecleándolo, hágolo igual que se escribe un manuscrito sobre el papel, con muchísimas abreviaturas —y sin cuidar ni la ortografía ni el distingo entre mayúsculas y minúsculas—. Para transformar ese texto en un formato presentable acudo a dos sucesivos procedimientos: el primero es, mientras estoy tecleando, usar mis macros de escritura (pulsando una tecla de atajo en lugar de la palabra o locución deseada). Aplico posteriormente el segundo procedimiento: una vez concluido el manuscrito (o una parte sustancial del mismo), encárgase una nueva macro de transformar en texto presentable (neto) lo hasta ese momento tecleado, que todavía era texto bruto.

Sin embargo, aún será menester, ulteriormente, realizar varias operaciones para dar al texto uno u otro formato final —pudiendo, evidentemente, un mismo documento recibir diversos formatos para diferentes entregas del mismo.

[86]. Únicamente el WordPerfect (en sus diversas y sucesivas versiones) ha implementado la función de mostrar códigos, para mí esencialísima y absolutamente indispensable. Gracias a ella podemos, p.ej., percatarnos de cuándo hemos dejado sin cerrar un código de negrilla —para no llevarlo a rastras allende el límite deseado; o ver si se ha agazapado un tabulador en medio de una línea, lo cual distorsiona el texto; etc. En mi configuración del WordPerfect 5.1 actívase esa función con una sola pulsación de la tecla <F11> y se desactiva del mismo modo.

Subsiste la función de revelar códigos en el WordPerfect 2021 —un programa gráfico que corre únicamente bajo Windows (W10 o W11)—, si bien, en general, trátase de una versión muchísimo menos amigable

05) Marcar la fuente (el tipo de letra) separada e independientemente del atributo, de suerte que no vengan afectadas las marcas de atributo por un ulterior reformateo del texto que altere la fuente (pasando, p.ej., de courier a garamond).

06) Escribir con atajos, de suerte que cualquiera de los vocablos más comunes de nuestro idioma (o de otros dos) aparezca en la pantalla al pulsar tan sólo una combinación de dos o tres letras, sin haber tenido que teclear toda la palabra.

07) Correr muchas otras macros que sirvan para: i) reformatear el documento según pautas prefijadas; ii) clasificar y seleccionar fichas de un fichero; iii) adecuar el texto a una u otra de varias opciones de impresión; iv) insertar en él subdocumentos; y, v) en suma, muchas otras tareas de edición.

08) Escribir, no sólo los caracteres griegos y la notación simbólica, sino también, cuando sea menester, todo el texto en lenguaje HTML.[87]

09) Disponer de auxilios de idioma (para español, francés e inglés): sinónimos y ortografía.

10) Ofrecer visualización *preprint*, e.d. que, si bien la edición del texto no es WYSIWYG, antes de mandarlo a la impresora resulte posible, sin salir del programa, ver cuál será el resultado en el papel.[88]

---

que el viejo WordPerfect 5.1. De las 17 cualidades que le reconozco a éste último, varias de las más importantes están faltando en el WordPerfect 2021.

Acerca de la función de revelar códigos, leemos en http://www.cv.nrao.edu/~pmurphy/doc-interchange.shtml:

> The ability to REVEAL CODES: when you engage this function, the window splits in two with the bottom part showing the various codes (font changes, hard returns and page breaks, boxes, etc.) You can step through these, delete unwanted or unneeded codes, and gain much more control over your document. Often, such codes can be the root of many document interchange and formatting problems. AFAIK this function is not available in Word or OpenOffice/StarOffice, or if so, in an extremely diluted manner.

87. Al referirme aquí al lenguaje HTML, quiero precisar: con toda la gama necesaria de caracteres, incluyendo el alfabeto griego más signos de lógica matemática o de otros formalismos. No pretendo agotar todo el espacio de grafías codificables en HTML —la mayor parte de las cuales desbordan en marco de mi escritura—, mas sí un amplio espectro, como aquel que puede comprobar el lector consultando los ficheros de extensión .htm o .html de mi portal http://sorites.org, donde está ubicada la revista digital *SORITES* de filosofía analítica, que produje entre 1995 y 2008.

88. A todo lo ya dicho contra el WYSIWYG, agrego ahora una nueva consideración. Estamos tan habituados a la diferencia entre letra romana, itálica (o cursiva), bastardilla, negrilla, versalita, etc, que nos da la impresión de que esas variantes son consustanciales a la escritura. No es así. Constituyeron un invento de los impresores posteriores a Gutenberg. Esos distingos no se dan en otras grafías diversas de la latina.

Además, si hasta ahora la escritura está, esencialmente, diseñada para volcarse al papel, en el futuro no ocurrirá así, sino que leeremos principalmente en pantalla, con las tabletas. Para visualizar un texto en pantalla,

11) Disponer de varios controladores de impresora, escogiendo uno u otro a tenor del resultado que se desee.

12) Poseer un programa anejo para editar esos controladores de impresora según las preferencias del usuario (como, v.g., retirar un determinado item de la lista de fuentes, reservándole la función de marcar un cierto atributo).

13) Alternar teclados costumbrizados: uno para el texto normal, otro para la notación simbólica, un tercero para la escritura en griego —y así cuantos nos resulten útiles—, en cada uno de los cuales estará fijada una determinada macro a las teclas que no sean alfanuméricas.

14) Disponer de menúes de texto desplegables, ocultos hasta que se pulse una tecla que los active,[89] de suerte que, al venir activados, resalten por el color aquellas pulsaciones de tecla que irán seleccionando las operaciones deseadas.[90]

15) Lanzar, a la vez, varias copias del mismo procesador de texto en sendos escritorios —conmutando entre ellos mediante la pulsación de las respectivas *hot-keys*—, distinguiéndose esas copias por una particular combinación de colores.[91]

16) Desplegar el texto que estamos tecleando, en la pantalla de edición, en formato 80X25, de suerte que la letra sea suficientemente grande y clara, aun para quien padezca alguna minusvalía óptica.

17) Prescindir del ratón (que puede dejarse totalmente desactivado), pudiendo

---

preferible resulta, en lugar de esa pluralidad del diseño de la letra, acudir a contrastes de color —usando, p.ej., el rojo donde solemos usar la itálica; el verde, en lugar de la negrilla; y otros colores para sendos atributos. (Eso facilitaría la lectura a quienes sufren alguna minusvalía óptica.)

89. En mi caso, la tecla que activa (y desactiva) los menúes es <Alt> (o, indistintamente, <AltGr> —o sea el <Alt> a la derecha del espaciador).

90. Puesto que el WordPerfect 5.1 que uso yo es el inglés (lo preferí a la mal traducida versión en español), el menú desplegable consta de los siguientes nueve ítemes: **F**ile, **E**dit, **S**each, **M**ark, **T**ools, F**o**nt, **G**raphics y **H**elp. (Resulta indistinto pulsar la mayúscula o la minúscula.)

91. Gracias a ese lanzamiento, a la vez, de tres copias clónicas del mismo programa WordPerfect 5.1, resulta posible simultanear la edición de hasta seis documentos (pasando bloques de texto del uno al otro). Evidentemente, en lugar de tres clones, podrían ser cuatro, cinco, seis o los que uno desee (siempre que el número sea compatible con los recursos de la máquina, que son finitos).

Desconozco si pueden lanzarse varias copias simultáneas del M$-Office o de otros *Offices*. Pensarán los usuarios de tales programas que no es menester, ya que, sin duda, tratándose de aplicaciones modernas y de más potencia, son capaces de simultanear la edición de un mayor número de documentos. Dudo, empero, que tal simultaneidad vaya acompañada de una diferencia de colorido, gracias a la cual, cuando yo paso de un clon a otro, salta a la vista en cuál de ellos estoy. También me pregunto si en un *Office* gráfico puede realizarse la permutación meramente pulsando *hotkeys* o si hay que recurrir al ratón.

así el escritor mantener ambas manos permanentemente sobre el teclado.

Además de esos diecisiete, existen otros dos motivos adicionales por los cuales soy adicto al WordPerfect 5.1.

El primero de ellos es tratarse de un procesador de texto que permite escribir sin estilos. En WordPerfect 5.1 puede el usuario, si lo desea, crear estilos, aplicándolos en su escritura. De hecho había yo acudido a ellos en los primeros años de mi práctica con este programa.

Luego me desengañé, dándome cuenta de su efecto nefasto. Al definir un estilo, encapsula en él el escritor determinados códigos y atributos; sin embargo, ulteriormente cambia de opinión, cuando aplica ese mismo estilo a otro fragmento de texto. Al final se van multiplicando los estilos; cualquier edición de un estilo acarrea consecuencias de formato que el escritor no tenía en su mente a la hora de hacer esa modificación. Además, el documento se va hinchando, convirtiéndose en un fichero desmesuradamente voluminoso para su contenido textual.

Cuando recibo un documento ajeno (o mío, mas tecleado por otro) —en uno u otro caso, a través de un filtro de conversión— tópome con un fichero mastodóntico, sobrecargadísimo de estilos. ¡Vaya trajín que implica limpiarlo, ir quitando códigos redundantes, inútiles o perjudiciales!

En lugar de los estilos, resulta mucho mejor ir adjudicándole a cada fragmento de texto, caso por caso, los atributos adecuados, sin encasillarlo en ningún estilo seleccionado de un elenco estilístico dado.

P.ej., habrá fragmentos que hayan de figurar en itálica y negrilla a la vez; resulta visible y sencillísimo asignarle ambos atributos (lo cual además se reflejará, en pantalla, en el color adscrito a esa combinación), en vez de embutirlo en uno de entre decenas de estilos. Los estilos son engorrosos, entorpeciendo la escritura, la corrección y el reformateo de los documentos.

El segundo motivo adicional es que el WordPerfect 5.1 lo conozco, lo domino, habiéndolo estudiado con ahínco (consagrándole, con aplicación, muchísimo tiempo), empollándome gruesos manuales y libros, ejercitándome con empeño en el aprendizaje experimental de su riquísimo lenguaje de macros —con el cual he escrito cerca de un millar de ellas, sin cuyo auxilio mi escritura sería mucho más lenta y difícil.[92]

Gracias a tal aprendizaje (que abarcó un trabajo de memorización), soy capaz, no sólo de teclear el texto con la celeridad que me permiten las macros a las cuales recurro, reformatearlo rápida y cómodamente de muy diversas

---

92. No obstante, honestamente he de reconocer que casi todas esas macros —que continúo usando— las escribí hace ya bastantes años, cuando —con soltura y hasta preciosismo— dominaba yo ese lenguaje de macros. Hoy mi conocimiento está oxidado. Sigo beneficiándome de lo que entonces aprendí; si tuviera que agregar nuevas macros (sobre todo de aquellas que son más difíciles y complicadas), me haría falta volver a estudiar.

maneras y aplicarle filtros que lo transforman (v.g. para convertirlo en texto codificado en lenguaje HTML), sino también insertar gráficos, crear tablas, columnas, recuadros de texto con ecuaciones, encabezamientos y notas a pie de página (transformables en notas finales y viceversa, reformateables de ahecho cuando sea menester). También manejo con inmediatez, sin tener que darle vueltas, los menúes desplegables y un montón de atajos de tecla.

Adicionalmente, con una idónea configuración del Dosemu y del WordPerfect 51, resultan posibles:

— salir transitoriamente del programa (sin perder nada) para ejecutar, en el inductor del DOS virtual, otras órdenes o aplicaciones DOS;

— preparar el texto tecleado para su impresión, directa o indirecta, en variados formatos (tanto de tamaño, márgenes y demás detalles de paginación cuanto de configuración de impresora, v.g. .hcl o .ps [exteriormente convertible en .pdf]).[93]

¿Posibilitan esas funciones otros programas actuales de tratamiento de texto? Desde luego no los que yo he ensayado o abierto. Incluso las pocas de tales funciones realizables con otros programas se llevan a cabo sin comodidad ni facilidad, no bastando la mera pulsación de unas cuantas teclas.

Aun en el contrafáctico supuesto de que fuera realizable todo eso con otro programa, habría yo de aprenderlo *ex novo*; por lo cual, aunque existiera otra aplicación con las enumeradas diecisiete características (más la posibilidad de prescindir de estilos), únicamente me sería útil mediante un aprendizaje previo, el cual requeriría un tiempo que ni puedo ni quiero consagrarle.

Hágome cargo de que, en la vida académica y profesional, hay que reconvertirse, no pudiendo uno permanecer dormido en los laureles, anclado en aquello que aprendió decenios atrás. Pienso haber cumplido esa norma en

---

[93]. Lamentablemente (a consecuencia de los cambios del kernel —y, en general, del entorno de usuario, seguramente motivados, a su vez, por las variaciones del *hardware*, al menos en parte), ya no resulta hoy el Dosemu tan agradable de usar como lo era en el último decenio del siglo pasado (un tiempo que posiblemente se extienda a lo largo del primer decenio del siglo XXI).

Recurriendo al permiso «s» —que hace de un binario ejecutable un programa que el usuario puede correr con los privilegios del *root*— podía entonces el Dosemu acceder directamente, en modo real, al *hardware*: el vídeo de la pantalla, el teclado y la impresora. Hoy hay que ejecutarlo en modo protegido, lo cual significa que se interpone una capa. No controlando ya ninguno de esos tres recursos, no tiene el Dosemu acceso al metal, sino que ha de comunicarse con esas piezas del *hardware* a través del filtro interpuesto por el sistema operativo.

En lo atinente a la impresión y al teclado resulta remediable tal aminoración de potencia. No así con relación a la visualización de texto. Lo cual se traduce en una imposibilidad de ver hoy en pantalla —correctamente reflejados e identificados a simple vista— los 512 caracteres extendidos del WordPerfect que antes era posible visualizar. Perdemos así varios importantes signos de griego y de notación lógico-matemática, como la disyunción, la conyunción, el condicional, los cuantificadores, así como los operadores de necesidad y de posibilidad.

El aprieto aguza el ingenio. He diseñado yo, con una macro nueva, un sucedáneo, que sustituye provisionalmente esos signos por otros visualizables (procediendo al reemplazo inverso a la hora de formatear definitivamente el texto). Mas no deja de ser un expediente para andar por casa.

mi propia vida vocacional.

Sin embargo, ¿también es preceptivo reciclarse en el uso de cualquier utillaje adquirido? Convertiríase tal preceptividad en una tarea imposible e indeseable. Quien haya aprendido en su juventud el alemán podrá tener que adquirir algún vocabulario marginal novedoso, mas no ha de volver a estudiar el idioma desde zero —ni otra nueva lengua por la cual decidieran reemplazar al alemán ciertos «expertos».

Dan a veces los informáticos la impresión de diseñar sus innovaciones como si fuera infinitamente extensible el tiempo de aprendizaje de los usuarios. Para atenuar esa exigencia ofrécennos programas dizque amigables e «intuitivos», que a mí no me sirven, careciendo yo de intuición. (Además no los encuentro amigables para nada.)

* * *

¿Qué habría hecho yo de haberme visto constreñido a renunciar al Dosemu —y, por lo tanto, al WordPerfect 5.1? Habría estudiado el TeX, lo único que prometía resultados parcialmente comparables —aunque dudo que con todas las diecisiete cualidades que acabo de enumerar. Habiendo abordado tal aprendizaje varias veces, no he perseverado nunca, al tener siempre el refugio del WordPerfect 5.1, que tanto he luchado por seguir usando.[94]

Ése es el motivo —que no la nostalgia— de que el autor de estas páginas sienta —y lo manifieste— un especial apego al DOS, a pesar de Micro$oft. Habiendo llevado a cabo la investigación conducente a redactar este ensayo, ya sé que el DOS se lo debo al brillante y desgraciado Gary Kildall, q.e.p.d.

---

94. La similitud entre los lenguajes TeX, HTML y WordPerfect póngola de manifiesto en un Anejo al presente ensayo, en el cual reproduzco un fragmento del mismo en esos tres formatos.

Capítulo XVI

Características del Unix/Linux (1ª parte): aspectos generales

Las esenciales características del Linux no son otras que las del Unix. Una buena parte del contenido del presente capítulo serviría igualmente para describir a cualquier distribución del Unix. Lo cual no impide para que, en su evolución, hayan ido desarrollando ciertas peculiaridades las distintas distribuciones del Linux y que, en general, el Unix/Linux de hoy no sea exactamente igual que el de los orígenes. Según era de esperar, algunas órdenes se han abandonado, modificado o reemplazado; otras se han añadido. Mas tales diferencias son marginales.

. . .

Primer rasgo llamativo: en Unix/Linux todo es un fichero. Cualquier elemento relacionado con el computador es un fichero, cualquier recurso o dispositivo es un fichero, cualquier periférico es un fichero.

En Unix/Linux no existen diferencias categoriales. Es una diferencia categorial aquella que opone a dos entidades, X, Z, cuando no tiene **sentido** siquiera negar de Z (ni afirmarlo) algo que se afirme o se niegue de X.

El DOS y el Windows son categoriales. En la sintaxis del DOS y del Windows no se puede siquiera *decir* que se copia un fichero a un conducto o dispositivo o que se borra una impresora. En Unix/Linux se puede copiar un dispositivo en un fichero normal y viceversa; se puede querer borrar un dispositivo de CDRom (sin éxito, pues es de sólo lectura).[95]

. . .

El **segundo rasgo** del Unix/Linux es su árbol de directorios, la estructura del sistema; un árbol que no ha permanecido inmutable, pero cuyo perfil esencial se ha mantenido inalterado desde los años setenta del pasado siglo. En el capítulo siguiente voy a presentarlo sinópticamente.

. . .

---

95. No vienen, en cambio, tratados como ficheros ni el propio computador ni los usuarios.

El **tercer rasgo** es que en Unix/Linux los ficheros vienen clasificados en virtud de sus atributos o permisos, no de su denominación ni, por lo tanto, de las desinencias.

Cada fichero viene atribuido a un dueño, su *user*, y a un determinado grupo que abarque a ese dueño. Conque el Unix/Linux traza tres círculos concéntricos, con permisos diversificados: primer círculo, el del dueño (*user*); segundo círculo, el del grupo; tercer círculo, el de los demás (*other*). Asignánsefe, positiva o negativamente, a cada uno los permisos de lectura, escritura y ejecución.

Ejecutar un directorio significa abrirlo, acceder a sus contenidos. Ejecutar un fichero normal sólo resulta posible cuando es uno de suyo ejecutable (al menos un *script* del *shell*).[96]

El atributo de lectura viene marcado por una puntuación de 4; el de escritura, por una de 2; el de ejecución, por una de 1. Y se suman. Forman así los tres atributos una serie de tres dígitos; cada dígito es: 1, 2, 3, 4, 5, 6 ó 7. El 7 significa una combinación de los tres permisos: el de lectura, el de escritura y el de ejecución.

De ese modo, si un fichero (p.ej. un directorio) tiene el permiso 750, es que el dueño (*user*) puede leerlo, escribir en él y abrirlo, mientras que los otros miembros del grupo pueden leerlo y abrirlo mas no escribir en él y cualesquiera otros usuarios no pueden ni escribir, ni leer ni siquiera acceder al directorio, que les permanece cerrado.

Ahora bien, ¿no es cierto que, además de la distinción por atributos, también se usa hoy en Linux la diferenciación y clasificación los ficheros por sus respectivas desinencias? Ya me he referido a esa cuestión. Efectivamente, empléanse hoy en Linux las desinencias para determinar si un fichero es, v.g., de carácter pictórico (un .jpg o un .png, entre otros muchos), si es un libro (un .epub, .mobi, .azw, etc), si es un PDF, si es un documento (.doc, .wp, .odt, .docx, …), si es un audio (un .m4a, un .wav, un .mp3, entre otros), si es un vídeo (un .avi, un .mp4 etc), si es un fichero de texto (un .txt o un .html) y así sucesivamente. Son indicaciones que se les ofrecen a los gestores de ficheros para que invoquen determinados programas a la hora de abrir sendos ficheros.

Habiéndose extendido tanto esa mala costumbre, dudo que sea desarraigable, si bien —por las razones más arriba evocadas— resulta perjudicial e inconveniente. Desde luego se opone completamente a la concepción básica del Unix sobre denominaciones de ficheros.

. . .

El **cuarto rasgo** del Unix/Linux es el de tratarse de un sistema

---

96. Según ya lo evoqué más arriba, una de las molestas manifestaciones de que un fichero provenga de haberse copiado desde el Windows o el Android es que exhibe el atributo o permiso de ejecutable, cuando es un audio, o un libro, o un PDF o lo que sea, no una aplicación ejecutable.

multitarea y multiusuario. Sepárase en eso del DOS, asemejándose al Windows, aunque difieran muchísimo sendas maneras de implementar tal característica.

Ahora bien, si el carácter multitarea es obviamente, no ya útil, sino necesario, hoy podríamos pensar que el carácter multiusuario va resultando superfluo. Cuando en un hogar había un único computador familiar (un verdadero *home computer*), tenía que compartirse. Hoy en muchas familias hay, no sólo un computador —o más— por usuario, sino también varios dispositivos.

Aun así, para ciertos propósitos resulta beneficioso que un mismo usuario físico se desdoble en varios usuarios virtuales (varias personas o máscaras) con utilizaciones alternas de la máquina. De esos desdoblamientos el más simple y usual es que un mismo usuario, juan, entre en el computador, según los casos, como juan y como root —*vide infra*— (por mucho que los desarrolladores nos prodiguen el acuciante consejo —casi un precepto— de no entrar nunca como root).[97]

Para aclarar lo que acabo de decir, hay que precisar que, de entre los varios usuarios, existe uno, el superusuario o *root* (literalmente «raíz»), que es el soberano. Enseñan todos los desarrolladores de distribuciones del Unix/Linux que debemos acceder a nuestras máquinas sólo lo indispensable como *root*, porque el *root* puede hacer muchas cosas —frecuentemente demasiadas. Consejo que, por bien intencionado y prudente, se agradece, pero sin olvidar que una de las cosas que nos hacen amar el Unix/Linux es la libertad, incluso la de poder hacernos mal a nosotros mismos.[98]

. . .

El **quinto rasgo** del Unix/Linux es la concha, el *shell*, que es un transmisor de órdenes del usuario a la máquina. Hay varias conchas, pero la más común es Bash (*Bourne Shell* en el viejo Unix y *Bourne Again Shell* en el Linux).

La idea y hasta el propio vocablo de «shell» fueron uno de tantos inventos del Multics. (Al principio habíase tratado de un simple mecanismo de teletipo, o télex.)

En sus primerísimos momentos (cuando apenas empezaba a existir, en torno a 1970) no había asumido todavía el Unix la necesidad de una concha, un shell, lo cual causaba una dificultad en la utilización del computador.

---

97. P.ej., en un período ya pretérito, había creado yo en mi computador varios usuarios virtuales, a cada uno de los cuales pertenecía su propio árbol de subdirectorios, con su particular especialización; p.ej., al producir yo, por aquellos años, la revista electrónica *Sorites*, había creado un usuario, sorites, cuyo /home/sorites abarcaba una enorme cantidad de directorios y ficheros —con los manuscritos sometidos, los informes evaluativos, las galeradas de cada número de la revista, la correspondencia etc.

98. Según ya lo señalé más arribe, una de las horribles cortapisas del Android es la imposibilidad de entrar como *root* en nuestro dispositivo (salvo *ruteándolo*, tarea dificilísima y, sobre todo, arriesgadísima —de lo cual puedo dar testimonio).

Pronto se vio que, para que el usuario pudiera interactuar con la máquina, era menester que existiera un programa que recibiera inductos del teclado, a tenor de los cuales corriera ejecutables o realizara él mismo determinadas tareas, emitiendo eductos que se reflejarían en la pantalla.

Un *shell* podía ser cualquier programa con esa triple virtualidad (recibir, ejecutar, emitir), lo cual significa que podría ser más o menos amplio el elenco de aquellas tareas directamente ejecutables por el programa al que se asignara función de *shell*. Un shell puede ser, así, más polifacético que otro. A cada shell le viene asociada una gramática de órdenes y, por lo tanto, de posibles *scripts* redactables y (a través del shell) ejecutables por el usuario.

Tiene cada usuario su propio directorio hogareño, su *home* (aunque se pueden crear usuarios *ad hoc* sin *home*). Existe un directorio que pende de la raíz y se llama, precisamente, «home», el cual alberga a los *homes* de todos los usuarios. P.ej. el directorio *home* de Juan será /home/juan. Ahí reside una configuración propia del *shell*, a través de un fichero *.bashrc* —libremente configurable— que hace correr determinadas aplicaciones cuando Juan inicia una sesión, asignándoles unos u otros valores a ciertas variables —lo cual será útil para la ejecución de otros programas en función de los valores adscritos a esas variables. También establece ciertos alias, que sirven para abreviar las instrucciones dadas a la máquina en el CLI.

Los *scripts* redactados según la gramática ofrecida por el *shell* desempeñan una función similar a la de los *batches* del DOS, o sea instrucciones en serie que —a través de esa interfaz que es el *shell*— imparte el usuario a la máquina para ejecutarlos. Es muy beneficioso saber escribir *scripts* para acelerar, aligerar y automatizar una pluralidad de tareas.

. . .

El **sexto rasgo** del Unix/Linux constitúyenlo sus órdenes. Algunas de ellas son bastante parecidas a varias de las órdenes o instrucciones del DOS, mas otras son peculiares del Unix/Linux. En el capítulo XVIII voy a presentar un elenco de órdenes —no exhaustivo, desde luego.

. . .

El **séptimo rasgo** del Unix/Linux es la existencia de enlaces, *links*, que son de dos tipos: duros y simbólicos. En virtud de un *link*, un mismo fichero, sin necesidad de duplicarse o copiarse, puede tener dos (o más) denominaciones e incluso ubicaciones. Si el enlace es simbólico, trátase de una mera anotación, de un *como si*. Los *hard links*, sin embargo, implican una ocupación de espacio en disco. Yo sólo uso enlaces simbólicos. (Jamás he usado *hard links*, no habiendo entendido del todo ni en qué consistan ni para qué sirvan.)

Así, supongamos que tenemos, en un directorio D1 —que puede incluso estar en un dispositivo externo, como una llave USB—, un fichero, xyzuv, pero que queremos también tenerlo en otro directorio, D2 —que puede estar ubicado

en otro dispositivo, v.g. en una partición del almacenamiento interno NVME0n1p2—, con otro nombre, bcdefg. Basta, para conseguirlo, dar la instrucción de que se cree un enlace simbólico D1/xyzuv → D2/bcdefg. (En Unix/Linux la separación entre un directorio y sus contenidos —sean subdirectorios o ficheros normales— la marca la barra inclinada «/», no su inversa, «\». Ésta última se emplea para otros usos.)

. . .

El **octavo rasgo** del Unix/Linux es la existencia de variables con asignaciones, algunas de las cuales son comunes —viniendo establecida por los ficheros de configuración general que establece el superusuario. Una de esas variables es el *path* o sendero, o sea un elenco de aquellos directorios donde pueden hallarse ejecutables que uno invoque. (¡Entendámonos! Pueden ubicarse ejecutables en cualquier directorio donde a uno le plazca, mas únicamente serán hallados —a menos que se escriba todo el recorrido de su ubicación— si el directorio está en el *path*. P.ej., un ejecutable, xswqm, que esté ubicado en el directorio /usr/local/bin, será invocable sin más que dar la orden «xswqm», mientras que, si se halla en /local/varios/novedades, únicamente se ejecutará si la orden impartida es «/local/varios/novedades/xswqm» (o si se ha activado ese directorio, lo cual se sabe con la orden «pwd»).[99]

[99]. En lo atinente a este octavo rasgo, el Unix/Linux es muy parecido al DOS, que lo había tomado del Unix, como tantas otras cosas.

Capítulo XVII

Características del Unix/Linux (2ª parte): el árbol de directorios

Voy a presentar un elenco sumario de los directorios del Unix/Linux, siguiendo la lista de Slackware; difiere algo (pero secundariamente) la de mi actual distribución, Lubuntu. (Más abajo también aclararé esas diferencias.)

Ante todo está el directorio raíz, «/». Es la cúspide, representando a la propia máquina. Prefijando al nombre de un directorio, ese signo, «/», indícase que se trata de un directorio que pende, sin intermediario alguno, de la raíz.

No se confunda ese directorio raíz —en inglés *root*— con el directorio propio del superusuario, del *root*, usuario con plena potestad (gobernador de la máquina y del sistema operativo). Ese directorio no es «/», sino «/root». Es el directorio del administrador del sistema, el cual no viene subsumido en el directorio «/home» (al cual en seguida me referiré) como los demás usuarios, sino aparte completamente.

Y es que «/home» puede montarse en una partición diferente del disco (diferente de la partición ocupada por «/»). Por algún percance podría suceder que no se montara; y, bajo ciertos sistemas de inicialización, puede montarse después. Si el «/root» fuera un subdirectorio del «/home», en ambas situaciones se seguirían consecuencias adversas e incluso una parálisis del funcionamiento del aparato, toda vez que sin el root, sin el piloto, el navío se va a pique. Puede haber rutinas de inicialización que invoquen ejecutables alojados en «/root/bin» (el subdirectorio del directorio «/root» con ejecutables cuya invocación le está exclusivamente reservada al superusuario), las cuales no serían halladas ni ejecutadas de pender ese directorio «/root» del «/home» (en el supuesto, repito, de que éste no estuviera montado, en el mismo instante del arranque, como ha de estarlo forzosamente el directorio raíz, «/»).

* * *

Tenemos, en segundo lugar, el directorio «/bin», que alberga esenciales ejecutables, que constituyen el mínimo cúmulo de programas requeridos para que pueda un usuario valerse del computador. Entre ellos están la concha (*shell*), e.d. el intérprete de órdenes, y los mandos del sistema de ficheros («ls», «cp», etc). El directorio «/bin» suele permanecer inalterado tras la instalación. O, si acaso, viene modificado por ascensos de algún paquete suministrados por

el desarrollador de la distribución.

En tercer lugar, está el directorio «/boot», donde se hallan aquellos ficheros que usa el cargador del núcleo Linux (antes LILO, ahora GRUB). También sufre este directorio pocas modificaciones tras la instalación del sistema operativo. Aquí está ubicado el kernel.

En Lubuntu hay más de un kernel presente: el último más, por si acaso, el precedente. Arriesgado resulta actualizar el kernel; yo no lo recomiendo salvo que exista una buena razón de peso. Existen dos peligros: 1) que, al arrancar, el gestor de arranque no lo encuentre; 2) que, por la causa que sea, no se adapte a la máquina igual que el anterior, v.g. desbordando la memoria. En ambos casos prodúcese un *kernel panic*, cuyas consecuencias serán deletéreas (cuando no se posee suficiente pericia, puede tener el usuario que reinstalar el sistema operativo).

En las modernas distribuciones, de ese directorio «/boot» pende un subdirectorio, «efi», donde se monta una pequeña partición de arranque del disco duro (o equivalente dispositivo de almacenamiento interno). (En seguida aclararé en qué consiste eso de *montar*.) Con la actualmente más utilizada aplicación de arranque, GRUB, es necesario ese mecanismo del subdirectorio «/boot/efi». Más sencillo era el anterior, con el programa de arranque Lilo (*Linux-Loader*), hoy poco utilizado.

* * *

Viene ahora un cuarto directorio, el «/dev». Para entender qué se aloja en él, hemos de recordar que el Unix/Linux trata todo como ficheros incluso los recursos o dispositivos de *hardware*, como los puertos seriales, los discos duros, cualesquiera otros dispositivos de almacenamiento, interno o externo (como los modernos módulos de memoria no volátil, NVME, los disquetes o *floppies* o los aparatos conectados por USB); son asimismo *device-nodes* las terminales o consolas virtuales —los «ttys»), las impresoras, las escrutadoras o cualesquiera periféricos a los que esté conectado el computador (por vía USB, puerto paralelo o serial e incluso por la red).

En aras de acceder a cualquiera de esos recursos o dispositivos, el computador ha de tener un fichero especial, un *device node*, que le esté consagrado. Todos esos *device-nodes* vienen alojados en el directorio «/dev».

Algunos de esos recursos o *devices* pueden montarse, o sea configurarse para venir accedidos como sitios de almacenamiento de ficheros. Montar en el directorio HHH un recurso, p.ej. /dev/sda1, requiere dar la orden «mount /dev/sda1 HHH», pudiendo HHH ser un subdirectorio (p.ej. si HHH=/home/alfredo/disco).

De ese montaje encárganse, en parte, los ficheros de configuración que ejecuta en su arranque el sistema operativo; otros montajes ha de hacerlos el usuario —si bien las modernas distribuciones montan automáticamente los recursos montables; p.ej., si insertamos una llave USB con una partición

formateada, el Ubuntu lo monta en un directorio que crea, el cual será un subdirectorio de «/media». Con Slackware, en cambio, el mero hecho de enganchar una llave USB no determinaba que viniera montada; únicamente lo sería cuando diera la instrucción oportuna el usuario. Siendo más cómodo el montaje automático (diseñado precisamente para usuarios inexpertos), no deja de encerrar sus peligros e inconvenientes en algunos casos.

Para poder montarse un recurso (un *device node*), o sea un poblador del directorio «/dev/», ha de ser un recurso de bloque, no uno de carácter.[100]

Los recursos (*devices*) disponibles en nuestra máquina, en una determinada sesión, podemos conocerlos de dos modos. Uno es mirando los contenidos del directorio «/dev». El otro es leyendo el fichero «/proc/devices». De ambas maneras podemos saber cuáles son de carácter y cuáles de bloque.

En mi computador, en la sesión en que estoy escribiendo estas líneas, son de carácter los siguientes: «mem» (la memoria, obviamente), «tty» (el terminal), «console» (la consola), «lp» (la impresora), «usb», «usb_device» y otros cuantos más cuya función resulta oscura sin avanzar en el estudio informático.

Entre los recursos de bloque ahora disponibles en mi computador figuran: «loop», «sd», «device-mapper» y «virtblk».

Si es condición necesaria para montar un *device* que éste sea de bloque, no es, en cambio, condición suficiente. Los recursos de bloque son piezas de *hardware* de uno de los dos siguientes tipos:

1º) Dispositivos montables como sistemas de ficheros (*file-systems*, o sea dispositivos formateados), o emulaciones virtuales de tales piezas (los *loops*, p.ej.);

2º) Dispositivos particionados o particionables, siendo, en este caso, esas particiones las que constituyen recursos montables.

Así, el *device* «/dev/sda», aun siendo un dispositivo de bloque, ha de particionarse, siendo las particiones formateadas aquellas que sí constituyen recursos montables (en este caso, «/dev/sda1», «/dev/sda2», etc.)

Otros recursos o *devices*, alojados en ese mismo directorio «/dev», son los siguientes: «/dev/mouse» (el ratón), «/dev/audio» (la tarjeta de sonido), «/dev/eth0» (la primera tarjeta ethernet), «/dev/eth1» (la segunda), «/dev/loop1» (el primer «loop», recurso reservado para montar un fichero cual si se tratara de un dispositivo), «/dev/loop2», «/dev/loop3», etc.

Hay recursos (*devices*) que desempeñan funciones peculiares, como «/dev/random» (que vuelca secuencias de números aleatorias) e igualmente dos nodos sin denotación —cuyo rol guarda cierta similitud con el de la clase vacía

---

100. Ese distingo técnico resulta arduo para un lego sin conocimientos técnicos. El lector hallará en el internet aclaraciones, si las desea.

en teoría estándar de conjuntos—: los *devices* «null» y «zero», que, pese a lo que sugerirían sus nombres, desempeñan papeles muy diversos uno del otro. El recurso «/dev/null» es un sumidero de datos (que hace desvanecerse cualesquiera datos que le enviemos, siendo así una especie de agujero negro; la orden «ls -l > /dev/null» envía la lista de los contenidos del presente directorio a ese sumidero; en tal caso su uso sería, obviamente, absurdo, mas hay órdenes tales que no queremos que aparezca en pantalla el educto de su ejecución). Esos dos recursos pueden considerarse, en cierto sentido, nodos ficticios.

Hay que introducir una precisión adicional. En las distribuciones tradicionales, como Slackware, los ficheros albergados en el directorio «/dev» existían, en el disco duro, permanentemente, una vez que habían sido creados, incluso cuando los dispositivos denotados hubieran dejado de estar presentes; v.g. un disco externo USB podía desconectarse. Sucedía empero que esos nodos eran vacíos cuando, por desconexión de tales dispositivos, dejaban de referirse a ellos.

En las distribuciones más recientes, como Ubuntu, todo el directorio «/dev» se crea en memoria volátil, RAM, aunque también se generan nodos vacuos. En una de las modernas distribuciones, al rebutearse el computador desaparecen todos los contenidos de ese directorio «/dev», para crearse de nuevo otro directorio «/dev», que no hereda nada del precedente.

Cerraré este apartado con una significativa precisión. Ciertamente todos los sistemas Unix (incluidas todas las distribuciones de Linux) han adoptado este tratamiento del directorio «/dev» y de sus contenidos (un tratamiento consistente en el principio de Unix de que «todo es un fichero» —un componente físico es fichero *sui generis*, un *device*, precisamente alojado en este directorio). Mas es eso lo que impugnan los programadores del moderno método de inicialización *systemd* (*vide infra*, capítulo XXI). Ocúrreseme que los desarrolladores de las distribuciones del Linux habrían de prestar mayor atención a este problema, no mirándolo como una logomaquia. Percataríanse tal vez así de los peligros que encierra el *systemd*.

* * *

Paso a considerar ahora un quinto directorio, el «/etc». Resulta engañoso su nombre, pudiendo sugerir que se trata de una colección residual de aquello que no se ha sabido o podido clasificar en directorios de más descriptivo perfil. No, no es eso.

El directorio «/etc» aloja principalmente los ficheros de configuración. Justamente aquí están los que encauzan y determinan los procesos de inicialización del computador, los ajustes del entorno gráfico, el catálogo de los usuarios y los grupos en que se congregan, los *scripts* que han de correr al ponerse la máquina en marcha. Varios de tales *scripts* ejecutan ciertos programas que han de permanecer corriendo, en transfondo, y a los que se llama «demonios» (piénsese, no en diablos, sino en los *dáimones* de la mitología

griega, como el célebre *daimon* de Sócrates, a los cuales más podríamos considerar ángeles de la guardia); su función es similar a la que en el DOS tenían los programas residentes en memoria.

Ha acarreado una radical mutación de este directorio «/etc» reemplazar el viejo sistema de inicialización de Unix por esa alambicada y complejísima innovación que es el *systemd*. Todo en él ha venido trastrocado. Entre otros nuevos subdirectorios que han surgido está uno que precisamente se llama «systemd», donde se desencadena paralelamente la acción de 69 «servicios» o «unidades». El resultado eleva a un exponente la complejidad del anterior y, por consiguiente, también causa un logarítmico incremento en la dificultad de su manejo.

* * *

Pasamos, a renglón seguido, al sexto directorio, el «/home». Siendo Linux un sistema operativo multiusuario, a cada uno de sus usuarios se le da una cuenta junto con un directorio propio para sus ficheros individuales, el cual viene denominado, normalmente, por el nombre del usuario. (En mis máquinas, v.g., yo soy el usuario «lorenzo», aunque cada quien puede escoger el nombre que guste.) Siendo yo, lorenzo, un usuario de mi computador, tengo un directorio propio así llamado (directorio lorenzo), que es subdirectorio de «/home» (es el «/home/lorenzo»). En él tengo almacenados mis datos (aunque también puedo colocarlos en otros directorios a los que tenga permiso de ejecución y escritura); uno de mis subdirectorios es «/home/lorenzo/bin» donde tengo mis propios *scripts* u otros binarios cuya ejecución me reservo, sin compartirla con otros eventuales usuarios de la misma máquina.

* * *

Tenemos ahora un séptimo directorio, el «/lib». Están aquí las librerías del sistema requeridas para la operación básica. Entre sus contenidos figuran la libraría C, el cargador dinámico y los módulos del kernel.

Pasamos al octavo directorio, el «/mnt», que contiene (o, más bien, solía contener) puntos de montaje temporales para dispositivos de almacenamiento que no sean el principal, como discos duros, discos USB removibles, CDRom y DVDRom. Ya sabemos que los ve el computador como ficheros, teniéndolos ya como nodos en el directorio «/dev». Ahora bien, para que la máquina acceda a los contenidos almacenados en tales dispositivos, éstos han de venir precisamente *montados* en otros ficheros especiales, que son los contenidos del directorio «/mnt».

También pueden montarse en otros directorios. En las modernas distribuciones úsase poco el directorio «/mnt», que ha venido suplantado por otro nuevo, «/media», que cumple la misma función.

El noveno directorio es «/opt», donde están alojados paquetes de *software* optativos. La idea que dio lugar a crear este directorio (que no existía en las primitivas versiones de Linux) es que se instale en «/opt/XXX» cualquier

paquete de *software* extrasistemático, XXX, pudiendo borrarse si el superusuario decide eliminarlo. Como su propio nombre indica, este directorio sería prescindible, al haber otras ubicaciones posibles para esos paquetes adicionales. Es más claro colocarlos en «/opt», para que estén juntos, el ejecutable y sus librerías. Yo, p.ej., tengo en «/opt/google/chrome» el navegador de Google Chrome; y en «/opt/vivaldi» el navegador Vivaldi. Cada uno con sus librerías y ficheros auxiliares.

El décimo directorio es «/proc» (que viene de «proceso»). Es un directorio muy *sui generis*, virtual. No forma parte del sistema de ficheros, del *filesystem*; por consiguiente reside únicamente en memoria (volátil) sin estar, pues, alojado en el disco duro (ni en ningún otro dispositivo de almacenamiento, ni interno ni externo), siendo su función dar acceso a información atinente al kernel. Éste necesita conocer varios elementos de información. Suminístraselos la máquina en este directorio «/proc». También puede enviar el usuario información a este directorio y consultarla (v.g. con la orden «cat «/proc/cpuinfo»).»

El undécimo directorio es «/sbin», reservado a ejecutables esenciales cuya ejecución está reservada al superusuario. (Otros usuarios pueden ejecutarlos suplantando provisionalmente el rol de superusuario mediante la orden «sudo».) Varios de ellos son demonios que se lanzan desde los ficheros de inicialización radicados en «/etc». Igualmente están aquí otros ejecutables particularmente delicados y sensibles (aquellos que posibilitan operaciones cuya inadecuada realización sería perjudicial para el sistema). Los usuarios normales no ejecutan programas ubicados en el directorio «/sbin».

* * *

El duodécimo directorio es «/tmp», que es un almacenamiento temporal, en cuyos subdirectorios van a ubicarse cuantos ficheros hayan de irse creando sobre la marcha, al compás del lanzamiento de unos u otros programas; v.g. ficheros de desbordamiento, de trueque y de almacenamiento provisional de datos. Todos los usuarios tienen permisos de lectura y escritura en este directorio.

Ahora bien, existe una importante novedad. En las viejas distribuciones, por muy temporal que conceptualmente fuera, el directorio «/tmp» estaba físicamente presente en el disco duro y en él se colocaban, también físicamente, los ficheros temporales. Algunos de ellos eran automáticamente borrados, al cerrarse las aplicaciones que los habían creado, mas otros permanecían, pudiendo quedar durmiendo para siempre el sueño de los justos, hasta que se hacía una limpia.

Hoy, en cambio, en las nuevas distribuciones todo el directorio «/tmp», junto con sus subdirectorios, existe únicamente en memoria RAM, volátil, extinguiéndose al apagarse o rebutearse la máquina. Lo cual presenta ventajas e inconvenientes. La ventaja de que no se amontone basura inservible. Los dos inconvenientes son los de que: 1º, se está ocupando memoria (lo cual hoy es asumible, mas no lo era tanto cuando se sufrían serias limitaciones de memoria

disponible); 2º, al haberse desvanecido todos los ficheros temporales de una sesión anterior, hemos perdido huellas que podían aportarnos a veces útil información con la cual resultaba posible comprender qué anomalías o malos funcionamientos se habían producido previamente o recordar qué se había hecho con la máquina en esas precedentes sesiones.

* * *

El decimotercer directorio es el importantísimo «/usr». («usr» aquí no abrevia a «user», sino que es una sigla que denota *unified system ressources*, los recursos unificados del sistema.)

Es un directorio muy voluminoso en cualquier sistema Linux. Ante todo, hay que destacar en él su principal subdirectorio, «/usr/bin», que alberga a los ejecutables del sistema, salvo los más esenciales (aquellos que se ubican en uno de los dos directorios ya mencionados, «/bin» y «/sbin»). También existe un subdirectorio «/usr/sbin» donde figuran binarios cuya ejecución queda, en principio, reservada al superusuario y que normalmente no son accesibles a los demás usuarios.

Otros subdirectorios de «/usr» son «/usr/share» —donde se alojan muchísimos ficheros compartibles entre diversas aplicaciones, agrupados en subsubdirectorios—, «/usr/include» —que engloba otros módulos auxiliares invocables por las aplicaciones según se ejecuten—, «/usr/lib» —las principales librerías del sistema, salvo las más vitales (o sea, aquellas, de las cuales ya hemos hablado, que se han situado en «/lib»), junto con nuevos directorios de librerías que se han ido agregando con el tiempo: «/usr/lib32», «/usr/lib64», «/usr/libexec» y «/usr/libx32». Los desarrolladores de cada distribución determinan ese elenco de directorios con sendas librerías en función de criterios de coherencia.

Del «/usr» pende el «/usr/src» (donde se albergan ficheros fuente, que sirven para recompilar; sólo que tal operación —que antiguamente realizábamos asiduamente los más atrevidos— hoy requiere muchísima pericia o excesiva temeridad).

Otro importantísimo subdirectorio es el «/usr/local», reservado a ejecutables agregados por el titular de la máquina. En mis tiempos de usuario de Slackware este subdirectorio lo utilicé muchísimo, pues agregaba yo un montón de ejecutables —muchos de ellos compilados por mí partiendo de sendos programas-fuente—, que se almacenaban en «/usr/local/bin», con sendas librerías, ubicadas en «/usr/local/lib». Otros subdirectorios del «/usr/local» son: «/usr/local/sbin» (con, p.ej., un programa para generar contraseñas pseudoaleatorias), «/usr/local/man» (con los manuales), «/usr/local/share» y «/usr/local/src».

El decimocuarto directorio es «/var». Aquí están muchos ficheros derivados, tales como registros (*logs*) del sistema, datos del *cache*, ficheros de bloqueo (*program lock files*). Es un directorio para datos que sufren frecuentes mutaciones, de donde viene su nombre, **variable**.

* * *

En Ubuntu 22.04 los directorios «/lib», «/lib32», «/lib64», «/bin» y «/sbin» son meros enlaces simbólicos, o sea alias, respectivamente, de «/usr/lib», «/usr/lib32», «/usr/lib64», «/usr/bin» y «/usr/sbin». Así, de un plumazo, se han despejado las dudas sobre si un ejecutable es tan vital como para ubicarlo en «/bin» antes que en «/usr/bin».

La consecuencia es que ya no resulta posible ubicar el directorio «/usr» en una partición separada del disco duro. Además, el viejo distingo deslindaba lo que es esencial e imprescindible para que funcione la máquina de lo que es útil sin ser necesario. En la vieja organización, podían cambiarse fácilmente los binarios alojados en «/usr/bin» —incorporando otros o suprimiendo algunos—, mas, en cambio, pocas veces se alteraba nada en los directorios «/bin», «/sbin» y «/lib», considerados inviolables.[101]

En mi actual distribución existen otros directorios: uno de ellos es «/virtual» —donde están las máquinas virtuales que puedo correr mediante el programa de Oracle VirtualBox (máquinas con sendas emulaciones de otros sistemas operativos o de otras versiones de mi mismo sistema operativo, el Unix/Linux). Otros directorios nuevos son «/run», «/srv», «/sys», «/snap» y «/media» (de éste último ya he hablado).

Salvo el «/snap» (destinado a las encapsulaciones *snap*, a las cuales me referiré en el capítulo XIX) y el «/srv» (que en mi máquina está vacío por completo, al haberse concebido para servidores), tienen los otros un contenido fugaz: el «/media» da alojamiento a dispositivos externos conectados (que pueden variar de una sesión a otra), al paso que «/run» y «/sys» contienen ficheros temporales que usa el sistema y que, normalmente, no necesita consultar para nada el usuario. (Yo apenas he entendido cuál sea su contenido; confieso no haberme esforzado.)

* * *

Creábase además en Slackware un directorio «/install», cuya única función era alojar provisionalmente paquetes binarios preparados para su inmediata instalación; tales paquetes venían en seguida borrados, dejando vacío el directorio.

Además de los directorios cuya lista figura en los párrafos precedentes, puede el usuario (o, más exactamente, el superusuario, el *root*) crear todos aquellos que tenga por conveniente. Yo, p.ej., desde hace muchos años tengo un directorio «/local», del cual penden subdirectorios con múltiples y variadísimos contenidos, al no desear que engruesen mi propio directorio «/home/lorenzo». (Lo he llamado así, habiendo podido llamarlo de cualquier

---

101. Si no estoy equivocado, esa reducción del «/bin», el «/sbin» y el «/lib» a «/usr/bin», «/usr/sbin» y «usr/lib», respectivamente, ha venido impuesta por la adopción del nuevo sistema de inicialización, *systemd* —cuyos inconvenientes voy a examinar más abajo, en el capítulo XXI.

otro modo.)

Era muy usual en Slackware dividir el disco duro en un número más o menos grande de particiones. Así, frecuentemente, el «/usr» y el «/home» tenían sus propias particiones. Yo había creado, no sólo particiones propias para el «/tmp» y para el «/opt», sino también particiones libremente denominadas por mí para almacenar, v.g., ficheros especialmente grandes (todo, claro, con relación a las capacidades de entonces); p.ej. podía haber una partición que se montara en un directorio especial, «/Verdi», con ficheros audio, mucho más voluminosos que los de texto; u otra montada en un directorio «/Tiziano» con ficheros gráficos; o una «/Visconti», con ficheros vídeo.

Al instalar Ubuntu, resulta un poco menos fácil particionar el disco duro (u otro almacenamiento interno) como a uno le dé la gana. Es, a mi juicio, preferible, para guardar grandes volúmenes de datos (sobre todo ficheros gráficos, de audio o, más aún, de vídeo), crear *links* simbólicos, apuntando a directorios de dispositivos externos de almacenamiento (tarjetas de memoria SD o discos conectados por USB), aligerando así la ocupación de los dispositivos de almacenamiento interno.

Capítulo XVIII

Características del Unix/Linux (3ª parte): elenco de órdenes en el CLI

En no pocas introducciones al Linux amablemente ofrecidas por voluntarios en el internet pónese el acento en mandos impartidos en el CLI que suelen ser dados por usuarios avanzados y por administradores del sistema con un grado alto de competencia.

Por el contrario yo me voy a centrar en órdenes de las más normalitas y corrientes, de esas que tiene que manejar cualquier usuario (cualquier usuario que se adentre en el CLI, no conformándose con el escritorio gráfico).

La primera de esas órdenes es «ls», un poco equivalente al «dir» del DOS. Sólo un poco, ya que la orden «ls» tiene muchos parámetros u opciones que le son propios. «ls -lr» instruye a la máquina para que enumere todos los ficheros que tiene en el directorio activado y en todos sus subdirectorios, facilitándonos, con relación a cada uno de ellos, sus atributos (a qué usuario y a qué grupo pertenece el fichero y cuáles son sus permisos). La opción «-a» determina que también se incluyan en la enumeración los ficheros ocultos, o sea aquellos cuyo nombre comience con un punto (p.ej. el «.bashrc» y en general todos los ficheros de configuración de cada *home* de usuario).

Parecido, sólo que muchísimo más potente, es el mando «find», que tiene una amplia gama de opciones, con efectos variadísimos. P.ej. puede seleccionar ficheros cuyos nombres compartan alguna ristra de caracteres; ficheros del mismo atributo; ficheros cuyos respectivos senderos (*paths*) tengan también algo en común (aunque los nombres de sendos ficheros no lo tengan). Con «find» también se puede dar la orden de que se ejecute un cierto mando para cada uno de los ficheros hallados. Lo cual causa una peligrosidad de esta orden, que ha de manejarse con prudencia, pensándolo bien (La orden «find * -name *a* -exec rm \{\} \;» elimina todos los ficheros cuyo nombre contenga una «a».)

Para encontrar el fichero «xinitrc» por defecto en el sistema úsase la siguiente orden: «find / -name xinitrc». Dado que ha de atravesar todo el árbol de directorios, su ejecución puede demorarse.

La orden «file» nos informa de qué tipo de fichero se trata; p.ej. si es un fichero binario, si es uno de texto ASCII, si es un directorio, si es un enlace

simbólico, etc.

La orden «cat» lee en pantalla el fichero indicado. Hay que evitar aplicarla a ficheros binarios, ya que volcar en pantalla un binario puede acarrear efectos indeseados e indeseables.

La orden «head» es como un «cat» truncado, pues vuelca en pantalla sólo unas pocas líneas iniciales de un fichero. La orden «tail», en cambio, nos vuelca en pantalla las líneas finales.

La orden «touch» asigna a un fichero la fecha de hoy —o aquella que expresamente se diga al emitir la orden—, creando el fichero (vacío, eso sí) cuando no exista. (P.ej. «touch 6uq.txt», no existiendo tal 6uq.txt, lo crea, pudiendo luego editarse con cualquier editor de texto para volcar a él un contenido, como puede ser un bloque tomado de otro escritorio).

La orden «date» nos informa de la fecha actual, mas, con ciertos parámetros, puede alterar la fecha que tenga registrada la máquina.

Úsase la orden «which» para localizar rápidamente un programa (un ejecutable). Busca en el sendero (*path*), devolviendo la primera instancia que halla junto con el directorio en que se ubica. P.ej. «which bash» nos dará como educto «/bin/bash».

La orden «whereis» es similar a «which», sólo que también puede buscar páginas de manual (*man*) y ficheros de fuente (*source files*).

La orden «clear» despeja la pantalla, siendo equivalente a la orden «cls» del DOS.

Las órdenes «more» y «less» sirven para pausar el despliegue en pantalla por páginas. Así «find --help | more» nos informará de cómo funciona esa orden tan sumamente fértil, suministrándonos la información por pantalladas sucesivas.

* * *

La orden «cp» sirve para copiar, sea un fichero, sea un directorio. Así, «cp actuales.doc /local/nuevo/» copiará el fichero «actuales.doc» del directorio activado a un (ya existente) directorio «/local/nuevo/». Como tantas órdenes del CLI, es potentísima, siendo aconsejable (sobre todo cuando uno es un neófito en Linux) dar esa orden con parámetros de cautela, como «cp -vbuip».

La orden «dd» sirve para copiar un fichero a un dispositivo o viceversa —y también un dispositivo a otro. Es prescindible, puesto que en Unix/Linux todo es un fichero, de suerte que esa función la desempeña igualmente la orden «cp». (P.ej. «cp /dev/sda /dev/sdb» equivale a «dd if=/dev/sda of=/dev/sdb».)

Otro mando de uso corriente es «mv» (*move*) —orden igualmente peligrosa—, que sirve para desplazar un fichero de un directorio a otro y también para renombrarlo.

Dado su riesgo, yo, para renombrar ficheros, uso otras aplicaciones que

me bajé hace tiempo, principalmente *ren*, la cual ofrece muchas posibilidades —además de no borrar nunca un fichero sin haber demandado al usuario que confirme esa eliminación, caso por caso. P.ej. la orden «ren "*_32_*.mp3" "#2.mp3"» elimina del nombre de cada fichero afectado toda la ristra inicial hasta la secuencia «_32_», inclusive. Es éste un ejemplo demasiado sencillo, siendo muchísimas —y enormemente complejas— las potencialidades de esa utilísima aplicación.

Para borrar un directorio, úsase «rmdir». Y para crearlo, «mkdir». Si queremos crear un subdirectorio de un todavía inexistente directorio, v.g. «/tmp/jornada/matutino/», agregamos el parámetro «-p» (*parents*): «mkdir -p /tmp/jornada/matutinos».

Bórrase un fichero (mas, en principio, no un directorio) con la orden «rm». Obsérvese, empero, que un fichero borrado con la orden «rm» podría desborrarse con ciertas utilidades —junto con una buena dosis de pericia. Por eso, cuando intentemos suprimir un fichero secreto (p.ej. uno que contenga contraseñas), nos convendrá usar la orden «shred», que lo hará ilegible. (Con el parámetro «-u», la orden «shred» lo extirpa y elimina a la vez.)

La orden «rm -fr *» elimina, sin pedir confirmación, no sólo todos los ficheros del actual directorio, sino también los subdirectorios con sendos contenidos.

Esas arriesgadas órdenes, «rm» y «mv», conviene —cuando todavía un usuario es inexperto— darlas siempre con el parámetro «-i» (de «interactividad»), que nos pide confirmación antes de consumar la destrucción de ficheros existentes.

En el CLI puede un usuario imprudente o aventurero dar órdenes como «sudo rm -fr /*», lo cual borrará la totalidad del contenido del disco (u otro dispositivo de almacenamiento interno). Quien emita tal orden ya no podrá rebutear. Se habrá cargado la instalación, teniendo que reinstalar el sistema operativo; lo peor no es eso, sino que, si, además, no había hecho un respaldo reciente de sus datos, también los habrá perdido irremediablemente para siempre.

No es ésa la única orden arriesgadísima y destructiva. Puesto que en Unix/Linux todo es un fichero, tiene sentido la orden «cp amistades.txt /dev/sda», la cual convierte el disco sda en una copia del fichero de texto «amistades.txt».

Otro modo de eliminar todos los contenidos del disco sda es la orden «dd if=/dev/zero of=/dev/sda»; alternativamente, la orden «echo "Hola" > /dev/sda». (Si el usuario no tiene permiso de escritura, puede preceder esas órdenes de

«sudo», arrogándose los privilegios del superusuario; *vide infra*.)[102]

Esas y parecidas órdenes pueden tener, no obstante, su sentido válido. Supongamos que a un doctorando le ha asignado el departamento universitario el uso de un computador, mas ha de devolverlo, limpio de sus propios contenidos, el día mismo de la colación de grado. Para eliminar, por completo, todos esos contenidos (entre los cuales pueden hallarse datos confidenciales o personalísimos que desee mantener en secreto), el doctorando tiene a su disposición esas órdenes (y varias más) para destruirlos, restituyendo a la Universidad el computador con un almacenamiento interno vacío. (A lo mejor el departamento usa Windows, mientras que, para su propio trabajo, el doctorando había quitado el sistema de Micro$oft, instalando el Linux.)

* * *

La orden «cd» sirve para pasar del directorio activo a otro, que con ese paso vendrá activado. Sin complemento, la orden pasa al directorio *home* del usuario. (Cuando yo imparto esa orden sin añadirle nada, activo mi directorio «/home/lorenzo».)

La orden «pwd» nos dice en qué directorio estamos, o sea cuál está activado. Resulta muy conveniente emitirla para cerciorarse de que la siguiente orden que vayamos a dar la estamos impartiendo en el directorio deseado —y no, por error o descuido, en otro, donde pudiera causar efectos indeseados e indeseables. (Cuál sea el directorio activado no suele figurar en el inductor; tal inductor o incitador [*prompt*] es libremente configurable por el usuario; personalmente, lo he configurado yo de manera que sí me diga, tras el nombre de la máquina, el directorio activado, seguido de la fecha y hora completas así como, ya en otra línea, mi nombre de usuario.)

La orden «mount» sirve para montar un recurso (un *device*) en un directorio. No se podrá acceder a los contenidos de un recurso mientras no esté montado; tal recurso será un dispositivo, una partición o un fichero —en este último caso, ha de tratarse de un fichero especial, que sea imagen de un disco o de una partición. Tratándose de un fichero, ha de montarse con el parámetro «-o loop» (o mediante la orden «losetup»). Podemos montar, de ese modo, imágenes de disquetes, de llaves USB, de CDRoms, de particiones. Así es posible modificar los contenidos de una imagen de disco antes de desmontarla y volcarla, con «dd», sobre un disco real. La orden «umount» desmonta ese recurso.

Gracias a la orden «cryptsetup» gestionamos los dispositivos encriptados, pudiendo así montarlos escribiendo la contraseña correcta.

---

[102]. Imitando a Micro$oft (la cual —en su insaciable concupiscencia— aspira a satelizar al Linux), en una pretensión de puerilizar al usuario —cercenando su libertad de usar su propia máquina como le dé la gana— tienden ahora los gurúes de las nuevas distribuciones del Linux a desincentivar que salgamos del GUI, en el cual estamos controlados y confinados.

* * *

La orden «lp» (o, equivalentemente, «lpr») envía el fichero a la impresora.

Otras órdenes necesarias son: «grep» para hallar ristras de texto dentro de ficheros; «locate», parecida a «grep», aunque con algunas diferencias; «sort» para volcar un fichero de texto sobre otro con el mismo contenido, mas clasificado (normalmente por orden alfabético, existiendo varios parámetros); «sed» para reemplazar, dentro de uno o varios ficheros, una ristra de texto por otra (si bien a mí nunca me gustó el *sed* —cuya sintaxis me resulta excesivamente complicada—, prefiriendo una aplicacioncita llamada «streplace», producida por un programador alemán, Johannes Overmann, utilísima y fácil de usar, incluso recursivamente en un directorio y sus subdirectorios).

La orden «reboot» rebutea la máquina, mientras que «shutdown» y «halt» sirven para apagarla. (Hay diferencias técnicas en las cuales no entro aquí.)

La orden «ps» nos da, como educto, una lista de los procesos en curso; particularmente interesante es agregarle el parámetro «xa». Podemos ver así, a veces, un proceso que está funcionando mal y matarlo con la orden «kill» junto con el parámetro «-9».

Alternativamente la orden «pid XXX» nos informa del número de ejecución del programa XXX; de ser, supongamos, tal número «3871», a continuación damos la orden «kill -9 3871» para matar esa aplicación XXX, lo cual resulta útil cuando se ha atascado, mas sigue ocupando un terminal y un buen trozo de memoria. También podemos dar la orden «killall XXX», si sólo tenemos ejecutándose una copia de esa aplicación.

La orden «lsof» nos informa, no sólo de los ficheros abiertos, sino también de los *unix-sockets* y las conexiones de red.

La orden «top» muestra las tareas en ejecución y el estado del sistema. La orden «dmesg» da como educto una larga serie de mensajes, volcando en la pantalla un registro de los últimos acaecimientos de la máquina, incluidos los errores, siendo particularmente útil justamente para detectar fallas.

Indágase con la orden «groups» a qué grupos pertenecemos. (Al dar yo esa orden, el educto es el siguiente: «lorenzo adm cdrom sudo dip plugdev lpadmin vboxuseres sambashare».)

Hay varias órdenes para averiguar datos de nuestro *hardware* y de nuestro *software*, como las siguientes: «hostname», «uname», «lshw», «lspci», «lsusb», «dmidecode», «biosdecode», «lsmod», «ip» (que nos informa de los recursos de red).

* * *

La orden «ifconfig» nos dice cuál es la configuración de nuestra máquina con relación a nuestra red (wifi o cableada), nuestro IP y la dirección MAC; son datos necesarios si queremos conectar con nuestro computador de sobremesa un dispositivo externo, como una tableta, por alguno de los protocolos samba,

ftp, telnet u otro. (La orden «iwconfig» nos da otra información, mas exclusivamente de nuestra conexión inalámbrica, en caso de tenerla —que es hoy lo más usual, no habiéndolo sido en los viejos computadores, que únicamente podían conectarse por cable ethernet.)

Permítenos la orden «telnet» acceder al CLI de otra máquina (que, siéndonos accesible, tenga activado el demonio telnetd).

Gracias a la orden «ftp» accedemos a otro computador (no forzosamente uno de nuestra propia intranet), siempre que tenga activado el demonio ftpd. (Es mejor la utilidad «lftp», mas hay que buscarla y bajársela.)

«Lynx» abre el navegador más clásico, en modo texto o carácter; resulta hoy limitada su antaña utilidad, al haberse ido, cada vez más, cargando de gráficos las páginas HTML accesibles en la *web*.

«wget» nos permite descargarnos un fichero que cuelgue, públicamente accesible, en un sitio *web*.

* * *

En las distribuciones derivadas de Debian (entre las cuales se encuentran los Ubuntus y Linux Mint) la orden «apt» nos sirve para saber qué actualizaciones están disponibles en los repositorios oficiales de nuestra distribución (u otros que voluntariamente hayamos agregado nosotros a la lista) para, si lo decidimos, bajárnoslos e instalarlos. (A diferencia de Windows, Linux nunca nos fuerza a actualización alguna. En todo momento somos libres y dueños de actualizar o de no actualizar.)

Muy parecida a «apt» de la rama Debian es la orden «yum» o «dnf» de la rama RedHat/Fedora. Son similares, pero no iguales, al poseer cada una sus características propias y sus singulares parámetros. (En Slackware no hubo, durante muchos años, nada similar; agregóse posteriormente el *slackpkg*, un gestor de paquetes, el cual, no obstante difícilmente puede considerarse equivalente a «apt» o a «yum», por varios motivos.) Otras distribuciones tienen sus propios gestores de paquetes y, por consiguiente, otras órdenes, en lugar de «apt» o «yum». (P.ej. en ArchLinux la orden correspondiente es «pacman» —mas de nuevo la equivalencia funcional no es exacta.)

* * *

La orden «reset» nos permite restablecer la configuración normal y esperada de la consola en el caso de que se haya corrompido; v.g., cuando hayamos cometido el error de dar la orden «cat XXX», siendo XXX un fichero binario —y no de texto—, nuestra pantalla se perturbará al punto de no dejarnos comunicarnos con la máquina; remédiase tal perturbación con el «reset».

La orden «ln» nos permite crear un vínculo o enlace, o sea un pseudofichero que apunta a otro. Así, cuando en un directorio, digamos «/local/audio/», tengamos un fichero cual pudiera ser *palabras.mp3* y

queramos acceder a él en nuestro propio directorio personal —en mi caso «/home/lorenzo»—, crearemos en este último directorio un enlace, ya sea con el mismo nombre del fichero aludido o con otro cualquiera, de suerte que, al abrirlo, aquello que, en realidad estaremos abriendo será el citado fichero del directorio «/local/audio/». Así podemos también dar varios nombres a un mismo fichero o directorio. Todo eso puede parecer ocioso, mas, a medida que se acostumbra uno, va percatándose de su enorme utilidad.

La orden «df» (abreviatura de «disk free») nos informa tanto de cuáles dispositivos están montados (los «file systems») cuanto del espacio ocupado y libre de cada uno. Con el parámetro «df -Th» la información recabada es la de cuál sea el sistema de ficheros de cada dispositivo montado además de cuánto espacio esté disponible, todo ello en formato humanamente legible.

Infórmanos la orden «blkid» de los UUIDs de los recursos del sistema tales como discos, otros dispositivos de almacenamiento y sus particiones. ¿Qué es un UUID? Es el *universally unique identifier* de un recurso de almacenamiento; a tenor del estándar RFC9562, trátase de una etiqueta de 128 bits resultante de concatenar 32 guarismos hexadecimales (de «0» a «f») —usualmente escritos en forma de cinco bloques desiguales unidos entre sí por guiones.

Otra orden parecida es «lsblk» —si bien cada una de ellas aporta su particular información.

Genérase aleatoriamente cada UUID —por un procedimiento que aproxima a zero las probabilidades de repetición. Por ello prefiérese hoy, para denominar a los dispositivos montables, llamarlos por sus UUIDs, que, en principio, son inmutables, mientras que no lo son otras denominaciones, como «sda1», «hdb3», etc, rótulos que se generan cada vez que arranca la máquina, la cual va asignándoles tales nombres sin tomar para nada en cuenta cuáles les había atribuido la vez pasada.

Cuando tengamos un script del Bash que corra automáticamente realizando tal o cual operación en un dispositivo, v.g. en «/dev/sdb2», podremos llevarnos la desagradable sorpresa de que, esta vez, la máquina haya asignado ese nombre a un dispositivo diferente del que esperábamos. Lo cual no sucede si llamamos al dispositivo por su UUID.

No obstante, el uso de UUIDs viene afectado por un inconveniente, a saber: cuando hayamos clonado (con la orden «dd») un dispositivo de almacenamiento en otro —p.ej., la partición de arranque de nuestro dispositivo principal de almacenamiento interno, como un NVME, en una partición de un disco interno añadido—, el programa de arranque confundirá ambos dispositivos, ya que el clonaje ha asignado al clon el mismo UUID del original.

¿Solución?[103] Hay que tener instalada la aplicación «e2fsprogs» y lanzar la orden «tune2fs» con unos determinados parámetros. Así dispondremos de dos alternativas para el arranque, en previsión de que una de ellas pudiera corromperse o no funcionar bien.

La orden «du» nos informa de cuánto espacio del disco ocupan el presente directorio (u otro especificado) y sus subdirectorios; tiene varios parámetros, como «du . -s -m», que nos informa de cuántos megabytes ocupa, en su conjunto, el actual directorio, omitiendo información particularizada sobre los subdirectorios.

Resulta «du» tremendamente útil para averiguar cuándo hemos sobrepasado un límite razonable. Si tenemos, v.g., un directorio con mensajes acumulados de correo electrónico (enviado y recibido), con el transcurso de los años puede adquirir un volumen gigantesco (sobre todo teniendo en cuenta que no pocos mensajes de email contienen anejos binarios, como ficheros de texto en PDF o en DOCX, gráficos —sea en PNG, en JPG o en cualquier otro formato—, audios, etc). Cuando hayamos rebasado un límite razonable, sería aconsejable (para liberar espacio en el almacenamiento interno) pasar a un disco externo de respaldo aquellos contenidos antiguos —y ya clausurados— cuya continuación no sea previsible.

Permítenos la orden «at» demorar la ejecución de un binario o de una orden hasta un momento posterior. (Desde luego también se puede planear la ejecución con un fichero configurativo gracias al *cron*, que yo no uso; la orden «at» la usé mucho en el pasado, por razones que ya no existen, habiendo, pues, cesado mi utilización de la misma.)

La orden «man» abre un fichero de texto, un manual, con aclaraciones sobre una orden, un ejecutable o un proceso. P.ej. «man cp» abre un texto aclaratorio de cómo funciona la orden «cp» diciéndonos cuáles son sus parámetros posibles.

Las órdenes «free» y «vmstat» nos informan de la memoria ocupada y de cuánta está libre; son parecidas, sin ser idénticas. La orden «history» nos muestra el historial de las órdenes que hemos ejecutado previamente.

Para archivar ficheros, comprimirlos y descomprimirlos, tenemos las órdenes «tar», «gzip», «gunzip», «zip» y «unzip».

* * *

Hay varias órdenes reservadas al administrador del sistema (*root*, o superusuario): «chmod» cambia los atributos de un fichero (permisos de lectura, escritura y ejecución del dueño del fichero, de su grupo y de los demás

---

[103]. Tómola del documento «Change the UUID of a Partition or Disk» redactado por Michal Albin, disponible en www.baeldung.com/linux/editor/michal-author. Refiérome aquí únicamente a *filesystems* del tipo más característico de Linux, a saber: ext2, ext3 o ext4. El mencionado artículo brinda útiles indicaciones para otros formateos, como fat, xfs y btrfs

usuarios); «chown» cambia el dueño del fichero; «chgrp» modifica el grupo al cual se adscribe el fichero. «passwd» cambia la contraseña; «adduser» agrega un nuevo usuario; «modprobe» carga un módulo; «rmmod» lo descarga.[104]

La orden «fdisk» nos permite, no sólo conocer las particiones de un dispositivo con sus respectivos sistemas de ficheros, sino también modificarlas; esto último resulta peligrosísimo, habiendo de hacerse a sabiendas y con el máximo tino, a falta del cual acarreamos un descalabro.

La orden «gdisk» es igual a «fdisk», con la sola diferencia de que ésta última se aplica a discos con una tabla DOS de particiones (o sea con un MBR, *master boot record*), al paso que «gdisk» se aplica a dispositivos GPT («GPT» significa «GUID partition table», donde «GUID» abrevia a *globally universal identifier*).[105]

La orden «mkfs» nos sirve para dar formato a una partición. Esa orden lleva desinencias indicando qué sistema de ficheros queremos para tal formateo.

Otras órdenes reservadas al *root* son las siguientes. Con «fsck» vemos si está limpio un sistema de ficheros para, si no lo está, poder limpiarlo (para lo cual es menester que no esté montado); la orden «sync» acelera la ejecución de órdenes pendientes que se hallaran en los tampones de memoria; resulta especialmente útil cuando queremos desmontar un sistema de ficheros, mas primero hemos de cerciorarnos de que no quedan tareas pendientes que lo afecten —v.g. que se hayan acabado de copiar los ficheros que queríamos volcar en él.

La orden «chroot» cambia el directorio raíz para ejecutar cierta orden; es una orden destinada a poquísimos casos y a la cual únicamente ha de acudirse cuando se tiene suficiente pericia, sabiendo perfectamente lo que uno está haciendo.

La orden «elabel» sirve para adscribir un nombre o rótulo (un *label*) a una partición con un sistema de ficheros que sea ext2, ext3 o ext4; para bautizar a una partición con sistema de ficheros FAT o VFAT úsase la orden «fatlabel»; otras órdenes sirven para denominar particiones con alguna de los nuevos sistemas de ficheros, btrfs y xfs; tales sistemas están en auge (previendo algunos que desplazarán a los sistemas de ficheros a los cuales estábamos habituados los linuxitas: ext3 y ext4).

* * *

---

104. Hay dos órdenes consustanciales al *systemd*; a mí todavía me resultan extrañas, no habiéndolas aprendido: «systemctl» —para averiguar y gestionar los «servicios» del sistema operativo— y «journalctl» (un poco parecida a «dmesg», ya citada).

105. Ahora todos los nuevos dispositivos vienen configurados como GPT, quedando el MBR para dispositivos viejos.

Hay órdenes que permiten a un usuario que no sea el *root* ejercer algunas funciones reservadas precisamente al *root* o suplantarlo. Son «super», «su», «sudo», cada una con sus peculiaridades. En Slackware usábase «super» mas en las modernas distribuciones se prefiere el «sudo», con el cual un usuario que no sea el *root* puede dar una orden privativa del *root*; eso sí, una por una; para ello ha de escribir su propia contraseña. Este mecanismo se ha establecido a fin de que ningún usuario entre en su máquina como *root*, sino que, para cada orden particular privilegiada que desee ejecutar, la dé como si fuera el *root*. (No estoy convencido de que sea la solución más segura o idónea, mas es la que ha prevalecido).

Están también disponibles muchas pequeñas aplicaciones ejecutables en el CLI, las unas en los propios repositorios oficiales de Ubuntu —o de otras distribuciones—, las otras descargables de sitios reputados y seguros, como GitHub, Sourceforge.net y Codeberg. Rinden importantes servicios.

Entre ese enorme cúmulo de utilidades disponibles para el Linux, mencionaré las del paquete *dateutils* (que yo me he bajado y he compilado hace unas semanas): «dateadd», «dateconv», «datediff», «dategrep», «dateround», «dateseq», «datesort», «datetest», «datezone».

Existen muchísimas más, como: «cal», «ren», «wren», «rename», «lowercase», «iso3166», «setfont», «replaceit», «romandate», «romannum», «strptime», «reminder», etc.

Para el manejo de ficheros audio (principalmente de ficheros mp3) contamos con utilidades como: «id3tool», «mp3cut», «mp3info», «mp3tag» y «mp3wrap».

Para manipulación de PDFs, el *pdftk* (y otras varias utilidades adicionales.)

* * *

Con excepción de aquellas distribuciones especiales del Linux destinadas exclusivamente a administradores de servidores, resulta hoy perfectamente posible ser un usuario de Linux sin abrir el CLI, sin saber siquiera que existe, desconociendo su sintaxis e ignorándolo todo sobre las órdenes que acabo de enumerar. (Ser usuario de Linux como, en la mayoría de los casos, son usuarios de Windows quienes permanecen bajo la servidumbre del sistema operativo de los de Redmond.)

Sin embargo, es infinitamente más fecunda la experiencia que del Linux se tiene en el CLI, abriendo posibilidades mayores, transmitiendo al usuario un sentimiento de potencia y de libertad, incentivando su creatividad, haciéndolo más competente, más capaz, más dueño del *software* que maneja y de la propia máquina.

Capítulo XIX

Las encapsulaciones

Otro rasgo del Unix/Linux es su sistema de librerías, al cual ya me he referido más arriba, más una consecuencia del mismo, que es el reciente surgimiento de las encapsulaciones (*containerizations*).

Típicamente ubícanse las librerías en los directorios «/lib», «/usr/lib», «/usr/local/lib», «/opt/lib» y algunos otros. Una variable de configuración establece dónde buscará u hallará las librerías el sistema operativo.

Según ya lo he señalado, esa cualidad de poder correr ejecutables mucho menos voluminosos mediante la invocación de librerías ofrece la ventaja de un menor tamaño de esos ejecutables y, por consiguiente, que no obstruyan o atasquen la memoria RAM y que sea rápida su iniciación.

Mas el precio a pagar es que cualquier actualización del sistema, cualquier renovación de las librerías, acarreará que quedan inservibles ciertos ejecutables que a uno le estaban prestando buen servicio y que deseaba seguir usando. Es lo que se llama «el infierno de las incompatibilidades».

En Windows cada nueva versión (del 95 al 98, al Millenium, al Vista, y así sucesivamente) suele comportar que dejan de funcionar ciertos ejecutables (y también *drivers*) que antes funcionaban. Windows manda; a los demás tócales obedecer. Los productores de esos *drivers* y ejecutables no tienen más que modificarlos para adaptarse a la vigente versión de Windows.

No así en Unix/Linux. Ni los usuarios ni los productores de *software* somos sus siervos ni sus vasallos. Además, esa renovación de librerías varía según las diversas distribuciones. Ningún desarrollador de un *software* puede estar al tanto de cómo anda la renovación de librerías en esta distribución o en aquélla.

Por eso púsose de manifiesto la conveniencia de, muchas veces, compilar un ejecutable estáticamente, lo cual implica que el propio ejecutable contendrá en su interior un duplicado de cada una de las librerías por él invocables en su ejecución. No siempre resulta factible ni deseable. A veces es posible convertir un ejecutable dinámicamente compilado (o sea que tiene, en su ejecución, que invocar librerías a él externas) en uno estáticamente compilado, ya sea acudiendo al programa *statifier* —me temo que poco eficaz

(pero, para quien lo quiera probar, disponible en los sitios github y sourceforge.net)—, ya al mágico Ermine.[106] Según su desarrollador,[107] el resultado de aplicar Ermine a un ejecutable dinámicamente compilado es una máquina virtual, no susceptible de los tropiezos e inconvenientes que afectan a otros procedimientos, como el del *statifier*.

Las nuevas distribuciones de Linux han buscado una solución a ese problema de las incompatibilidades de librería y de las disparidades entre distribuciones diversas (justamente por su diferente elenco de librerías, no actualizadas de modo sincronizado). La han creído hallar en las encapsulaciones (*containerization*).

Una encapsulación no es forzosamente un mero ejecutable estáticamente compilado, pudiendo ser un paquete que contiene en sí las librerías precisas para su respectivo ejecutable, prescindiendo de librerías ajenas. (Al menos en teoría o sobre el papel, pues acaso las cosas sean más complicadas). Se ha dicho que las encapsulaciones generan ejecutables inmutables (adjetivo sin duda hiperbólico) y agnósticos (en el sentido de que pueden correr en distribuciones dispares).

Hay tres principales modalidades de encapsulación actualmente: *snap* *flatpak* y *appimage*. (A diferencia de las otras dos encapsulaciones, *appimage* sí compila el ejecutable estáticamente, formando un único fichero como un bloque monolítico, en vez de una colección de ficheros.)

Snap es propio de la casa Canonical, viniendo cada vez más utilizado en las nuevas versiones de Ubuntu (y de sus varios sabores). En la versión 22.04 de Lubuntu —que es aquella que está usando el autor de estas páginas— es escasísima, casi marginal, la presencia de *snaps* (sobre todo tras haber eliminado el Firefox y el Chromium, navegadores hinchados y no tan amigables como suele creerse —que, en todo caso, resultaban para mí redundantes y ociosos).

Flatpak es la encapsulación preferida en el grupo de distribuciones cuyo eje es RedHat, como Fedora, aunque también se han desarrollado para Debian, Mageia, Solus e incluso Ubuntu.

Son durísimamente criticados los *snaps* por quienes sienten animadversión a Canonical y menosprecio por Ubuntu. De esas críticas, selecciono cuatro que me parecen más pertinentes. (Otras ni siquiera las he entendido.)

1ª.— El *software* utilizado para generar snaps es privativo de Canonical (y no libre).

Lo privativo de Canonical no son los *snaps*, ni siquiera el procedimiento

---

106. Accesible en magicermine.com/sesl.html.

107. Un tal Valery, de Tel Aviv.

de encapsulación, sino exclusivamente el *software* usado en su plataforma para poner los *snaps* a disposición de los usuarios. No son privativos ni los programas encapsulados, ni el producto de la encapsulación —el propio *snap*— ni el demonio que corre en nuestra máquina y con el cual actívanse los *snaps* ni el *software* usado para generar *snaps*, el *snapcraft* —un programa que yo no tengo (dudo que sea compatible con mi versión del Lubuntu, requiriendo la 24.04, más reciente). Todo eso es *software* «libre». Y el *software* «libre» no se esclaviza por alojarse en un envoltorio cuyo productor use un *software* no-libre (igual que la carta que enviamos —cuyo contenido libremente escribimos nosotros mismos— no se somete a la servidumbre ni del fabricante del sobre ni de la agencia de transporte que la transmite a su destinatario).

Es más, con la difusión del programa *snapcraft* —que puede correrse en versiones más recientes de Ubuntu—, el usuario que lo desee puede, si lo desea, producir sus propios *snaps*, proponiéndolos para su aceptación en el portal de Canonical. Disponiendo de suficientes conocimientos informáticos, puede también modificar una dirección ínsita en el *snapd* (e imagino que también en el *snapcraft*) que orienta al portal de Canonical como fuente de los *snaps*; así, el creador de un *snap* propio podrá distribuirlo por su cuenta, totalmente al margen de la empresa Canonical.

Al usar *software* propio y privativo en su procedimiento de difusión, Canonical no está transgrediendo ninguna de las obligaciones asumidas con respecto al *software* libre que distribuye. Ni la versión 2 ni la 3 de la licencia GPL de la FSF (*Free software Foundation*) imponen deber alguno de no usar *software* no libre en el proceso de distribución del *software* libre,[108] sino únicamente abstenerse de que el contenido distribuido abarque *software* no-libre.[109]

Lo que es cierto es que *snaps*, por ahora, sólo los hay en el repositorio de Canonical (si bien eso puede dejar de ser así en cuanto alguien se anime a modificar los programas *snapd* y *snapcraft*, eliminando la prefijada dirección al portar de Canonical), mientras que, al parecer, los otros dos tipos de encapsulaciones pueden hallarse en varios repositorios.

2ª.— Canonical fuerza a sus usuarios a usar *snaps*.

No es cierto. Ningún usuario de Ubuntu está constreñido a usar *snaps*.

---

[108]. Dígolo, no obstante, sin pleno convencimiento, ya que, meditando hondamente en torno a esas licencias (sobre todo la versión 3) y leyendo informaciones sobre las peleas exegéticas a que han dado lugar (potencialmente conducentes a pleitos), me doy cuenta de lo sutilísimas que son sus implicaciones y de la ancha franja de vaguedad que rodea sus cláusulas. Sin embargo, hasta donde yo sé, los guardianes de la ortodoxia liberista, como la *Free software Foundation*, no ha impugnado la implementación de los *snaps* por Canonical.

[109]. Deberíamos escribir «libre» entre comillas, no sólo porque el *software* no puede gozar de libertad, sino, además, porque la difusión del llamado «*software* libre», en su versión *copyleft* [la de la FSF], no otorga al receptor libertad, como sí hacen la licencias llamadas «permisivas»; el *copyleft* concede al receptor determinadas libertades, rehusándole otras.

Quien no los quiera puede perfectamente suprimirlos, uno por uno, para, en su lugar, descargar y usar otro *software*, o el mismo no encapsulado en ningún *snap*. Yo mismo —además de haber eliminado de mi máquina el más sonado *snap*, el Firefox, un navegador que no me gusta—, me bajé e instalé un *flatpak* (aunque, tras haber comprobado su falta de utilidad para mí, lo acabé borrando).

3ª.— El lanzamiento de ejecutables así encapsulados, en sendos *snaps*, resulta más lento que el de los mismos ejecutables sin encapsular.

A mi juicio reviste escasísima importancia perder unos pocos segundos frente a las ventajas de la encapsulación: la integridad del ejecutable con sus módulos o librerías y la prevención de colisión en las dependencias.

4ª.— Cada *snap* monta un dispositivo virtual, un *loop*. Voy a explicarlo.

En Unix/Linux un dispositivo montado es lo equivalente a un conducto o *drive* del DOS y del M$-Windows (denomínanse esos conductos, en DOS y en Windows, «A:», «B:», «C:», «D:», etc.)

Normalmente lo que se monta es un *block device*, o sea un disco o una partición de disco (o de otro soporte de almacenamiento, como un NVME o una tarjeta de memoria). Sólo que en Unix/Linux también pueden montarse dispositivos virtuales, o sea ficheros que vienen montados en sendos *loops* (bucles), con la orden *losetup*.

Siendo finito el número de tales posibles *loops*, el que los *snaps* acaparen unos cuantos de ellos disminuye la reserva de *loops* a disposición del usuario.

Dudo que eso pueda ser serio inconveniente. Salvo la queja en abstracto, no he hallado, en ningún foro, narración concreta de algún usuario a quien se le haya quedado corto el número de *loops* disponibles por culpa de los *snaps.* Y es que el tope a ese número viene fijado en el kernel o en un módulo; el sistema operativo, sin intervención nuestra, lo determina automáticamente a un nivel holgado para, tras haber montado todos los *snaps*, dejar a nuestra disposición *loops* de sobra. (El usuario puede crear *loops*, aunque seguro que sólo hasta donde alcance el límite prefijado, que debe ser siempre más alto de cualesquiera necesidades previsibles.)

En mi propio uso, no me ha ocasionado dificultad alguna la existencia de unos cuantos dispositivos virtuales montados como loops por sendos *snaps*.[110]

¿Cuál de los tres procedimientos de encapsulación es mejor? Francamente considero que es una cuestión técnica que merece venir debatida

---

110. Resulta muy corriente y práctico montar ficheros como *loops*; v.g. imágenes de disquetes (de aquellos floppies de antaño), o de llaves USB, o de particiones enteras, o de CDs o de tarjetas de memoria. Yo lo hago a menudo sin que los *snaps* me lo hayan obstaculizado ni dificultado.

y estudiada por expertos en ingeniería informática.

En mi propia distribución, Lubuntu 22.04, pocos son los ejecutables encapsulados. (Ésos pocos, *snaps*.)

* * *

¿Tendré que actualizar el sistema operativo? ¿Resultará entonces afectado por el infierno de incompatibilidades aquel programa que mimo yo como las niñas de mis ojos, el Dosemu? Para prevenirme de tal eventualidad compré en junio de 2024 una licencia para usar el Ermine, habiendo así *erminiado* mi Dosemu (en la versión 3.6.24). Lamentablemente, se ha saldado por el desengaño el ensayo realizado de hacerlo correr en el Lubuntu 24.04 (a través de una máquina virtual, con el programa VirtualBox de Oracle).[111]

Ante tal decepción, recurrí a un intento desesperado, que causó la congelación y parálisis completas de mi máquina, lo cual me forzó a reformatear el almacenamiento interno, reinstalando, desde zero, el sistema operativo.[112]

¿Puede un usuario encapsular ejecutables en su propio computador? Ya he dicho que actualmente puede crear sus propios *snaps*.

También puede crear una *appimage* —la cual, por otro lado, semeja poder aplicarse únicamente a ejecutables de modo gráfico, a programas de desinencia «.desktop». El procedimiento de encapsulación por *appimage* resulta arduo y complejo; además, hasta donde he alcanzado a ver, no es posible encapsular un ejecutable que yace en el propio computador, sino que ha de aplicarse esa operación a un paquete, v.g. un fichero *.deb*, bajado de un repositorio. (De momento, llévanme, provisionalmente, mis indagaciones a la conclusión de que para esa operación hace falta una pericia que rebasa mi nivel de competencia como usuario.)[113]

---

111. No puedo, por consiguiente, estar satisfecho con el Ermine, al no haber colmado mis expectativas.

112. Achaco yo la culpa a la estructura *systemd*, de la cual voy a ocuparme en el capítulo XXI). (Alternativamente podría ser responsable de ese descalabro la aplicación inmunitaria AppArmor.)

113. Invito al lector a consultar la información comparativa brindada en «Comparison Between Snaps, Flatpak, and AppImage Packages», https://www.baeldung.com/linux/snaps-flatpak-appimage.

Capítulo XX

El problema de la seguridad electrónica

Está en la mente de todos el problema de la seguridad, evidenciado por el apagón electrónico del 19 de julio de 2024, que paralizó a todo el Occidente.

Sólo que únicamente afectó a servidores que corrían Windows. Ninguno con Linux.

Han aducido los turiferarios de Micro$oft que ni era culpa de la propia Micro$oft (sino de la compañía dizque de seguridad electrónica, Crowd Strike) ni el Linux es inmune a posibles ataques así.

Según las declaraciones de unos y otros, resulta palmario que Micro$oft había confiado a CrowdStrike el secreto que le permitió perpetrar el ataque. Alega Micro$oft que quedaba obligada a ello por un acuerdo al que había llegado con la autoridad de la Unión Europea, tendente a no monopolizar los servicios de seguridad de su propio sistema operativo. En Bruselas desmienten esa lectura del acuerdo, que se ha hecho público en la página *web* de Micro$oft.

El hecho es que siempre han argumentado los de Redmond que una ventaja de que su *software* no sea de fuente abierta es el secreto mismo de su estructura, que dificultaría o impediría los asaltos; a lo cual se añaden las ulteriores y reiteradas actualizaciones de seguridad (imperativamente impuestas a los usuarios, quiéranlas o no —ya que el Windows no se compra, sino que se alquila, reservándose el arrendador permanente derecho a alterar el producto —alteraciones que el arrendatario tiene, contractualmente, deber de soportar).

Mas, cuando ese secreto —presuntamente protector para los consumidores— lo revela Micro$oft a terceros, tenemos un secreto compartido, que deja de ser secreto. Con todo, algo de razón llevan, por una vez, los de Redmond. Un usuario de Windows está confiando en Micro$oft; confiar también en unos oferentes de más protección implica someterse a varios superiores, entre los cuales puede haber colisiones.

Desconfío yo totalmente de la protección que brindaría el secreto del *software* de Micro$oft, sabiendo de sobra que hasta las encriptaciones presuntamente invulnerables pueden descifrarse, según es bien sabido por la experiencia histórica.

No es que profese yo la teoría opuesta de que el *software* de fuente

abierta está a salvo justamente porque cualquiera lo puede revisar, advirtiendo sus fallas. Demasiado optimista es esa apreciación, dada la enorme dificultad de lectura de los millones de líneas del código del kernel de Linux. Mucho tiempo y muchos conocimientos hacen falta para esa inspección. Ni un servidor ni, verosímilmente, el amable lector estaríamos en condiciones de emprender tal tarea.

Todo el *software* es vulnerable, pero, hasta donde yo sé, ya blindan suficientemente al usuario de Linux las medidas de protección incorporadas al kernel (a las versiones actualizadas del kernel) más otras suplementarias que contienen las distribuciones recientes, sin que sea razonable que ningún administrador de un servidor con Linux entregue las llaves a terceros.

Y ¿quiénes son esos terceros? La firma CrowdStrike la fundó y la acaudilla el bachiller George Kurtz; individuo cuyos estudios se limitan a un modestísimo grado en contabilidad, sin que se sepa de capacitación suya en informática.

En octubre de 2009 la empresa productora de antivirus McAfee lo había nombrado jefe tecnológico y vicepresidente ejecutivo. Seis meses después McAfee causó el derrumbe de sus clientes en todo el mundo, al lanzar una actualización de su *software* que borraba ficheros críticos del sistema Windows XP, impidiéndoles, además, rebutear. Comentándolo, escribió Ed Bott en *ZDNet*: «No estoy seguro de que un productor de virus haya sido capaz de desarrollar un *malware* que haga tanto daño como el causado hoy por McAfee».

Según *Forbes* la fortuna del bachiller George Kurtz sumaba 3.628 millones de dólares antes del cataclismo de julio, habiéndose reducido un poco después.

Por otro lado, resulta extraño que Micro$oft se someta a las pretensiones de Bruselas en todo el mundo. Puede perfectamente distribuir en Europa una versión de su kernel (incorporado a Windows), emitiendo otra en el resto del mundo. Eso habría evitado la catástrofe en USA, Australia, la India, etc. (De ser verdadera la defensa de Micro$oft, todos los afectados han de saber que la comisión de la Unión Europea es cómplice del agravio.)

Aun en el supuesto de que sea verídico y exacto lo que alega Micro$oft en su propia defensa, ¿cómo así no interpone, entre las actualizaciones de tales vigilantes y su admisión en los computadores con Windows, un peldaño de precaución, que sería su previa inspección por ingenieros de la propia Micro$oft?

Nada exonera, pues, a Micro$oft de la mala calidad de su sistema operativo tan fácilmente explotable, tan vulnerable.

En lo atinente a Linux, en absoluto está éste igualmente desamparado, ni siquiera en aquellos casos de administradores de servidores Linux que se han dejado seducir por los embaucadores de Crowd Strike (o de cualquier otro *strike*). Tiene el Linux sus propias protecciones.

Lo malo de las mismas es que son a veces tan rigurosas que con ellas sucede lo que con el sistema inmunitario que nos defiende de las infecciones, pero cuyas nada anómalas disfunciones causan también las graves enfermedades autoinmunes.

Con relación al computador doméstico —el que Ud tiene en su casa, amigo lector o el que tengo yo— peca tal vez el Linux de exceso de protecciones; exceso, insisto, para el usuario casero —a menos que sea un cabeza loca o un atolondrado que, de puro desprevenido, pique en cualquier anzuelo (como responder a mensajes de *wasap* o de SMS sobre un paquete postal que no se ha podido entregar o sobre una devolución tributaria).

Son, en cambio, adecuadas las precauciones para entornos donde se da exposición al peligro (principalmente servidores de empresas o entidades abiertas al público, sean oficiales o privadas). Y las distribuciones de Linux las han tomado muy en serio.

1º. Ofrecen un parapeto el sistema de propiedad de los ficheros y de diferenciados permisos de usuario (permisos de escritura, lectura y ejecución). Si Ud cae en la trampa de, abriendo el email, en un momento de inatención, picar en un enlace malévolo que le hinque en su máquina un *malware*, existen posibilidades de que quede contenido por la barrera de permisos de usuario (a menos, claro, que abra el email con los privilegios del superusuario, lo cual dudo que haga nadie en su sano juicio).

2º. En la gran mayoría de los casos, aquellas aplicaciones que vaya a correr en su máquina un linuxita se las habrá bajado de repositorios oficiales u oficiosos de su respectiva distribución, donde se han filtrado los programas. Claro que eso no constituye una plena garantía, ya que los administradores de tales repositorios no han podido revisar cada línea de código; siendo tantas las aplicaciones que ofrecen, también ellos pueden incurrir en descuido, dando, desprevenidamente, su visto bueno a alguna aplicación maligna. Sin embargo, eso resulta improbable (de hecho no sé que haya sucedido). Aun en ese supuesto, sería inverosímil que tal aplicación pudiera ejecutarse con los privilegios del kernel.

En cuanto a los otros sitios de los cuales nos bajamos aplicaciones adicionales algunos usuarios (entre ellos el autor de estas páginas), generalmente los filtramos con sumo cuidado, en virtud de su merecida reputación —como, v.g., la de sourceforge.net.

3º. En las modernas distribuciones del Linux tienen potestad de policía ciertos programas, como el SELinux (*Security Enhanced Linux*) —usado en RedHat, Fedora y varias distribuciones más, principalmente en servidores— y el AppArmor, usado en otras distribuciones, entre ellas Ubuntu (AppArmor parece estar más orientado al *desktop*, al computador casero o personal.)

Es misión de ambos programas vigilar a las demás aplicaciones, matando aquellas cuyo comportamiento no se ajuste a las reglas y levantando un muro protector del kernel contra cualesquiera intromisiones.

¿Que hace concretamente el AppArmor? Cito su propia autodescripción:

> AppArmor is an easy-to-use Linux Security Module implementation that restricts applications' capabilities and permissions with profiles that are set per-program. It provides mandatory access control (MAC) to supplement the more traditional UNIX model of discretionary access control (DAC). (…) It uses profiles of an application to determine what files and permissions the application requires. Some packages will install their own profiles, and additional profiles can be found in the apparmor-profiles package.

Según esa confesión, el AppArmor limita nuestra libertad de uso de nuestra propia máquina, para protegernos, por nuestro bien. Sólo que sentirse uno controlado resultará admisible y natural para quienes estén habituados al puerilizante Windows. No así a quienes venimos de muchos años de libertad con Slackware, el cual, sin barreras, nos dejaba actuar a nuestras anchas. Toca resignarse al AppArmor, mas añoramos la libertad, que nos parecía consustancial con el Linux.[114]

Para los servidores —sobre todo para aquellos que custodian instalaciones de responsabilidad colectiva o que dan acceso a las mismas—, todas las precauciones son pocas; para tales fines está probada —y viene generalmente reconocida— la fiabilidad de Linux (no inexpugnable mas sí bien fortificado contra cualesquiera asaltos). Ofrecen una significativa tranquilidad las dos arquitecturas de seguridad (SELinux y AppArmor); en el supuesto de que —por inadvertencia propia o por malevolencia ajena— se hayan introducido en nuestra máquina aplicaciones que atenten a la integridad del sistema, cumplen esos programas su función de vigilancia y protección del kernel, neutralizando tales aplicaciones potencialmente dañinas.[115]

4º. Al haber muchas distribuciones de Linux —por más que, en el fondo, todas sean Linux (y por consiguiente iguales en un 80%)—, tienen sus diferencias, quedando cada una sujeta a unas pautas propias de actualización, incluyendo la securitaria.

De haberse dirigido contra sistemas Linux un ataque como el lanzado por Crowd Strike el 19 de julio de 2024, es dudoso que hubiera podido, indistintamente, afectar a los RedHat, a los Fedora, a los Arch, a los Slackware, a los Debian, a los Ubuntu, y así sucesivamente.

Además, los computadores con una misma distribución no están unidos

---

114. Me temo que esos controles, junto con la preeminencia excesiva del entorno gráfico y el arrinconamiento del CLI, son derivas que hacen al Linux de hoy más parecido al Windows, del cual no quisimos saber nada los viejos linuxitas.

115. A esos dos programas de seguridad, AppArmor y SELinux, agrégase la guardia montada por el *systemd*, del cual en seguida voy a hablar mal (mal porque reduce la libertad del usuario en su propia máquina). ¿Habría, por mor de la objetividad, que reconocerle un lado positivo, a saber: que, con *systemd*, queda permanentemente sujeto a control el funcionamiento del computador? Sin duda es ése uno de los argumentos que esgrimen sus adeptos, mas la experiencia lo desmiente (v. en el capítulo XX lo que sucedió con las utilidades *xz*).

en ninguna red ni son iguales entre sí. Cada administrador es perfectamente libre de recompilar el kernel, de agregar o quitar módulos, de eliminar o añadir aplicaciones y de ligarse con una red o no hacerlo. Conque un ataque como el de CrowdStrike contra, v.g., Ubuntu de ningún modo iba a afectar a todos los Ubuntus.

Esa variedad interna del Linux ofrece una barrera al contagio similar a aquella que los cultivadores forestales, escarmentados, saben practicar intercalando en los bosques diversas especies de árboles, lo cual no sólo frena la propagación de epidemias sino que también puede servir de obstáculo frente a los incendios.

Ya sé que no faltan gurúes de servidores Linux —generalmente amedrentados o coaccionados por sus empleadores— que, tal vez cediendo a exigencias de compañías de seguros, se someten a la intromisión de esa, como mínimo, sospechosa firma Crowd Strike, en vez de confiar en los propios dispositivos de seguridad que les ofrece su sistema operativo.

De hecho, en abril y mayo ya había lanzado Crowd Strike sendos asaltos contra algunos de esos servidores. Tuvo el primero como víctimas a ciertos servidores con la distribución Debian. Jack Cushman, director del Harvard Library Innovation Lab, informó: «CrowdStrike lanzó una actualización el viernes al anochecer. Resultaba incompatible con la última versión estable de Debian», lo cual hizo colgarse a todos los servidores conectados con CrowdStrike. Pronto se averiguó la causa del mal, subsanándose sin haberse producido ninguna desgracia.

En mayo, CrowdStrike atacó a servidores con otra distribución de Linux, la RockyLinux. El más notorio de los afectados fue un sanatorio. Pronto se descubrió la raíz del problema, remediándose sin consecuencias mayores. Bastaba volver al kernel precedente o quitar el Falcon (el *software* presuntamente securitario emitido por CrowdStrike.)

Aunque hubiera sido más generalizado el impacto, únicamente habría afectado, a lo sumo, a todos los servidores con una particular distribución (todos aquellos que, además, imprudentemente se hubieran sometido a CrowdStrike). El impacto sobre los servidores con Debian no afectaba a aquellos que corren Ubuntu o RedHat o LinuxMint o Fedora. (Más significativamente, el ataque a RockyLinux no parece haber afectado a otras distribuciones de la misma rama familiar, RedHat, Suse y OracleLinux.)

Ya he dicho que aquello que me lleva a confiar en Linux no es que sea de fuente abierta, sino los cuatro argumentos que he expuesto, junto con la demostrada honradez y valía de Linus Torvalds (cuya supervisión abarca las aportaciones de 25.000 colaboradores)[116] junto con la gran profesionalidad

---

[116]. Cualidades sobradamente demostradas, que no obstan a defectos que se le han reprochado, como su chovinismo, su altivez e insolencia —que lo llevan a veces a la grosería—, así como su carencia de pulcritud deontológica para deslindar lo profesional de lo político. Tales anomalías han dado lugar a broncas y rupturas

de los planteles al frente de las distribuciones de mayor reputación (incluyendo las tres que yo he usado: Slackware, Fedora y Ubuntu).

En todo caso, dentro del horizonte Linux hay recursos de protección adicionales, sin duda oportunos para entornos de riesgo. P.ej., la empresa tecnológica rusa Rusbitech produce un *soft/hardware trusted boot control module*, el MAKSIM-M1, que impide el acceso no autorizado ofreciendo otros rasgos de acentuada seguridad digital. Diseñado para la distribución rusa Astra Linux, también da soporte ese módulo a los demás sistemas con el kernel de Linux (del 2.6.x al 5.x.x) e incluso al mismísimo Windows.[117]

Notemos que el asalto del 19 de julio de 2024 no ha causado muertes. Y es que ningún servidor de vital importancia corre con Windows. Todos usan Linux (y los pocos que no lo hacen usan otro Unix).

* * *

La única falla seria de seguridad registrada en Linux ha sido la de las utilidades XZ, premeditada y alevosamente causada por un adiestradísimo y artero infiltrado, Jia Tan —prontamente descubierta y atajada en marzo de este mismo año, 2024.

Constituyen esas utilidades un módulo de compresión usado en el acceso a un servidor remoto. En la versión binaria de tal módulo (no así en la fuente) había embutido el simulado Jia Tan una cuña, gracias a la cual, al establecerse una conexión por SSH, quedaría abierta una portezuela trasera por donde, eventualmente, podría penetrar en el servidor un intruso, hincando en él algún programa maligno (*malware*).

Respecto al o a la presunt@ Jia Tan (nombre obviamente usurpado por el o la maleante/a), desconócense su identidad y sus motivaciones. CrowdStrike sostuvo que se trataba de un *hacker* profesional a sueldo de los servicios secretos rusos (¿cómo no?). Alegaron otros que, a la vista del caso Snowden, no resultaría inverosímil que se tratara de un jaqueo de la NSA estadounidense. ¿O fue un agente chino? A cada una de esas hipótesis pueden oponerse serias y convincentes objeciones. En China y Rusia (y en la propia América) úsanse versiones propias del Unix/Linux cuya seguridad hubiera podido verse indirectamente comprometida de haberse propagado esa referida cuña de las utilidades *xz*.

---

evitables. Peor que eso es haber constituido un factor del episodio más bochornoso en la historia del kernel Linux, a saber: la discriminatoria exclusión de los contribuidores rusos el 22 de octubre de 2024, presuntamente en acatamiento a un decreto de Biden, sin que se hayan aclarado las exactas obligaciones legales que hacían imperativa tal exclusión —masiva y generalizada— precisamente en ese momento, en vísperas de unas elecciones cuyo previsible resultado podría traducirse en una revocación del citado decreto (*executive order*). (Hasta donde yo sé, nadie ha probado que, en el ordenamiento jurídico de USA, fueran inviables recursos administrativos o contenciosos para esquivar la perentoria y deletérea ejecución de ese malhadado decreto.)

117. Sólo que claro, ¿quién no teme hoy venir hostigado si recurre a soluciones rusas?

Han conjeturado otros que Jia Tan no trabajaba para nadie, siendo posiblemente un programador despechado, que, tras años contribuyendo, gratis, al desarrollo de algún *software*, se vengó por no haber podido acceder a una plaza remunerada.[118]

Trátase de fantásticas elucubraciones. Resúltame intrigante que Crowd Strike —una firma ciertamente no *al di sopra di ogni sospetto*— emita una hipótesis sobre un asunto en el cual, presuntamente, no tendría arte ni parte.

Asómbrame que no se haya debatido adónde apuntarían las sospechas de plantearse la cuestión: *Cui bono?* No «¿quién hubiera podido, de haber prosperado la contaminación —manteniéndose oculta durante suficiente tiempo—, aprovecharse de ella para, eventualmente, infiltrarse en determinados servidores a través del protocolo SSH?» (lo cual resulta especulativo, indirecto y a medio/largo plazo), sino, más rectamente y a brevísimo plazo, «¿a quién habría beneficiado que la contaminación pasara desapercibida —por algún tiempo, el suficiente para facilitar que se comprometiera efectivamente la seguridad de algún servidor, lo cual habría desprestigiado al sistema operativo Unix/Linux y al *software* gratuito?».

Lo que ha sucedido, sin embargo, es todo lo contrario. La infección sólo había afectado a versiones preliminares (pre-beta) de las distribuciones Debian, Fedora, Ubuntu y Kali. A ninguna distribución efectivamente difundida al público. Habíase detectado (y rápidamente subsanado) la disfunción antes de sendas difusiones. (Canonical aplazó en una semana la difusión de la versión beta de Ubuntu 24.04, habiéndose cerciorado previamente de su completa limpieza.)

Más concretamente, la infección únicamente amenazó a versiones preliminares, de mero ensayo, de aquellas distribuciones que compartían dos rasgos comunes:

1º. Que su *software* viniera empaquetado con uno de los dos procedimientos de empaquetación *deb* y *rpm* —lo cual excluye a los de la familia Arch, a los Slackware y otros.

2ª. Que fueran distribuciones con *systemd*, ya que, en verdad, lo contaminado fue un módulo de *systemd*.[119]

---

[118]. De paso así se ataca al *software* gratuito que, en parte, viene posibilitado por las voluntarias aportaciones de muchos programadores benévolos.

[119]. Desencadenóse, a este respecto, una polémica entre partidarios y adversarios de *systemd*. Afirmaron los últimos que el *systemd* no puede librarse de la sospecha de haber contribuido a la infección —sin quererlo, evidentemente—, puesto que, constituyendo un mecanismo envolvente que extiende sus tentáculos a todas las áreas del funcionamiento del sistema operativo, presta el flanco a que se propaguen las fallas. A lo cual responden los primeros que el haber sufrido infección no es culpa del infectado, pudiendo haberse dado que, aun sin el *systemd*, se hubiera consumado la contaminación.

En vez de una disputa de opinantes foreros, merecería un examen técnico tal tema de discusión. Lo que,

Ningún servidor ha sido víctima de asalto por esa vía. La experiencia ha corroborado la seguridad y la fiabilidad de Linux.

en cualquier caso, me queda claro es que no se han probado las virtudes defensivas del *systemd*.

Capítulo XXI

Inicialización y control: la polémica sobre *systemd*

Es propio del Unix/Linux su sistema de inicialización. Mejor dicho, sus varios sistemas. Aquí entramos en zona de trincheras, en campo de contienda: la guerra entre partidarios y adversarios del nuevo sistema de inicialización, el desafortunado *systemd*, el cual ha ido sustituyendo, en la gran mayoría de las distribuciones, al viejo sistema heredado del Unix.

Tradicionalmente, todas las distribuciones del Linux habían retomado el sistema de inicialización del Unix, si bien con dos modalidades un poquito diversas: el Slackware había tomado el sistema de inicialización de Berkeley, el del BSD, al paso que las demás distribuciones (ignoro por qué) habían preferido el de System V (o sea el de la AT&T —al cual ya he tenido ocasión de referirme más arriba).

Habiendo sido yo, durante 26 años, usuario de Slackware (una distribución de la cual se ha dicho que es más UNIX que Linux), estaba tan familiarizadísimo con su sistema de inicialización que ahora, con este nuevo (y, para mí, poco deseable), el *systemd*, me siento coartadísimo y hasta atemorizado. De hecho sé que debo comportarme muchísimo más disciplinadamente, no pudiendo transgredir como antes los límites que me tracen los desarrolladores del sistema, so pena de catástrofe. (Con el agravante de que tales límites no constan en ningún catálogo que fije las prácticas prohibidas, habiendo de adivinarse.)

Está concebido este nuevo sistema para regimentar a los usuarios, por su propio bien, por su seguridad, al precio de que actuaciones no autorizadas —que antes, simplemente, causaban una molestia pasajera— ahora puedan forzarnos a reformatear el almacenamiento interno para reinstalar el sistema operativo. Cualquier usuario razonable será, pues, obediente, respetando esas prohibiciones. (Lo malo es que —como lo acabo de decir ya— han de conjeturarse, habiéndolas guardado en su pecho los conceptores del nuevo sistema.)

* * *

Seguramente está anhelante el lector de que le diga qué diantre sea un sistema de inicialización. Es lo que ejecuta el computador en saliendo del kernel. Le pido paciencia, ya que, para explicarlo, hay que dar un rodeo.

Al encenderse el computador, primero entra en ejecución el *firmware* preinstalado de fábrica (tradicionalmente era el BIOS, pero ahora es otra cosa, el UEFI; omito aclarar la diferencia entre ambos, que yo mismo he entendido mal, por lo cual carecerían de valor mis explicaciones).

Para avanzar desde el *firmware*, es menester un gestor de arranque (*boot manager*).

En muchos casos, el gestor de arranque es también un cargador de buteo, *boot loader*, que toma posesión de la memoria, preparándola para la introducción del kernel, al cual cede el paso a continuación.

Esa doble misión —la de *boot manager* y la de *boot loader*— asúmenla programas como el viejo Lilo de Slackware y el hoy más generalizado —del cual personalmente tengo yo una dolorosa experiencia—, el GRUB (un *boot manager*, para mí, errático y propenso a dejar en la estacada al usuario).

Por el contrario hay programas que únicamente actúan de *boot manager*. No toman posesión de la memoria, limitándose a abrir la puerta al kernel, el cual actúa como su propio *boot manager* —o, para usar tal metáfora, como su propio acomodador. Aunque el kernel actual tiene esa virtualidad, no sucedía lo propio antes, lo cual justificaba la necesidad de un *boot manager* que también actuara como *boot loader*.

Un *boot manager* que no es un *boot loader* es el amigable rEFInd, sencillísimo de usar; por él he optado yo en cuanto lo he conocido —habiendo, así, puesto fin a las tribulaciones y zozobras que, durante varios meses, me había causado el desafortunado GRUB.

Un segundo *boot manager* sin *boot loader* será NMBL («nimble» —o sea «ágil»—, acrónimo de «No More Boot Loader»), que está siendo desarrollado ahora por un equipo de RedHat, encabezado por la ingeniera Marta Lewandowska; consistiría en dejarle cargarse al kernel por si solo.[120] (¿No es justamente eso lo que hace el rEFInd? (A éste, empero, no se refirió Marta, si escuché atentamente toda la grabación.) Ni en ella se hizo eco de la diferencia conceptual entre *boot manager* y *boot loader*.)

* * *

Tras la ejecución de un *boot manager* o un *boot loader*, entra en escena el kernel. Éste carga en memoria lo principal para que funcione el sistema operativo. Concluida esa tarea, surge el espacio de usuario. Éste comienza con un programa de inicio, el *init*.

Como lo he dicho ya unos pocos párrafos más atrás, Slackware sigue la pauta heredada del Unix/BSD, a tenor de la cual el primer programa que ejecutaba la inicialización —en saliendo de la ejecución del kernel— era *init*,

---

120. Eso es, al menos, lo que he creído entender visionando su filmada conferencia del 13 de junio de este año 2024. V. https://www.youtube.com/watch?v=ywrSDLp926M. V. también https://lwn.net/Articles/979789/.

regulado en el fichero «/etc/inittab». Yo lo edité a menudo en función de una cambiante pluralidad de necesidades.

Daba inmediatamente paso el *init* a la ejecución de los escritos configurativos del subdirectorio «/etc/rc.d». El primero de ellos en correr era «/etc/rc.d/rc.S», el cual habilitaba la memoria virtual, montaba los sistemas de ficheros (o sea las particiones del almacenamiento interno), limpiaba los directorios de registros (*logs*) e inicializaba los recursos *Plug&Play*, cediendo en seguida el sitio a otros escritos.

Esos otros ficheros de configuración —todos agrupados en el subdirectorio «/etc/rc.d»—, eran escritos con instrucciones cuya comprensión resultaba obvia, sin apenas aprendizaje previo (o poquísimo), siendo, todos ellos, sencillísimamente modificables por el usuario. Regulaban las sucesivas fases de la inicialización, lanzando los ejecutables residentes en memoria —o *demonios*, según vienen denominados en el Unix/Linux.

Todos esos ficheros de configuración de la iniciación podía configurarlos el usuario.[121] Solía yo configurar principalmente el «rc.modules» y, desde luego siempre, el «rc.local», el último de todos ellos en ejecutarse, donde colocaba instrucciones para que lanzara determinados programas —que yo deseaba tener corriendo en transfondo— y creara ciertos directorios temporales, adscribiéndoles determinados atributos y una titularidad, o sea un *owner* y un *group*. Sencillo, directo, sin darle vueltas ni buscar tres pies al gato.

Una vez ejecutados todos ellos, aparecía el inductor (el *prompt*), e.d. ya se estaba en la consola (el CLI). Podían, según la configuración pre-delineada, además de abrirse, simultáneamente, varias consolas (hasta nosecuántas, pero, en todo caso, más de 12), lanzarse —en una de ellas— el entorno gráfico con la orden «starx».

Esos ficheros de configuración del directorio «/etc/rc.d/» podían volver a ejecutarse en cualquier momento (por el *root* o superusuario) con las órdenes «/etc/rc.d/rc.XXX stop» y «/etc/rc.d/rc.XXX start» (siendo «XXX» una variable). Uno detenía su ejecución para modificarlos y luego los volvía a poner en ejecución.

Tal sistema de configuración no era intrusivo. Desde la ejecución inicial de los mismos, su único efecto era el correr en transfondo de los demonios. Por lo demás quedaba el usuario libre, sin ataduras, sin ingerencias, dueño y señor de la máquina para hacer en ella lo que le diera la gana.

* * *

Bajo la impetuosa acometida de un joven e imperioso programador alemán, el diplomado Lennart Poettering (un hombre de muchísimo talento,

---

121. De ellos mencionaré: «rc.0», «rc.4», «rc.6», «rc.K», «rc.M», «rc.inetd», «rc.inet1», «rc.inet2», «rc.wireless», «rc.samba», «rc.hotplug», «rc.modules», «rc.sysvinit», «rc.font» y «rc.local».

anteriormente empleado de RedHat y, desde julio del 2022, a sueldo de Micro$oft), varias distribuciones reemplazaron ese sistema de inicialización por otro nuevo, el *systemd*.

Cundió el mal ejemplo, siendo hoy pocas las distribuciones que se resisten al *systemd*. (Aunque, entre tanto, se han inventado otras alternativas, no han logrado prosperar.)

Entre las distribuciones que no se han sometido figuran las 15 siguientes: Devuan, AntiX, MX Linux, Nitrux, Void Linux, GoboLinux, Alpine Linux,[122] Artix, TinyCore Linux, Chimera Linux, Venom Linux, Kiss Linux, PCLinuxOS, Linux From Scratch y Slackware.[123]

Es elocuente ese elenco, justamente por lo que no contiene. Esas 15 distribuciones juntas suman un pequeño porcentaje de usuarios del Linux, lo cual demuestra cuán difícil resulta, en la práctica, librarse del fatídico *systemd*.[124]

¿Qué hace exactamente el *systemd*? Citaré un fragmento de su autopresentación:[125]

> *systemd* (…) provides aggressive parallelization capabilities, uses socket and D-Bus activation for starting services, offers on-demand starting of daemons, keeps track of processes using Linux control groups, maintains mount and automount points, and implements an elaborate transactional dependency-based service control logic. *systemd* (…) works as a replacement for sysvinit. Other parts include a logging daemon, utilities to control basic system configuration like the hostname, date, locale, maintain a list of logged-in users and running containers and virtual machines, system accounts, runtime directories and settings, and daemons to manage simple network configuration, network time synchronization, log forwarding, and name resolution.

O sea, *systemd*, no sólo rige la inicialización del espacio del usuario (el *user space*, e.d. aquel en el cual, habiendo completado su ejecución, el kernel cede la máquina a las aplicaciones que quiera ejecutar el usuario), sino que,

---

122. Que, según ya hemos visto más arriba, no es Linux, al no contener el software GNU. Es un sistema operativo diferente, que comparte con Linux el núcleo.

123. Corren rumores —sin confirmar— de que la próxima versión de Slackware posiblemente adoptaría *systemd*; a mi juicio, trátase de mera especulación.

124. Faltaría a la equidad si omitiera, en este punto, la ardiente defensa de su sistema que —siempre en su vehemente estilo de discusión sin concesiones— ofrécenos su inventor, el diplomado Lennart Poettering, en «The Biggest Myths». V. http://0pointer.de/blog/projects/the-biggest-myths.html.

El hecho de que su diatriba refute treinta objeciones o reproches contra *systemd* muestra cuán extensa es la oposición a esa innovación, que sigue siendo mal aceptada por muchísimos usuarios, aunque se resignen, como hago yo, al no quedarnos otro remedio.

125. https://systemd.io/.

además, permanece enseñoreado como gobernador y centinela, que vigila, controla y regula, permanentemente, todo el funcionamiento de la máquina, restringiendo la acción de los ejecutables lanzados por el usuario (incluso por el *root* o superusuario), matándolos si, según su propio criterio, se comportan mal.

Systemd ha sido acusado de:

1º) Sacrificar lo que, hiperbólicamente, se llamaba «la filosofía Unix».

2º) Causar una complejización excesiva, una auténtica hinchazón o inflamación, sin genuina necesidad.

3º) Hacer la configuración de la máquina mucho más abstrusa y enigmática para el usuario (quien —para desenvolverse en esa selvática maraña— necesitaría seguir un curso o, al menos, empollarse un buen manual).

4º) Provocar algunos resultados perversos (v.g. para el kernel).

5º) Imponer una configuración manifiestamente desagradable (o incluso odiosa) para muchos linuxitas, sin que los desarrolladores hayan accedido a tomar en cuenta esas reacciones adversas.

De esas cinco objeciones asumo yo la tercera. He de mencionar, empero, la contraobjeción del diplomado Poettering (secundada por bastantes desarrolladores), a saber: lejos de ser difícil, resultaría, según ellos facilísimo, no sólo escribir y editar los pequeños ficheros de configuración de *systemd*, sino también aprender el elenco de órdenes en el CLI que hacen falta para manejarlo.

Cada vez que se esgrime el criterio de facilidad o de dificultad, hay que preguntarse: fácil o difícil ¿para quién? Para un italiano es fácil aprender español —al tratarse de dos idiomas hermanísimos, que comparten al menos un 66'6% de fonética, vocabulario y gramática—, siendo más difícil aprender sueco o, peor, chino. Para un noruego es más fácil aprender inglés que malgache.

Partiendo de zero, ¿resúltale más fácil o más difícil a un aprendiente enterarse de cómo se redactan los ficheros configurativos del *systemd* que los *scripts* del Bash o los del sistema de Slackware que he resumido? No lo sé, ni creo que se haya llevado a cabo sobre esa cuestión ningún estudio serio, con evidencia científicamente corroborable.

En cualquier caso ¿es eso relevante? No partimos de zero. La mayoría de los usuarios normalitos de cualquier distribución donde se ejecute el *systemd* no van a molestarse en reescribir o modificar nada de la configuración por defecto del *systemd*. Quienes sí querríamos alterar esos ficheros de configuración —y hasta quitar algunos, reemplazándolos por otros— verosímilmente estamos familiarizados (en mayor o menor medida) con los *scripts* del Bash, resultándonos llano y acostumbrado el texto de los ficheros de configuración del Unix, al paso que todo en el *systemd* nos resulta ajeno y

misterioso.

A esa objeción contra el *systemd*, agregaré otras tres.

1ª. Habiendo leído varios alegatos de los propugnadores de *systemd*, no he conseguido entender qué se gana con él ni qué razones había para esa hondísima mutación. He creído comprender que su adopción ha facilitado la labor de los desarrolladores. ¡Sea! No puedo opinar. Mas, ¿valía la pena un cambio que ha alterado tantísimo todo el funcionamiento del sistema operativo?

2ª. Todo eso se hace, al menos en gran medida, en aras de la seguridad, mas esa obsesión securitaria puede provocar desastres (como a mí me ha sucedido). La seguridad es un imperativo para los servidores abiertos al público, pero no tiene tanta relevancia para el *desktop* de un humilde usuario en su casa (e incluso en su oficina) o para el NAS doméstico de nuestra red wifi particular, inaccesible fuera de nuestro hogar.

3ª. Sin duda en aras de la seguridad, el *systemd* impone una severa restricción al libre manejo de su máquina por el usuario. Lo voy a ejemplificar con dos consideraciones: 1ª, la drástica restricción que nos impone el *systemd* sobre el número de tareas que nos permite ejecutar en nuestra propia máquina, de la cual somos administradores; 2ª, mi incidente con librerías incompatibles colocadas en el directorio /usr/lib, que relato en el transiguiente apartado de este mismo capítulo.

* * *

Empiezo por la primera de las dos consideraciones indicadas en la tercera objeción. Trátase de la restricción que nos impone el *systemd* acerca de cuántas tareas se nos autorizan. Si emitimos en el CLI la orden «systemctl status user-$UID.slice», hallaremos, en ese mismo CLI, un educto como (en mi caso):

```
user-1000.slice - User Slice of UID 1000
Loaded: loaded
Drop-In: /usr/lib/systemd/system/user-.slice.d
└─50-TasksMax.conf
Active: active since Fri 2024-09-06 09:53:24 CEST; 2h 17min ago
Tasks: 281 (limit: 41068)
Memory: 1.6G
```

Significa eso que el *systemd* nos autoriza a ejecutar un máximo de (en mi caso) 41068 tareas, cifra enorme en comparación con el modestísimo número de las que estoy actualmente realizando, que son sólo 281. Ahora bien, ¿cuántas tareas podría ejecutar mi computador simultáneamente? El triple. Permítenos el dictatorial *systemd* ejecutar tan sólo un 33% de las tareas posibles. (En máquinas más modestas, o menos modernas, el límite autorizado

podía ser, v.g., 10267.)

Resultando suficiente ese límite para un uso normal del computador, no deja de constituir un constreñimiento eso de no poder ejecutar un número de tareas que exceda el 33% de aquellas que nuestro aparato sea capaz de realizar.

¡Es por nuestro bien! Trátase de ponernos en guardia contra nosotros mismos, librándonos de caer en tentaciones. V.g., la tentación de ejecutar en el CLI la temida *fork bomb (bomba de bifurcación)*,[126] consistente en una orden que, por autorreproducirse viralmente, desborda los recursos de la máquina, conduciendo a su parálisis —lo cual acarrea vernos forzados a apagarla. (Circunscríbese, empero, el daño a la sesión actual, no afectando al disco ni, por consiguiente, impidiendo arrancar, con toda normalidad, una nueva sesión.)

¿Cuál es esa pavorosa *fork bomb*? Es la curiosa orden «:(){:|:&};:». ¡Qué orden más rara ¿verdad?! (En lugar del pintoresco «:» podríamos usar cualquier otro signo, v.g. «μ».)[127]

Lo que hace esa orden es: 1º, define la función «:» como aquella función f el educto de cuya ejecución es, a su vez, inducto de sí misma, sólo que ejecutada en transfondo (sin reflejarse, por lo tanto, en la pantalla del CLI); 2º, manda que se ejecute tal función f.

Hay que entender que tal orden está redactada en el lenguaje de la concha *Bash*, afín al cálculo lambda y a la lógica combinatoria. En esos sistemas, una función puede aplicarse a sí misma. V.g., la función λx.x aplicada a sí misma da como resultado esa misma función.[128]

Y es que, a diferencia de la teoría estándar de conjuntos, no se da desnivelamiento o estratificación en esos sistemas.

Igual que el cálculo cuantificacional de Russell y Whitehead, rígese la teoría estándar de conjuntos por el principio [de exclusión] del círculo vicioso,

---

126. Creada, al parecer, por un informático radicado en Amsterdam, de apodo «Jaromil».

127. No se crea que la bomba de bifurcación es una maligna posibilidad sólo del Linux, ya que en Windows puede también realizarse, corriendo un *script* (un .bat o un .cmd) cuyo contenido sea «%0|%0». Su significado es: «doy la orden de que se ejecute esta orden y que, en habiéndose ejecutado, su educto invoque una nueva ejecución de la misma orden». ¡Paradojas de la recursividad!

128. El cálculo lambda (sin tipos) es casi equivalente a la lógica combinatoria, cuyo vocabulario únicamente está formado por combinadores y cuya sintaxis preceptúa que, cuando A y B sean combinadores, también lo será la concatenación AB. Para esa lógica, no hay diferencia entre funciones, elementos o individuos y estados de cosas (ni entre términos y asertos); según lo dijo uno de sus máximos cultivadores, Frederic Fitch, todo es un hecho (sin dejar de ser una función). Para introducirse en esa lógica, vale comenzar leyendo el artículo que le consagra la enciclopedia Stanford de filosofía. Leibniz fue precursor de esa lógica con su escrito *Generales inquisitiones de analysi notionum et veritatum*, a cuyo estudio dediqué yo mi artículo «De la logique combinatoire des *Generales Inquisitiones* aux calculs combinatoires contemporains, *Theoria* Nº 14-15, pp. 129-59 (ISSN 0495-4548.), 1991.

a saber: no ha de definirse función alguna de tal modo que, directa o indirectamente, venga involucrada en la definición la existencia de la propia función que se está definiendo.

Identificamos un conjunto con su función característica (la cual toma como argumentos objetos y como valores funcionales los dos valores veritativos, la Verdad y la Falsedad [o, admitiéndose una lógica multivalente, cualquier valor veritativo en el intervalo entre 0 y 1]). Lo que nos impone ese principio es que un conjunto cualquiera, *C*, no abarque a ente alguno de nivel igual o superior al del propio *C*.

La teoría estándar de conjuntos estratifica así la realidad en niveles. Están los individuos, entes del nivel 0; luego están los conjuntos de individuos, entes de nivel 1; a continuación están los conjuntos de conjuntos de individuos, entes de nivel 2; y así sucesivamente.[129] No se pueden mezclar los niveles. Por lo tanto exclúyese un conjunto como $\mu=\{\mu,a,b\}$.[130]

Por extraña que les parezca a los neófitos, resulta, en cambio —en el lenguaje de programación—, perfectamente legítima y válida esa orden, «:(){:|:&};:»,[131] aun siendo dañina, por el indeseable efecto ya señalado.[132]

¿Podría picarnos la curiosidad de ejecutarla, a ver qué sucede? Podría. Sufriríamos, como penitencia, la paralización de la máquina, teniendo que apagarla. Sálvanos de esa tentación —y de otras parecidas— nuestro demonio de la guardia, el *systemd*, imponiéndonos dejar inactivas 66% de las tareas simultáneamente ejecutables en nuestro aparato. Además de resultar una limitación excesiva, ¿no constituye tal restricción un acto de paternalismo, impidiéndonos aprender por la experiencia y hasta forzar los límites de nuestra

---

[129]. Viene a presuponer la teoría estándar de conjuntos una visión idealista o constructivista (sin extraer las últimas consecuencias de tal presuposición). En cierto sentido, los individuos estarían ahí, preexistentes; operando con ellos, iría el sujeto creando los conjuntos, nivel por nivel, al agruparlos mentalmente.

[130]. V. mis dos artículos «Lógica combinatoria y teoría estándar de conjuntos» (https://digital.csic.es/handle/10261/10408) y «Características técnicas y significación filosófica de un cálculo lambda libre» (http://lorenzopena.es/articles/logica/lambda.pdf). En el primero de ellos ofrezco argumentos filosóficos a favor del reconocimiento de cúmulos como $\mu=\{\mu,a,b\}$. Sea, v.g., una sociedad limitada, *S*, creada por Ernestina, Tarsicio y Matilde; ésta última, decidiendo irse, vende su participación, que viene comprada por la propia sociedad, la cual, desde ese momento, abarcará a tres socios, a saber: Ernestina, Tarsicio y le propia *S*, que será miembro de sí misma.

[131]. Es válida en el sentido de que no vulnera la gramática del lenguaje de programación.

[132]. Si un infiltrado nos mete esa orden, arruina nuestra actual sesión de trabajo. El *software* maligno suele buscar un estrago más hondo y duradero, que afecte al disco; las bombas de bifurcación son meras gamberradas. Ahora bien, un *malware* podría alterar algún fichero de configuración de arranque de nuestro sistema, hincando en él una bomba de bifurcación; con lo cual, cada vez que arrancáramos, nos enfrentaríamos al mismo efecto deletéreo. A mi juicio, para hacer frente a tal amenaza, hay que prevenir las infiltraciones, sin ponerle trabas al usuario y dueño de la máquina.

máquina, que para algo es nuestra?

Felizmente hay cómo burlar esa supervigilancia, a saber con la siguiente orden:

systemctl --runtime set-property user-<uid>.slice TasksMax=<value>

donde, evidentemente, tanto «<uid>» cuanto «<value>» son variables que cada quien ha de reemplazar por lo que corresponda en su caso (en el mío <uid>=1000, mientras que he escogido fijar <value>=66666). (Salta a la vista, no obstante, que eso hay que haberlo aprendido, empollándose los manuales —o buscando orientación en foros de linuxitas por internet).[133]

* * *

Paso ahora a la segunda de las dos consideraciones indicadas en la tercera objeción (al final del apartado anteprecedente). Esperando yo inmutabilizar el Dosemu para poder ejecutarlo en cualquier distribución del Unix/Linux —actual o futura—, lo encapsulé con el programa Ermine. Sin embargo, al ir a ejecutarlo en una máquina virtual con Lubuntu 24.04 (una versión más actualizada que la mía, que es la 22.04), resultó inejecutable por faltar, en esa versión, dos librerías. Retornando a mi máquina física, busqué en vano ambas librerías.

Averigüé entonces que se trataba de librerías de versiones antiguas del sistema operativo, presentes en mi anterior Slackware.

Todavía estaba vigente mi licencia del Ermine, de suerte que intenté re-erminiar el Dosemu con esas dos librerías. No tuve éxito.

Lo peor fue que, tras otros intentos fallidos, me decidí a copiar ambas librerías al directorio «/usr/lib», para ver si ahí las reconocería el Ermine. En cuanto lo hube hecho, dejó de funcionar la máquina, no pudiendo ni siquiera rebutearse. Tuve que apagarla para después reinstalar el sistema operativo.

¿A qué lo atribuyo? A un exceso de celo, ya sea del AppArmor, ya del propio *systemd*, el cual monta guardia permanente, escudriñando cuanto sucede en la máquina para poner a raya a cualquier aplicación cuyo funcionamiento juzgue irregular o incorrecto. Se ve que una de las intolerables incorrecciones es que en el directorio «/usr/lib» se encuentren librerías incompatibles.

Con Slackware, desde luego, no me habría sucedido eso. El sistema operativo no se habría dado por enterado de tales librerías forasteras hasta que yo, en el CLI, hubiera dado la orden «ldconfig» (cosa que, evidentemente, me habría rigurosamente abstenido de hacer). Y aun entonces, el sistema habría emitido una queja, sin provocar catástrofe alguna.

Concedo que no debí realizar una maniobra como la relatada, sobre todo

---

133. V.g. en https://itsfoss.com.

cuando carezco de suficientes conocimientos técnicos. Sólo que, desde que hube iniciado mi uso de computadores, en febrero de 1989, he sido siempre muy emprendedor y aventurero. Unas veces me ha ido bien, otras mal. (Infrecuentísimo ha sido, no obstante, que me vaya tan mal como en la ocasión que acabo de relatar.)

He escarmentado. Seré más prudente. No se han vuelto a producir descalabros.

Insisto en que no puedo afirmar rotundamente que ese vigilante que pecó por exceso de celo fuera el *systemd*. El único sospechoso alternativo sería el AppArmor. Inclínome, habiendo leído varias páginas donde se explica qué tareas realizan el uno y el otro, a opinar que el culpable fue el *systemd*.

Admitamos que, con el *systemd* y con el AppArmor, sea más seguro el manejo del computador. Sólo que seguro ¿en qué sentido? Seguro ¿de qué peligros? Probablemente de disfunciones causadas por aplicaciones malignas o perjudiciales que hubiera podido meternos una indeseable infiltración; pagando el precio de que, a cambio, nos dejan hacer menos cosas, cercenando nuestra libertad. O sea, expiamos el gozar de esa adicional seguridad viendo, a cambio, disminuida nuestra libertad.

Mas lo peor ni siquiera es eso, sino que, por su exceso de celo, de alguna operación imprudente nuestra —que, de suyo, pudiera ser benigna— uno u otro de esos guardianes derivan consecuencias catastróficas, un descalabro total. Si pudiera, renunciaría yo con gusto a esa seguridad, que para mí (hegelianamente) se transmuta en inseguridad.

* * *

Siendo ardua la discusión técnica —en sus pormenores difícil de comprender por el autor y, seguramente, también por el lector del presente ensayo—, pienso que conviene fijarse en la ideología que alienta esa innovación, la cual corrobora plenamente la primera de las objeciones esgrimidas contra *systemd*, a saber: su incompatibilidad con la «filosofía UNIX» —o, diría yo, más sencillamente, el espíritu Unix, el ideario Unix.

Confírmanlo las propias declaraciones del diplomado Poettering. Leo un suelto titulado «Lennart Poettering on *systemd*'s Tumultuous Ascendancy», del 26 de enero del 2017, escrito por Swapnil Bhartiya.[134]

> (Según Poettering) uno de los más conocidos conceptos que a menudo se citan sobre el Unix es que todo es un fichero, personalmente pienso (sigue diciendo Poettering) que es una completa imbecilidad. No cualquier cosa es un fichero. Mi impresora no es un fichero y, si Unix pretende que mi impresora es un fichero, eso es pura necedad, ya que un fichero es algo muy diferente».
>
> (Sigue diciendo Poettering que) la mayoría de los programadores

134. https://thenewstack.io/unix-greatest-inspiration-behind.

> acabarán aprendiendo la programación orientada a objetos, en la universidad o en otras escuelas; se enterarán de que el más abstracto concepto de la programación orientada a objetos es (…) ver todo como un objeto. En Unix es el fichero. Estoy bien seguro (agrega) de que es un completo sinsentido poner el fichero como el concepto más básico; el fichero es algo muy específico, algo sumamente diverso de una impresora o de una tarjeta de sonido.

Pero ¡vamos a ver! Desconociendo yo qué sea la programación orientada a objetos, paréceme curioso que se atribuya a los brillantísimos originadores y desarrolladores del Unix la creencia de que una impresora es un fichero, una especie de idealismo berkeleyano que concebiría que el mundo se limita a lo que está en el computador accesible desde el sistema operativo.

La idea del Unix es que, a efectos del tratamiento informático, todo se puede reducir (ficticiamente, claro está) a un fichero. Todo. Una impresora, una escrutadora, una tarjeta ethernet, un disco externo USB o cualquier otro periférico.

¿Qué es un fichero? Es, para la informática, la unidad mínima de almacenamiento. Desde el punto de vista del sistema operativo, trátanse todos los objetos como si fueran ficheros, unidades mínimas de almacenamiento. Copiar un fichero en la impresora es mandarlo a la impresora, o sea imprimirlo.

Desde fuera de la labor informática, la impresora es un trozo de materia con su estructura y su configuración propias, que efectúa su específica función material: imprimir sobre hojas de papel. Mas eso no lo ve el usuario que, con las manos en el teclado, mira la pantalla; lo que ve es una notificación en el sentido de «fichero copiado en impresora».

¿Qué sucede si, por buscarle tres pies al gato, al usuario se le ocurre copiar la impresora en otro fichero? Sencillamente que crea otro fichero con las mismas características del fichero *printer* (o comoquiera que se denomine el correspondiente *device* ubicado en el directorio «/dev)».

El sistema operativo no trata como un fichero mi tableta de chocolate. Ocúpase únicamente de objetos con los cuales tenga que ver, ya sean piezas de *hardware* internas del computador, ya sean periféricos. Cada uno de ellos viene representado por un fichero peculiar alojado en el directorio «/dev», resultando irrelevante para el sistema operativo cuál sea su realidad física.

Gracias a ese enfoque ha podido el Unix desarrollar su potencia. Por el contrario, son incapaces de esa abstracción sistemas como el Windows, no pudiendo ni siquiera concebir como ficheros los discos ni sus particiones.

La sintaxis del Windows es pluricategorial; la del Unix es monocategorial. Si la programación orientada a objetos no consigue entenderlo, lejos de constituir un avance, es un retroceso.

En otra de sus inflamadas prolaciones proclama el diplomado Poettering:

> Unix tiene un diseño de los años setenta. Los computadores de hoy funcionan de manera muy diferente de los de entonces. Quienes piensan que los computadores funcionan exactamente como funcionaban en los 70 deben de tener poca experiencia en el manejo de computadores, desconociendo cuánto han cambiado.
>
> En *systemd* afanámonos, con empeño, en observar otros sistemas operativos (…), consagrando muchísimo tiempo a estudiar Windows para hallar en él ideas interesantes.

Está clara la radical enemiga al Unix del diplomado Poettering; desde luego, está en su perfecto derecho. Únicamente cabe reprocharle la insinceridad o duplicidad de sus esporádicos juramentos de fidelidad al Unix, un sistema que aborrece, que considera basado en un disparate y que, además, le resulta periclitado y obsoleto.

Inicialmente disimulada, esa hostilidad al Unix proclámanla ya los enardecidos desarrolladores de *systemd*, al estar su líder encastillado en la fortaleza Micro$oft. Leo —un par de meses antes de escribir yo este ensayo— un suelto de Liam Proven con el título «Versión 256 of *systemd* boasts '42% less Unix philosophy'».[135] En efecto, de ser un 42% menos Unix vanaglóriase el propio anuncio del equipo desarrollador del *systemd*, a cuya cabeza sigue hallándose el empleado de Micro$oft, diplomado Poettering.

* * *

Una de las novedades de esta última versión del *systemd* es eliminar la orden *sudo* que permite a un mero usuario realizar una tarea concreta con los privilegios del superusuario o *root*, caso por caso.

Jáctase de ello el diplomado Poettering en su característica prosa: «Let's throw out the concept of SUID on the dump of UNIX' bad ideas». Propone reemplazar el «sudo» por el «*systemd*-run» redenominado «run0»; según leo en su página en la red social Mastodon (suelto del 29 de abril de este mismo año 2024), «[run0] behaves a lot like a sudo clone, but with one key difference: it's **not** in fact SUID; instead it just asks the service manager to invoke a command or shell under the target user's UID». (Careciendo yo de competencia para entender su explicación, resúltame palmario que el cambio alterará hondamente la relación entre un usuario, digamos «jesus», y su máquina, dificultando considerablemente su posibilidad de introducir en ella modificación alguna fuera de su propio directorio «/home/jesus».)

En las viejas distribuciones del Unix/Linux, había dos modos de entrar en la máquina: como *root* y como mero usuario. Dos pasarelas las ofrecían las órdenes «su» y «super», con las cuales —dentro de márgenes marcados por el *root*— podía un usuario realizar tareas propias del superusuario. Posteriormente se adoptó otra orden, «sudo», que limitaba ese salto a la realización separada de cada tarea.

---

135. https://www.theregister.com/2024/06/13/versión_256_systemd/.

Mas siempre en Unix/Linux ha sido y es posible que el usuario de la máquina (que, en el caso de los computadores domésticos, suele ser el dueño) pueda administrarla, configurándola según lo tenga por conveniente.

No así, bien lo sabemos, en Windows, donde es casi nulo el margen de intervención del usuario. El sistema operativo viene ya fijado y configurado por la empresa productora, quedándole únicamente al usuario —y eso ciñéndose a la predeterminada e inmutable configuración— usar aquellas aplicaciones que, o bien vengan ya contenidas en el sistema operativo en sí, o puedan agregarse, producidas por terceros; usarlas sin alterar absolutamente nada.[136]

Lo que ahora quiere imponernos el diplomado Poettering es que el Linux rompa ya del todo la continuidad con el Unix para ser como Windows. Habrá dejado de contar con ese utensilio, la orden «sudo», el usuario normalito, a quien resultará de todo punto imposible modificar la configuración de su propia máquina.

Que ésa es la intención del *systemd* (en su última versión al menos) confírmalo esta otra declaración del diplomado Poettering, quien nunca se queda corto explayándose sobre sus planes.[137]

> During his talk, Poettering shared an even more ambitious vision. «I personally think we should go towards systems where etc/ is generally not writeable» (…) In this world you're not just making a copy of your home directory on a USB stick; instead that portable storage medium becomes your home directory. Meaning that you can plug it into this laptop today, log in and it just works. And then you can take it out and put it in another laptop tomorrow, log in there, and it all just works.

¡Más claro agua! Trátase de que sea inmutable la máquina; el usuario no tiene voz ni voto sobre qué aplicaciones estén en ella instaladas. Limítase a usarlas sin tocar ni manchar el aparato, no pudiendo modificar nada fuera de su directorio personal, que estaría ubicado en una llave USB.

Si todavía permaneciera alguna duda acerca de lo que, para nuestro querido sistema operativo, han tramado los de Micro$oft —representados por el diplomado Poettering—, quedará despejada al leer lo siguiente (que tomo de la citada bitácora de Herr Poettering y de sus sueltos de la red Mastodon):

> Hermetic /usr/
>
> First of all the OS resources (code, data files, …) should be hermetic in an immutable /usr/. (…) This /usr/ tree can then be mounted read-only into the writable root file system that then will eventually carry the local

---

136. La intangibilidad de la configuración es incluso mayor hoy que en el viejo WindowsXP o en los precedentes, que todavía toleraban un relativo margen de acción del usuario.

137. La tomo de la página https://thenewstack.io/systemds-lennart-poettering-wants-to-bring-linux-home-directories-into-the-21st-century/.

> configuration, state and user data in /etc/, /var/ and /home/ as usual. (…)
>
> In my model a hermetic OS is hence comprehensively defined within /usr/: combine the /usr/ tree with an empty, otherwise unpopulated root file system, and it will boot up successfully, automatically adding the strictly necessary files, and resources that are necessary to boot up.
>
> Monopolizing vendor OS resources and definitions in an immutable /usr/ opens multiple doors to us:
>
> We can (…) guarantee structural, cryptographic integrity on the whole vendor OS resources at once, with full file system metadata.
>
> We can (…) update the OS resources atomically and robustly, while leaving the rest of the OS environment untouched.
>
> We can implement factory reset easily: erase the root file system and reboot. The hermetic OS in /usr/ has all the information it needs to set up the root file system afresh, exactly like in a new installation.

En resumen, tendríamos un sistema operativo cuyo funcionamiento se parecería al de Windows. El árbol binario sería intangible, al usuario le estaría vedado hacer nada en la máquina fuera de su propio HOME personal. Las únicas aplicaciones serían las que nos hubiera puesto el desarrollador de nuestra distribución, las cuales quedarían grabadas, de una vez por todas, al actualizar tal distribución del sistema operativo. Estaríamos sentados ante nuestro propio computador como ante uno ajeno, cuyo uso nos han prestado.

* * *

Cabe preguntarse cómo así Micro$oft consiente que su empleado Lennart Poettering siga consagrándose a desarrollar un *software* específico para Linux, el *systemd*, verosímilmente a tiempo completo.

El diplomado Poettering goza del favor de sus nuevos empleadores, quienes apadrinan y auspician sus iniciativas, propulsándolo como un aspirante al liderazgo en el horizonte Linux.

Así,[138] convoca anualmente una conferencia en Berlín con el título «All Systems Go!», «a conference focused on foundational user-space Linux technologies», con el patrocinio de las siguientes firmas: Chorus One, Meta (FaceBook), Pantheon, Solar, Lunar, Terrestrial y, sobre todo, Microsoft.

Hay que entender en su pleno contexto ese apadrinamiento. Lleva años Micro$oft infiltrándose en el mundo Linux. El primer paso fue la absorción del sitio Github, una plataforma para desarrolladores de *software* Linux (31 millones según leo), creada por iniciativa de Linus Tolvalds, constituyendo una especie de repositorio auxiliar de las distribuciones.

Cuando en Ubuntu damos la orden «sudo apt update», infórmasenos únicamente de aquellas actualizaciones que figuren, sea en los repositorios

---

138. V. su bitácora https://0pointer.de/blog/.

oficiales de Ubuntu, sea en otros que nosotros mismos hayamos (incautamente) agregado a la /etc/apt/sources.list. (P.ej., los repositorios personales de algunos desarrolladores, los *ppa*, que es razonable añadir para bajarnos algún *software* concreto que nos haga falta; sólo que, tras la descarga e instalación de tal *software*, resulta juicioso editar esa lista, borrando de ella cualesquiera repositorios que no sean los oficiales de Ubuntu; incluso el de Canonical puede causarnos incompatibilidades —ya que no es un repositorio oficial de Ubuntu.)

Por consiguiente, al actualizar nuestro *software*, con dicha orden «sudo apt update», no se tiene en cuenta para nada el contenido de GitHub. No obstante, a sabiendas de su preeminencia en el mundo Linux, es frecuentísimo acudir a GitHub para bajarse algún *software* que no figure en los repositorios de nuestra distribución.[139]

Hízose por dos etapas la adquisición de GitHub por Micro$oft. La primera (a modo de paso preparatorio) fue el linchamiento moral, en 2014, de su fundador y CEO (*chief executive officer*, o sea director general), Tom Preston-Werner, acusado por la erinias, ora de acoso sexual o «basado en el género», ora de represalia o de crear un ambiente sexista. Desmintió tales calumnias la investigación interna. Reconocieron los indagantes no haber podido probar ni una sola conducta ilícita (en particular ninguna discriminación «basada en el género», ni acoso, ni represalia ni ofensa); sólo que, buscando tres pies al gato, sí les había parecido «encontrar errores de juicio». Ese dictamen forzó a Preston-Werner a dimitir.[140]

La destitución de Tom Preston-Werner allanó la ocupación de su cargo por Chris Wanstrath, quien perpetró la segunda etapa de la operación, vendiendo GitHub a Micro$oft cuatro años después.[141]

Wansrtrath cedió entonces el puesto de máximo directivo al hombre de Micro$oft Nat Friedman, obteniendo, en compensación, el cargo de *Microsoft technical fellow* encargado de «iniciativas estratégicas de *software*.[142]

Ya tenemos así a los de Redmond enseñoreados de la principal plataforma de *software* Linux. No para aquí la cosa. Micro$oft, además, habiéndose percatado de que su propio sistema operativo, Windows, no vale para gobernar grandes servidores, estando reservado a Linux ese espacio, creó su propio gran servidor de nube, Azure, ¡con Linux! En lugar de usar otra

---

[139]. No es el único de esos almacenes de *software* Linux gratuitos, mas sí el de más alta reputación —y verosímilmente el mejor aprovisionado. Yo, personalmente, prefiero sourceforge.net, no habiéndome adaptado a GitHub, para mi gusto menos amigable.

[140]. V. https://github.blog/news-insights/the-library/results-of-the-github-investigation/.

[141]. Habiendo obtenido la venia de la Unión Europea para tal absorción, que le costó al monopolio de Redmond 7 miliardos y medio de dólares.

[142]. V. https://techcrunch.com/2018/10/26/microsoft-closes-its-7-5b-purchase-of-code-sharing-platform-github/.

distribución, creó la suya propia, el Linux-Azure, que, irónicamente, carece de entorno gráfico o GUI, habiendo de usarse únicamente en CLI y en TUIs (interfaces textuales de usuario).[143]

A la apropiación de Github y al lanzamiento de la distribución Azure y del servidor de nube de igual denominación, agrégase la creación del WSL, *Windows subsystem for Linux* (en dos versiones, 1 y 2), para correr, dentro de Windows, alguna distribución de Linux. Trátase de tener así a los linuxeros enjaulados, bajo supremacía de Windows, siendo éste el sistema operativo que controla la máquina.

Todo coronado por el eslogan exhibido en el portal de Micro$oft: «Microsoft loves Linux». ¡Ver para creer!

¡Cuánta agua ha corrido bajo los puentes desde los «Halloween documents» (de agosto de 1998 a junio de 2004)! Constituían un compendio del denigramiento de Linux por Micro$oft, que, en su campaña de descrédito, había recurrido a sus malas artes (presuntas averiguaciones que probarían cuán insolvente, frágil y destartalado sería Linux.)

Cuatro lustros han transcurrido. En la nueva estrategia de satelización de Linux, Micro$oft actúa con su consabida sagacidad. Su propio Linux, el Azure, es más Linux que los de muchas distribuciones, a la vez que, a través de su empleado Poettering, inflige a Linux una involución que lo haga parecerse, cada vez más, al Windows, haciéndole perder su fisonomía —y, con ella, su razón de ser.

Un Linux reducido a la miserable condición de Windows-bis, Windows para pobres, ya no gozaría del halo de prestigio que hoy tiene; ya no sería un Unix, ya no brillaría en el firmamento informático como la estrella del *software* avanzado y potente.

---

143. No ha alcanzado —ni, seguramente, aspirado a alcanzar— popularidad alguna ese Linux de Micro$oft (disponible en GitHub). Sólo algún que otro extravagante lo ha descargado e instalado.

Capítulo XXII

Linux y el movimiento del *software* libre

Creado bajo iniciativa y liderazgo de Richard Stallman el 27 de septiembre de 1983, propúsose el proyecto GNU construir un sistema operativo completamente «libre», que se llamaría igual que el propio proyecto, «GNU».

Establécese en 1985 la FSF, *Free Software Foundation* (Fundación para el Software Libre), la cual brinda su apoyo al proyecto GNU. La Fundación retribuirá el trabajo de algunos de los muchos programadores cuya labor contribuye a plasmar y desarrollar el cúmulo de utilidades GNU. Va a atraer la creciente reputación del proyecto a firmas comerciales, animándolas a hacer donaciones a la Fundación para beneficiarse de su producto.

Nunca se ha conseguido llevar a cabo el plan inicial de un completo sistema operativo libre, porque siempre ha faltado la clave de bóveda, o sea el kernel o núcleo.[144] Por eso Linux Torvalds inventó su propio kernel, cuyo mestizaje con las utilidades GNU ha dado como fruto nuestro sistema operativo Linux.

§1.— Los orígenes del *software* «libre»

Siempre se han ofrecido las utilidades GNU como *software* libre (prescindiré de las comillas en lo sucesivo). Stallman estaba espantado por el deletéreo acaparamiento del *software* como objeto de propiedad intelectual por las firmas desarrolladoras.

Sentía escozor por una experiencia reciente: habiendo aportado mejoras a un cierto *software* de la casa Symbolics, no se retribuyó su favor dándole a conocer el código resultante de introducir las mejoras. Eso violaba el hasta entonces guardado canon —no escrito— de reciprocidad y de colaboración entre programadores.

Reflexionando sobre ese contratiempo, ahondó sus ideas, perfilando la ideología del *software* libre. La regla definitoria no habría de ser la reciprocidad,

144. Aquel que se diseñó fue el HURD (tras haberse intentado, sin éxito, otro, el TRIX). Sin embargo, ocho lustros después, el HURD sigue sin estar a punto.

sino la transitividad. No se trataría únicamente de que, si yo te aporto unas mejoras a tu programa, tú, en contrapartida, me des a conocer el resultado de incorporar mis mejoras, sino de algo de alcance infinitamente mayor: tu programa habrá de quedar accesible a todos, de tal manera que quienquiera podrá leer el programa-fuente (el escrito sobre el teclado con caracteres alfanuméricos ASCII, o sea con un lenguaje de programación de alto nivel —entonces era el **C**—) para, eventualmente, modificarlo, si lo tiene por conveniente.

¿En virtud de qué? Hacer preceptiva la apertura del código fuente y otorgar a cualquier tercero la libertad de reescribirlo como le plazca era una nueva máxima que, hasta los años setenta, puede que hubiera coincidido con un uso *de facto*, mas nunca había venido formulada ni deliberadamente asumida. Ni siquiera podemos hablar de una costumbre en el sentido jurídico de la palabra, que exige, junto con la habitualidad de la conducta, su realización con *opinio juris seu necessitatis*, e.d. con la conciencia (explícita o implícita) de estar cumpliendo un deber que se impone.

¿En qué se fundamentaría esa nueva máxima? Para Stallman en un principio de libertad. Atenerse a esa máxima era un imperativo derivado de un derecho a la libertad de todos los seres humanos; no ya la libertad genérica de vivir su propia vida según sus preferencias, sino una, más concreta, de que el manejo de los aparatos electrónicos no esté sujeto a imposiciones de los diseñadores y fabricantes, sino que, para cualquier ser humano, se realice según sus propios deseos.

Fue por entonces cuando Stallman cayó en la cuenta de que esos derechos de libertad de todos (programadores e igualmente usuarios) chocaban con la propiedad intelectual del *software*. La alternativa parecía ser renunciar a la propiedad intelectual, poniendo el *software* en el público dominio (*public domain*), lo cual —según la legislación estadounidense, diversa de la europea— significa que, por decisión del creador del programa, éste pasa a ser un bien mostrenco, sin dueño, a disposición de cualquiera para lo que desee. (En los países donde se aplica el convenio de Berna sobre la propiedad intelectual a ningún autor le está permitido renunciar a esa propiedad ni, por consiguiente, colocar su producción intelectual en el público dominio.)

§2.— Las cuatro libertades stallmanianas

¿Quedará el *software* en el dominio público?[145] Entonces cualquiera podrá adueñarse de una copia de tal *software*, reelaborarla, produciendo, gracias a ella, un *software* derivado, ocultando la fuente resultante y vendiendo el programa compilado.

---

145. Insisto: en el concepto jurídico de *public domain* del derecho norteamericano —o, quizá más ampliamente, de la *common law*, de la cual se ha separado, en este punto, la legislación británica actual.

Así, colocar un *software* en el dominio público no preservaría los derechos de libertad de los usuarios. Uno se habría aprovechado, privando a los demás de un derecho igual. Lo cual era visto por Stallman como jugar con reglas dispares: gracias a que es libre (pues lo que está en el dominio público es de libre uso para todos), yo me he beneficiado para producir otra obra sin reconocer sobre ella a los demás un derecho como aquel del cual he disfrutado yo.

Cierto, esa operación de adueñamiento no afecta exactamente al mismo ente, el *software* original, sino a otro. Mas ese otro, derivado del original, no hubiera podido producirlo yo sin el original. La «comunidad» me ha ofrecido ese *software* primitivo y yo, beneficiario de tal ofrecimiento, no he correspondido de la misma manera.[146]

¿Cómo se articula, cómo se detalla o se analiza ese derecho a la libertad del *software*?

Vino plasmada tal articulación en dos documentos fundacionales: el Manifiesto GNU y la GPL (*General Public License*). En el Manifiesto, Stallman enumera los cuatro derechos de libertad atinentes al *software*:

— Nº 0.— Ejecutar el *software* como cada cual quiera y para lo que quiera.

— Nº 1.— Estudiar el *software* y modificarlo según sus propias preferencias y para cualquier fin que tenga por conveniente.

— Nº 2.— Transmitir y propagar el *software* recibido.

— Nº 3.— Difundir copias del resultado de ejercer el derecho Nº 1.

En esa lista no aparece, sin embargo, un componente que para Stallman formaba parte inseparable de ese haz de derechos: la recepción del código fuente, sin la cual el titular de los derechos no podría disfrutar, plena e irrestrictamente, del derecho Nº 1.

De conformidad con el espíritu de ese cúmulo, no sólo al primer receptor le asistirá el derecho a recibir el código fuente, sino que —por el principio de transitividad— idéntico derecho tendrán los sucesivos receptores con relación, no sólo al producto originario, sino también a los derivados.

Siendo el citado Manifiesto tan sólo un texto programático, parece preferible atenerse a su implementación jurídica en la GPL (Licencia Pública General), en la versión 2, de 1991 (ya que la más reciente, la 3 de 2007, no ha sido aceptada por todos los integrantes del movimiento GNU/Linux —al no haber otorgado su anuencia Linus Torvalds, quien sigue emitiendo las sucesivas versiones del kernel bajo la versión 2).[147]

---

146. Subsumiendo al originador del primer *software* en la «comunidad», Stallman interpreta lo que, en rigor, es una regla de transitividad como regla de reciprocidad.

147. Huelga entrar aquí en la controversia suscitada por el paso de la versión 2 a la 3.

¿Qué es la GPL? Es un documento legal tal que quien emita o difunda un *software* declarando colocarlo bajo la GPL está ejecutando un acto jurídico por el cual renuncia a algunos de los derechos que otorga la Ley a los creadores intelectuales (el *copyright* en el derecho anglosajón y la propiedad intelectual en el derecho europeo), mas conservando otros de esos derechos e imponiendo, a cambio, a los receptores del *software* ciertos deberes.

Lo cual significa que la renuncia a algunos de los derechos de un propietario intelectual queda sujeta a la expresa condición de que el beneficiario de la licencia asuma y cumpla los correspondientes deberes.

Concibiéndose tales deberes en los generosos términos del espíritu del *software* libre, entiéndese la enunciación de la GPL —al igual que la de cualesquiera otras licencias de la propia FSF— como una inversión del *copyright*, que ahora se ejercerá para beneficiar, no al creador del contenido, sino a los destinatarios, sólo que a condición de que se comporten ellos con igual altruismo.

Para dar un nombre a esa singularidad los de la FSF acuñaron el neologismo «copyleft».[148]

---

§3.— Las cláusulas de la GPL y la cuestión de la gratuidad

Aunque, en la ideología liberista de Richard Stallman, una licencia no es un contrato ni se funda en la propiedad intelectual (concepto que él repudia, juzgándolo una aberrante amalgama), han señalado —con razón, a mi modo de ver— otros abogados de ese tipo de licencia[149] que, en rigor, el *copyleft* hunde sus raíces en la propiedad intelectual, constituyendo un ejercicio de tal propiedad; asimismo el acto jurídico de emitir un *software* bajo la GPL se completa con el acto jurídico del destinatario al usarlo, constituyendo así un contrato, o sea una concordancia de voluntades que establece recíprocos deberes y derechos de ambos contratantes.[150]

---

148. Un juego de palabras basado en el doble sentido de «right», que también tenemos en español: «*derecho*» frente a «prohibición» y «*derecho*» frente a «izquierdo».

149. U otras de orientación parecida, cual las *Creative Commons* ideadas por el jurista Lawrence Lessig.

150. Traduzco un extracto de un suelto de Bruce Perens publicado en 2017: «Artifex demandó recientemente a Hancom, alegando que estaba usando Ghostscript en uno de sus productoss, sin haber comprado una licencia comercial y sin tampoco cumplir la GPL. Aún no se ha resuelto este pleito, mas el tribunal ya ha decidido que la GPL es un contrato, por lo cual ha admitido la demanda como un litigio por violación de un contrato, no meramente por infracción de derechos de autor»:

> Defendant contends that Plaintiff.s reliance on the unsigned GNU GPL fails to plausibly demonstrate mutual assent, that is, the existence of a contract. Not so. The GNU GPL, which is attached to the complaint, provides that the Ghostscript user agrees to its terms if the user does not obtain a commercial license. Plaintiff alleges that Defendant used Ghostscript, did not obtain a commercial license, and

Hállanse en la GPL muchas consideraciones y muchas cláusulas, con una prolijidad cuyo análisis desbordaría los límites de este capítulo. La Licencia afirma:

> No quedan cubiertas por esta Licencia actividades distintas de la copia, distribución y modificación del programa, hallándose fuera de su alcance. Tampoco está restringida la ejecución del programa; cúbrese el resultado de tal ejecución únicamente cuando su contenido constituya un trabajo basado en el programa.

En sustancia, el clausulado de la GPL no hace sino glosar las cuatro libertades ya enumeradas, sólo que desmenuzándolas y previniendo una variedad de circunstancias.

Son siete las cláusulas. Sin embargo, a mi juicio hay redundancia; a menos que yo la haya leído mal, la cláusula 3ª meramente explicita lo ya estipulado en las dos precedentes.

La cláusula 0ª, al precisar qué actividades vienen cubiertas por la Licencia, otorga a los receptores del *software* el derecho a usarlo como les dé la gana y para los fines que tengan por convenientes.

Limítase la cláusula 4ª a prohibir «copiar, modificar, sublicenciar o distribuir el programa salvo según lo expresamente dispuesto en esta Licencia»; aun si se omitiera tal cláusula, sería igual su efecto jurídico, ya que la legislación del *copyright* y de la propiedad intelectual hacen ilícita cualquier copia, modificación o distribución de una obra intelectual sin el consentimiento del autor y dueño intelectual.

Aclara la cláusula 5ª que el receptor no está obligado a aceptar la licencia, salvo si realiza alguno de los actos en ella cubiertos (de nuevo una redundancia).

La cláusula 6ª precisa el principio de transitividad: el receptor de una obra derivada adquiere los mismos derechos y asume los mismos deberes que el de la obra original; lo cual es asimismo redundante, puesto que la obra derivada tiene que someterse a la misma licencia que la originaria (en virtud de la cláusula 2ª.b). (No obstante, hay un precepto contenido en esa cláusula 6ª que, aun siendo redundante, es importantísimo: la prohibición de imponer ninguna otra condición al ulterior adquirente.)

Quédannos así dos cláusulas principales, a saber:

---

represented publicly that its use of Ghostscript was licensed under the GNU GPU. These allegations sufficiently plead the existence of a contract.

Puede que la noción de *contrato* tenga un perfil menos claro en la *common law*, pero en los ordenamientos jurídicos de raigambre romano-germánica hay múltiples formas de instituir la vigencia de un contrato. Los hay que se contraen por escrito, *scriptis*; de palabra, *verbis*; por la realización de ciertos hechos, *factis*; por la mera colocación o retirada de ciertos objetos, *rebus*. Nuestros códigos civiles prevén asimismo contratos atípicos, no proscribiendo otras formas de manifestar el consentimiento contractual, cual ha sucedido con el internet (y, mucho antes, había ocurrido con el telégrafo y el teléfono).

1ª.— Es lícito al receptor del programa copiar y distribuir copias textuales del mismo así como de su código fuente siempre que publique un correcto aviso de derechos de autor, mantenga intactos todos los avisos y entregue a cualquier destinatario del programa una copia de esta Licencia.

2ª.— Es lícito modificar las copias del programa o de cualquier parte del mismo, elaborando así una obra derivada; igualmente es lícito copiar y distribuir dichas modificaciones o derivaciones, siempre que se cumplan tres condiciones: a) los ficheros modificados o derivados llevarán avisos que indiquen los cambios; b) la obra derivada se emitirá exactamente bajo esta misma Licencia *verbatim*; c) la obra derivada exhibirá claramente tres avisos: (1) apropiado reconocimiento de los derechos de autor; (2) no garantía; (3) clara indicación de cómo acceder al texto de la Licencia.

Nótese que, mientras la cláusula 1ª atañe únicamente a la obra originaria —autorizando su ulterior distribución a terceros (bajo las condiciones estipuladas)—, la cláusula 2ª es atinente a la elaboración de obras derivadas o modificadas; compónese esta cláusula 2ª de dos partes netamente diferenciadas; una mejor redacción, en realidad, debería dividir la cláusula en dos.

Refiérese la primera de esas dos subcláusulas al hecho en sí de la creación de una obra derivada. Tal creación viene, en cambio, tajante y conminatoriamente prohibida en el *software* normal (el que los liberistas llaman «propietario», adjetivo usualmente traducido a nuestro idioma como «privativo»), cuyos productores y vendedores amedrentan truculentamente al comprador con la amenaza de que sobre él caerá todo el peso de la ley si osa elaborar una obra derivada, distribúyala o no.[151]

¿Hacía falta ese permiso de elaboración de obra derivada? Con otras palabras, cuando el dueño intelectual de una obra de su propia creación la distribuya (gratis o no) sin prohibir la elaboración de obras derivadas, ¿es lícita tal elaboración? Requeriría un estudio detallado de la legislación aplicable, mas me inclino a opinar que tal elaboración es lícita cuando no la haya prohibido el dueño intelectual de la obra.

Permite la segunda subcláusula propagar la obra derivada, siempre que se haga en los mismos términos que la obra originaria, e.d. bajo la misma licencia (la GPL), cumpliendo además otros requisitos de equidad: correcta y expresa atribución de autoría; información sobre los cambios introducidos; y exclusión de garantía.

Cuestión ésta de la no-garantía que, si bien perpetuamente reiterada en

---

151. El *software* «propietario» o privativo suele prohibir asimismo la ingeniería inversa, o sea la averiguación del código fuente, no ya utilizando algún programa descompilador y desensamblador, sino incluso por inferencias abductivas.

todas las versiones —no ya de esta licencia GPL, sino de las demás también llamadas «libres»—, merece un comentario especial. En estos términos viene expresada la exclusión de garantía en la licencia GPL v.2 (junio de 1991):

> Comoquiera que el programa venga licenciado gratis («free of charge»), no existe garantía para el mismo: en la medida permitida por la ley aplicable, excepto cuando, por escrito, se haya indicado lo contrario, tanto los titulares de los derechos de autor cuanto quienquiera haya modificado o transmitido el programa según los términos estipulados en la licencia lo proporcionan «tal cual», sin garantía de ningún tipo, ni expresa ni tácita, lo cual abarca, entre otras cosas, cualesquiera garantías implícitas de comerciabilidad o idoneidad para un determinado propósito. Recae enteramente sobre el adquirente el riesgo atinente a la calidad y al rendimiento del programa. De resultar defectuoso, asumirá el adquirente el pleno costo del necesario servicio, reparación o corrección.

Nótese que la versión posterior de la GPL, la v.3 (29 de junio de 2007), al declarar la ausencia de garantía, ha omitido la oración subordinada causal «comoquiera que el programa venga licenciado gratis». Lo cual nos lleva a un punto sumamente importante y delicado: la gratuidad.

El preámbulo de la Licencia contiene una explícita autorización a los receptores del *software* de cobrar por las copias que ellos distribuyan, ya sea del mismo o de las obras derivadas. Repítese ese permiso en la cláusula 1ª.[152]

Mas lo que nos interesa es que la versión de 1991 presupone que la obra colocada bajo la GPL se ha entregado gratis al adquirente (aun permitiendo cobrar por el servicio de distribución el precio que le plazca al distribuidor, caro o barato), siendo ésa la justificación de la falta de garantía. La GPL de 1991 constriñe así a quien, cobrando un precio, distribuya una copia del *software*, original o modificado (exigiendo así un pago al adquirente) que declare implícitamente que «el programa viene licenciado gratis ('free of charge')».

¿Qué es venir licenciado? ¿En qué difiere, con respecto a un *software*, venir licenciado de venir distribuido o transmitido? Si tengo que pagar un dinero al distribuidor para que me entregue él una copia del *software*, ¿en qué sentido cabe decir que se me ha licenciado gratis?

Hasta la saciedad (*usque ad nauseam*) han repetido todos los documentos emanados de la FSF, de la pluma o la palabra de Richard Stallman y de los demás adalides del *software* «libre», que la libertad preconizada es aquella que se expresa en el adjetivo inglés «free» en el sentido de «free speech»,

---

152. Trátase de una de las autorizaciones ociosas, ya que la legislación sobre la propiedad intelectual no prescribe la no-venalidad de la copia a menos que la haya preceptuado el autor.

no de «free beer»; *libre*, no *gratuito*.[153]

Ya he abordado este tema, más arriba, en el capítulo III de este libro. En mi opinión, lo que no es gratis no es libre; permítese su realización únicamente bajo una condición, el pago. Cuando a las féminas les estaba prohibido entrar en una iglesia sin llevar la cabeza cubierta, su entrada no era libre, como sí lo era para los varones. Que la condición impuesta ataña al atuendo, al pago o a cualquier otra circunstancia, el hecho es que un derecho condicional no es una libertad.

¿Por qué insisten Stallman y todos sus seguidores liberistas en la plena compatibilidad de la no-gratuidad con la libertad, en su concepción? ¿Por qué no colisiona, según ellos, con la libertad del «free software» exigir dinero a cambio de su uso, mientras que, en cambio, sus licencias prohíben a los distribuidores o desarrolladores de un producto derivado imponer otras condiciones, sean las que fueren —p.ej. que, cuando los subdestinatarios se lucren gracias a ese producto, paguen algo a sus desarrolladores?[154]

En efecto, en su penúltima frase estipula la cláusula 6ª: «You may not impose any further restrictions on the recipients' exercise of the rights granted herein», lo cual, leído en el contexto, significa que al distribuidor del programa prohíbesele exigir el cumplimiento de condición alguna (salvo el pago). Por el principio de transitividad, implica eso que tampoco podrán imponerse otras condiciones a la ulterior recepción de obras derivadas —las cuales, a tenor de la Licencia, habrán de sujetarse a esa misma licencia de la obra original.

Condicionar al desembolso de un dinero no cercenaría, pues, la libertad mas condicionar de cualquier otro modo sí la cercenaría.

Aunque los CDRoms de Slackware siempre fueron baratos (y además en seguida pudo descargarse gratis esa distribución mediante el ftp), otras distribuciones fueron de pago; en particular Yggdrasil.[155]

Era esencial en la ideología liberista ese explícito permiso de la

---

153. Para mayor claridad y rotundidad, los más entusiastas seguidores de esa ideología liberista han incorporado a la lengua inglesa el galicismo o hispanismo «libre» (que reemplaza a «free» o se le añade). Esfuerzo posiblemente vano, pues también en nuestro idioma hay usos de «libre» que significan «gratuito»: «libre acceso», «ingreso libre», «estacionamiento libre», «asistencia libre», «libre lectura».

154. De donde se sigue esta paradoja: si soy yo un desarrollador de *software* derivado de un *software* libre, me es lícito exigir un pago a quienes lo quieran adquirir —adquirir pagando—; mas me está prohibido imponerles la obligación de que, en el supuesto de que ellos se lucren gracias a mi programa, entonces, sólo entonces, me compensen económicamente por mi trabajo. V. *infra* la discusión sobre la Licencia de Aladdin.

155. V. Peter H. Salus, *The Daemon, the GNU and the Penguin* (Reed Media Services, 2008), capítulo 20, «Proliferating Penguins, Part I» (disponible en Kindle), donde se afirma que el precio de venta de los CDRoms de Yggdrasil era —en aquellos años noventa— de 99 dólares, suma entonces nada desdeñable, aproximadamente equivalente a unos 200 dólares de hoy (habiendo sido de un 2'52% anual la depreciación del dólar en los últimos cinco lustros).

venalidad, constituyendo el fanal anunciador de que no se trataba de ninguna corriente hostil al mundo del dinero, al capitalismo, a la empresa privada, al mercado.

A lo largo de ocho lustros, en sus múltiples alocuciones públicas, el apóstol del movimiento, Richard Stallman, ha reiterado los fundamentos y las consecuencias de la ideología subyacente a la propuesta de la GPL. En su binaria visión sólo existen dos alternativas: o bien adóptase para el *software* acogerse al *copyleft* —cuya más idónea, clara y precisa enunciación está encarnada en la GPL—, o, si no, viviremos bajo el dominio de las grandes firmas (*corporations*) que nos impondrán qué *software* usar, de qué manera, bajo qué condiciones, para qué fines, vigilándonos a fin de que cumplamos tales imperativos; o sea viviremos sin lo más sagrado y precioso, la libertad, único fin y valor de la vida humana.

Una sociedad libre será, según esa concepción, un cúmulo de individuos libres, cuya vida dependa, exclusivamente, de su propia voluntad. Cuando nuestro manejo de aparatos electrónicos esté sujeto a la voluntad ajena —o sea, a las imposiciones de esas grandes firmas desarrolladoras de *software* «propietario» o privativo—, entonces no seremos libres ni lo será la sociedad que, juntos, constituyamos.

La libertad es, así, el único valor que guía este movimiento del *software* «libre». La única finalidad de haberlo puesto en marcha y de seguir propulsándolo y defendiéndolo es preservar la libertad. Nada más. No entran aquí para nada otros valores, cual pudiera ser el bienestar de los individuos; menos aún la armonía o el equilibrio social o el bien común de la sociedad.

---

§4.— De cómo nos sojuzgan los oligopolios del *software* privativo

Asoma una paradoja en el hecho de que, mientras este movimiento profesa la libertad como único valor —coincidiendo en eso con el libertarianismo—, viene, así y todo, a rezumar como un recelo frente al gran capital, justamente porque, en este campo, tienden las más poderosas firmas a convertirse en monopolios u oligopolios que cercenan nuestra libertad.

Esas grandes firmas resguardan su *software* como un bien «propietario», suyo, vendiéndonoslo en condiciones contrarias al disfrute de las cuatro libertades proclamadas en el Manifiesto GNU. Repróchaseles, no que lo vendan (caro o barato), sino que nos impongan condiciones, lo cual choca con nuestra presunta libertad a usar el programa como nos dé la gana. (Las condiciones han ido amontonándose al arbitrio de los magnates de MATAFA).

Comparando las agobiantes condiciones de uso de un programa «propietario» o privativo de hoy —particularmente el horrendo Windows— con las del DOS de hace siete lustros —e incluso las de otros programas de pago de entonces, como el WordPerfect o el Back-and-Forth—, caigo en la cuenta de que en aquella época podía uno copiar el programa (de pago, ciertamente) en

cuantas máquinas quisiera, así como en dispositivos de almacenamiento externos (que inicialmente eran tan sólo los disquetes o *floppies*) para, posteriormente, instalarlo de ahí en un nuevo computador; podía hacerlo en el doble sentido de que ni se lo impedía artilugio alguno ni se lo prohibía ninguna cláusula contractual.[156]

---

§5.— La *Open Source Initiative*

La mencionada paradoja dio lugar al surgimiento, en 1998, de una desviación del movimiento del *software* libre: la OSI, *Open Souce Initiative*, bajo el doble impulso de dos informáticos, el libertario Eric Raymond —autor de la célebre obra *The Cathedral and the Bazaar*— y Bruce Perens. Ese nuevo grupo lanzó OSD, «Open Source Definition»; cualquier *software* que se emitiera en conformidad con esa directiva de la OSD se llamaría «*software* de fuente abierta».

Es jugoso mencionar que Perens, anteriormente, había redactado, para la distribución Debian, el «Debian Social Contract», cuyo eje lo constituían las «Debian Free Software Guidelines» (también de su autoría), las cuales, por un lado, radicalizaban el espíritu liberista de Stallman, mas, por el otro, abandonaban el requisito del *copyleft*, considerando igualmente «libre» el *software* difundido bajo las llamadas «licencias permisivas» (las del BSD, el MIT, Apache etc), las cuales autorizan a los receptores del *software* libre a producir obras derivadas bajo licencias no-libres.

Habiendo un parcial solapamiento entre el *software* emitido bajo *copyleft* y bajo la OSD, existe una diferencia esencial, ya que la OSD permite, mas no impone, que las obras derivadas se distribuyan bajo los mismos términos de la licencia original. La OSI abandona el discurso stallmaniano sobre las cuatro libertades, recalcando, en cambio, la exhortación a publicar el código fuente, no por mor de respetar derechos, sino en aras de la eficacia, ya que, cuantos más ojos escudriñen, mejor.

Disgustaba a los promotores de la OSI (principalmente a E. Raymond) una larvada hostilidad a la empresa comercial no sólo en el «GNU Manifesto», sino en todo el discurso de la *Free Software Foundation* (aunque doctrinalmente no fuera ésa la intención que había animado al equipo GNU).

Reaccionó Stallman insistiendo en que lo importante no era la eficacia, sino el respeto a las libertades. ¿Llevaba razón? Si el argumento correcto a favor de adoptar *software* de fuente abierta fuera el de la eficacia, eso habría de notarse en los resultados. Muy difícil es cerciorarse de esa ventaja cuando no

---

156. La única excepción, en mi caso, fue el carísimo primer OCR, el ReadStar —comprado en noviembre de 1990—, que únicamente podía funcionar en una máquina en la cual se hubiera adosado al puerto paralelo —aquel que entonces transmitía los datos a la impresora— un *dongle*, cuya detección requería, además, tener la impresora encendida.

tenemos dos productos comparables, el uno de fuente abierta y el otro no. Así, Windows, obviamente, es de fuente cerrada y Linux de fuente abierta. ¿Cuál es mejor? ¿Qué imparcial jurado emitirá un veredicto? Para los linuxitas es mejor el Linux, mas para los ventaneros lo es el Windows.

Si el motivo a favor del *software* libre es un argumento deóntico, la libertad, entonces (al menos deónticamente) es preferible ese *software*. Le doy la razón a Stallman. En verdad juzgo al Windows un pésimo programa mas no así a todo el *software* «propietario» o no-libre (v.g el WordPerfect 2021 de la casa Alludo [ex-Corel]). Por mucho mejor que fuera el Windows —y peor el Linux— subsistiría, para mí, la ventaja de la libertad de uso del Linux (si bien —según se va a ver en seguida— mi visión de la deseable libertad discrepa de la del ideario liberista).

---

§6.— Defensa del *freeware* frente al *software* «libre»

Aun entendiendo contraria al ideario liberista cualquier licencia que no sea de *copyleft*, ha acuñado ese movimiento una definición de «*software* libre» que también se aplica al emitido bajo licencias permisivas. En cambio rechaza como «propietario» todo *software* que se emita bajo una licencia que imponga a los receptores cualquier otra obligación así como cualquier *software* que, siendo gratuito, no se publique con licencia libre.

Stallman y sus seguidores han llevado una campaña contra el *freeware*, que tanto usé yo en mis años con el DOS y el OS/2, o sea programas cuyos autores los difundían en aquellos CDRoms de Walnut Creek, gratis, ya pidiendo que el usuario les retribuyera con alguna donación (*giftware*), ya ofreciendo un producto mejor a trueque de un pago o autorizando el uso gratuito sólo por determinado tiempo (*shareware*) o para uso personal (exigiendo un pago cuando el usuario fuera un establecimiento.)

A mi modo de ver, beneficiase —y mucho— el usuario final normalito y modesto con el *freeware*, siendo eso mucho más importante que su libertad de estudiar y modificar el código fuente y de producir obras derivadas; la mayoría de los usuarios no vamos a hacer nada de eso; limitámonos a usar los programas. Apreciamos, sí, la libertad, mas otras libertades:

— La libertad de instalar el programa en cuantas máquinas queramos y guardarlo en dispositivos de almacenamiento externo.

— La libertad de descargar aplicaciones diversas, usarlas o desecharlas, ya sea incondicionalmente (plena libertad), ya sea bajo condiciones asumibles (p.ej. módico precio a pagar tras un período de prueba).

— La libertad de configurar el programa a nuestro propio modo.

— La libertad de poner o quitar programas en nuestra máquina sin dar cuentas a nadie, sin tener que registrarnos como usuarios.

Ese *freeware* nos es favorable, mientras que no lo sería uno carísimo por

mucho que el vendedor nos permitiera a nosotros redistribuirlo gratis y producir obras derivadas.

¿Por qué prefiero yo el *freeware* gratuito o barato al «free software» caro? Porque el único derecho que reclamo es el de beneficiarme del bien común. Una parte del bien cumulativo de la sociedad es la producción intelectual. Otra es la producción de bienes materiales. Como miembro de la sociedad, por el pacto social o vínculo sinalagmático de solidaridad,[157] asísteme un derecho fundamental único: el de participar del bien común.

Por otro lado (según lo voy a demostrar) mi derecho de libertad no comprende ninguna de las cuatro libertades de Stallman. Ni las reclamo yo ni se las reconozco a los demás con respecto a mis propias obras intelectuales, las cuales, al revés, se difunden gratuitamente (en lo que de mí depende), mas con condiciones.[158]

Si mi derecho a la libertad no comprende ni una sola de las cuatro libertades stallmanianas, existe, en cambio, un derecho de creación intelectual de los inventores de *software*, que sí constituye un derecho fundamental, extendiéndose al de —en el marco de la legislación vigente— difundir su propia obra como quieran. Si aceptáramos la vigencia de las cuatro libertades de Stallman, tendríamos que recusar ese derecho del creador de una obra intelectual de escoger los modos y las condiciones de su difusión.

---

§7.— Trece objeciones al ideario liberista

1ª Objeción.— ¿En qué se basa la afirmación de que cada individuo (humano) es titular de esos cuatro derechos de libertad?

No he leído en los textos del movimiento GNU y de sus seguidores ningún argumento a favor de esa atribución de derechos; ni en las conferencias de Richard Stallman he escuchado tampoco razonamiento alguno, salvo estos dos:

1º) todos merecen disfrutar de esos cuatro derechos;

2º) la alternativa al reconocimiento de tales derechos es una sociedad en la que nuestras vidas estén dominadas por la voluntad de los oligopolios electrónicos e informáticos.

El primer argumento suena a mera petición de principio. ¿Hay diferencia entre tener un derecho y merecer disfrutar de él? Con todo, ese uso del verbo

---

157. Que he defendido en mi libro *Estudios Republicanos*, Madrid: Plaza y Valdés, 2009.

158. Cierto es que Stallman, en un principio, entendía sus cuatro libertades atinentes sólo al *software*, no a cualquier obra intelectual; el GNU acuñó otra licencia para obras intelectuales que no fueran *software*: la «Free documentation License», GFDL (cuyo uso ha sido, en buena medida, desplazado por la amplia aceptación de otro tipo de licencias «libres», las de la organización *Creative Commons*, originada por el jurista Lawrence Lessig).

«merecer» me causa algún malestar. En mi captación semántica, hay dos sentidos de «merecer»: uno, redundante, el de «tener derecho» («todos merecen no ser humillados ni maltratados»). El segundo sentido es el del mérito o merecimiento, propiamente dicho, un título que se adquiere en virtud de algún esfuerzo, alguna contribución propia; tal alumno merece una buena nota o tal acusado merece una pena apropiada a su conducta delictiva.

Si el primer sentido no nos sirve aquí de nada para justificar la atribución a todos de las cuatro libertades del liberismo, tampoco nos proporciona basamento alguno el segundo sentido. ¿Quiénes han acumulado méritos que los hagan acreedores de esas cuatro libertades? Los habrá, no digo que no, pero ¿todos? ¿Por el mero hecho de nacer miembro de la especie humana, o, más bien, de la subespecie de *homo sapiens sapiens*?

Falla, pues, totalmente el primer argumento, salvo como flagrante petición de principio.

Al segundo argumento cabe oponerle que hay otras alternativas. Una de ellas es la que, efectivamente, ofrecen casi todas las distribuciones de Linux, que combinan *software* «libre» —en el sentido liberista del *copyleft*— con otro que no lo es. Hácenlo para dar un buen servicio a los usuarios a fin de que puedan valerse del sistema operativo para manejar sus computadores, sus pantallas, sus impresoras y demás periféricos, estando ellos mismos constreñidos por lo que hayan querido suministrarles los fabricantes de ese *hardware*.

A lo cual hay que agregar la posibilidad de adjuntar a tales distribuciones programas adicionales que, siendo compatibles con el Linux, sean emitidos por sus respectivos desarrolladores bajo condiciones particulares.[159],[160]

No ha conseguido ofrecer el movimiento liberista ni una sola justificación de las cuatro libertades que preconiza. ¿Se lo ha propuesto? Lo dudo. Quizá sus adalides son intuyentes; los intuyentes intuyen, no molestándose en demostrar.

. . .

2ª Objeción.— ¿A qué orden normativo pertenece esa atribución de derechos y sus correspondientes deberes? (Establécese la correspondencia en

---

159. Lo cual ha sucedido en varios casos —como el CDRtools del difunto Jörg Schilling y el Ghostscrip de Aladdin (sobre el cual voy a volver más abajo).

160. El equipo GNU rechaza esa manga ancha de la mayoría de las distribuciones del Linux, considerando inmoral usar *software* no libre. Avala únicamente las siete siguientes distribuciones: Dragora GNU/Linux-Libre, gNewSense, GNU Guix System, Hyperbola GNU/Linux-libre, Parabola GNU/Linux-libre, Trisquel GNU/Linux y Ututo. Aun careciendo de ese aval, ufánanse, no obstante, de ajustarse a la exclusión de *software* no-libre otras ocho: PureOS, Arch Linux, Canaima GNU/Linux, Gentoo Linux, Dyne:bolic16, GNUinOS, ProteanOS y Uruk GNU. De ésas quince, una, Arch Linux, goza de popularidad.

virtud de un principio lógico-jurídico de necesaria correlación entre derechos y deberes.[161])

En el uso común y corriente, solemos usar los verbos «debe», «debería», «derecho» —y muchos otros con significación parecida o afín— sin precisar qué orden deóntico tomamos como marco de referencia —como dando, acaso ingenuamente, por supuesto que hay uno solo.

No es así para el teórico de la axiología y de la normatividad. Existe un orden ético o moral; existe un orden jurídico (que se subdivide en el jurídico-natural y el jurídico-positivo); existe, verosímilmente un orden de deontología política, o tal vez varios; puede que exista un orden deontológico y axiológico de relaciones sociales (p.ej. uno de buenas prácticas en los negocios, uno de buena vecindad, uno de compañerismo entre colegas, etc, los cuales carecen de refrendo jurídico).

¿Tienen todos derecho a ejecutar cualquier *software* como quieran y para lo que quieran, a estudiarlo, a modificarlo, a propagarlo, alterado o sin alterar? Por consiguiente, ¿tienen los desarrolladores de cualquier *software* obligación de permitir y aun facilitar a todos y cada uno el disfrute de esos cuatro derechos? ¿En qué orden normativo?

¿Es antijurídico, ilegal, incumplir esa obligación? Desde luego no lo es en el ámbito del derecho positivo, mas podríamos concebir que lo fuera en el del Derecho Natural.

Únicamente voy a plantear el problema desde mi personal teoría del derecho natural, el jusnaturalismo aditivo del bien común, o «jusnaturalismo laurentino», cuyo eje es la lógica nomológica.[162]

Según esta teoría, existe un único deber de los miembros y moradores de la República, que es el de contribuir al bien público. Y existe un único derecho axiomático, que es el de participar de ese bien mismo público o bien común.

De ese único derecho primitivo derívanse dos grupos de derechos. El primero es el de aquellos cuyo disfrute es colectivo y no distributivo; trátase, pues, de una titularidad y un goce *pro indiviso*, en mancomún.[163]

---

[161]. V. Lorenzo Peña, «La correlación lógico-jurídica entre deberes y derechos», *Persona y Derecho: Revista de fundamentación de las Instituciones Jurídicas y de Derechos Humanos*, Nº 61, 2009, pp. 73-102. ISSN 0211-4526. (Accesible en http://jurid.net/articles/ius/deberes.htm [HTML] y en http://hdl.handle.net/10261/19862 [PDF].)

[162]. V. Lorenzo Peña, *Visión lógica del derecho: Una defensa del racionalismo jurídico*. Madrid: Plaza y Valdés Editores. ISBN 9788417121068. (Acc. en http://jurid.net/books/vision/VLD1.pdf [formato B5 no imprimible].)

[163]. Para el individualismo, los derechos del hombre son de titularidad y disfrute puramente individuales. Una radicalísima expresión de tal individualismo hallámosla en el pseudoliberalismo de John Rawls, para quien la sociedad correcta es aquella que podría establecerse a tenor de un unánime convenio de todos, buscando cada cual únicamente su propio interés —sólo que ignorándolo todo sobre sí mismo, salvo su propia existencia. Rawls,

Resúmense esos derechos en uno: el de ser miembro y habitante de una República con estas siete cualidades:[164] 1ª, humanista —y, por lo tanto, abierta, acogedora—; 2ª, rica y próspera;[165] 3ª, internacionalmente respetada y segura;[166] 4ª, floreciente en obras públicas,[167] en producción industrial y agraria así como en artes, letras, ciencias, tecnología, saber y cultura; 5ª, bien gobernada; 6ª, cuya administración proporcione a la población un denso entramado de eficientes servicios públicos;[168] y 7ª, donde impere la justicia —lo cual significa tratar a todos según reglas acordes con el bien común.[169]

---

concretamente, proscribe la envidia (y así, indirectamente, cualquier inquietud por las desigualdades, para las cuales su «teoría de la justicia» no prevé tope alguno —siempre, eso sí, que redunden en provecho de los más desfavorecidos [o sea, de los más desfavorecidos de entre los más desfavorecidos]).

Con ese mismo criterio, cabe excluir, no sólo el altruismo, sino también el cuidado por el bien conjunto.

[164]. No desconozco que ese ideal es inalcanzable para los habitantes de los microestados, desde Barbados a Tuvalú, pasando por Lesoto, Malawi o las Islas Lucayas. E incluso para muchos otros. Frente a la quimera rawlsiana de un ficticio pacto social ubicado en un pasado mítico, propongo yo que los pequeños y medianos Estados se vayan fusionando en otros más viables. Nada de organizaciones supranacionales —que no resuelven las dificultades e incluso las agravan—, sino una genuina fusión estatal, sin marcha atrás. (En lugar de promover esa racionalización de las relaciones internacionales, la máxima potencia mundial actúa para el desmembramiento y la fragmentación de varios Estados actualmente existentes.)

[165]. Lo cual significa, no tanto haber conseguido ya ser rica, cuanto aplicar políticas que, mediante el crecimiento de las fuerzas productivas, garanticen a la población una perspectiva de vida mejor. Prosperidad más en sentido dinámico que estático.

[166]. Seguridad frente a peligros y amenazas del exterior y del interior —v.g., el riesgo de secesiones. Para que sea un motivo de orgullo ser miembro de una República, ésta ha de ser amada pero también temida por los enemigos del bien común. Siguiendo la doctrina de Fray Santiago Ramírez Dulanto, O.P., es el mejor régimen aquel que más firmemente garantiza la unidad de la comunidad política, sin la cual destrúyese el bien común.

[167]. Unas obras públicas que no deberían ignorar los imperativos de belleza. Siempre los tuvieron en cuenta las repúblicas antiguas, pero hoy, cuando se evalúan las políticas gubernamentales, ni siquiera entra en consideración la hermosura del espacio público.

[168]. Según lo sostuve en la *Introducción* del volumen colectivo *Ética y servicio público* (ed. por L. Peña, T. Ausín & O. Diego, Plaza y Valdés, 2010):

> es nuestro propósito contribuir a una reflexión doctrinal que abogue por un servicio público revigorizado, sobre bases sólidas, orientado por (…) diez criterios de buena regulación: Eficacia, calidad, suficiencia, igualdad, continuidad, rapidez, seguridad, economía, adaptatividad e innovación.

[169]. ¿Cuáles reglas son idóneas para el bien común? Acaso no sea ocioso recordar lo que decía Karl Marx en sus *Glosas al Programa de Gotha*, en 1875. A pesar de su repugnancia a las predicciones, lo había forzado la polémica con los lassallianos a diseñar una perspectiva de futuro, que él partía en dos. A corto plazo, el socialismo, donde, exigiéndose de cada quien según sus aptitudes, se remuneraría a cada uno según su trabajo (fórmula indeterminada, pero que, al menos, marca una pauta orientativa). A largo plazo, el comunismo —una vez que manaran a raudales las fuentes de la abundancia—; en él, de nuevo, cada quien aportaría a la sociedad según sus capacidades pero, en cambio, recibiría según sus necesidades.

Con otras palabras, es el derecho a que existan suficientes razones para enorgullecerse uno de pertenecer a tal República.[170]

El segundo grupo es el de los derechos cuyo disfrute es distributivo; subdivídense en derechos de bienestar y derechos de libertad.

Los primeros son aquellos cuyo disfrute es necesario para —gracias a las prestaciones de la comunidad o de determinadas personas particulares, físicas o jurídicas— conseguir una cantidad y calidad de vida acordes con el nivel de crecimiento que histórico-socialmente hayan alcanzado las fuerzas productivas. Un derecho de bienestar es un derecho a algo, un derecho cuyo contenido se expresa por un enunciado existencial (p.ej. que haya una casa en la que vivamos).[171] ¿Cuáles son tales derechos? Eso dependerá del nivel de crecimiento de las fuerzas productivas y del desarrollo social. En la época actual hay diez derechos de bienestar generalmente aceptados: empleo

---

Incluso de entre quienes siguen proclamándose sus lejanos seguidores, pocos hoy creen en el advenimiento de esa sociedad de plena abundancia. Mi opinión personal es que los criterios distributivos y redistributivos de la sociedad real han de ajustarse a parámetros que tengan en cuenta tanto las necesidades cuanto el merecimiento (positivo o negativo). Según en qué casos y en qué circunstancias, más lo uno o más lo otro. En última instancia, el criterio no puede ser más que uno: el bien común.

Un bien común que resulta incompatible, no sólo con excesivas desigualdades de fortuna, sino también con cualquier acaparamiento particular de bienes materiales que empequeñezca la riqueza pública.

170. Marca el párrafo que acabo de escribir —y que perfila, por vez primera, la concepción republicana de los derechos del hombre— una ruptura radical y absoluta con todas las demás concepciones de los derechos humanos (e incluso una revisión de cuanto, en pasados ensayos, propuse al respecto yo mismo).

Para todas las demás visiones de los derechos humanos, éstos son de naturaleza y disfrute plena y exclusivamente individual; o sea, son distributivos, jamás colectivos. De reconocerse, v.g., un derecho a la justicia, será únicamente el de cada quien a venir justamente tratado, no el derecho a vivir en una sociedad donde impere la justicia. Asimismo, en abordándose —en el contexto del tratamiento de los derechos del hombre— el tema de los servicios públicos, será, nada más, en el sentido del derecho individual mío a beneficiarme de tales servicios (mío y de Ud. y de cada cual, por separado —*singillatim*, no *conjunctim*). Desde la teoría de los derechos del hombre sería indiferente que la sociedad fuera pobre o rica, oscurantista o ilustrada, caótica o funcional. A lo sumo, hallaríase en toda esa problemática una cuestión de medios o instrumentos para la satisfacción de los derechos del hombre, mas no, de suyo, una de derechos humanos fundamentales.

Lo que estoy defendiendo es el derecho, no a estar orgulloso uno de vivir en su República, sino a que existan suficientes razones para ese sentimiento. (Así como el derecho individual a, cuando falten tales razones, ingresar en otra República donde sí las haya.)

Ese derecho colectivo es de bienestar, o sea de prestación. Trátase del bienestar colectivo, del bien de la República. Siendo un derecho colectivo de prestación, el acreedor es la República —o sea todos los miembros del cuerpo político colectivamente unidos—, al paso que el deudor es cada uno por separado.

171. V. Lorenzo Peña, «La fundamentación jurídico-filosófica de los derechos de bienestar», en *Los derechos positivos: Las demandas justas de acciones y prestaciones*, ed. por Lorenzo Peña y Txetxu Ausín. México/Madrid: Plaza y Valdés, 2006. ISBN 9788493439552, pp. 163-386 (acc. en http://hdl.handle.net/10261/10601 [PDF]).

(decentemente remunerado),[172] abastecimiento, protección civil, vivienda, movilidad, salud, instrucción, cultura, esparcimiento y descanso (lo cual incluye un seguro vitalicio de vejez e invalidez).

Los derechos de libertad son aquellos cuyo correlato es la abstención ajena de constreñirnos; su contenido es disyuntivo, ya que sólo tenemos una libertad de A cuando la tenemos igualmente de no-A. Son derechos de libertad derivados del fundamental derecho a participar del bien común únicamente aquellos sin cuyo goce viviríamos más oprimidos que beneficiados por la República de la cual somos miembros.

En virtud del desarrollo y el progreso sociales, varía históricamente el elenco de los derechos de libertad, mas, en la actual época histórica, reputamos tales los once derechos siguientes: 1º, hacer cuanto, sin ser preceptivo, tampoco sea ilegal;[173] 2º, ir y venir; 3º, fijar nuestra residencia; 4º, contraer matrimonio;[174] 5º, preservar nuestra intimidad individual y familiar; 6º decidir nuestra profesión; 7º, asegurar nuestra defensa propia y ajena; 8º, crear obras —inalienables e imprescriptibles— fruto de nuestro trabajo intelectual; 9º, decir y publicar francamente nuestras opiniones;[175] 10º, juntarnos con

---

172. No forzosamente por cuenta ajena. Podríamos llamarlo «el derecho a ganarse la vida con su propio trabajo». Sólo que ese *ganarse la vida* encierra dos importantes connotaciones: 1ª, con perspectiva de una vida mejor; 2ª, no sólo de uno mismo, sino de los suyos, de su familia.

173. Mas, por otro lado, siendo el buen gobierno un requisito esencial del bien público —a cuya existencia tenemos derecho—, no hay buen gobierno cuando el poder legislativo haya edictado leyes arbitrarias, que nos impongan prohibiciones no justificadas en aras del bien común. El art. 5 de la «Declaración de los derechos del hombre y del ciudadano» del 26 de agosto de 1789 enuncia, conjuntamente, ese doble principio: tanto la interdicción de prohibiciones innecesarias para la pública utilidad cuanto el derecho de todos a realizar cualquier conducta no prohibida por la ley.

174. V. mi ensayo «Disputación sobre el fundamento y el alcance del derecho a contraer matrimonio», DOI: 10.13140/RG.2.2.28878.46409, http://hdl.handle.net/10261/137546. Estriba el contenido de esa libertad en instituir, de mutuo acuerdo, una sociedad conyugal con derecho de ambos y cada uno a permanecer en ella. (Nótese que el acuerdo es para un hecho conyuntivo: entrar y permanecer —no tratándose, pues, de dos acuerdos sucesivos. El consentimiento implica un compromiso.)

175. Hablar sinceramente es hacerlo verazmente. No se trata de un derecho a proferir cuanto a uno le dé la gana. (Ni a realizar otros comportamientos vagamente subsumibles bajo una genérica «libertad de expresión».)

Notemos, empero, que a menudo son confusas y confundentes las formulaciones de este derecho en los textos jurídicos que lo reconocen. Así, la primera enmienda de la Constitución federal estadounidense prohíbe al congreso restringir la libertad de palabra («freedom of speech»); y, claro, la palabra puede no ser veraz; al hablar podemos contar embustes. ¿Era intención del constituyente norteamericano permitir también el derecho a la mentira como un derecho fundamental? Resúltame dudoso.

La «Déclaration des droits de l'homme et du citoyen» del 26 de agosto de 1789 prescribe en su art. 11: «La libre communication des pensées et des opinions est un des droits les plus précieux de l'homme». Aquí, al igual que en otros textos, agazápase el equívoco de los *pensamientos*, puesto que «pensar» tiene dos sentidos: el uno es el de creer, opinar; mas el otro es el de ocurrírsele a uno, pasarle por las mientes, concebir (concebir a título de algo meramente concebible, así sea ficticiamente). ¿Está la Asamblea de Versalles considerando

otros (como no sea para fines deshonestos); 11º, vivir según nuestras propias y libérrimas convicciones.

No veo cómo deducir las cuatro libertades stallmanianas de esos once derechos fundamentales de libertad. Ni, menos aún, de los diez derechos fundamentes de bienestar.[176] Que yo no vea cómo deducirlos no implica forzosamente que sean indeducibles. Juzgo, empero, sumamente inverosímil tal deducibilidad. Es fácil imaginar escenarios en los cuales no se reconocen ni se respetan las cuatro libertades stallmanianas pero sí (en mayor o menor medida) los veintiún derechos fundamentales que he enumerado.

Resúltame claro que incumbe la carga de la prueba a los stallmanianos, quienes habrán de demostrar que, sin esas cuatro libertades atinentes al *software*, es imposible gozar de los veintiún derechos fundamentes y, por lo tanto, participar del bien público.

---

precioso el derecho a afirmar lo que a uno se le ocurra a sabiendas de su absoluta falsedad?

El Convenio europeo de 1950 para la salvaguardia de los derechos del hombre (Convenio de Roma), en su art. 10, afirma que toda persona tiene derecho a la libertad de expresión, especificando que ese derecho comprende la libertad de opinión y la de comunicar ideas. ¿También ideas que el comunicante sepa, o crea saber, que son falsas? Entendido en su conjunto, parece claro que el citado artículo instituye la libertad de opinar y de expresar tales opiniones, no la de proferir embustes.

El art. 19 de la Declaración universal de los derechos del hombre de 1948 dice que todo individuo tiene derecho a la libertad de opinión y de expresión. ¿Opinión *y* expresión? ¿Es un par de derechos? ¿O es un único derecho, el de opinar y decir lo que uno opina?

El Pacto internacional sobre los derechos civiles y políticos de 1966 dice en su art. 19: «1. Nadie puede ser inquietado por sus opiniones. 2. Toda persona tiene derecho a la libertad de expresión». Entiendo que es idéntico el alcance subjetivo de ambas cláusulas y que la 2ª ha de comprenderse en relación con la primera; o sea, que toda persona tiene derecho a expresar sus opiniones, no a contar embustes. (Claro que puede equivocarse; es ésa una cuestión absolutamente diversa e independiente.)

El art. 34 de la Constitución republicana española de 1931 manda: «Toda persona tiene derecho a emitir libremente sus ideas y opiniones». ¿Ideas *y* opiniones? ¿En qué estriba ese distingo? Sea como fuere, no se otorga a nadie el derecho a proferir lo que no cree.

Llegamos al art. 20.1.a de la actual Constitución borbónica de 1978, que nos reconoce el derecho a «expresar y difundir libremente los pensamientos, ideas y opiniones»? ¿Dónde está la diferencia semántica entre «pensamiento», «idea» y «opinión»?

Si algún legislador ha deseado conferir a los legislados un derecho a la mentira, no se ha atrevido a enunciarlo. Sería fácil: «Todos tienen derecho a decir y publicar lo que quieran, veraz o mendazmente».

Todo derecho de libertad es un derecho a hacer o no hacer; en este caso, a decir sus opiniones o callarse. (Nadie está obligado a decir lo que piensa.) No a decir lo que opina o lo que no opina.

(Notemos que, si la libertad de palabra y publicación atañen exclusivamente a la emisión de las opiniones, entonces resulta dudoso que puedan entenderse como derechos también de las personas jurídicas.)

176. ¿Podríamos conjeturar que las cuatro libertades stallmanianas se derivaran directamente del derecho al bien común —o sea a la existencia de razones suficientes para sentirse uno orgulloso de ser miembro de su República? Eso significaría que fuera colectivo el disfrute de tales libertades, no distributivo, no individual; lo cual se halla en las antípodas de la ideología liberista. ¿Sería mejor una sociedad con las cuatro libertades stallmanianas atinentes al *software*? Resulta dudoso, a la luz de la colisión entre ese cuarteto, por un lado, e intereses legítimos, no sólo de los creadores intelectuales, sino también de los usuarios; colisión que analizo en las páginas siguientes.

Queda así descartado que las libertades stallmanianas vengan preceptuadas por el Derecho Natural o por el Derecho positivo.

¿Vienen prescritas por un código ético o moral o por alguna otra regulación deontológica? ¡Tal vez! Mas sería menester saber cuál es esa regulación o cuál es ese código ético, cómo se fundamente y cuál sea su fuerza de obligar. Dudo mucho que quepa hallarlo partiendo de éticas de mayor o menor aceptación en varias escuelas filosóficas, como la aristotélica, la estoica, la epicúrea, la escolástica, la utilitarista, la aretaica («ética de las virtudes») o la intrinsecista a lo Kant (mal llamada «deontológica»). No recuerdo ningún libro de ética que hable del *software* (ni de nada afín).

. . .

3ª Objeción.— ¿Cómo así unos derechos de libertad acarrean, correlativamente, un deber de prestación?

En efecto, según los textos examinados (el Manifiesto y la GPL), la libertad nº1 —la de estudiar el *software* y modificarlo según sus propias preferencias y para cualquier fin— implicaría el deber del desarrollador del *software* de publicar el código fuente, o sea el programa escrito en un lenguaje de alto nivel (con papel y pluma o con pulsaciones de el teclado). Lo que se le está exigiendo al programador es que, además de entregar (gratis o no) el programa debidamente ensamblado y compilado, transmita también al destinatario el código fuente. Trátase de una obligación de hacer y de dar.

Mas ninguna libertad acarrea deberes ajenos de dar ni de hacer (salvo únicamente el del poder público de proteger al individuo frente a coacciones ajenas que le impidan o estorben ejercer esa libertad). Fúndase esta correlación en uno de los principios de la lógica nomológica, el de no impedimento: es ilícito impedir o estorbar el disfrute de derechos ajenos.[177]

¿Hay casos en los que esté prohibida una omisión en virtud de ese principio de no impedimento —presupuesta, claro, la existencia de un derecho correlativo? Ciertamente, mas únicamente cuando el derecho correlativo sea de bienestar, por lo cual su correspondiente deber será de prestación. Mas ningún liberista ha afirmado el derecho al *software* como derecho de bienestar.

Según su concepción, el inventor de un *software* no viola derecho alguno de nadie si se abstiene de comunicarlo o publicarlo. Sólo incurre en conducta censurable si, comunicándolo o publicándolo, gratis o no, omite facilitar el código fuente. Inferir de un derecho de libertad un deber de prestación es incurrir en un *non-sequitur*.

. . .

---

177. V. Lorenzo Peña, «Normatividad y contingencia», en *Aproximaciones a la contingencia*, ed. por Concha Roldán & Óscar Moro, Madrid: Los libros de la Catarata, 2009, pp. 25-64. ISBN 9788483194379. (Acc. en http://digital.csic.es/handle/10261/13295 [PDF].) Hay que agregar el principio de que incumbe al poder público la tarea de prohibir y perseguir toda violación de derechos.

4ª Objeción.— ¿Cómo se demuestra que para «estudiar el *software* y modificarlo» sea necesario poder leer el código fuente? ¿No es posible acudir a la ingeniería inversa?

Cierto, eso no está al alcance de todos, mas tampoco lo está entender el código fuente ni, menos, conseguir alterarlo con algún provecho. Tales posibilidades quedan reservadas a individuos con buenos conocimientos de programación. Los cuales, sin disponer del código fuente, pueden (esforzándose, claro está) estudiar y modificar el programa, aun ignorando su código fuente, mediante ingeniería inversa.

Conque, aun admitiendo la libertad de estudiar y modificar el *software*, lo único que se seguiría sería la ilicitud de las cláusulas contractuales que prohíben la ingeniería inversa.[178]

. . .

5ª Objeción.— ¿Por qué existirían esas cuatro libertades en lo que atañe al *software* mas no con respecto a las demás obras del espíritu? Un programa es una obra intelectual, con peculiares características y funciones. Mas esas peculiaridades no alteran lo esencial: ser una obra del intelecto humano, como lo es un tratado de química, una introducción a la lógica matemática, un libro de filosofía, una novela, una tesis doctoral de historia, una pieza de teatro, una partitura musical, un panfleto o un artículo de revista.

Aceptar las cuatro libertades stallmanianas con relación a cualesquiera obras del espíritu es negar la propiedad intelectual. Richard Stallman la niega. En el derecho estadounidense no se reconoce tal propiedad.[179] Mas en el Derecho internacional existe, en virtud del convenio de Berna.

Yo he defendido la propiedad intelectual.[180] Y sigo defendiéndola. Mas

---

178. Cláusulas posiblemente ilícitas, de todos modos, por chocar con el derecho fundamental a estudiar, a aprender, a instruirse. Tales cláusulas yo las considero nulas. Si tuviera conocimientos informáticos, acudiría gustoso a ingeniería inversa, aun por el puro placer de ejercitar mi intelecto y mi derecho a entender.

179. Cuando los políticos norteamericanos denigran a quienes, según ellos, violan la propiedad intelectual, suelen referirse más bien a la propiedad industrial (patentes y marcas).

180. V. Lorenzo Peña,«Propiedad intelectual privada frente a la investigación como servicio público», *Arbor*, Nº 730, 2008, pp. 307-332. ISSN 0210-1963. (Acc. en http://jurid.net/articles/ius/intelect.pdf y http://hdl.handle.net/10261/7274). ¿Cómo así la defiendo cuando me pronuncio a favor de la abolición de la propiedad privada de bienes materiales? Porque la única justificación persuasiva que he hallado de esta última propiedad es la de Locke: es legítimamente suyo de uno aquello que él ha fecundado o enriquecido con su trabajo. Argumento que me parece sujeto a un serio reparo: quien así se adueña de un bien, que no le pertenecía, está arrogándose un derecho exclusivo sobre algo preexistente a su esfuerzo, mientras que el creador intelectual no se apodera de nada más que del producto de su propio trabajo cerebral. (Bien lo supo ver Diderot, uno de los creadores del concepto de propiedad intelectual.)

(A la objeción de que también el creador intelectual utiliza elementos preexistentes —pues no creamos de la nada—, respondo que no nos apropiamos de nada preexistente, sino que únicamente reivindicamos como

no como propiedad ilimitada, sino sujeta a servidumbres, en virtud de la obligación de cualquier creador intelectual de contribuir al bien común.

Una de esas servidumbres es la de posibilitar la lectura y el estudio de su obra (v.g. autorizando la difusión gratuita en ciertos casos y, en otros, no buscando ningún lucro excesivo).

Mas falta por demostrar que, siendo una obra del espíritu, un *software* venga regulado por un código normativo que no se aplique a las demás obras del intelecto.

. . .

6ª Objeción.— ¿Por qué ese haz de derechos en torno a programas destinados a correrse bajo computadores y no a otros aparatos electrónicos: lavadoras, receptores de radio y TV, enchufes programables, aspiradores, lavaplatos, microondas, secadoras, lavadoras y demás electrodomésticos electrónicos?

No han faltado quienes han exigido aplicar al *firmware* de cualesquiera aparatos electrónicos las prescripciones de «libre *software*», ya que, al fin y al cabo, de *software* se trata. Suelen ser adeptos del liberismo, sólo que de su ala más extremista, quienes abogan por lo que llaman «*software* ético». (Durante algún tiempo, esa tendencia radical giraba en torno al grupo Debian, pero hoy se ha dispersado.)

Reclamando yo otras libertades (no las cuatro stallmanianas), sí vería con buenos ojos que alguna de **mis** libertades se aplicara también al *firmware* de aparatos electrónicos que antes nos dejaban un mayor margen de decisión; p.ej. en una máquina de lavado, podían programarse separadamente la temperatura y la duración, mientras que ahora suele ofrecérsele al usuario un estrechísimo abanico de dos o tres opciones empaquetadas. Se ha cercenado su libertad de uso.

Poco parece afectar eso, sin embargo, al tipo de libertades de la FSF. No resultaría práctico que los fabricantes de lavadoras nos facilitaran el código fuente de su *firmware* para que pudiéramos modificarlo. ¿Juzgamos, por ello, que nuestras vidas están en manos de los fabricantes y diseñadores de su *firmware*?

Una respuesta razonable a esta interpelación sería decir que, mientras es muy limitado y circunscrito el uso de esos aparatos, resulta muchísimo más polifacético y múltiple el de los computadores. ¡Cierto! También es verdad que con el computador tenemos modos de actuar como el teclado. No obstante, trátase de diferencias prácticas, no de principio. Si tan cuestión de principios es abogar por el «*software* libre», no se entiende del todo bien que no se preconice igualmente el «*firmware libre*».

---

nuestra la obra que, a partir de tales elementos, hayamos creado.)

. . .

7ª Objeción.— Con relación a un *software* determinado, ¿en qué momento surgen y cuándo se extinguen nuestros cuatro derechos de libertad stallmaniana?

¿Surgen cuando el creador del *software* lo concibe? ¿Cuando lo escribe tecleándolo? ¿Cuando lo ensambla y compila generando así un binario ejecutable? ¿O únicamente cuando transmite ese resultado? Pero cuando lo transmite ¿a quién? ¿Será cuando lo transmite a un amigo? ¿O sólo cuando lo emite públicamente? ¿Y si lo emite en un sitio web de acceso reservado?

Sea como fuere, ¿qué sucede si el autor del *software* lo retira, valiéndose del derecho de arrepentimiento que el convenio de Berna reconoce a los creadores de obras intelectuales —o sea el derecho de sacar su obra de la circulación? ¿Cesan entonces de existir nuestros cuatro derechos de libertad stallmaniana?

Hágome cargo de que ese derecho de retirada o arrepentimiento es una consecuencia del concepto mismo de propiedad intelectual, espúreo y deplorable para Richard Stallman.[181] El creador podrá suprimir el *software*

---

[181]. Mientras estoy escribiendo estas líneas, entérome de que hace exactamente tres días, el lunes 14 de octubre de 2024, bajo el manto del anonimato, una panda de cobardes acorraladores han lanzado, en la red social Mastodon, una nueva andanada contra el benemérito Richard Stallman; para dar mayor difusión a su repugnante campaña de odio y estigmatización, han abierto un portal en la web, http://stallman-report.org, donde se puede leer (no sin arcadas) su monstuoso «Stallman report».

Habían empezado en 2019 la difamación y el anatema contra el fundador del *software* libre, cuando los linchadores consiguieron que la FSF lo echara. Dos años después dio marcha atrás la Fundación, restituyendo a Richard en su comité —aunque ya sin dejarle volver a ocupar la presidencia.

A raíz de ese retorno difundiéronse dos manifiestos, firmados por millares de individuos y colectividades, el uno en contra y el otro a favor de Richard Stallman.

No se han dado por vencidas las erinias, tornando ahora al asalto para pisotear, triturar y aniquilar al insigne abogado del liberismo (un liberismo que comparten, presuntamente, los nefarios autores del malhadado texto).

Exigen un nuevo «código ético» de la FSF con arreglo al cual vendrá duramente castigado quienquiera que ose pensar como Richard Stallman. Pónenseme los pelos de punta cada vez que oigo hablar de un código ético, pues eso es liberticida, conculcando la libertad de hacer cuanto no esté legalmente prohibido. ¡Cómo me acuerdo del sabio presagio de John Stuart Mill al avisarnos de que la presión de la turbamulta (la «opinión») puede ser una apisonadora más opresiva que los edictos de un déspota!

¿En qué se basa esta nueva anónima acusación? «This report catalogues [...] statements from Richard Stallman [...] on subjects including rape, sexual assault, child sexual abuse, sexual exploitation of children, and more».

Pues bien, he de decir que yo comparto la mayoría de los asertos de Richard Stallman que se le reprochan. P.ej., concuerdo con él en que sólo hay violación cuando hay violentamiento; en que un adolescente de 12, 13, 14 ó 15 años no es un niño; en que son absurdas muchas leyes penales edictadas bajo la presión del puritanismo misándrico; en que los jóvenes de ambos sexos han de gozar de libertad sexual; y en que no deberían penalizarse conductas que ni perjudican al bien común ni tienen víctimas.

Sin embargo, coincida yo o no con sus ideas, lo esencial estriba en que este hostigamiento, este acoso

de su propio repositorio mas no exigir a los demás que prescindan de él. Subsiste, empero, el problema de que el Derecho internacional (valga lo que valiere, seguramente poco) sí reconoce esa propiedad y ese derecho de arrepentimiento.

De todos modos persiste la cuestión del surgimiento. Los liberistas nunca nos han dicho que el creador de un *software* tenga obligación alguna (ni moral ni otra) de difundir su creación. Si, siendo un perfecto egoísta, decide disfrutar de él en solitario, o compartirlo con un puñado de allegados o socios, en círculo cerrado, no habría en su conducta nada reprochable.

Limítase nuestro derecho de cuadrilibertad stallmaniana a exigir que, o bien no se nos dé a conocer el *software* —que podría resultarnos útil—, o bien se nos entregue (aunque sea por un precio elevado) con una licencia de uso del *copyleft*.

¡Vaya! ¡Sea! Preferiría yo que, en vez de esa prescripción, se postulara la de que, teniendo todos el deber de contribuir al bien común, el creador de un programa útil haya de compartirlo, a salvo de su libertad de cada creador de reservar, posponer o terminar la circulación del programa —así como igualmente cohonestando ese deber con la legítima aspiración de ver retribuido su trabajo.

. . .

8ª Objeción.— Ya hemos visto que el ideario liberista somete al creador intelectual de un programa al dilema de, o no difundirlo, o hacerlo bajo licencia de *copyleft*.

Imaginemos que alguien inventa un programa que nos permitiría alterar nuestros documentos PDF reemplazando en ellos las fuentes (tipos de letra) por otras más a nuestro propio gusto. ¿Opta por usarlo sin compartirlo con nadie a fin de epatar con sus propios documentos? ¡Nada que reprochar según el liberismo! Mas supongamos que lo difunde como *freeware*: no ofrece el código fuente (que se guarda para sí), permitiendo usar su programa siempre que sea: 1) sin fin lucrativo; y 2) para usos honestos (a tenor de su propia concepción de la honestidad). Entonces está actuando de manera condenable según el liberismo.

Desconozco cuál será la reacción de los demás. La mía es que ese creador de *software* actuará loablemente difundiéndolo como *freeware* y censurablemente quedándoselo para él solito.

. . .

---

mediático de l@s malign@s detractore/as es una fechoría más de la cultura de la cancelación, imperante en el mundo académico anglosajón —cultura que desmiente las cacareadas libertades de opinión y palabra de las cuales, infundadamente, se ufana el mal llamado «mundo libre», el Occidente.

(Que, en estos temas suela estar yo de acuerdo con Richard Stallman no me impide discrepar absolutamente de la gran mayoría de los juicios políticos que a diario vierte en su portal http://stallman.org.)

9ª Objeción.— La normativa impuesta por el *copyleft* colisiona con los derechos de contribuidor individual a un *software* colectivo gestionado por un establecimiento, tenga o no ánimo de lucro. De hecho aniquila la legítima pretensión a ser retribuido por su contribución.

También en la práctica viene reducido a casi nada el derecho del contribuidor al público reconocimiento de su contribución. (Volveré más abajo sobre esta doble cuestión.)

. . .

10ª Objeción.— El ideario liberista desconoce el deber de contribuir al bien común.

Si lo tuviera en cuenta, no juzgaría irreprochable la conducta de aquel creador intelectual que, ora —sea antojadizamente o por misantropía— prefiere no compartir su obra, ora exige un abultadísimo precio, inabordable para gente modesta.

. . .

11ª Objeción.— Ateniéndonos al clausulado de la GPL y del *copyleft*, podría surgir esta situación absurda. Imaginemos que el equipo GNU vende a alguien el uso de su cúmulo de utilidades por 100 millones de dólares; eso sí, bajo la Licencia GPL (versión 2 o versión 3). Ese alguien puede, claro está, vender mil copias por un precio de 100.500 dólares, logrando así un beneficio; cada subadquirente puede, a su vez, vender mil copias por 120 dólares, también recibiendo una ganancia. Andarán en circulación un millón de copias.

Sólo que ¿quién se atreverá a invertir su dinero en tales operaciones? Al hacer la primera venta, GNU, en cumplimiento de su propia licencia del *copyleft* GPL, tiene que reservarse el derecho de vender otras copias por mucho menos o incluso regalarlas. Ninguna de esas ventas puede ser exclusiva.

Es más, supongamos que GNU hace sólo una única venta, la ya mencionada, sólo que uno de los subadquirentes, que ha comprado su copia por 1005 centenares de dólares, es un riquísimo mecenas, decidiendo regalar a quienquiera copias gratuitas. Sus competidores, que han invertido lo mismo que él, se verán arruinados.

De hecho, ya he hablado de los CDRoms de la distribución pionera Yggdrasil que, costaban, a fines del pasado siglo, 99 dólares. Mas el 20 de octubre de 2004 se lanzó el Ubuntu (versión 4.10, la primera), al precio de 0 dólares. Enviábanse a quien los pidiera CDs gratis, sin costo de envío siquiera y cuantas veces lo solicitara. ¿Quién podía competir con Ubuntu?

Si vamos a la plaza, hallamos puestos de fruta y verdura. Los tomates se venden en el puesto A a 3€ y en el B a 3'25. Las zanahorias están más caras en B que en A. Por la razón que sea, tenemos el hábito de comprar en B. Mas imaginemos que hay un puesto, C, donde los tomates se dan gratis, todos los que pidan los compradores. Está claro que ningún otro frutero seguirá

vendiendo tomates.

Eso es lo que determina la paradoja de la GPL. Nadie puede arriesgarse a vender copias de un programa licenciado bajo esa licencia, porque se enfrenta a una posible y probable competencia ruinosa. (Cuando pida un precio por un CDRom con ese contenido, o por descargárselo de un portal de la red, de hecho estará implorando un donativo.)

El único uso comercial viable de un *software* licenciado bajo la GPL es el soporte posventa (sólo que sin haber precedido venta alguna), como lo ofrecen RedHat (hoy IBM) y Canonical.

En tales condiciones, resulta irrelevante que se consigne en esa licencia el permiso de vender copias del programa, cuando el clausulado, tomado en conjunto, hace infactible semejante venta.

No deja, además, de resultar asombrosa la situación original. El primer creador del programa (o cúmulo de programas), el equipo GNU, podría haber exigido, por una copia, la citada suma o cualquier otra —según sus propias reglas—. ¿Habría hallado compradores? ¡No! Unos segundos de reflexión llevan a la conclusión de que sería una inversión ruinosa.

De ahí que, por mucho que los originadores del movimiento liberista se hayan empeñado en que no van contra el capitalismo ni contra la empresa privada, que defienden la libertad como en *free speech*, no *free beer*, todos hayan comprendido que, mercantilmente, es inviable un *software* así —salvo por el servicio de soporte, que también puede ofrecerse para *software* no-libre.

. . .

12ª Objeción.— Si el creador y desarrollador de un programa de *software* lo coloca bajo la licencia GPL, sitúase exactamente en la misma condición que la evocada en la objeción precedente. No podrá vender a nadie ni una copia.

Imaginemos que, siendo modesto y poco ambicioso, ese creador pone a la venta las copias de su producto al módico precio de 10 €. Sólo que uno de los compradores, mejor ubicado comercialmente por su red de distribución, decide vender copias de su propia copia por 5 €. Ante esa ruinosa competencia, el creador del programa ya no volverá a ganar nada por su obra.

Eso significa que el *copyleft* es esterilizante de la creación de pequeño y modesto *software*, que, en cambio, tiene un campo de oportunidades con variedades de *freeware* o de *shareware*.

. . .

13ª Objeción.— La promulgación de la GPL y del ideario del *copyleft* no ha impedido la proliferación de licencias «libres». Existen ahora más de un centenar.

Son «libres» las licencias que autoricen al receptor del programa cuanto

le permite la GPL, exonérenlo o no de los deberes que le impone esa licencia, siempre y cuando no le imponga ninguna otra obligación.

Son, pues, no-libres cuantas licencias de *software* prohíban, ora modificaciones del programa, ora un uso no-comercial, ora usos inmorales, ora usos para fines éticamente censurables (en el concepto del licenciante). También lo es una licencia que exija reconocer el mérito del creador del *software* de un modo determinado.[182]

Gracias a la GPL ha sido posible aunar, en un sistema operativo, un kernel valiosísimo, el de Linus Torvalds, más un cúmulo igualmente precioso de utilidades, las del GNU, así como una muchedumbre de programas cuya autoría corresponde a terceros. Si se hubiera emitido cada pieza bajo su propia licencia, no se habría podido formar ese armónico conjunto.

Mas, de cara al futuro, ¿por qué ha de imponerse ese modelo para cualesquiera nuevas adiciones útiles?

Dadas todas las desventajas de la GPL y del *copyleft* que he venido enumerando, juzgo preferible abrir el paso a pequeños creadores de *software* adicional, cada uno de los cuales produzca su propia obra, decidiendo libremente su propia licencia, «libre» o no.

Añoro yo, en verdad, aquella variedad de *freeware* y *shareware* para el DOS de los CDRoms de Walnut Creek de los años noventa del pasado siglo (para el sistema operativo DOS). *Software* que podría ser incluso de pago, cual lo es el *shareware*, por precios asequibles y que beneficien a creadores intelectuales, no a los oligopolios transnacionales.

A ninguna distribución le impide ofrecer ese tipo de servicio —enlazando al consumidor final con pequeños desarrolladores de *freeware* (para el sistema operativo Unix/Linux)— el mero hecho de que se acojan a la GPL (en la versión 2 o en la 3) tanto el kernel cuanto las utilidades integradas en la distribución. Mas, al considerar inmoral la producción (y aun el uso) de *software* no-libre, el credo liberista desanima y dificulta la prestación de tal servicio. Es de esperar que en el futuro las cosas cambien.[183]

---

182. Así, mi propia *Licencia laurentina* (https://www.researchgate.net/publication/318902057_Licencia_laurentina), aquella vigente para toda mi obra intelectual (salvo la que, por haber sido publicada comercialmente, está sujeta —al margen de mi voluntad— al respectivo contrato de edición).

183. Atreveríame, en verdad, a proponer algo muchísimo más osado (y, desde luego, a corto plazo, inatendible, bien lo sé): que, en el futuro, el Linux (incluyendo las indispensables utilidades GNU) se distribuya bajo una licencia más similar a la de Aladdin —de la cual voy a hablar en seguida—, de suerte que, cuando una firma mercantil obtenga ganancias explotando el sistema operativo —según lo hacen Canonical y la IBM mediante la oferta de soporte a los usuarios—, reciban también una remuneración los equipos desarrolladores del kernel y de las utilidades esenciales. Habría de transmitirse hacia abajo esa remuneración para retribuir el trabajo de los programadores que nos consagran su tiempo y su esfuerzo, mereciendo una recompensa económica y no sólo nuestro afecto de gratitud.

§8.— La Licencia de Aladdin

Un palmario ejemplo de qué conflictos suscitan la GPL y, en general, el recetario liberista de la FSF lo tenemos en la Licencia Aladdin de Ghostscript. Ghostscript es un cúmulo de utilidades para la manipulación y conversión de ficheros de impresión en formatos .ps (post-script) y PDF así como para procesar imágenes.[184]

La más ampliamente usada de esas utilidades es *ps2pdf*, a la cual recurro yo desde hace decenios, ya que mis controladores de impresión del WordPerfect 5.1, capaces de producir ficheros de impresión .ps (PostScript), no generan, en cambio, documentos en .pdf. De ahí que necesite ese programa externo, GhostScript, para convertir mis ficheros .ps en documentos .pdf, desplegables en la Web.[185]

Autor de Ghostscript fue Peter Deutsch, quien inicialmente produjo ese programa para el GNU, emitiéndolo bajo la GPL. Posteriormente Deutsch fundó su propia empresa, Aladdin, emitiendo nuevas versiones de Ghostscript bajo el sistema de licencia dual (optativa): o bien la AGPL (pequeña variante de la GPL) o bien su propia licencia, *Aladdin Free Public Licence* (Licencia pública libre de Aladdin).[186] El meollo de esa Licencia está en estos párrafos:

> Por la presente, siempre que cumpla Ud. con todas las condiciones estipuladas en esta Licencia, le otorga el licenciante los siguientes derechos (…): (a) Copiar y distribuir copias literales (…) del código fuente del Programa (…). (B) Modificar el Programa, crear obras basadas en el Programa y distribuir copias de las mismas en todo el mundo, en cualquier medio.
>
> 2. Condiciones. Queda sujeta esta licencia a las siguientes condiciones: (a) Prohíbese distribuir el Programa, o cualquier trabajo basado en el Programa, por parte de una firma mercantil a un tercero cuando tenga lugar algún pago en relación con dicha distribución (…) (b) No están sujetas a esta Licencia, cayendo fuera de su alcance, las actividades distintas de la copia, distribución y modificación del Programa. (…) (c) Debe cumplir con todas las siguientes condiciones con respecto a cualquier obra que distribuya o publique y que (…) se derive del Programa (…) (ii) Debe hacer que la Obra tenga licencia en su conjunto y sin costo alguno para todos los terceros según los términos de esta Licencia (…)

---

184. La más reciente versión es la 10.04.0 del 18 de septiembre de 2024, que yo —tras haberme bajado la fuente— he compilado hace unos días en mi computador.

185. No así para imprimir, puesto que mis impresoras actuales admiten el formato .ps.

186. En 2002 Deutsch dejó la programación, para consagrarse a la música, vendiendo el GhostScript a Artifex Software Inc., con sede social en San Francisco, cuyo dueño es el surcoreano Jeong Hee Kim, quien ha abandonado ahora la licencia Aladdin, ofreciendo su *software* en dos modalidades: la una bajo la licencia AGPL y la otra, bajo un contrato mercantil.

> (iv) Debe acompañar la Obra con el código fuente completo correspondiente legible por máquina, entregado en un medio utilizado habitualmente para el intercambio de software. (…) (vi) No puede imponer más restricciones al ejercicio, por parte del destinatario, de los derechos otorgados en este documento.

Resulta obvia la similitud de espíritu entre esta Licencia Libre de Aladdin y las licencias aprobadas por la FSF, salvo en un punto preciso: prohíbe la Licencia Aladdin lucrarse con ese *software* a las empresas mercantiles. Permítelo, en cambio, a individuos y entidades colectivas sin fin de lucro. También autoriza una amplísima variedad de usos por firmas comerciales, siempre que no se persiga ninguna ganancia por la redistribución del programa.

¿Qué opción les queda a las firmas mercantiles deseosas de beneficiarse con tal redistribución? Entrar en contacto con Aladdin, suscribiendo un contrato idóneo.

Dio lugar esa minúscula restricción a litigios entre Aladdin y la FSF. No me ocuparé del recorrido de tales pleitos ni de su desenlace. Felizmente —ya superada la incertidumbre— podemos los linuxitas seguir usando libremente el Ghostscript.

Lo único que deseo señalar es cuán razonable resulta, a mi juicio, lo que estipulaba esa Licencia Aladdin. Era gratuita, era libre, nos otorgaba enormes libertades y únicamente prohibía a organizaciones profesionalmente afanosas de lucro conseguirlo vendiendo el producto de Aladdin cuando lo habían obtenido gratis. Sólo esas organizaciones, y para ese fin concreto, estaban obligadas a negociar con Aladdin un acuerdo comercial. Aladdin altruistamente nos regalaba su *software*, mas no se lo regalaba a empresas que lo fueran a vender; ésas habrían de comprarlo.

Los abogados de Aladdin alegaban cuán injusto resultaba que, habiendo hecho donación de su producto esa empresa, otras firmas se lucraran vendiéndolo, sin haber pagado nada por adquirirlo y sin que, por consiguiente, de esa venta se derivase ni la más mínima compensación para los creadores del producto, quienes habrían trabajado gratis a favor de esas otras firmas.

Pues bien, esa Licencia Aladdin vino considerara no-libre y también ese producto (el ramal de GhostScript bajo Licencia Aladdin) como ajeno al *software* de fuente abierta (*open source*), a pesar de que está clarísimo que se ofrecía gratis con su código fuente.

Según ya lo dije más arriba, personalmente prefiero yo la Licencia Aladdin, pareciéndome más idónea para el bien común, al propiciar una remuneración para los programadores cuando el adquirente sea una firma mercantil cuya línea de negocio consista en ofrecer, venalmente, un servicio cuyo objeto sea el *software* en cuestión.

§9.— Los derechos de los autores

He venido reflexionando, en lo que precede, sobre cuál Licencia sería más razonable y beneficiosa para el bien común —para el bien del público, de los usuarios, de la comunidad social, de las entidades desarrolladoras del *software*, sin olvidar a los programadores, cuyo mérito debería venir retribuido.

Pásase por alto, a menudo, el trabajo de los programadores. No se acuerdan de ellos quienes niegan la propiedad intelectual (reconociendo la propiedad material, mucho menos justificable).

Hará unos cinco lustros, pensaba yo que el Unix/Linux era producto de investigadores universitarios que, sin buscar otra remuneración que su salario, nos ofrecían, como gratuito servicio público, ese magnífico *software*. Ni siquiera entonces era así. Hoy mucho menos, según lo hemos estudiado unos capítulos más atrás.

Vienen producidos por grandes establecimientos colectivos tanto el núcleo o kernel cuanto las propias utilidades GNU —que forman el esqueleto del sistema operativo—. Otro tanto cabe afirmar de muchos otros programas que, sin constituir, en sentido estricto, parte del sistema, son anejos al mismo, viniendo ofrecidos al público, ora en repositorios de las distribuciones, ora en otros portales de *software* libre —como Github y SourceForge.

Cuando quien ofrece un binario ejecutable no es un individuo, sino una organización, hay que pensar que ésta únicamente produce lo que hayan hecho, gracias a su trabajo, ciertos individuos, ya sean miembros de la organización o trabajadores a sueldo o programadores que han contribuido a la obra de la organización, sea gratis, sea a cambio de algún emolumento.

Lamentablemente la actual legislación reguladora de la propiedad intelectual ha caído en el desatino de reconocer a las empresas la autoría de las obras intelectuales producidas por sus empleados en el ejercicio de su función. Eso es lamentable, demostrando que el legislador está a sueldo de los magnates del capital. ¡Tanto individualismo para, a la postre, ningunear y eclipsar la obra de los individuos, cuyo mérito viene desleído y olvidado!

Por lo tanto, de los programadores que trabajan para los oligopolios como Micro$oft podemos olvidarnos; permanecen en la sombra, siendo piezas fungibles, anónimas, ignotas de un engranaje donde manda el dinero.

¿Podemos esperar que no sea así en el *software* libre? Lamentablemente la situación, sin ser igual de mala, dista de ser correcta.

Seguramente cuando echó a andar el proyecto GNU, los autores de los programas eran, todos ellos, miembros del mismo equipo académico, aunque pronto se les adjuntaron, sin lugar a dudas, colaboradores benévolos.

Con el transcurso de los años y decenios, mucho ha ido evolucionando la relación entre los colaboradores externos y los miembros de los equipos que pilotan Linux. Esos equipos son, actualmente: el equipo GNU y su proyección

jurídica, la FSF (fundación para el *software* libre); el equipo que, bajo los auspicios de la Fundación Linux, encabeza Linus Torvalds, desarrollando el núcleo o kernel del sistema operativo; los equipos al frente de las múltiples distribuciones (y aun de las subdistribuciones —así, dentro de la constelación Ubuntu, tenemos los equipos del Kubuntu, del Lubuntu etc); posiblemente alguno más, que produce cúmulos de utilidades que luego vienen incorporadas a unas u otras distribuciones.

Dado que la mayoría de las distribuciones corren a cargo de entidades sin ánimo de lucro y dada la gratuidad del *software* producido, resulta difícil remunerar el trabajo de los programadores integrantes de esos equipos, aunque verosímilmente reciben una buena retribución quienes programan dentro de varios de esos equipos, como los empleados de la Fundación Linux, los de la empresa RedHat, los de Canonical y otros generosamente subvencionados. No obstante (sin que yo tenga modo alguno de calcular su número), son colaboradores benévolos muchos autores de programas, o de partes significativas de esos programas.

Actualmente el contrato de colaboración que ha acabado prevaleciendo es el llamado «Acuerdo de Licencia Fiduciaria 2.0». Surgió como obra de la «Free Software Foundation Europe», al caer en la cuenta de que, hasta entonces, los conceptores de tales acuerdos habían ignorado los derechos morales de autor, que constituyen un componente esencial del derecho de propiedad intelectual según la legislación internacional (convenio de Berna) y según las legislaciones nacionales de los Estados europeos.

Las tres principales cláusulas de ese acuerdo fiduciario son:

1ª. Licencia de derechos de autor para la empresa o equipo que gestiona el *software*: «Con sujeción a los términos y condiciones de este Acuerdo, por la presente otorga [el colaborador] [a dicha empresa o entidad] una licencia mundial, libre de regalías, exclusiva, perpetua e irrevocable [...] con derecho a transferir un número ilimitado de licencias no exclusivas o a otorgar licencias a terceros, bajo los Derechos de Autor que cubren la colaboración, para utilizarla por todos los medios, incluidos, entre otros: publicarla, modificarla, preparar trabajos derivados de la misma y combinarla con otros materiales, reproducirla, en forma original o modificada, distribuirla, ponerla a disposición del público, mostrarla y ejecutarla públicamente en forma original o modificada».

2ª. Derechos morales: «Los derechos morales no se ven afectados en la medida en que están reconocidos y no renuncian a ellos por la ley aplicable. No obstante, puede Ud. agregar su nombre al mecanismo de atribución utilizado habitualmente en los materiales a los que colabora como el encabezado de los ficheros de código fuente de su colaboración; respetaremos esta atribución cuando la utilicemos».

3ª. «Licencia de derechos de autor. Tras dicha concesión de derechos a Nosotros [a la empresa o entidad], le otorgamos inmediatamente una

licencia mundial, libre de regalías, no exclusiva, perpetua e irrevocable, con derecho a transferir un número ilimitado de licencias no exclusivas o a otorgar sublicencias a terceros, bajo el *Copyright* que cubre la colaboración para utilizarla por todos los medios».

Detrás de toda esa enrevesada jerigonza, lo que resulta es lo siguiente:

1º.— El autor de la colaboración conserva el derecho de usarla como quiera siempre que no otorgue a nadie una licencia exclusiva. Eso significa, claro, que, en realidad, pierde cualquier posibilidad real de ganar dinero gracias a su creación (ya que nadie le va a comprar una licencia no exclusiva).

2º.— La empresa o entidad adquirente de los derechos patrimoniales de autor será la dueña del *software* a todos los efectos —con las dos únicas excepciones de los derechos morales y de que el autor de la colaboración conserva el derecho indicado en el punto nº1.

3º.— El ejercicio de los derechos morales de autor por aquel del cual es obra intelectual la colaboración queda circunscrito a escribir su propio nombre en el código fuente, garantizándole la entidad o empresa que no lo borrará (no lo borrará de ese código fuente); o sea, el nombre del autor no se dará a conocer a los usuarios del *software*, salvo si alguno de ellos se decide a leer todo el código fuente, en cuyas honduras irá hallando, en algún renglón, los nombres de los autores de las colaboraciones acumuladas en el *software* final.

Los derechos morales de autor son legalmente tres: el de reclamar el debido reconocimiento de su autoría; el de oponerse a que su obra sea modificada o alterada; el de arrepentimiento. (Además, son imprescriptibles e inalienables, al derivarse de uno de los derechos fundamentales del hombre, el de creación intelectual.)

A pesar del tenor literal de este Acuerdo dizque fiduciario, a mi juicio conculca varios de esos derechos.

Comprendo que sería problemático querer hacer respetar el derecho al arrepentimiento o la retractación. En la ley de propiedad intelectual, ese ejercicio está condicionado a, en caso necesario, indemnizar a terceros perjudicados. Tratándose de una contribución individual a un *software* colectivo, serían incalculables el monto del perjuicio infligido y del daño causado; ni se vislumbra cuál sería el remedio; dado lo cual, ese derecho, por mucho que lo proclame la ley, resulta en este caso inaplicable. (De hecho la propia ley tiene previstas cláusulas atinentes a obras colectivas; no entraré aquí en esos detalles jurídicos.)

Sin embargo, juzgo que este «Acuerdo fiduciario», tras su aparente equidad, es leonino y violatorio de los derechos individuales de los colaboradores.

Reconozco, empero, la enorme dificultad de compaginar la GPL con otro

tipo de contrato menos lesivo para los autores.

Entiendo que, reemplazando la GPL por una Licencia parecida a la Aladdin, podría ofrecerse una solución más adecuada para los colaboradores, que respetara mejor sus derechos, ofreciéndoles:

1º.— Cualquier *software* derivado de su colaboración o que la incorpore —modificada o sin modificar— se entregará siempre al público junto con una hoja, en formato texto, cuyo único contenido será una lista de los colaboradores (enumerados, ora por orden alfabético, ora por la estimada importancia de su respectiva colaboración o por cualquier otro criterio) —con sus nombres y apellidos; tal lista se desplegará, ostensiblemente, en cualquier repositorio o pág[a] web donde sea accesible dicho *software*.

2º.— En el caso de que el *software* en cuestión genere alguna ganancia o se utilice mercantilmente con fin de lucro (sea directa o indirectamente, v.g. mediante un servicio de soporte a los usuarios, cual es el caso de Suse, Ubuntu y RedHat), ello redundará en una retribución al trabajo de los colaboradores, la cual habrá de negociarse con ellos (o con un sindicato que represente sus intereses).

A cambio de ello, podría exigirse a los colaboradores suscribir un contrato de exclusiva con la empresa o entidad encargada de incorporar su colaboración a un proyecto más amplio de *software* —aunque tan sólo cuando estuviera suficientemente justificada esa cesión, en virtud de su estricta necesidad para que prosperase dicho *software*.

Capítulo XXIII

¿Triunfo de Linux?

¿Cómo puede hablarse de victoria de Linux cuando tal sistema operativo viene usado únicamente por un exiguo porcentaje de usuarios? A lo sumo corre en un 4% de los computadores personales (de sobremesa o portátiles). Sigue siendo, en cambio, Windows el sistema operativo usado por un 72% de los usuarios; y la mayoría del restante 28% usa el Mac.[187]

Siendo todo eso verdad, hay que mirar la tendencia. Es creciente el número de usuarios de Linux según las estadísticas disponibles. Una de las fuentes que manejo es la que ofrece la firma Statcounter, basándose en detección de navegadores por los motores de busca en la red. (Un método, sin duda, propenso a falsos positivos y falsos negativos, mas no por ello totalmente descartable.)

Atribuye Statcounter al Linux un porcentaje de usuarios ligeramente por encima del 4% (en 2024), mientras que hace tres años era del 1'5%.

Datos adicionales y más detallados ofrécelos Bryan Lunduke,[188] quien recopila informaciones de cuatro fuentes: el ya citado Statcounter, Steam (plataforma de juegos), Statista y Gartner, las cuales operan con diferente método.

Lo más interesante es aquello de lo que nos enteramos gracias a Gartner, a saber: en el mundo el número de computadores personales (de sobremesa o portátiles) es de unos 1.399 millones.

De ser así, aplicando las apreciaciones estadísticas de las otras tres fuentes, tendríamos que el número de computadores personales con Linux oscilaría entre 56 y 62 millones. (Es sumamente dudoso que se trate del número de usuarios, ya que, verosímilmente, hay usuarios que poseen varias máquinas con Linux.)

Lo acaso más sorprendente es que, según la plataforma de juegos Steam, haya 2.745.600 linuxeros entre sus clientes (cifra modesta mas

---

187. Concretamente un 15% de los computadores personales corren con el sistema de Apple.

188. En su sitio lunduke.locals.com.

reveladora, pues daba la impresión de que Linux y juego resultaban incompatibles).

Aun tratándose de una evidencia observacional meramente anecdótica, he de mencionar que, cuando consulto en Amazon páginas de venta de aparatos electrónicos, leo las opiniones de los clientes, lo cual me ha hecho descubrir que hoy muchos compradores quieren instalar Linux.

Además, la presencia cultural de Linux es muy superior al porcentaje de usuarios. Y es que, entre los aficionados a la informática, son muchos los linuxitas, al paso que no tantos usuarios de Windows aspiran a considerarse informáticos *amateurs*; limítanse, en general, a usar el sistema que vino preinstalado en su máquina.

Hay que fijarse también en el uso del Linux en los servidores y grandes computadores no domésticos. Tomo unos datos facilitados en el artículo «Linux Statistics 2024 by Market Share» de Barry Elad.

Linux pilota el 96'3% de los más potentes servidores de red en el mundo, cuya cifra se eleva a un millón aproximadamente. Con Linux funcionan Twitter, Yahoo, Amazon y eBay. De los servidores de nube, usan Linux: Amazon, Alibaba Cloud, Google Cloud Platform y hasta el mismísimo Microsoft Azure. En conjunto, casi el 90% de los servicios de nube operan con Linux. Un 70% de los servidores de red operan con Linux.

Con Linux funcionan un 91'5% de los 500 máximos y más veloces supercomputadores del mundo y el 76'6% de la *web* (con servidores Nginx, Apache y Litespeed), mientras que con Windows sólo funciona un 4'5% de la *web*.

Con Linux —o, en algunos casos, con algún otro sistema Unix— funcionan servidores vitales: los que auxilian a los controladores aéreos (por eso no se cayó ningún avión el 19 de julio), a la NASA (y demás entidades similares de otros países), a las grandes obras públicas, al depósito y el manejo de los armamentos, a las centrales eléctricas (más aún a las nucleares), a los movimientos de tropas —en general a todo lo militar—, a la ingeniería, a la protección civil, al salvamento marítimo y a la guardia costera y portuaria.[189]

El prestigio del cual goza actualmente Linux confírmalo la reciente ley suiza EMBAG (Ley federal sobre el uso de medios electrónicos para la realización de tareas de gobierno), la cual prohíbe a los establecimientos públicos de la Confederación Helvética usar *software* que no sea de fuente

---

189. Funcionan, en cambio, con Windows: la Bolsa, los bancos —o un número de ellos—, las entidades gestoras de tarjetas de pago (Visa, MasterCard etc), las compañías de seguros, los servicios de atención al paciente en los sanatorios (no así los de control de aparatos en los quirófanos ni nada de esa gravedad), los mostradores para pasajeros en los aeropuertos. Todo eso se paralizó el 19 de julio —no habiendo afectado la parálisis a Rusia, al usarse allá únicamente el Linux, en una u otra de sus varias distribuciones, tras haber abandonado el país Micro$oft en 2022.

abierta; cuando los únicos sistemas operativos de fuente abierta son los Unix/Linux, una galaxia en la cual es aplastante el predominio del Linux.

Lamentablemente, al estudiar jurídicamente esa Ley salta a la vista lo sumamente limitado de su alcance, por dos motivos, siendo el primero que sólo afecta a organismos dependientes del poder federal, no de los cantones. El segundo y principal motivo es que, en el Derecho administrativo suizo, únicamente se rigen por el Derecho público los contratos entre la Administración y las empresas privadas cuando atañen al ejercicio de funciones de autoridad, no a la prestación de servicios públicos ni a cualesquiera obras —por mucho que la ejecución de éstas vaya a redundar, después, en el cumplimiento de tales servicios. Lo cual significa que se rigen por el Derecho privado casi todos los contratos entre la Administración federal helvética y las empresas privadas; mas la Ley EMBAG únicamente afecta al Derecho público.

Suiza es un país precioso, limpio, funcional y, diría yo, casi de ensueño, gozando de la única democracia del mundo y pudiendo enorgullecerse de su avanzadísima cultura, de su destacadísima investigación científica y de sus envidiables logros industriales. Mas es también un país donde, a ciertos efectos, mejor se practica el disimulo, contando, a veces, más la apariencia que la realidad.

Aun con esas limitaciones, son dos triunfos de Linux los hechos de que Suiza haya promulgado esa Ley y de que el Land germano de Schleswig-Holstein haya mandado, recientemente, que, en adelante, sus oficinas no usen Windows sino Linux —decisión tomada en vista de las elevadas exigencias de *hardware* del Windows-11, que forzarían a desechar la mayoría de los computadores de que disponen esos establecimientos.

Yendo más allá, el ministro de digitalización de ese Land, Herr Dirk Schrödter, ha aducido que la medida va en el sentido de la soberanía digital. Lo cual significa reconocer que el Linux da libertad, mientras que el Windows sojuzga. Ha recordado, además, que, a la hora de las economías presupuestarias, hace daño el gasto en licencias de *software* privativo, o sea Windows y las aplicaciones diseñadas para correrse únicamente en Windows.[190]

Llévame todo eso a la conclusión de que Linux ha ganado la batalla cultural, es respetado como el sistema más seguro, poderoso y eficiente. Paulatinamente, va penetrando incluso en ese reducto que parecía inexpugnable, el de los usuarios domésticos.

Frente a taimados y prepotentes enemigos, frente a la pantoplutocracia, no puede ser ni total ni definitiva la victoria. No podemos confiar en que vaya

---

190. Verdad es que en Alemania existe un doloroso precedente, el de la capital bávara que, habiendo abrazado Linux en 2003, retornó a la servidumbre de Micro$oft por presión de los medios de negocios y de la clase política —particularmente del alcalde, Dieter Reiter—, habiendo convenido a cambio, con Micro$oft que ésta fijara en Munich su sede alemana.

a continuar la actual tendencia, siendo posible una involución.[191]

Ya lo veremos. En todo caso, resultan clamorosos esos triunfos de Linux, teniendo en cuenta cómo se ha enfrentado a la supremacía de los oligopolios, al poder de las grandes fortunas, habiéndose ganado a pulso, por sus propios méritos, cuanto ha logrado, contra viento y marea, contra el boicot de muchos fabricantes de *hardware* (y de *software*, como Adobe), contra denigraciones e intentos de asfixia y —lo peor— contra el prejuicio de que uno tiene lo que ha pagado, siendo así forzosamente malo cuanto sea gratuito.

[191]. Máxime a la vista de la infiltración que ya sufre nuestro sistema operativo, principalmente por obra y gracia del ínclito *systemd*, mas también por otras vías, según lo comenté al final del capítulo XXI.

Capítulo XXIV

Conclusiones: ¡Volver al Unix!

¿Qué conclusión extraigo de mis veintiséis años de usuario del Linux (como único o, cuando menos, como principalísimo sistema operativo de mis computadores —de sobremesa o portátiles)?

Ha constituido para mí el Linux una encantadora experiencia, que me ha producido enorme gozo, por un incentivador y estimulante aprendizaje, por el descubrimiento de mis propias capacidades de solventar problemas y dificultades, de inventar soluciones, de indagar fuentes de información atendibles y de hallar suministradores de valioso *software* (casi siempre totalmente gratuito; excepcionalmente de costo módico).

Gracias al Linux me he enriquecido y desarrollado mentalmente, me he formado, convirtiendo aquello que, inicialmente, semejaba ser una tarea, no ya de valor meramente instrumental, sino además ardua y engorrosa, en un medio de autoperfeccionamiento.

Son las delicias del Linux inseparables de las penalidades que hay que ir venciendo y superando, toda vez que aquello que más fruición acarrea es justamente hacer frente a los escollos, ensayar salidas, inventar.

No piense, empero, el lector que tales penas hayan sido tan hondas que me hayan dejado amargo recuerdo. En conjunto, han sido las experiencias agradables abrumadoramente mayoritarias; las pasajeras zozobras únicamente se han llevado una pequeña fracción del tiempo consagrado a la utilización del sistema operativo.

Las más veces (pero no siempre) he salido triunfante de esa lucha. A menudo me han ayudado las opiniones halladas en foros linuxitas (no faltando, así y todo, los malos consejos, cuyo seguimiento se ha revelado deletéreo —lo cual me hizo escarmentar para reflexionar y ponderar los riesgos antes de atenerme a cualesquiera instrucciones, por muy expertos que parecieran ser quienes las impartían).

A lo largo de mi vida, siempre he hallado, en mis muchos estudios, una fuente de regocijo (en general tanto mayor cuanto más esfuerzo me hubieran costado el aprendizaje y la solución de las dificultades). Cuando (con demora, relativamente a mis colegas de la institución en la cual prestaba mis servicios

académicos) abordé la utilización de aparatos electrónicos y, junto con ella, el aprendizaje del *software* necesario para su uso, distaba yo de sospechar que esa nueva ocupación pudiera transformarse en un genuino estudio, similar a mis otras actividades intelectuales; estudio que, igual que los otros, exigía ejercitar, con denuedo y empeño, la imaginación, la memoria y el razonamiento (deductivo, inductivo y abductivo).

Una labor que fui desarrollando desde mis primeros pinitos en el manejo de computadores —en aquel ya lejano año de 1989— y que seguí ahondando y extendiendo en mi etapa como usuario del DOS, hasta 1998, era la de ensayar aplicaciones hallando los trucos para usarlas con provecho

No se produjo ningún salto con la adopción del Linux hace más de cuatro lustros; sólo que si, de un lado, surgieron, con ese tránsito, nuevos y más difíciles desafíos, a la vez experimenté, cada vez más, un aprendizaje y una capacitación para afrontar dificultades (averiguando mis propias aptitudes intelectuales, forjando mi tenacidad y construyéndome personalmente).

De esa larga y fructífera experiencia formó parte la labor de compilación, que me vi forzado a cesar cuando la creciente complejidad del *hardware* y del *software* frustraron mis esfuerzos, impidiéndome continuar la reelaboración de los programas según mis personales preferencias. Compilar resultaba una tarea a veces fácil y a veces difícil, pero siempre inmensamente satisfactoria, con el resultado de que, una vez acabada la labor, tenía uno un *software* mucho más suyo, mucho más propio, fruto, en parte, del personal esfuerzo y de las personales preferencias.

A través de esa apropiación del *software* iba adquiriendo yo una sensación de dominio también del *hardware*, de poder sobre mi propia máquina, configurada a mi modo, por mí, de conformidad con mis gustos y mis necesidades.

Ha sido mi experiencia del Linux muy diversa de la de tantos otros linuxitas —la abrumadora mayoría de los influenciadores, opinantes y foreros— que llegaron a Linux unos cuantos años después (generalmente varios lustros más tarde), empezando por Ubuntu (al paso que con Slackware comencé yo para terminar con Ubuntu); muchos de ellos sólo han instalado en sus máquinas aplicaciones ofrecidas en los repositorios oficiales de sus respectivas distribuciones —y suelen proclamar que una de las ventajas del Linux es precisamente no buscar *software* fuera del que les ofrece su distribuidor acreditado—. Leyendo o visionando sus emisiones, ha sido infrecuente hallar algo equivalente a mi entusiasmo por triunfar sobre las dificultades, compilar, personalizar («costumbrizar») las aplicaciones e idear modos innovadores y personales de hacer las cosas.

Desde luego hay excepciones. Yo mismo habría fracasado en varios de mis esfuerzos si no hubiera podido aprovecharme —como efectivamente lo hice frecuentísimamente— de consideraciones, aclaraciones y explicaciones ofrecidas en los foros por linuxitas infinitamente más sabios que yo, muchísimo más

duchos en informática.

No resulta raro encontrar en los foros y canales linuxitas proclamas sobre la facilidad y la belleza del sistema operativo (belleza que, a veces, parece referirse a la del gráfico del escritorio), enseñándoles a los adictos del Windows que no va a resultarles penosa ni ardua la transición al Linux. Para satisfacer a esa clientela proveniente de Windows se han construido y adaptado cada vez más las nuevas versiones y distribuciones. (Ahora hace falta perforar la superficie para poder manejar el sistema operativo como lo manejábamos los del viejo Slackware, en consolas, en modo texto.)

No ha sido ésa la única influencia negativa del Windows en Linux. Otra es el creciente cercenamiento de nuestra libertad de usuarios, con las aplicaciones de seguridad, por nuestro bien, paternalísticas, que nos imponen los desarrolladores de las distribuciones hoy en boga.

Muchos linuxeros no quieren esa libertad. Con filial obediencia, acatan gustosos las imposiciones de los desarrolladores, confiando en su mejor criterio y no aspirando al autoperfeccionamiento ni al dominio que han sido para mí los dos encantos del Linux. (A mí, en cambio, me desagrada tan paternal protección.)

En seis motivos fúndase mi apreciación del Linux:

1º. Lo racional de su construcción, plasmado en cuatro rasgos:

(i) su estructura arbórea de directorios;[192]

(ii) su gama de atributos;

(iii) la articulación entre el kernel y las utilidades adicionales;

(iv) los *symbolic links*, o enlaces simbólicos.[193]

2º. El poderío de sus múltiples programas, brillantes obras de genuina ingeniería informática (comenzando por las utilidades del GNU).

3º. Lo multifacético y eficaz del CLI, con los *scripts* de la concha, el Bash, que facilitan al usuario la realización de una muchedumbre de tareas diversas.

---

192. Refiérome con eso, no al mero hecho de tener alguna estructura arbórea, sino justamente a la que tiene el Unix/Linux, donde obedecen a criterios racionales y claros la agrupación y el nivel jerárquico de los directorios, subdirectorios, subsubdirectorios etc. (Las pocas veces que entro en Windows resúltanme enigmáticas, arbitrarias e incomprensibles [¿por falta de costumbre?] la denominación y la ubicación de los directorios, en sus relación verticales y horizontales.)

193. Gracias a los enlaces simbólicos puedo, físicamente, tener todo un árbol de directorios en un dispositivo de almacenamiento externo sin dejar de actuar con y los ficheros de tal árbol como si estuvieran en mi propio directorio *home/lorenzo*. Lo cual conlleva ventajas: 1, para la custodia de directorios encriptados; 2, para los respaldos; y 3, para las eventuales reinstalaciones —cuando, por una u otra causa, hasta por alguna torpeza mía, resulten necesarias.

4º. La transparente perspicuidad de su funcionamiento (frente a lo arcano e inescrutable de Windows [y, en cierta medida, también de Android]).

5º. La posibilidad de ejecutar, gracias al Dosemu, mi aplicación preferida e inseparable compañera de trabajo: el WordPerfect 5.1.[194]

6º. La compatibilidad entre esa potencia del sistema —enriquecido con las aplicaciones adicionales— y un amplio margen de libertad del usuario en decidir qué programas ejecutar y, hasta cierto punto, cómo configurarlos.

* * *

Evidentemente ¡hay tantas cosas que estudiar y aprender! ¿Ha valido la pena haber consagrado tanto esfuerzo a lo que, a la postre, no deja de ser un mero instrumento, el manejo de unas máquinas a través del *software*? ¿No habría sido más provechoso haber dedicado ese tiempo a dominar el griego o la teoría de números o el álgebra universal o el cálculo lambda o el Derecho Romano o la música o el árabe o el chino o la «ciencia política», o …?

Posiblemente. Sólo que, de un lado, se me presentó esta tarea; se cruzó en mi vida. Podía haberla abordado más a la pata la llana, desde luego; sólo que no hubiera sido eso muy acorde con mi personalidad, con mi talante, con mi idiosincrasia. De otro lado, me repugnó la alternativa más comúnmente seguida, la del entorno gráfico del Windows. (Ignoro si tendrá tal rechazo algo que ver con una de mis características psíquicas.)

Sea como fuere, ha de quedar claro que de ningún modo estoy pregonando ni recomendando a nadie que siga mis pasos. Primero, porque creo en la libertad, convencido de que cada quien ha de trazar su propia senda en la vida, incluso en lo atinente al manejo de computadores. Segundo, porque sería quimérico que alguien, hoy, abordara lo atinente a la informática como yo tuve que abordarlo hace siete lustros; un abordaje que constituye el contexto de mi experiencia linuxita.

A quienes estén satisfechos con su Windows, deséoles que lo sigan disfrutando. Quienes hayan ingresado en el Linux pero por otra vía (toda esa masa de linuxitas que jamás han manejado Slackware ni compilado ni fuñado)

---

194. Leyendo algunas páginas del portal del Prof. Edward Mendelson, infiero que también resulta eso posible (no sin trabajo) en varios entornos Windows, hasta en el hodierno Windows11. No lo he probado. (Hace bastantes años intenté, sin conseguirlo, correr el WordPerfect 5.1 bajo el Windows de entonces —no recuerdo cuál era.)

El Profesor Edward Mendelson enseña Literatura inglesa en la la Universidad de Columbia en Nueva York. Es asimismo un informático, habiendo aportado valiosas contribuciones para facilitar la ejecución del WordPerfect (versiones 5.1 y 6) bajo Windows y bajo Linux. Lamentablemente no he logrado yo hacer funcionar el llamado «Dosemu2», pese a sus meticulosas instrucciones; salvo en ese punto, en los demás siempre he sacado provecho de sus benévolas aclaraciones en el portal http://mendelson.org.

Acaso seamos Edward Mendelson y un servidor los dos restantes entusiastas del WordPerfect, cual dos insumergibles náufragos en una marejada que se ha llevado a pique a tantísimos miles de usuarios avanzados del WordPerfect, quienes han sucumbido ante el arrebatador huracán de los entornos gráficos.

harán bien continuando por ese camino.

Los más inquietos, los más emprendedores, los más dispuestos a afrontar riesgos y dificultades para perfeccionarse y dominar mejor el *software* y el *hardware* ganarían —sea cual fuere la distribución que a la postre prefieran— ensayando aplicaciones ajenas a las ofrecidas por su distribuidor, tratando de compilar, personalizando los programas que usen, trabajando en el CLI y con aplicaciones de modo texto, prescindiendo lo más posible del escritorio y del modo gráfico.

Lo que se gana con esa opción es resucitar la experiencia del Unix. Consíguese así ser más que un simple linuxita: ser un unixita.[195]

Dejo para el final otro de los motivos de satisfacción. La gratuidad. He venido alabando la libertad del linuxita frente a la servidumbre del usuario de Windows. Libertad que comporta desafíos y tropiezos, cuya superación es justamente lo que más placer genera. Sólo que poco de eso tendríamos si hubiera que pagar. Porque resultaba gratis he podido ensayar muchas aplicaciones, saltar de una distribución a otra, explorar alternativas, reconfigurar los programas a mi modo. Si, habiéndome bajado un número significativo de distribuciones, las he ido barajando hasta decidir, en cada caso, cuál me resultaba más idónea, es porque no tenía que pagar nada. Si, para la realización de una tarea, he ido tanteando varias aplicaciones hasta fijarme en una que me resulte preferible, ha sido porque no vendían el derecho a realizar tales ensayos.

Muchísimas son las ventajas del Linux, una construcción intelectual de grandísimo valor y de enorme utilidad así como un sistema operativo gracias al cual puede el usuario gozar de un ancho margen de libertad. Sin la gratuidad escaso sería el disfrute de dicha libertad.

---

195. Es una lástima que el Slackware, el más Unix de todos los Línuxes, resulte tan difícil de compatibilizar con las modernas configuraciones del *firmware*, con sus nuevos ajustes de UEFI y sus nuevos métodos de particionamiento de disco. De no ser por tal obstáculo, de buena gana aconsejaría yo a los amantes del Linux una temporada de familiarización con el Slackware —aunque, a la postre, cada cual adoptara aquella distribución que más le gustara.

Podría comparar este consejo mío al que prodigaba el difunto Mario Bunge a quienes se inclinan a un filosofar descuidado y con peligro de caer en verborrea: pasar un tiempo sujetándose a la disciplina de la lógica matemática y esforzándose por encauzar a través de la misma la propia corriente de ideas; una experiencia temporal, tras la cual podrían buscar otros itinerarios que más les pluguieren.

Apéndice gráfico

Lámina N° 1

En la mitad superior:

Varios modelos de microcomputadores de finales de los setenta y comienzos de los ochenta.

En la mitad inferior:

Los primeros IBM-PCs y uno de los primeros clónicos (de Compaq) en el ángulo inferior derecho.

Lámina N° 2

En la mitad superior:

Un suelto periodístico de finales de los setenta anunciando el computador doméstico del Altair 8800.

En la mitad inferior:

Anuncio del Windows-98 por 392 dólares. Más otros modelos de los primeros «home computers» junto con un juego de cintas magnéticas (*cassettes)* como único dispositivo de almacenamiento entonces.

Lámina N° 3

Arriba izquierda: Anuncio de *Digital Research* sobre sus computadores domésticos con su propio sistema operativo, el CP/M.

A la derecha y debajo: reproducción de otros aparatos pioneros de la computación casera más una foto de un *minicomputer* de la NSA, 1971.

En la fila segunda por abajo a la derecha: Un anuncio del TRS-80 por 600$ (otro de los *home computers* de los últimos setenta).

Lámina N° 4

Continuamos viendo modelos de los últimos *home computers* y los primeros *personal computers* más un minicomputador de los años 60/70 (en el cual la interacción humana con la máquina se hace de pie).

Lámina N° 5

Mitad superior:

Árbol de subdistribuciones del Ubuntu —incluyendo ramificaciones que se han acabado desgajando.

Mitad inferior:

Precios de *software* Unix en 1984. Para una institución administrativa o académica, la copia cuesta 16.000 $, pero para una empresa el precio es de 43.000$ (siempre que se vaya a usar en una sola máquina).

Lámina N° 6

Arriba: árbol de directorios del Linux.

Centro: árbol de ramificaciones de la distribución Slackware (incluyendo derivadas que posteriormente han seguido su propio rumbo).

Abajo: Historia gráfica del Unix, con sus familias así como los nexos de derivación y de influencia mutua.

Lámina N° 7

Arriba:

Una amplia gama de logotipos de distribuciones de Linux

Abajo:

A la izquierda pantalladas de escritorios de varias distribuciones.

A la derecha fotocopia del famoso mensaje de email de Linus Torvalds en 1991-07-03 anunciando su proyectado *kernel*.

Lámina N° 8

Fotografías de: Don Fernando Corbató (en tiempo de su jubilación); el juvenil grupo de Palo Alto de la Xerox (años 70, con moda de la época); Ian Murdock (1973-2015 —creador de la distribución Debian); Ken Thomson (uno de los fundadores del Unix); cuatro fotos de Linus Torvalds.

Abajo: Patrick Volkerding (*the Man*, inventor y productor del Slackware) y Richard Stallman.

Lámina N° 9

Fotos de: Mark Shuttleworth (creador del Ubuntu), Jörg Schilling, Dennis Ritchie (creador del lenguaje de programación **C** y cofundador del Unix) y Lennart Poettering (el creador de *systemd*).

Mitad inferior: otra foto de Richard Stallman.


commodore
PET
personal computer

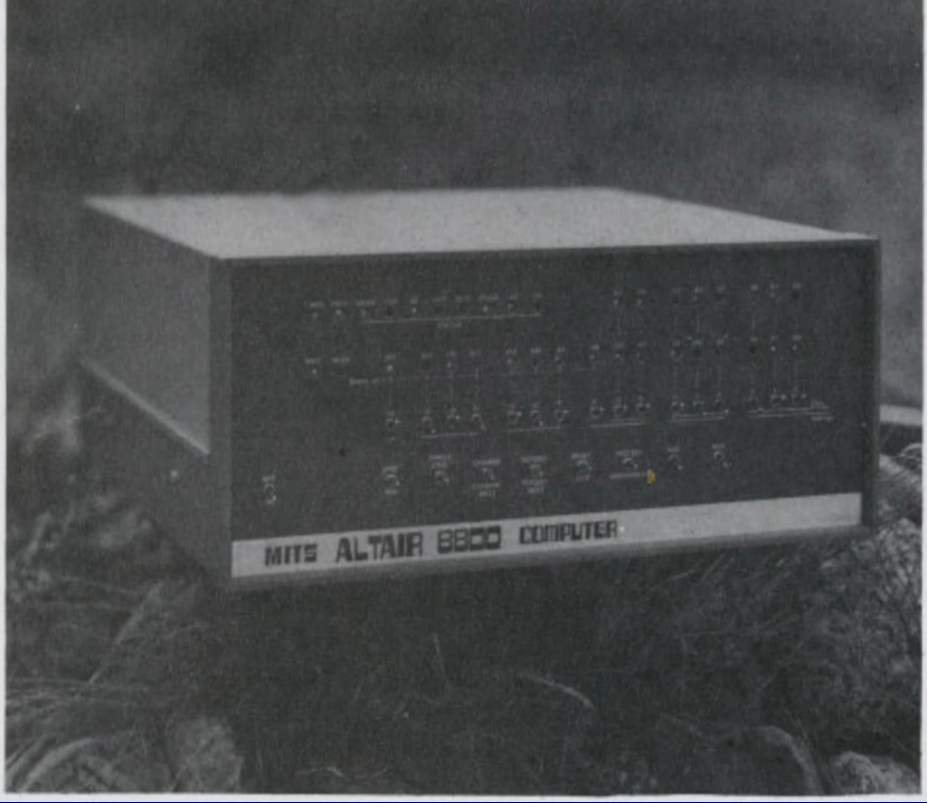

IBM

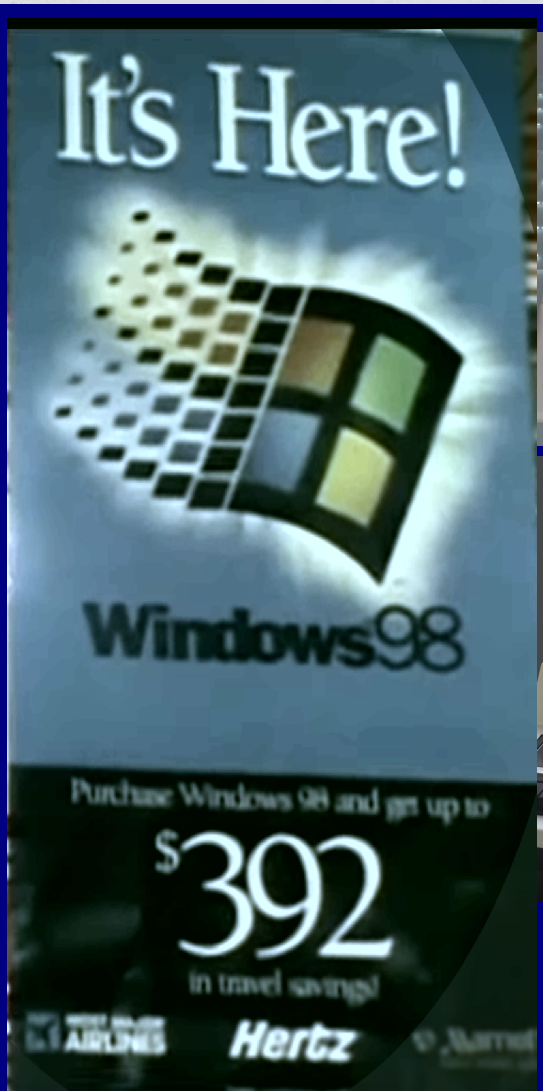
A COMPUTER CONCEPT BECOMES AN EXCITING REALITY.

Not too long ago, the thought of an honest, full-blown computer that sells for less than $500 would have been considered a mere pipe dream.

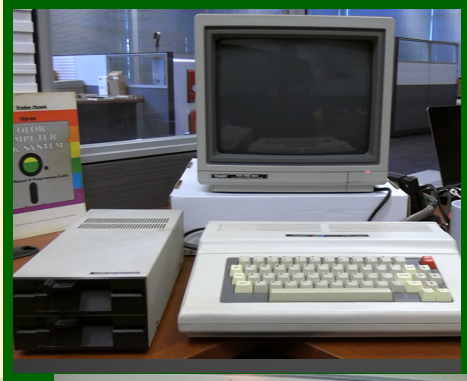
Everyone knows that computers are monstrous, box-shaped machines that sell for 10's and 100's of thousands of dollars.

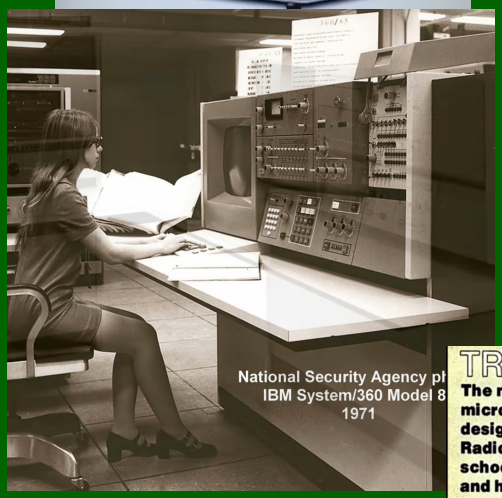
Pipe dream or not, *MITS*, the quality engineering oriented company that pioneered the calculator market, has made the *Altair 8800* a reality. It is the realization of that day when computers are accessible to almost anyone who wants one.






CP/M® GRAPHICS™
Your ticket to success.
Take the lead in microcomputer applications with powerful graphics software from Digital Research. CP/M and GSX are the keys to your graphic future. GSX is a logical extension of CP/M which many OEMs are adopting to standardise graphic device I/O. Computers with GSX allow your programs to take advantage of integrated graphic displays and peripherals like plotters, printers and CRT terminals. Together, CP/M and GSX deliver the same vital portability for your programs and data that has made CP/M the most accepted operating system in microcomputer history.
We also supply GSS-KERNEL™ a library of graphic commands for drawing lines, polygons, and text according to the emerging ISO standard: GKS (Graphical Kernel System). We also offer GSS-PLOT™ a library designed to let you create bar graphs, pie charts, histograms, and scatter plots. Both of these libraries can be linked with CBASIC™ Compiler, Pascal/MT+™, PL/I and FORTRAN on 8- and 16-bit systems. When you put it all together, the Digital Research graphics family is the most complete system you can buy for development and execution of graphic-oriented applications. Whether you're an application developer, OEM or user of micro-computers, call Digital Research for your ticket to graphic success.
Digital Research (UK) Limited, Oxford House, Oxford Street, Newbury, Berkshire, RG13 1JB. Telephone: 0635 35304.
DIGITAL RESEARCH
CP/M GRAPHICS
GSS-KERNEL
GSS-PLOT
UK Distributors
Tansys Limited
Windsor
Tel: 07535 56747
The Software Connection Limited
Fareham
Tel: 0329 230025
Xitan Systems Limited
Southampton
Tel: 0703 334711

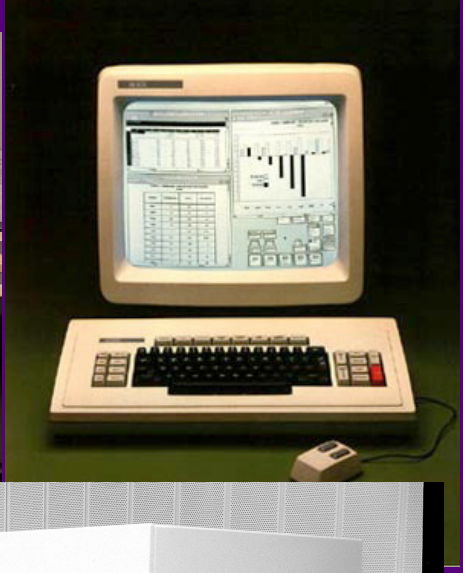


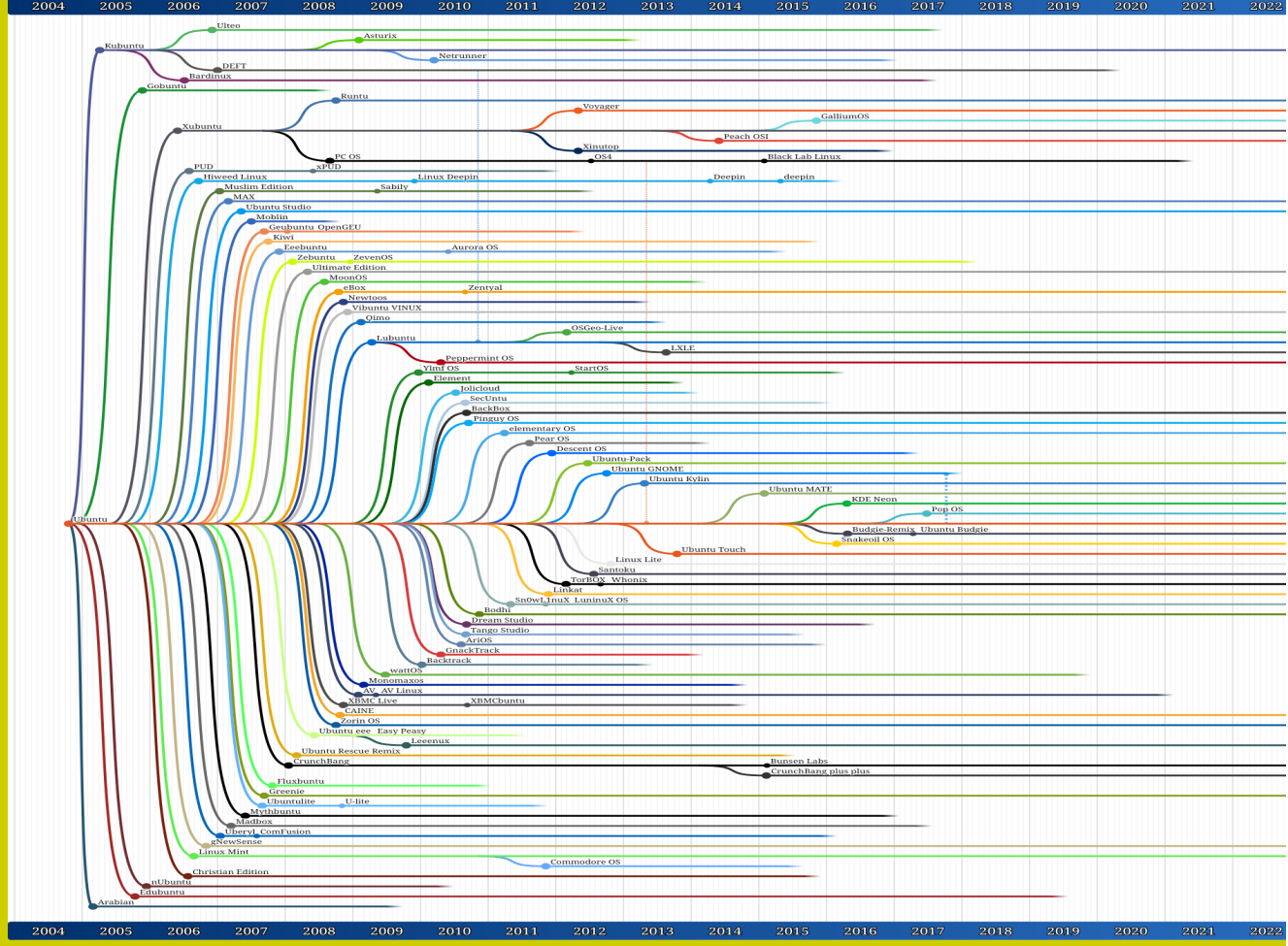

National Security Agency ph
IBM System/360 Model 8
1971


TRS-80
The new, low-cost micro computer system designed and built by Radio Shack® for school, home, office and hobbyist!
• Wired and tested
• NOT a kit
• U.L. listed
• E-x-p-a-n-d-a-b-l-e
Leadership in electronics: It's a tradition at Radio Shack, and the TRS-80 shows we're keeping it that way. Others talk of "home computers of the future." Ours is ready now, complete and fully assembled! You just plug it in and begin exploring its amazing power. No previous knowledge required!
Almost limitless capabilities. Great for family use, hobbyists, educators, engineers, businessmen. Learn to program in easy BASIC or use our applications programs. For small business, scientific computations, teaching functions, even home bookkeeping and innumerable games—it's all possible on this computer. Check it out—you'll agree that The Micro Computer Age has finally arrived with the powerful, affordable TRS-80.
TRS-80 Price List
Complete Computer (less display) $399.95
12" Video Display 199.95
Realistic® CTR-41 Cassette Recorder 49.95
Separate Components Price $649.85
Complete System Price 599.95
See next page for TRS-80 computer details and specifications.


VIXEN


disk II
disk II
Apple II



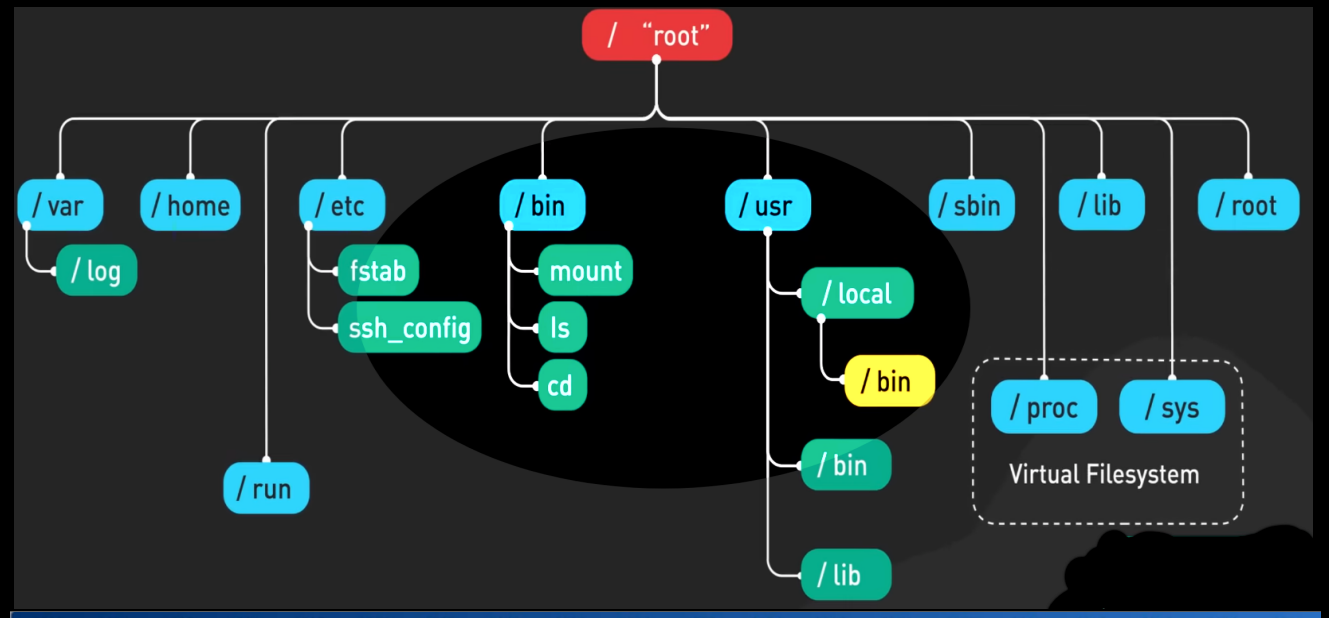


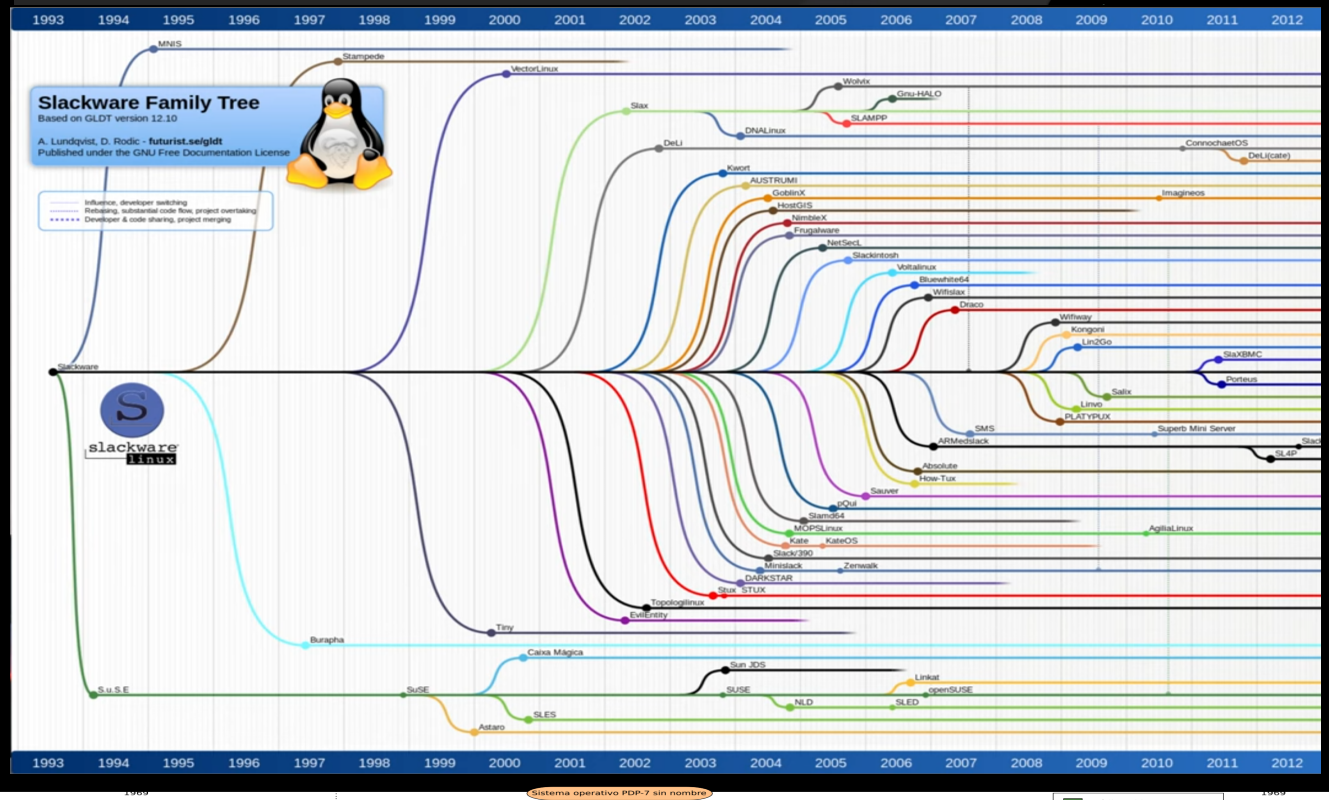
2/24/84

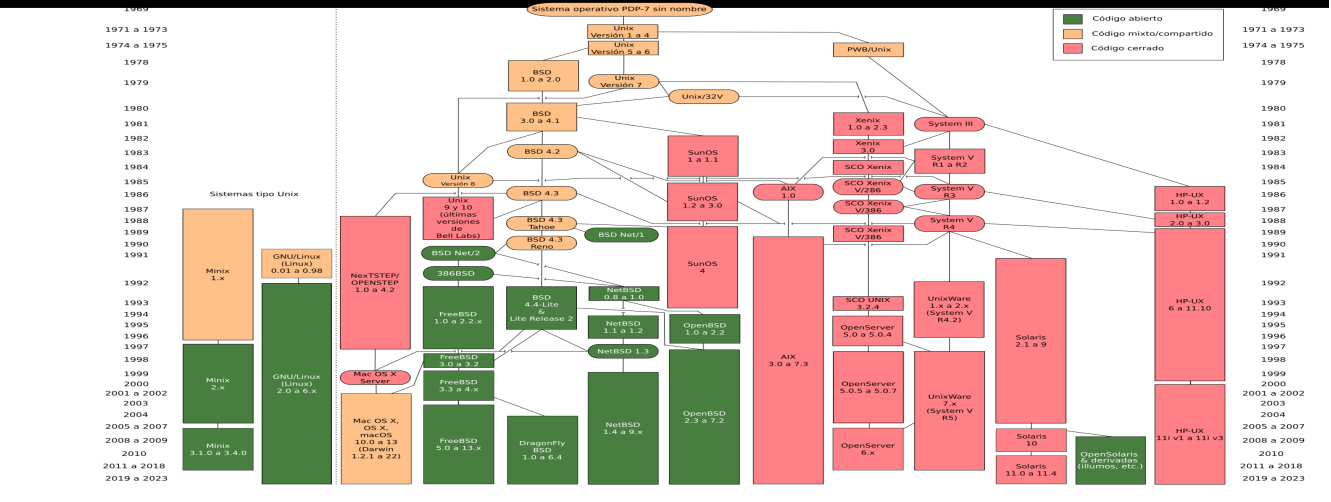
AT&T INTERNATIONAL

UNIX* SYSTEM V AND ADD ON APPLICATIONS PRICES

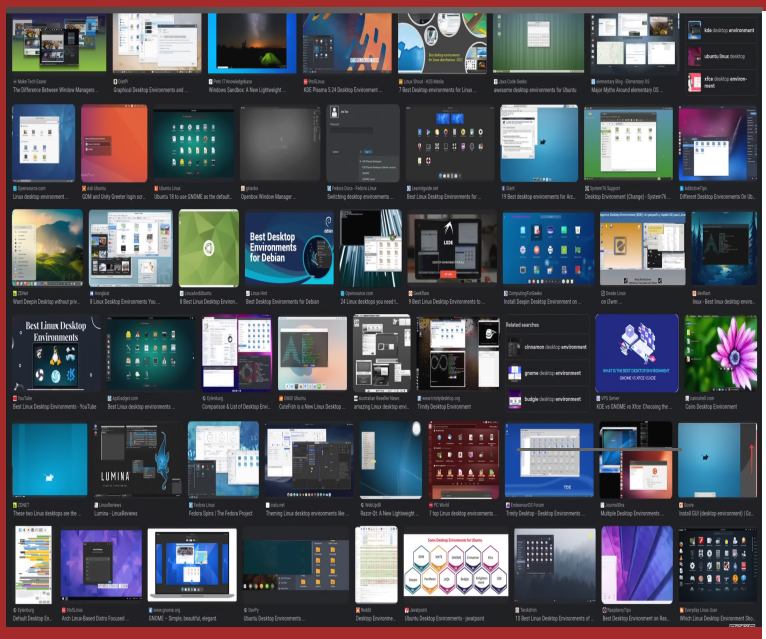
Item	Commercial	Educational
UNIX System V, Release 2.0, Source Code (1)	$43,000.00	$800.00
Each Additional CPU	$16,000.00	$400.00
Customer Provisions Supplemental Agreement	$25,000.00	-
Multiple Source Licenses	1-32 Users $1000 1-64 Users $3500 64+ Users $7000	-
UNIX System V, Release 2.0 Administrative-Educational	-	$16,000.00
Upgrade Fees from UNIX System V, Release 1.0 (1) (2)	$2,500.00 All CPU's	$800.00 All CPU's
Add on packages (source code only.) BASIC Interpreter. First CPU.	$5,000.00	$2,500.00
BASIC Interpreter Additional CPU's	$2,500.00	$1,250.00
Customer Provisions Fee	$10,000.00	-


/ "root"
/ var
/ log
/ home
/ etc
fstab
ssh_config
/ run
/ bin
mount
ls
cd
/ usr
/ local
/ bin
/ bin
/ lib
/ sbin
/ lib
/ root
/ proc
/ sys
Virtual Filesystem


1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012
Slackware Family Tree
Based on GLDT version 12.10
A. Lundqvist, D. Rodic - futurist.se/gldt
Published under the GNU Free Documentation License
Influence, developer switching
Rebasing, substantial code flow, project overtaking
Developer & code sharing, project merging
MNIS
Stampede
VectorLinux
Slax
Wolvix
Gnu-HALO
SLAMPP
DNALinux
DeLi
ConnochaetOS
DeLi(cate)
Kwort
AUSTRUMI
GoblinX
HoseGIS
NimbleX
Frugalware
NetSecL
Slackintosh
Voltalinux
Bluewhite64
Wifislax
Draco
Imagineos
Wifiway
Kongoni
Lin2Go
SlaXBMC
Porteus
Salix
Linvo
PLATYPUX
SMS
ARMedslack
Superb Mini Server
Absolute
How-Tux
Sauver
pQui
Slamd64
MOPSLinux
Kate
KateOS
AgiliaLinux
Slack/390
Minislack
Zenwalk
DARKSTAR
Stux
STUX
Topologilinux
EvilEnrelity
Tiny
Burapha
Caixa Mágica
Sun JDS
Linkat
openSUSE
SUSE
NLD
SLED
SuSE
S.u.S.E
SLES
Astaro
slackware linux


Sistema operativo PDP-7 sin nombre
Unix Versión 1 a 4
Unix Versión 5 a 6
PWB/Unix
BSD 1.0 a 2.0
Unix Versión 7
Unix/32V
BSD 3.0 a 4.1
BSD 4.2
SunOS 1 a 1.1
Xenix 1.0 a 2.3
Xenix 3.0
System III
SCO Xenix
System V R1 a R2
Unix Versión 8
BSD 4.3
SunOS 1.2 a 3.0
AIX 1.0
SCO Xenix V/286
System V R3
Unix 9 y 10 (últimas versiones de Bell Labs)
BSD 4.3 Tahoe
BSD 4.3 Reno
BSD Net/1
SCO Xenix V/386
System V R4
HP-UX 1.0 a 1.2
HP-UX 2.0 a 3.0
BSD Net/2
386BSD
SunOS 4
NextSTEP/ OPENSTEP 1.0 a 4.2
NetBSD 0.8 a 1.0
BSD 4.4-Lite & Lite Release 2
FreeBSD 1.0 a 2.2.x
NetBSD 1.1 a 1.2
OpenBSD 1.0 a 2.2
NetBSD 1.3
FreeBSD 3.0 a 3.2
FreeBSD 3.3 a 4.x
Mac OS X Server
Mac OS X, OS X, macOS 10.0 a 13 (Darwin 1.2.1 a 22)
FreeBSD 5.0 a 13.x
DragonFly BSD 1.0 a 6.4
NetBSD 1.4 a 9.x
OpenBSD 2.3 a 7.2
AIX 3.0 a 7.3
SCO UNIX 3.2.4
OpenServer 5.0 a 5.0.4
OpenServer 5.0.5 a 5.0.7
OpenServer 6.x
UnixWare 1.x a 2.x (System V R4.2)
UnixWare 7.x (System V R5)
Solaris 2.1 a 9
Solaris 10
Solaris 11.0 a 11.4
OpenSolaris (y derivadas illumos, etc.)
HP-UX 6 a 11.10
HP-UX 11i v1 a 11i v3
Sistemas tipo Unix
Minix 1.x
Minix 2.x
Minix 3.1.0 a 3.4.0
GNU/Linux (Linux) 0.01 a 0.98
GNU/Linux (Linux) 2.0 a 6.x
Código abierto
Código mixto/compartido
Código cerrado
1969
1971 a 1973
1974 a 1975
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001 a 2002
2003
2004
2005 a 2007
2008 a 2009
2010
2011 a 2018
2019 a 2023

Linux Distributions



From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: Gcc-1.40 and a posix-question
Message-ID:
Date: 3 Jul 91 10:00:50 GMT

Hello netlanders,

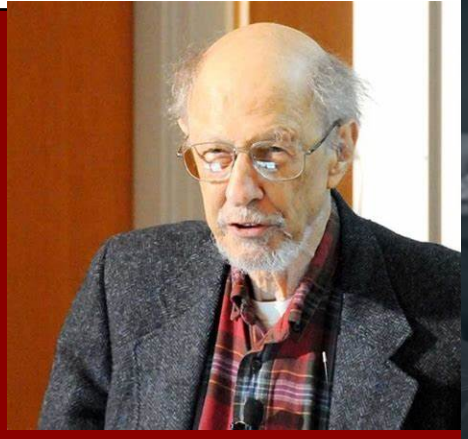
Due to a project I'm working on (in minix), I'm interested in the posix standard definition. Could somebody please point me to a (preferably) machine-readable format of the latest posix rules? Ftp-sites would be nice.

The project was obviously linux, so by July 3rd I had started to think about actual user-level things: some of the device drivers were ready, and the harddisk actually worked. Not too much else.

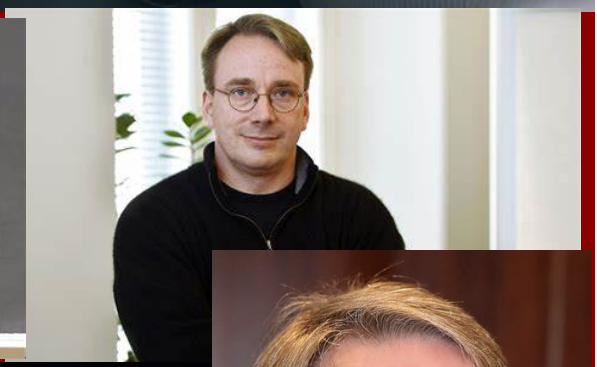
As an aside for all using gcc on minix - [deleted]

Just a success-report on porting gcc-1.40 to minix using the 1.37 version made by Alan W Black & co.

Linus Torvalds torvalds@kruuna.helsinki.fi




PARC




LINUX
Come visit us @
www.cdrom.com/
What is
SLACKWARE
LINUX?
LINUX
SLACKWARE