Programari lliure

Jesús M. González Barahona
Joaquín Seoane Pascual
Gregorio Robles

P07/M2001/02709

UOC
www.uoc.edu

Índex

1. Introducció

> "If you have an apple and I have an apple and we exchange apples, then you and I will still each have one apple. But if you have an idea and I have an idea and we exchange these idees, then each of us will have two idees."
>
> "Si tu tens una poma i jo tinc una poma i les intercanviem, seguirem tenint una poma cada un. Però si tu tens una idea i jo tinc una idea i les intercanviem, cada un de nosaltres tindrà dues idees. "
>
> Atribuït a Bernard Shaw

Què és el programari lliure? Què és i quines implicacions té la llicència d'un programa lliure? Com s'està desenvolupant el programari lliure? Com es financen els projectes de programari lliure i quins models de negoci que s'hi relacionen s'estan experimentant? Què motiva els desenvolupadors, especialment els que són voluntaris, a involucrar-se en projectes de programari lliure? Com són aquests desenvolupadors? Com es coordinen en els seus projectes i com és el programari que produeixen? En resum, quin és el panorama general del programari lliure? Aquest és el tipus de preguntes que tractarem de respondre en aquest text. Perquè encara que el programari lliure és cada vegada més present en els mitjans de comunicació i en les converses dels professionals de la informàtica, i encara que fins i tot comença a estar en boca dels ciutadans en general, encara és un desconegut per a molts. I els que el coneixen moltes vegades no en saben més que d'alguns dels seus aspectes, i en desconeixen completament d'altres.

Per començar, en aquest capítol presentarem els aspectes específics del programari lliure, centrant-nos fonamentalment a explicar les seves bases per als que s'aproximin al tema per primera vegada i a motivar la seva importància. Entre aquestes bases ens detindrem en la definició del terme (per saber de què parlarem) i en les conseqüències principals de l'ús (i de la mera existència) del programari lliure.

1.1. El concepte de *llibertat* en el programari

Programari privatiu

En aquest text utilitzarem el terme *programari privatiu* per a referir-nos a qualsevol programa que no pugui considerar-se programari lliure, d'acord amb la definició que s'ofereix més endavant.

Des de principis dels anys setanta ens hem acostumat que qui comercialitza un programa pugui imposar (i imposi) les condicions sota les quals pot usar-se. Pot prohibir, per exemple, que sigui deixat a un tercer. Malgrat que el programari és l'element tecnològic més flexible i adaptable que tenim, pot imposar-se (i és comú imposar) la impossibilitat d'adaptar-lo a unes necessitats concretes, o de corregir-ne els errors, sense el permís explícit del productor, que normalment es reserva en exclusiva aquestes possibilitats. Però aquesta és només una de les possibilitats que ofereix la legislació actual: el *programari lliure*, al contrari, atorga les llibertats que el *programari privatiu* nega.

1.1.1. Definició

Així doncs, el terme *programari lliure* (o *programes lliures*), tal com va ser concebut per Richard Stallman en la seva definició (Free Software Foundation, "Free software definition" –http://www.gnu.org/philosophy/free-sw.html– [120]), fa referència a les llibertats que pot exercir qui el rep, concretament quatre:

1) Llibertat per a executar el programa en qualsevol lloc, amb qualsevol propòsit i per sempre.
2) Llibertat per a estudiar-lo i adaptar-lo a les nostres necessitats. Això exigeix l'accés al codi font.
3) Llibertat de redistribució, de manera que se'ns permeti col·laborar amb veïns i amics.
4) Llibertat per a millorar el programa i publicar-ne les millores. Això també exigeix el codi font.

El mecanisme que s'utilitza per garantir aquestes llibertats, d'acord amb la legalitat vigent, és la distribució mitjançant una llicència determinada, com veurem més endavant (capítol 3). En aquesta l'autor plasma el seu permís perquè el receptor del programa pugui exercir aquestes llibertats, i també les restriccions que pugui voler aplicar (com donar crèdit als autors originals en cas de redistribució). Perquè la llicència sigui considerada lliure, aquestes restriccions no poden anar en contra de les llibertats esmentades.

L'ambigüitat de *free*

El terme original en anglès per a *programes lliures* és *free software*. Tanmateix, *free*, a més de 'lliure', significa 'gratis', la qual cosa genera gran confusió. És per això que sovint en anglès s'agafen paraules manllevades espanyoles i es parla de *libre software*, en contraposició amb *gratis software*, igual com nosaltres prenem prestada la paraula *software*.

Així doncs, les definicions de *programari lliure* no fan cap referència al fet que pugui aconseguir-se gratuïtament: el programari lliure i el programari gratuït són coses ben diferents. Tanmateix, dit això, cal explicar també que a causa de la tercera llibertat, qualsevol pot redistribuir un programa sense demanar contraprestació econòmica ni permís, la qual cosa fa pràcticament impossible obtenir grans guanys simplement per la distribució de programari lliure: qualsevol que l'hagi obtingut pot redistribuir-lo al seu torn a preu més baix, o fins i tot gratis.

Nota

Malgrat que qualsevol pot comercialitzar un programa donat a qualsevol preu, i això fa que teòricament el preu de redistribució tendeixi cap al cost marginal de còpia, hi ha models de negoci basats precisament a vendre programari, perquè hi ha moltes circumstàncies en les quals el consumidor està disposat a pagar si rep certes contraprestacions, com per exemple una certa garantia, encara que sigui subjectiva, sobre el programari que adquireix o un valor afegit en forma de selecció, actualització i organització d'un conjunt de programes.

Des d'un punt de vista pràctic, hi ha diversos textos que defineixen més precisament quines condicions ha de complir una llicència per ser considerada de programari lliure. Entre ells, destaquen per la seva importància històrica la definició de *programari lliure* de la Free Software Foundation (http://www.gnu.org/philosophy/free-sw.html) [120], les directrius de Debian per a decidir si un programa és lliure (http://www.debian.org/social_contract.html#guidelines) [104] i la definició del terme *open source* per l'Open Source Initiative (http://www.opensource.org/docs/definition_plain.html) [215], molt similar a les anteriors.

Nota

Per exemple, les directrius de Debian entren en el detall de permetre que l'autor exigeixi que els codis font distribuïts no siguin modificats directament, sinó que els originals s'acompanyin de pedaços separats, i que els programes binaris es generin amb noms diferents de l'original. A més, exigeixen que les llicències no contaminin altres programes distribuïts en el mateix mitjà.

1.1.2. Termes relacionats

Equivalent de *programari lliure* és el terme *open source software* ('programes de font oberta'), promogut per Eric Raymond i l'Open Source Initiative. Filosòficament, el terme és molt diferent, ja que fa èmfasi en la disponibilitat de codi font, no en la llibertat, però la seva definició és pràcticament la mateixa que la de Debian ("The open source definition", 1998 –http://www.opensource.org/docs/definition_plain.html–) [183]. Aquest nom és més políticament asèptic i recalca un aspecte tècnic que pot donar lloc a avantatges tècnics, com ara millors models de desenvolupament i negoci, major seguretat, etc. Durament criticat per Richard Stallman ("Why *free programari* is better than *open source*") [204] i la Free Software Foundation (http://www.fsf.org) [27], ha trobat molt més ressò en la literatura comercial i en les estratègies de les empreses que d'una manera o una altra donen suport al model.

Altres termes relacionats d'alguna manera amb el programari lliure són els següents:

Freeware	Són programes gratuïts. Normalment es distribueixen només en binari, i es poden obtenir sense cost. De vegades s'aconsegueix també permís de redistribució, però d'altres no, de manera que llavors només es poden obtenir del lloc "oficial" mantingut a aquest efecte. És habitual que es facin servir per a promocionar altres programes (típicament amb funcionalitat més completa) o serveis. Exemples d'aquest tipus de programes són Skype, Google Earth o Microsoft Messenger.
Shareware	No és ni tan sols programari gratis, sinó un mètode de distribució, ja que els programes, generalment sense codis font, es poden copiar lliurement, però no utilitzar continuadament sense pagar-los. L'exigència de pagament pot estar incentivada per funcionalitat limitada, missatges molestos o una simple apel·lació a la moral de l'usuari. A més, les estipulacions legals de la llicència podrien utilitzar-se en contra de l'infractor.

Charityware, careware	Es tracta generalment de *shareware* el pagament del qual es demana per a una organització *caritativa* patrocinada. En molts casos, el pagament no s'exigeix, però se sol·licita una contribució voluntària. Algun programari lliure, com **Vim**, sol·licita contribucions voluntàries d'aquest tipus (Brian Molenaar, "What is the context of charityware? ") [173].
Domini públic	L'autor renuncia absolutament a tots els seus drets en favor del comú, la qual cosa ha de quedar explícitament declarat en el programa, ja que si no es diu res, el programa és propietari i no es pot fer res amb aquest. En aquest cas, i si a més es proporcionen els codis font, el programa és lliure.
Copyleft	Es tracta d'un cas particular de programari lliure la llicència del qual obliga que les modificacions que es distribueixin siguin també lliures.
Propietari, tancat, no lliure	Es tracta de termes usats per a denominar el programari que no és lliure ni de font oberta.

1.2. Motivacions

Com hem vist, hi ha dues grans famílies de motivacions per al desenvolupament de programari lliure que donen lloc, així mateix, als dos noms amb què se'ls coneix:

- La motivació ètica, abanderada per la Free Software Foundation (www.fsf.org) [27], hereva de la cultura *hacker* i partidària de l'apel·latiu *lliure*, que argumenta que el programari és coneixement que ha de poder difondre's sense traves i l'ocultació del qual és una actitud antisocial, i que afirma a més que la possibilitat de modificar programes és una forma de llibertat d'expressió. Pot aprofundir-se en aquest aspecte en els assajos de Stallman (*Free Software, free society. Selected essays of Richard M. Stallman*) [211] o en l'anàlisi de Pekka Himanen (*The hacker ethic and the spirit of the information age*. Random House, 2001) [144].

- La motivació pragmàtica, abanderada per l'Open Source Initiative (www.opensource.org) [54] i partidària de l'apel·latiu *font oberta*, que argumenta avantatges tècnics i econòmics que repassarem a l'apartat següent.

A part d'aquestes dues grans motivacions, la gent que treballa en programari lliure pot fer-ho per moltes altres raons, que van des de la diversió (Linus Torvalds i David Diamond, Texere, 2001) [217] fins a la mera retribució econòmica, possiblement deguda a *models de negoci* sostenibles. Al capítol 4 s'aprofundeix en aquestes motivacions a partir d'anàlisis objectives.

1.3. Conseqüències de la llibertat del programari

El programari lliure implica nombrosos avantatges i pocs desavantatges, molts dels quals han estat exagerats (o falsejats) per la competència propietària. D'aquests, el que té més fonament és l'econòmic, ja que com hem vist, no és possible obtenir molts diners de la distribució i aquesta la pot i sol fer algú diferent de l'autor. És per això que es necessiten models de negoci i altres me-

canismes de finançament que es desenvolupen al capítol 5. Altres desavantatges, com ara la falta de suport o la qualitat escassa, estan relacionats amb el finançament, però a més en molts casos són falsos, ja que fins i tot un programari sense cap tipus de finançament sol oferir molt bon suport gràcies a fòrums d'usuaris i desenvolupadors, i moltes vegades té gran qualitat.

Tenint presents les consideracions econòmiques, hem d'observar que el model de costos del programari lliure és molt diferent del model de costos del programari propietari, ja que gran part d'aquest s'ha desenvolupat fora de l'economia formal monetària, moltes vegades amb mecanismes de canvi: "jo et dono un programa que t'interessa i tu l'adaptes a la teva arquitectura i li fas millores que t'interessen". Al capítol 7 s'expliquen mecanismes d'enginyeria de programari apropiats per a aprofitar aquests recursos humans no pagats i amb característiques pròpies, mentre que al capítol 8 s'estudien les eines usades per a fer efectiva aquesta col·laboració. Però, a més, gran part dels costos disminueixen pel fet que és lliure, ja que els programes nous no han de començar des de zero, sinó que poden reutilitzar programari ja fet. La distribució té també un cost molt menor, ja que es fa per Internet i amb propaganda gratuïta en fòrums públics destinats a això.

Una altra conseqüència de les llibertats és la qualitat derivada de la col·laboració voluntària de gent que contribueix o que descobreix i notifica errors en entorns i situacions inimaginables pel desenvolupador original. A més, si un programa no ofereix la qualitat suficient, la competència pot prendre'l i millorar-lo partint del que hi ha. Així la *col·laboració* i la *competència*, dos poderosos mecanismes, es combinen per aconseguir una millor qualitat.

Examinem ara les conseqüències beneficioses segons el destinatari.

1.3.1. Per a l'usuari final

L'usuari final, ja sigui individual o empresa, pot trobar verdadera competència en un mercat amb tendència al monopoli. En particular, no depèn necessàriament del suport del fabricant del programari, ja que hi pot haver múltiples empreses, potser petites, que disposin del codi font i de coneixements i que puguin fer negoci mantenint determinats programes lliures.

Ja no es depèn tant de la *fiabilitat* del fabricant per a intentar deduir la qualitat d'un producte, sinó que la guia ens la donarà l'acceptació de la comunitat i la disponibilitat dels codis font. A més, ens podem oblidar de *caixes negres*, en les quals cal confiar "perquè sí", i de les estratègies dels fabricants, que poden decidir unilateralment deixar de mantenir un producte.

L'avaluació de productes abans de la seva adopció és ara molt més senzilla, ja que n'hi ha prou d'instal·lar els productes alternatius en el nostre entorn real i provar-los, mentre que per al programari propietari cal fiar-se d'informes externs o negociar proves amb els proveïdors, la qual cosa no sempre és possible.

Atesa la llibertat de modificar el programa per a ús propi, l'usuari pot personalitzar-lo o adaptar-lo a les seves necessitats, corregint errors si en tingués. El procés de correcció d'errors descoberts pels usuaris en programari propietari sol ser extremadament feixuc, si no impossible, ja que si aconseguim que es reparin, sovint això es farà en la versió següent, que pot trigar anys a sortir i que de vegades, a més, caldrà comprar de nou. En programari lliure, tanmateix, podem fer nosaltres la reparació esmentada, si estem qualificats, o contractar el servei fora. També podem, directament o contractant serveis, integrar el programa amb un altre o auditar la seva qualitat (per exemple la seguretat). El control passa, en gran manera, del proveïdor a l'usuari.

1.3.2. Per a l'Administració pública

L'Administració pública és un gran usuari de característiques particulars, ja que té obligacions especials amb el ciutadà, sigui proporcionant-li serveis accessibles, neutrals respecte als fabricants, o garantint la integritat, la utilitat, la privadesa i la seguretat de les seves dades a llarg termini. Tot això l'obliga a ser més respectuosa amb els estàndards que les empreses privades i a mantenir les dades en formats oberts i manipulats amb programari que no depengui d'estratègia d'empreses, generalment estrangeres, certificat com a assegurança per auditoria interna. L'adequació a estàndards és una característica notable del programari lliure no tan respectada pel programari propietari, generalment àvid de crear mercats captius.

Així mateix, l'Administració té una certa funció d'aparador i guia de la indústria que el fa tenir un gran impacte que s'hauria de dirigir a la creació d'un teixit tecnològic generador de riquesa nacional. L'esmentada riquesa pot crear-se mitjançant el foment d'empreses el negoci dels quals sigui, en part, el desenvolupament de nou programari lliure per a l'Administració o el manteniment, l'adaptació o l'auditoria del programari existent. Al capítol 6 ens estenem més en aquesta qüestió.

1.3.3. Per al desenvolupador

Per al desenvolupador i productor de programari, la llibertat canvia molt les regles del joc. Amb ella li és més fàcil competir essent petit i adquirir tecnologia punta. Pot aprofitar-se del treball dels altres, competint fins i tot amb un altre producte mitjançant la modificació del seu propi codi, si bé també el competidor copiat s'aprofitarà del nostre codi (si és *copyleft*). Si el projecte es porta bé, pot aconseguir-se la col·laboració gratuïta de molta gent, i a més, es té accés a un sistema de distribució pràcticament gratuït i global. No obs-

tant això, queda pendent el problema de com obtenir recursos econòmics si el programari realitzat no és fruit d'un encàrrec pagat. Al capítol 5 es tracta en detall aquest tema.

1.3.4. Per a l'integrador

Per a l'integrador, el programari lliure és el paradís. Significa que ja no hi ha més caixes negres per intentar encaixar, sovint amb enginyeria inversa. Pot llimar asprors i integrar trossos de programes per aconseguir el producte integrat necessari, en disposar d'un patrimoni ingent de programari lliure d'on extreure les peces.

1.3.5. Per a qui proporciona manteniment i serveis

Disposar del codi font ho canvia tot i ens situa gairebé en les mateixes condicions que el productor. Si no són els mateixos és perquè fa falta un coneixement profund del programa que només el desenvolupador posseeix, per la qual cosa és convenient que el mantenidor participi en els projectes que es dedica a mantenir. El valor afegit dels serveis és molt més apreciat, ja que el cost del programa és baix. Aquest és actualment el negoci més clar amb programari lliure i amb el qual és possible un major grau de competència.

1.4. Resum

Aquest primer capítol ha servit com a presa de contacte amb el món del programari lliure. El concepte, definit per Richard Stallman, es basa en quatre llibertats (llibertat d'execució, llibertat d'estudi, llibertat de redistribució i llibertat de millora), dues de les quals suposen l'accés al codi font. Aquesta accessibilitat i els seus avantatges motiven un altre punt de vista menys ètic i més pragmàtic, defensat per l'Open Source Initiative, que ha donat lloc a un altre terme: *programari de font oberta*. S'han comentat també altres termes relacionats per similitud o contraposició, i que permeten aclarir els conceptes. Finalment, s'ha parlat de les conseqüències de la llibertat del programari per als principals actors implicats.

2. Una mica d'història

"When I started working at the MIT Artificial Intelligence Lab in 1971, I became part of a software-sharing community that had existed for many years. Sharing of software was not limited to our particular community; it is as old as computers, just as sharing of recipes is as old as cooking. But we did it more than most. [...] We did not call our software *free software*, because that term did not yet exist; but that is what it was. Whenever people from another university or a company wanted to port and us a program, we gladly let them. If you saw someone using an unfamiliar and interesting program, you could always ask to see the source code, so that you could read it, change it, or cannibalize parts of it to make a new program."

"Quan vaig començar a treballar al Laboratori d'Intel·ligència Artificial del MIT el 1971, vaig passar a formar part d'una comunitat de programari compartit que existia des de feia molts anys. Compartir codi no era una cosa específica de la nostra comunitat: és una cosa tan antiga com els ordinadors, de la mateixa manera que compartir receptes és tan vell com cuinar. Però nosaltres ho fèiem més que la majoria. [...]No anomenàvem el nostre programari *programari lliure* perquè aquest terme encara no existia, però això és el que era. Quan algú d'una altra universitat o d'una empresa volia migrar i fer servir un programa, nosaltres li deixàvem fer-ho gustosament. Si veies algú utilitzant un programa estrany i interessant, sempre podies demanar-li veure'n el codi font, per a poder llegir-lo, canviar-lo o canibalitzar-ne parts per fer un programa nou. "

Richard Stallman, "The GNU Project" (publicat originalment al llibre *Open sources*) [208]

Encara que totes les històries relacionades amb la informàtica són forçosament breus, la del programari lliure és una de les més llargues entre aquestes. De fet, podria dir-se que als seus començaments, pràcticament tot el programari desenvolupat complia les definicions de *programari lliure*, encara que el concepte ni tan sols existia encara. Més tard la situació va canviar completament, i el programari privatiu va dominar l'escena, pràcticament en exclusiva, durant bastant de temps. Va ser durant aquesta època quan es van establir les bases del programari lliure com l'entenem avui dia, i quan, a poc a poc, van començar a aparèixer programes lliures. Amb el temps, aquests començaments s'han convertit en una tendència que ha anat creixent i madurant fins a arribar a la situació actual, en la qual el programari lliure és una possibilitat que cal considerar en gairebé tots els àmbits.

Aquesta història és bastant desconeguda, fins al punt que per a gran part dels professionals informàtics el programari privatiu és el programari "en el seu estat natural". Tanmateix, la situació és més aviat la contrària, i les llavors del canvi que es va començar a entreveure en la primera dècada del 2000 havien estat plantades ja a començaments dels anys vuitanta.

Bibliografia

No hi ha gaires històries exhaustives sobre el programari lliure, i les que hi ha són articles normalment limitats amb la finalitat del seu estudi. En qualsevol cas, el lector interessat pot completar el que s'ha exposat en aquest capítol consultant "Open Source Initiative. History of the OSI" [146] (http://www.opensource.org/docs/history.php), que posa l'èmfasi en l'impacte del programari lliure al món empresarial entre els anys 1998 i 1999; "A brief history of free/open source software movement" [190], de Chris Rasch, que cobreix la història del programari lliure fins a l'any 2000, o "The origins and future of open source software" (1999) [177], de Nathan Newman, que se centra en gran manera

en la promoció indirecta que el Govern dels EUA ha fet del programari lliure o sistemes similars durant les dècades de 1970 i 1980.

2.1. El programari lliure abans del programari lliure

El programari lliure com a concepte no va aparèixer fins a principis de la dècada de 1980. Tanmateix, la seva història pot traçar-se des de bastants anys abans.

2.1.1. I en el principi va ser lliure

Durant els anys seixanta el panorama de la informàtica estava dominat pels grans ordinadors, instal·lats fonamentalment en empreses i centres governamentals. IBM era el principal fabricant, amb gran diferència sobre els seus competidors. En aquesta època, quan s'adquiria un ordinador (el maquinari), el programari venia com un acompanyant. Mentre es pagués el contracte de manteniment, es tenia accés al catàleg de programari que oferia el fabricant. A més, no era comuna la idea que els programes fossin una cosa "separada" des d'un punt de vista comercial.

En aquesta època el programari es distribuïa habitualment juntament amb el seu codi font (en molts casos només com a codi font), i en general, sense restriccions pràctiques. Els grups d'usuaris com SHARE (usuaris de sistemes IBM) o DECUS (usuaris de DEC) participaven d'aquests intercanvis, i fins a cert punt els organitzaven. La secció "Algorithms" de la revista *Communications of the ACM* era un altre bon exemple de fòrum d'intercanvi. Podria dir-se que durant aquests primers anys de la informàtica el programari era lliure, almenys en el sentit que els que hi tenien accés podien disposar habitualment del codi font, estaven acostumats a compartir-lo, a modificar-lo i a compartir també aquestes modificacions.

El 30 de juny de 1969 IBM va anunciar que, a partir de 1970, vendria part del seu programari separadament (Burton Grad, 2002) [131]. Això va suposar que els seus clients ja no podien obtenir, inclòs en el preu del maquinari, els programes que necessitaven. El programari es va començar a percebre com alguna cosa amb valor intrínsec i, com a conseqüència, es va fer cada vegada més habitual restringir escrupolosament l'accés als programes i es van limitar, tant com es va poder (tècnicament i legalment), les possibilitats dels usuaris per a compartir, modificar o estudiar el programari. Dit d'una altra manera, es va passar a la situació que continua essent habitual al món del programari a començaments del segle XXI.

Bibliografia

El lector interessat en aquesta època de transició pot llegir, per exemple, "How the ICP Directory began" [226] (1998), on Larry Welke comenta com va néixer un dels primers catàlegs de programari no lligats a un fabricant, i com en aquest procés va descobrir que les empreses estaven disposades a pagar per programes no fets pel fabricant dels seus ordinadors.

A mitjan dècada de 1970 era ja absolutament habitual, en qualsevol àmbit informàtic, trobar-se amb programari privatiu. Això va suposar un gran canvi cultural entre els professionals que treballaven amb programari i va ser l'origen del floriment d'un gran nombre d'empreses dedicades al nou negoci. Faltava encara gairebé una dècada perquè comencés a aparèixer, de forma organitzada i com a reacció a aquesta situació, el que avui coneixem com a *programari lliure*.

2.1.2. Dècada dels setanta i principis de la dècada dels vuitanta

Fins i tot quan la tendència aclaparadorament majoritària era explorar el model de programari privatiu, hi havia iniciatives que mostraven algunes característiques del que després es consideraria programari lliure. De fet, alguna va arribar a produir programari lliure tal com el definim avui dia. Entre elles es pot destacar SPICE, TeX i Unix, el cas del qual és molt més complex.

SPICE (Simulation Program with Integrated Circuit Emphasis) és un programa desenvolupat a la Universitat de Califòrnia, a Berkeley, per a simular les característiques elèctriques d'un circuit integrat. Va ser desenvolupat i posat en el domini públic pel seu autor, Donald O. Pederson, el 1973. SPICE va ser originalment una eina docent, i com a tal es va estendre ràpidament a moltes universitats de tot el món. En elles va ser utilitzat per molts estudiants de la incipient disciplina, en aquell temps, del disseny de circuits integrats. Ja que estava en el domini públic, SPICE podia redistribuir-se, modificar-se, estudiar-se... Fins i tot es podia adaptar a unes necessitats concretes i vendre aquesta versió com a producte privatiu (la qual cosa han fet durant la seva història dotzenes de vegades una gran quantitat d'empreses). Amb aquestes característiques, SPICE tenia totes les paperetes per a convertir-se en l'estàndard de la indústria, amb les seves diferents versions. I efectivament, això va ser el que va passar. Probablement aquest va ser el primer programa amb característiques de programari lliure que durant un temps va copar un mercat, el dels simuladors de circuits integrats, i sens dubte va poder fer-ho precisament pel fet de tenir aquestes característiques (a més de les seves innegables qualitats tècniques).

Bibliografia

Es pot consultar més informació sobre la història de SPICE a "The life of SPICE", treball presentat el 1996, Bipolar Circuits and Technology Meeting Minneapolis, MN, EUA, el setembre de 1996 [175].

La pàgina web de SPICE: http://bwrc.eecs.berkeley.edu/Classes/IcBook/SPICE/.

Donald Knuth va començar a desenvolupar TeX durant un any sabàtic, el 1978. TeX és un sistema de tipografia electrònica molt utilitzat per a la producció de documents de qualitat. Des del principi, Knuth va utilitzar una llicència que avui seria considerada de programari lliure. Quan el sistema es va considerar raonablement estable, el 1985, va mantenir aquesta llicència. En aquella època, TeX era un dels sistemes més grans i més coneguts que es podia considerar programari lliure.

Bibliografia

Algunes fites de la història de TeX poden consultar-se en línia a http://www.math.utah.edu/software/plot79/tex/history.html [39]. Per a saber més detalls, també és útil l'article corresponent de Wikipedia, http://www.wikipedia.org/wiki/TeX [233].

2.1.3. Desenvolupament primerenc de Unix

Unix, un dels primers sistemes operatius portables, va ser creat originalment per Thompson i Ritchie (entre d'altres) en els Bell Labs d'AT&T. El seu desenvolupament ha continuat des del seu naixement, cap a 1972, i ha donat lloc a innombrables variants comercialitzades per (literalment) desenes d'empreses.

Durant els anys 1973 i 1974, Unix va arribar a moltes universitats i centres d'investigació de tot el món, amb una llicència que en permetia l'ús per a finalitats acadèmiques. Encara que hi havia certes restriccions que impedien la seva lliure distribució, entre les organitzacions que disposaven de la llicència el funcionament va ser molt similar al que es va veure més tard en moltes comunitats de programari lliure. Els que tenien accés al codi font de Unix van tenir un sistema que podien estudiar, millorar i ampliar. Al voltant d'aquest va aparèixer una comunitat de desenvolupadors que aviat va començar a girar entorn del CSRG de la Universitat de Califòrnia, a Berkeley. Aquesta comunitat va desenvolupar la seva pròpia cultura, que com veurem més endavant, va ser molt important en la història del programari lliure. Unix va ser, fins a cert punt, un assaig primerenc del que es va veure amb GNU i Linux diversos anys més tard. Estava confinat a una comunitat molt més petita i era necessària la llicència d'AT&T, però en altres aspectes el seu desenvolupament va ser similar (en un món molt menys comunicat).

Maneres de desenvolupament pròpies del programari lliure

A *Netizens. On the history and impact of Usenet and the Internet* (IEEE Computer Society Press, 1997 [139], pàgina 139) poden llegir-se unes línies que es podrien referir a molts projectes lliures: "Les contribucions al valor de Unix durant el seu desenvolupament primerenc van ser moltes, gràcies al fet que el codi font estava disponible. Podia ser examinat, millorat i personalitzat".

A la pàgina 142 de la mateixa obra es diu el següent: "Els pioners com Henry Spencer estan d'acord en com va ser d'important per als que pertanyien a la comunitat Unix tenir el codi font. Ell remarca com la disposició dels codis font feia possible la identificació i la reparació dels errors que descobrien. [...]. Fins i tot a finals de 1970 i a principis de 1980, pràcticament cada lloc Unix tenia fonts completes".

Encara més explícit és el text de Marc Rochkind "Interview with dick haight" *Unix Review* (maig 1986) [198]: "aquesta era una de les grans coses sobre Unix en els primers dies: la gent realment compartia amb els altres. [...] No solament apreníem molt aquells dies del material compartit, sinó que mai no havíem de preocupar-nos sobre com funcionaven realment les coses perquè sempre podíem anar a llegir el codi font".

Amb el temps, Unix va ser també un exemple primerenc dels problemes que podien presentar els sistemes privatius que a primera vista "tenien alguna característica del programari lliure". A finals de la dècada de 1970, i sobretot durant la de 1980, AT&T va canviar la seva política, i l'accés a noves versions de Unix es va convertir en una cosa difícil i cara. La filosofia dels primers anys,

que havia fet tan popular Unix entre els desenvolupadors, va canviar radicalment fins al punt que el 1991 AT&T va posar una demanda a la Universitat de Berkeley per haver publicat el codi de Unix BSD que el CSRG de Berkeley havia creat. Però aquesta és una altra història, que reprendrem més endavant.

2.2. El començament: BSD, GNU

Tots els casos descrits en l'apartat anterior van ser iniciatives individuals o no complien estrictament els requisits del programari lliure. Fins a principis de la dècada de 1980 no van aparèixer, de forma organitzada i conscient, els primers projectes per a la creació de sistemes compostos de programari lliure. En aquella època es van començar també a fixar (la qual cosa probablement és més important) els fonaments ètics, legals i fins i tot econòmics que s'han anat desenvolupant i completant fins avui dia. I com que el nou fenomen necessitava un nom, durant aquells anys es va encunyar també el terme mateix *programari lliure*.

2.2.1. Richard Stallman, GNU, FSF: neix el moviment del programari lliure

A començaments de 1984, Richard Stallman, en aquella època empleat en l'AI Lab del MIT, va abandonar la seva feina per començar el projecte GNU. Stallman es considerava un *hacker* dels que gaudien compartint les seves inquietuds tecnològiques i el seu codi. Veia amb desgrat com la seva negativa a firmar acords d'exclusivitat i de no-compartició l'estaven convertint en un estrany al seu propi món, i com l'ús de programari privatiu en el seu entorn el deixava impotent davant de situacions que abans podia resoldre fàcilment.

La seva idea en abandonar el MIT era construir un sistema de programari complet, de propòsit general, però totalment lliure ("The GNU Project", DiBona *et al*.) [208]. El sistema (i el projecte que s'encarregaria de fer-lo realitat) es va dir GNU (acrònim recursiu, "GNU's not Unix"). Encara que des del principi el projecte GNU va incloure en el seu sistema programari ja disponible (com ara TeX o, més endavant, el sistema X Window), hi havia molt per construir. Richard Stallman va començar a escriure un compilador de C (GCC) i un editor (Emacs), tots dos encara en ús (i molt populars).

Des del principi del projecte GNU, Richard Stallman es va preocupar per les llibertats que tindrien els usuaris del seu programari. Estava interessat en el fet que no només els que rebessin els programes directament del projecte GNU, sinó qualsevol que el rebés després de qualsevol nombre de redistribucions i (potser) modificacions, continués podent gaudir dels mateixos drets (modificació, redistribució, etc.). Per a això va escriure la llicència GPL, probablement la primera llicència de programari dissenyada específicament per a garantir que un programa fos lliure en aquest sentit. El mecanisme genèric que utilitzen les llicències de tipus GPL per a aconseguir aquestes garanties, Richard Stall-

man el va anomenar *copyleft*, que continua essent el nom d'una gran família de llicències de programari lliure (Free Software Foundation, GNU General Public License, versió 2, juny 1991) [118].

Richard Stallman també va fundar la Free Software Foundation (FSF) per aconseguir fons, que dedica al desenvolupament i a la protecció del programari lliure, i va establir els seus fonaments ètics amb "The GNU Manifesto" (Free Software Foundation, 1985) [117] i "Why software should not have owners" (Richard Stallman, 1998) [207].

Des del punt de vista tècnic, el projecte GNU va ser concebut com un treball molt estructurat i amb metes molt clares. El mètode habitual estava basat en grups relativament petits de persones (normalment voluntaris) que desenvolupaven alguna de les eines que després encaixarien perfectament al trencaclosques complet (el sistema GNU). La modularitat de Unix, en la qual s'inspirava el desenvolupament d'aquest projecte, coincidia totalment amb aquesta idea. El mètode de treball generalment implicava l'ús d'Internet, però davant de la seva escassa implantació en aquells dies, la Free Software Foundation també venia cintes en les quals gravava les aplicacions, de manera que va ser probablement una de les primeres organitzacions que es va beneficiar econòmicament (encara que de manera bastant limitada) de la creació de programari lliure.

A començaments de la dècada de 1990, uns sis anys després del seu naixement, el projecte GNU estava molt a prop de tenir un sistema complet similar a Unix. Tot i així, fins aquell moment encara no havia produït una de les peces fonamentals: el nucli del sistema (també anomenat *kernel* la part del sistema operatiu que es relaciona amb el maquinari, l'abstreu, i permet que les aplicacions comparteixin recursos i, en el fons, funcionin). Tanmateix, el programari de GNU era molt popular entre els usuaris de les diferents variants de Unix, en aquell temps el sistema operatiu més usat en les empreses. A més, el projecte GNU havia aconseguit ser relativament conegut entre els professionals informàtics, i molt especialment entre els que treballaven en universitats. En aquella època, els seus productes ja tenien una merescuda reputació d'estabilitat i qualitat.

2.2.2. El CSRG de Berkeley

El CSRG (Computer Science Research Group) de la Universitat de Califòrnia, a Berkeley, va ser des de 1973 un dels centres on més desenvolupament relacionat amb Unix es va fer durant els anys 1979 i 1980. No solament es van migrar aplicacions i se'n van construir altres de noves per al seu funcionament sobre Unix, sinó que es van dur a terme importants millores en el nucli i s'hi va afegir molta funcionalitat. Per exemple, durant la dècada dels vuitanta, diversos contractes de DARPA (dependent del Ministeri de Defensa dels EUA) van finançar la implementació de TCP/IP, que ha estat considerat fins als nostres dies com la referència dels protocols que fan funcionar Internet (vinculant,

de passada, el desenvolupament d'Internet i l'expansió de les estacions de treball amb Unix). Moltes empreses van utilitzar com a base de les seves versions de Unix els desenvolupaments del CSRG, que van donar lloc a sistemes molt coneguts en l'època, com ara SunOS (Sun Microsystems) o Ultrix (Digital Equipment). D'aquesta manera, Berkeley es va convertir en una de les dues fonts fonamentals de Unix, juntament amb l'"oficial", AT&T.

Per poder utilitzar tot el codi que produïa el CSRG (i el dels col·laboradors de la comunitat Unix que ells d'alguna manera coordinaven), feia falta la llicència de Unix d'AT&T, que cada vegada era més difícil (i més cara) d'aconseguir, sobretot si es volia l'accés al codi font del sistema. Tractant d'evitar en part aquest problema, el juny de 1989 el CSRG va alliberar la part de Unix relacionada amb TCP/IP (la implementació dels protocols en el nucli i les utilitats), que no incloïa codi d'AT&T. Va ser l'anomenada *Networking Release 1* (Net-1). La llicència que es va alliberar va ser la famosa llicència BSD, que llevat de certs problemes amb les seves clàusules sobre obligacions d'anunci, ha estat considerada sempre un exemple de llicència lliure minimalista (que a més de permetre la redistribució lliure, permetia també la seva incorporació a productes privatius). A més, el CSRG va provar un nou mètode de finançament (que ja estava provant amb èxit l'FSF): venia cintes amb la seva distribució per 1.000 dòlars. Encara que qualsevol podia redistribuir al seu torn el contingut de les cintes sense cap problema (ja que la llicència ho permetia), el CSRG va vendre cintes a milers d'organitzacions, amb la qual cosa va aconseguir fons per continuar desenvolupant.

Veient l'èxit de la distribució de Net-1, Keith Bostic va proposar reescriure tot el codi que encara quedava del Unix original d'AT&T. Malgrat l'escepticisme d'alguns membres del CSRG, va realitzar un anunci públic en el qual demanava ajuda per realitzar aquesta tasca, i a poc a poc les utilitats (reescrites a partir d'especificacions) van començar a arribar a Berkeley. Mentrestant, es va realitzar el mateix procés amb el nucli, de manera que es va reescriure de forma independent la major part del codi que no havien realitzat Berkeley ni col·laboradors voluntaris. El juny de 1991, després d'aconseguir el permís de l'administració de la Universitat de Berkeley, es va distribuir la *Networking Release 2* (Net-2) amb gairebé tot el codi del nucli i totes les utilitats d'un sistema Unix complet. De nou el conjunt es va distribuir sota la llicència BSD i es van vendre milers de cintes al preu de 1.000 dòlars la unitat.

Només sis mesos després de l'alliberament de Net-2, Bill Jolitz va escriure el codi que faltava en el nucli perquè funcionés sobre arquitectura i386, alliberant 386BSD, que va ser distribuït per Internet. A partir d'aquest codi van sorgir més tard, en successió, tots els sistemes de la família *BSD: primer va aparèixer NetBSD com una recopilació dels pedaços que s'havien anat aportant a la Xarxa per a millorar 386BSD; més endavant va aparèixer FreeBSD com un intent de suportar fonamentalment l'arquitectura i386. Diversos anys més tard es va formar el projecte OpenBSD, amb èmfasi en la seguretat. I també hi va haver una distribució propietària basada en Net-2 (encara que era certament origi-

nal, ja que oferia als seus clients tot el codi font com a part de la distribució bàsica), que va realitzar de forma independent la desapareguda empresa BSDI (Berkeley Software Design Inc.).

En part com a reacció a la distribució feta per BSDI, Unix System Laboratories (USL), subsidiària d'AT&T que tenia els drets de la llicència de Unix, va posar una demanda judicial, primer a BSDI i després a la Universitat de Califòrnia. Els acusava de distribuir la seva propietat intel·lectual sense permís. Després de diverses maniobres judicials (que van incloure una contrademanda de la Universitat de Califòrnia contra USL), Novell va comprar els drets de Unix a USL, i el gener de 1994 va arribar a un acord extrajudicial amb la Universitat de Califòrnia. Com a resultat d'aquest acord, el CSRG va distribuir la versió 4.4BSD-Lite, que aviat va ser utilitzada per tots els projectes de la família *BSD. Al cap de poc (després d'alliberar encara la versió 4.4BSD-Lite Release 2), el CSRG va desaparèixer. En aquell moment hi va haver qui va témer que fos el final dels sistemes *BSD, però el temps ha demostrat que segueixen ben vives, amb una nova forma de gestió més típica de projectes lliures. Encara en la dècada del 2000 els projectes que gestionen la família *BSD són dels més antics i consolidats al món del programari lliure.

Bibliografia

La història de Unix BSD és molt il·lustrativa d'una forma peculiar de desenvolupar programari durant els anys setanta i vuitanta. Qui hi estigui interessat pot gaudir de la lectura de "Twenty years of Berkeley Unix" (Marshall Kirk McKusick, 1999) [170], en la qual es pot seguir la seva evolució des de la cinta que va portar Bob Fabry a Berkeley amb la idea de fer funcionar en un PDP-11 una de les primeres versions del codi de Thompson i Ritchie (comprada conjuntament pels departaments d'informàtica, estadística i matemàtiques), fins a les demandes judicials d'AT&T i els últims alliberaments de codi que van donar lloc a la família de sistemes operatius lliures *BSD.

2.2.3. Els començaments d'Internet

Gairebé des del seu naixement, a començaments de la dècada de 1970, Internet va tenir molta relació amb el programari lliure. D'una banda, des dels seus començaments, la comunitat de desenvolupadors que la van construir va tenir clars diversos principis que després es farien clàssics al món del programari lliure; per exemple, la importància que els usuaris puguin ajudar a depurar errors o la compartició de codi. La importància de BSD Unix en el seu desenvolupament (en proporcionar durant els anys vuitanta la implementació més popular dels protocols TCP/IP) va fer que molts costums i formes de funcionament passessin fàcilment d'una comunitat, la de desenvolupadors al voltant del CSRG, a una altra, la dels que estaven construint el que llavors era NSFNet i després seria Internet, i viceversa. Moltes de les aplicacions bàsiques en el desenvolupament d'Internet, com Sendmail (servidor de correu) o BIND (implementació del servei de noms), van ser lliures i, en gran manera, fruit d'aquesta col·laboració entre comunitats.

Finalment, a finals dels anys vuitanta i en la dècada dels noranta, la comunitat del programari lliure va ser una de les primeres que va explorar fins al fons les noves possibilitats que permetia Internet per a la col·laboració entre grups geogràficament dispersos. Aquesta exploració va fer possible, en gran manera, la mateixa existència de la comunitat BSD, la FSF o el desenvolupament de GNU/Linux.

Un dels aspectes més interessants del desenvolupament d'Internet, des del punt de vista del programari lliure, va ser la gestió completament oberta dels seus documents i les seves normes. Encara que avui pugui semblar una mica normal (ja que és l'habitual, per exemple, en l'IETF o en el World Wide Web Consortium), en la seva època la lliure disposició de totes les especificacions i documents de disseny, incloses les normes que defineixen els protocols, va ser una cosa revolucionària i fonamental per al seu desenvolupament. A *Netizens. On the history and impact of Usenet and the Internet* [139] (pàgina 106) podem llegir:

> "Aquest procés obert promovia i portava a l'intercanvi d'informació. El desenvolupament tècnic té èxit només quan es permet que la informació flueixi lliurement i fàcilment entre les parts interessades. La promoció de la participació és el principi fonamental que va fer possible el desenvolupament de la Xarxa."

Podem observar com aquest paràgraf podria ser subscrit, gairebé amb tota seguretat, per qualsevol desenvolupador en referir-se al projecte de programari lliure en el qual col·labora.

En una altra citació, a "The evolution of packet switching" [195] (pàgina 267) podem llegir:

> "Com que ARPANET era un projecte públic que connectava moltes de les principals universitats i institucions d'investigació, els detalls d'implementació i rendiment es publicaven àmpliament."

Òbviament, això és el que sol ocórrer en els projectes de programari lliure, on tota la informació relacionada amb el projecte (i no solament la implementació) sol ser pública.

En aquest ambient, i abans que Internet, ja ben entrats els anys noranta, es convertís sobretot en un negoci, la comunitat d'usuaris i la seva relació amb els desenvolupadors era clau. En aquella època moltes organitzacions van aprendre a confiar no en una única empresa proveïdora del servei de comunicació de dades, sinó en una complexa combinació d'empreses de serveis, fabricants d'equips, desenvolupadors professionals i voluntaris, etc. Les millors implementacions de molts programes no eren les que venien amb el sistema operatiu que es comprava amb el maquinari, sinó implementacions lliures que ràpidament les substituïen. Els desenvolupaments més innovadors eren fruit

no de grans plans d'investigació en empreses, sinó d'estudiants o professionals que provaven idees i recollien la realimentació que els enviaven cents d'usuaris dels seus programes lliures.

Com ja s'ha dit, Internet també va proporcionar al programari lliure les eines bàsiques per col·laborar a distància. El correu electrònic, els grups de notícies, els serveis d'FTP anònim (que van ser els primers magatzems massius de programari lliure) i, més tard, els sistemes de desenvolupament integrats basats en web han estat fonamentals (i imprescindibles) per al desenvolupament de la comunitat del programari lliure tal com la coneixem i, en particular, per al funcionament de la immensa majoria dels projectes de programari lliure. Des del principi, projectes com GNU o BSD van fer un ús massiu i intens de tots aquests mecanismes, desenvolupant, alhora que les empraven, noves eines i sistemes que al seu torn milloraven Internet.

Bibliografia

El lector interessat en una història de l'evolució d'Internet, escrita per diversos dels seus protagonistes, pot consultar "A brief history of the Internet" (comunicació de l'ACM, 1997) [166].

2.2.4. Altres projectes

Durant la dècada de 1980 van veure la llum altres importants projectes lliures. Entre ells destaca, per la seva importància i projecció futura, l'X Window (sistema de finestres per a sistemes de tipus Unix), desenvolupat en el MIT, que va ser un dels primers exemples de finançament a gran escala de projectes lliures amb recursos d'un consorci d'empreses. També val la pena esmentar Ghostscript, un sistema de gestió de documents PostScript desenvolupat per una empresa, Aladdin Software, que va ser un dels primers casos de recerca d'un model de negoci basat en la producció de programari lliure.

A finals dels anys vuitanta hi ha ja en marxa tota una constel·lació de petits (i no tan petits) projectes lliures. Tots ells, juntament amb els grans projectes esmentats fins aquí, van establir les bases dels primers sistemes lliures complets, que van aparèixer a començaments de la dècada de 1990.

2.3. Tot en marxa

Cap a 1990, gran part dels components d'un sistema informàtic complet estaven ja llestos com a programari lliure. D'una banda, el projecte GNU i les distribucions BSD havien completat la major part de les aplicacions que componen un sistema operatiu. De l'altra, projectes com X Window o el mateix GNU havien construït des d'entorns de finestres fins a compiladors, que moltes vegades estaven entre els millors del seu gènere (per exemple, molts administradors de sistemes SunOS o Ultrix substituïen per als seus usuaris les aplicacions propietàries del seu sistema per les versions lliures de GNU o de BSD). Per a tenir un sistema complet construït només amb programari lliure faltava únicament un component: el nucli. Dos esforços separats i independents van omplir aquest buit: 386BSD i Linux.

2.3.1. A la recerca d'un nucli

A finals dels vuitanta i principis dels noranta, el projecte GNU comptava amb una gamma bàsica d'utilitats i eines que permetia tenir el sistema operatiu d'una manera completa. Ja llavors, moltes aplicacions lliures, entre les quals va ser especialment interessant el cas d'X Window, eren les millors en el seu camp (utilitats Unix, compiladors...). Tanmateix, per a completar el trencaclosques faltava una peça essencial: el nucli del sistema operatiu. El projecte GNU estava buscant aquesta peça amb un projecte anomenat Hurd, que pretenia construir un nucli amb tecnologies modernes.

2.3.2. La família *BSD

Pràcticament en la mateixa època, la comunitat BSD estava també en camí cap a un nucli lliure. En la distribució Net-2 només faltaven sis fitxers per a tenir-lo (la resta ja havia estat construïda pel CSRG o els seus col·laboradors). A començaments de 1992, Bill Jolitz va completar aquests fitxers i va distribuir 386BSD, un sistema que funcionava sobre arquitectura i386 i que amb el temps donaria lloc als projectes NetBSD, FreeBSD i OpenBSD. El desenvolupament durant els mesos següents va ser ràpid, i a finals d'any ja era prou estable per a ser usat en producció en entorns no crítics, que incloïen, per exemple, un entorn de finestres gràcies al projecte XFree (que havia portat X Window a l'arquitectura i386) o un compilador de gran qualitat, GCC. Encara que hi havia components que utilitzaven altres llicències (com els procedents del projecte GNU, que usaven la GPL), la major part del sistema es distribuïa sota la llicència BSD.

Bibliografia

Alguns dels episodis d'aquesta època són il·lustratius de la potència dels models de desenvolupament de programari lliure. És ben conegut el cas de Linus Torvalds, que va desenvolupar Linux mentre era estudiant de segon curs de la Universitat d'Hèlsinki. Però no és l'únic cas d'un estudiant que es va obrir camí gràcies als seus desenvolupaments lliures. Per exemple, l'alemany Thomas Roel va migrar X11R4 (una versió del sistema X Window) a un PC basat en un 386. Aquest desenvolupament el va portar a treballar a Dell, i més endavant, a ser fundador dels projectes X386 i XFree, fonamentals perquè GNU/Linux i els *BSD tinguessin aviat un entorn de finestres. Pot llegir-se més sobre la història d'XFree i el paper que hi va fer Roel a "The history of xFree86" (*Linux Magazine*, desembre 1991) [135].

Després va venir la demanda d'USL, que va fer que molts usuaris potencials temessin ser demandats al seu torn si la Universitat de Califòrnia perdia el judici, o simplement que el projecte es parés. Potser aquesta va ser la raó que més endavant la base instal·lada de GNU/Linux fos molt més gran que la de tots els *BSD combinats. Però això és una cosa difícil d'assegurar.

2.3.3. GNU/Linux entra en escena

El juliol de 1991 Linus Torvalds (estudiant finès de vint-i-un anys) posa el primer missatge on esmenta el seu aleshores projecte de fer un sistema lliure similar a Minix. Al setembre allibera la primeríssima versió (0.01), i cada poques

setmanes apareixen noves versions. El març de 1994 va aparèixer la versió 1.0, la primera denominada *estable*, però el nucli que havia construït Linus era utilitzable des de feia bastants mesos. Durant aquest període, literalment centenars de desenvolupadors es bolquen sobre Linux, i integren al seu voltant tot el programari de GNU, XFree i molts altres programes lliures. A diferència dels *BSD, el nucli de Linux i gran part dels components que s'integren al voltant d'ell es distribueixen amb la llicència GPL.

Bibliografia

La història de Linux és probablement una de les més interessants (i conegudes) al món del programari lliure. Es poden trobar molts enllaços a informació sobre aquesta a les pàgines del desè aniversari del seu anunci, encara que probablement la més interessant és "History of Linux", de Ragib Hasan [138]. Com a curiositat, pot consultar-se el fil en el qual Linus Torvalds anunciava que estava començant a crear el que després va ser Linux (al grup de notícies comp.os.minix) a http://groups.google.com/groups?th=d161e94858c4c0b9. Allà explica com porta treballant en el seu nucli des d'abril i com ja ha portat algunes eines del projecte GNU sobre ell (concretament, esmenta Bash i GCC).

Entre els molts desenvolupaments apareguts al voltant de Linux, un dels més interessants és el concepte de *distribució*[1]. Les primeres distribucions van aparèixer aviat, el 1992 (MCC Interim Linux, de la Universitat de Manchester; TAMU, de Texas A&M, i la més coneguda, SLS, que més tard va donar lloc a Slackware, que encara es distribueix en la dècada del 2000), i han suposat l'entrada de la competència al món de l'empaquetament de sistemes al voltant de Linux. Cada distribució tracta d'oferir un GNU/Linux preparat per utilitzar, i partint totes del mateix programari, han de competir en millores que la seva base d'usuaris consideri importants. A més de proporcionar paquets precompilats i preparats per utilitzar, les distribucions solen oferir les seves pròpies eines per gestionar la selecció, la instal·lació, la substitució i la desinstal·lació d'aquests paquets, així com la instal·lació inicial en un ordinador, i la gestió i l'administració del sistema operatiu.

(1)Aquest concepte s'explica en detall en l'entrada corresponent de Wikipedia, www.wikipedia.org/wiki/Linux_distribution

Amb el temps, unes distribucions han anat succeint-se a unes altres com les més populars. Entre totes aquestes, es poden destacar les següents:

1) Debian, desenvolupada per una comunitat de desenvolupadors voluntaris.

2) Red Hat Linux, que primer va desenvolupar internament l'empresa Red Hat, però que més endavant va adoptar un model més comunitari i que va donar lloc a Fedora Core.

3) Suse, que va donar lloc a OpenSUSE, en una evolució similar a la de Red Hat.

4) Mandriva (successora de Mandrake Linux i de Conectiva).

5) Ubuntu, derivada de Debian i produïda a partir d'ella per l'empresa Canonical.

2.4. Temps de maduració

A mitjan dècada 2000, GNU/Linux, OpenOffice.org o Firefox tenen una presència relativament habitual en els mitjans de comunicació. La immensa majoria d'empreses utilitza programari lliure almenys per a alguns dels seus processos informàtics. És difícil ser un estudiant d'informàtica i no utilitzar programari lliure en grans quantitats. El programari lliure ha deixat de ser una nota a peu de pàgina en la història de la informàtica per convertir-se en una cosa molt important per al sector. Les empreses d'informàtica, les del sector secundari (les que utilitzen intensivament programari, encara que la seva activitat principal és una altra) i les administracions públiques comencen a considerar-lo com un fet estratègic. I està arribant, a poc a poc però amb força, als usuaris domèstics. En línies generals, s'entra en una època de maduració.

I en el fons, comença a sorgir una pregunta molt important, que d'alguna forma resumeix el que està passant: "som davant d'un nou model d'indústria programari?". Encara podria passar, potser, que el programari lliure no arribés a ser més que una moda passatgera que amb el temps només fos recordada amb nostàlgia. Però també podria ser (i això sembla cada vegada més plausible) un nou model que és aquí per quedar-se, i potser per canviar radicalment una de les indústries més joves, però també de les més influents.

2.4.1. Finals de la dècada dels noranta

A mitjan dècada dels noranta, el programari lliure oferia ja entorns complets (distribucions de GNU/Linux, sistemes *BSD...) que permetien el treball diari de molta gent, sobretot de desenvolupadors de programari. Encara hi havia moltes assignatures pendents (la principal, disposar de millors interfícies gràfiques d'usuari en una època en què Windows 95 era considerat l'estàndard), però ja hi havia uns quants milers de persones a tot el món que només usaven programari lliure en el seu treball diari. Els anuncis de nous projectes se succeïen i el programari lliure començava el seu llarg camí cap a les empreses, els mitjans de comunicació i, en general, el coneixement públic.

D'aquesta època és també l'enlairament d'Internet com a xarxa per a tots, en molts casos de la mà de programes lliures (sobretot en la seva infraestructura). L'arribada del web a les llars de milions d'usuaris finals consolida aquesta situació, almenys referent a servidors: els servidors web (HTTP) més populars sempre han estat lliures (primer el servidor del NCSA, després Apache).

Potser el començament del camí del programari lliure fins a la seva posada de llarg en la societat pot situar-se en el cèlebre assaig d'Eric Raymond, "La catedral y el bazar" (Eric S. Raymond, 2001) [192]. Encara que molt del que s'hi exposa era ja ben conegut per la comunitat de desenvolupadors de programari lliure, reunir-lo en un article i donar-hi una gran difusió el va convertir en una influent eina de promoció del concepte de *programari lliure* com a mecanisme

de desenvolupament alternatiu a l'emprat per la indústria del programari tradicional. Un altre article molt important en aquesta època va ser "Setting up shop. The Business of open-source software" [141], de Frank Hecker, que per primera vegada va exposar els models de negoci possibles entorn del programari lliure i que va ser escrit per a influir en la decisió sobre l'alliberament del codi del navegador de Netscape.

Si l'article de Raymond va suposar una gran eina de difusió d'algunes de les característiques fonamentals del programari lliure, l'alliberament del codi del navegador de Netscape va ser el primer cas en què una empresa relativament gran, d'un sector molt innovador (la llavors naixent indústria del web), prenia la decisió d'alliberar com a programari lliure un dels seus productes. En aquella època, Netscape Navigator estava perdent la batalla dels navegadors web davant el producte de Microsoft (Internet Explorer), en part per les tàctiques de Microsoft de combinar-lo amb el seu sistema operatiu. Per a molts, Netscape va fer l'única que podia fer: tractar de canviar les regles per a poder competir amb un gegant. I d'aquest canvi de regles (tractar de competir amb un model de programari lliure) va néixer el projecte Mozilla. Aquest projecte, no sense problemes, ha portat diversos anys després a un navegador que, si bé no ha recuperat l'enorme quota de mercat que va tenir al seu dia Netscape Navigator, sembla que tècnicament és almenys encara millor que els seus competidors privatius.

En qualsevol cas, i amb independència del seu èxit posterior, l'anunci de Netscape d'alliberar el codi del seu navegador va suposar un fort impacte en la indústria del programari. Moltes empreses van començar a considerar el programari lliure com a digne de consideració.

També els mercats financers es van començar a ocupar del programari lliure. En plena eufòria de les puntcom, diverses empreses de programari lliure es converteixen en objectiu d'inversors. Potser el cas més conegut és el de Red Hat, una de les primeres empreses que van reconèixer que la venda de CD amb sistemes GNU/Linux preparats per a utilitzar podia ser un model de negoci. Red Hat va començar distribuint la seva Red Hat Linux, fent gran èmfasi (almenys per a l'habitual en l'època) en la facilitat de maneig i manteniment del sistema per persones sense coneixements específics d'informàtica. Amb el temps, va anar diversificant el seu negoci, mantenint-se en general en l'òrbita del programari lliure, i el setembre de 1998 va anunciar que Intel i Netscape hi havien invertit. "Si és bo per a Intel i Netscape, segur que és bo per a nosaltres", devien pensar molts inversors. Quan Red Hat va sortir a borsa a l'estiu de 1999, l'oferta pública d'accions va ser subscrita completament, i aviat el valor de cada títol va pujar espectacularment. Va ser la primera vegada que una empresa aconseguia finançament del mercat de valors amb un model basat en el programari lliure. Però no va ser l'únic: el mateix van fer més tard d'altres com VA Linux o Andover.net (que més tard va ser adquirida per VA Linux).

Nota

Red Hat proporciona una llista de fites històriques relacionades amb la seva empresa a http://fedora.redhat.com/about/history/.

En aquesta època van néixer també moltes empreses basades en models de negoci entorn del programari lliure. Sense sortir a borsa i sense aconseguir capitalitzacions tan formidables, van ser tanmateix molt importants per al desenvolupament del programari lliure. Per exemple, van aparèixer moltes empreses que van començar distribuint les seves pròpies versions de GNU/Linux, com SuSE (Alemanya), Conectiva (Brasil) o Mandrake (França), que més tard s'uniria a l'anterior per formar Mandriva. D'altres proporcionaven serveis a empreses que ja demanaven manteniment i adaptació de productes lliures: LinuxCare (EUA), Alcove (França), ID Pro (Alemanya), i moltes més.

Per la seva part, els gegants del sector també van començar a posicionar-se davant del programari lliure. Algunes empreses, com IBM, el van incorporar directament en la seva estratègia. Altres, com Sun Microsystems, van mantenir amb ell una curiosa relació, de vegades de suport, de vegades d'indiferència, de vegades d'enfrontament. La majoria (com Apple, Oracle, HP, SGI, etc.) van explorar el model del programari lliure amb diverses estratègies, que anaven des de l'alliberament selectiu de programari fins a la simple migració a GNU/Linux dels seus productes. Entre aquests dos extrems, hi va haver moltes altres línies d'acció, com la utilització més o menys intensa de programari lliure als seus productes (com va ser el cas del Mac OS X) o l'exploració de models de negoci basats en el manteniment de productes lliures.

Des del punt de vista tècnic, el més destacable d'aquesta època va ser probablement l'aparició de dos ambiciosos projectes amb la finalitat d'aconseguir migrar el programari lliure a l'entorn d'escriptori (*desktop*) dels usuaris no gaire versats en la informàtica: KDE i GNOME. Dit de forma molt simplista, l'objectiu final era que no s'hagués d'usar la línia d'ordres per a interaccionar amb GNU/Linux o *BSD ni amb els programes sobre aquests entorns.

KDE va ser anunciat l'octubre de 1996. Utilitzant les biblioteques gràfiques Qt (en aquell temps un producte privatiu de l'empresa Trolltech, però gratuït per al seu ús sobre GNU/Linux[2]), van iniciar la construcció d'un conjunt d'aplicacions d'escriptori que funcionessin de forma integrada i que tinguessin un aspecte uniforme. El juliol de 1998 van alliberar la versió 1.0 del K Desktop Environment, que va ser aviat seguida de noves versions cada vegada més completes i madures. Les distribucions de GNU/Linux aviat van incorporar KDE com a escriptori per als seus usuaris (o almenys com un dels entorns d'escriptori que els seus usuaris podien triar).

[(2)]Més tard, Qt va passar a ser distribuït sota una llicència lliure, la QPL (Qt Public License), no compatible amb la GPL, fet que causava algun problema, ja que la major part de KDE estava distribuït sota la GPL. Amb el temps, finalment Trolltech va decidir distribuir Qt sota la llicència GPL, amb la qual cosa aquests problemes van acabar.

En gran manera com a reacció a la dependència que tenia KDE de la biblioteca propietària Qt, l'agost de 1997 es va anunciar el projecte GNOME (Miguel de Icaza, "The story of the GNOME Project") [101], amb fins i característiques molt similars als de KDE, però amb l'objectiu explícit que tots els seus com-

ponents fossin lliures. El març de 1999 es va alliberar GNOME 1.0, que amb el temps també s'aniria millorant i estabilitzant. A partir d'aquell moment, la major part de les distribucions de sistemes operatius lliures (i molts derivats de Unix privatius) van oferir com a opció l'escriptori de GNOME o el de KDE i les aplicacions d'ambdós entorns.

Simultàniament, els principals projectes de programari lliure que ja funcionaven continuen amb bona salut i sorgeixen nous projectes cada dia. A diverses vetes de mercat s'observa com la millor solució (reconeguda per gairebé tothom) és programari lliure. Per exemple, Apache ha mantingut gairebé des de la seva aparició l'abril de 1995 la major quota de mercat entre els servidors web; XFree86, el projecte lliure que desenvolupa X Window, és amb diferència la versió d'X Window més popular (i, per tant, el sistema de finestres per a sistemes de tipus Unix més estès); GCC és reconegut com el compilador de C més portable i un dels de millor qualitat; GNAT, sistema de compilació per a Ada 95, aconsegueix la major part del mercat de compiladors Ada en pocs anys; i així successivament.

El 1998 es va crear l'Open Source Initiative (OSI), que va decidir adoptar el terme *open source software* ('programari de font oberta') com una marca per a introduir el programari lliure al món comercial, tractant d'evitar l'ambigüitat que en anglès suposa el terme *free* (que significa tant 'lliure' com 'gratis'). Aquesta decisió va suposar (i encara suposa) un dels debats més enverinats del món del programari lliure, ja que la Free Software Foundation i d'altres van considerar que era molt més apropiat parlar de *programari lliure* (Richard Stallman,"Why *free software* is better than *open source*", 1998) [206]. En qualsevol cas, l'OSI va realitzar una fructífera campanya de difusió de la seva nova marca, que ha estat adoptada per molts com la forma preferida de parlar del programari lliure, sobretot en el món anglosaxó. Per a definir el programari *open source*, l'OSI va utilitzar una definició derivada de la que utilitza el projecte Debian per a definir el programari lliure (Debian Free Software Guidelines" http://www.debian.org/social_contract.html#guidelines) [104], que d'altra banda reflecteix amb bastant d'aproximació la idea de la FSF sobre això ("Free software definition", http://www.gnu.org/philosophy/free-sw.html) [120], per la qual cosa, des del punt de vista pràctic gairebé qualsevol programa que és considerat programari lliure és també considerat *open source* i viceversa. Tanmateix, les comunitats de programari lliure i de programari de font oberta (o almenys les persones que s'identifiquen com a part d'una o d'una altra) poden ser profundament diferents.

2.4.2. Dècada de 2000

A començaments de la dècada del 2000 el programari lliure és ja un seriós competidor en el segment de servidors i comença a estar llest per a l'escriptori. Sistemes com GNOME, KDE, OpenOffice.org i Mozilla Firefox poden ser utilitzats per usuaris domèstics i són suficients per a les necessitats de moltes em-

preses, almenys pel que fa a ofimàtica. Els sistemes lliures (i sobretot els basats en Linux) són fàcils d'instal·lar, i la complexitat per a mantenir-los i actualitzar-los és comparable a la d'altres sistemes privatius.

En aquests moments, qualsevol empresa de la indústria del programari té una estratègia respecte al programari lliure. La majoria de les grans multinacionals (IBM, HP, Sun, Novell, Apple, Oracle...) incorpora el programari lliure amb més o menys decisió. En un extrem podríem situar empreses com Oracle, que reaccionen simplement portant els seus productes a GNU/Linux. En l'altre podria situar-se IBM, que té l'estratègia més decidida i ha realitzat les majors campanyes de publicitat sobre GNU/Linux. Entre els líders del mercat informàtic, només Microsoft s'ha significat amb una estratègia clarament contrària al programari lliure i en particular al programari distribuït sota llicència GPL.

Quant al món del programari lliure en si mateix, malgrat els debats que de tant en tant sacsegen la comunitat, el creixement és enorme. Cada vegada hi ha més desenvolupadors, més projectes de programari lliure actius, més usuaris... Cada vegada més el programari lliure està passant de ser una cosa marginal a convertir-se en un competidor que cal tenir en compte.

Davant d'aquest desenvolupament, apareixen noves disciplines que estudien específicament el programari lliure, com l'enginyeria del programari lliure. A partir de les seves investigacions comencem, a poc a poc, a entendre com funciona el programari lliure en els seus diferents aspectes: models de desenvolupament, models de negoci, mecanismes de coordinació, gestió de projectes lliures, motivació de desenvolupadors, etc.

En aquests anys comencen a veure també els primers efectes de la *deslocalització*, que permet el desenvolupament de programari lliure: països considerats "perifèrics" participen en el món del programari lliure de forma molt activa. Per exemple, és significatiu el nombre de desenvolupadors mexicans o espanyols (ambdós països amb poca tradició d'indústria programari) en projectes com GNOME (Lancashire, "Code, culture and cash: the fading altruism of open source development", 2001) [164]. I és encara més interessant el paper del Brasil, que està tenint una gran quantitat de desenvolupadors i experts en tecnologies de programari lliure i un decidit suport per part de les administracions públiques. Esment a part mereix el cas de gnuLinEx, molt significatiu de com una regió amb poca tradició de desenvolupament de programari pot tractar de canviar la situació amb una estratègia agressiva d'implantació de programari lliure.

Des del punt de vista de la presa de decisions a l'hora d'implantar solucions programari, cal destacar el fet que hi ha certs mercats (com els serveis d'Internet o l'ofimàtica) en els quals el programari lliure s'ha convertit en una opció natural la no-consideració de la qual és difícil de justificar quan s'està estudiant quin tipus de sistema utilitzar.

En el costat negatiu, aquests anys han vist com l'entorn legal en el qual es mou el programari lliure està canviant ràpidament a tot el món. Per una part, les patents de programari (patents de programació) són considerades cada vegada a més països. Per una altra, les noves lleis de protecció de drets d'autor dificulten o fan impossible el desenvolupament d'aplicacions lliures en alguns àmbits, els més coneguts dels quals és el dels visors de DVD (a causa de l'algoritme CSS d'ofuscació d'imatges que s'utilitza en aquesta tecnologia).

gnuLinEx

A començaments del 2002 la Junta d'Extremadura va donar a conèixer públicament el projecte gnuLinEx. La idea era simple: promoure la creació d'una distribució basada en GNU/Linux amb l'objectiu fonamental d'utilitzar-la en els milers d'ordinadors que s'instal·laran en els centres educatius públics de tota la regió. Extremadura, situada en la part occidental d'Espanya, fronterera amb Portugal, compta amb aproximadament un milió d'habitants i mai no s'havia destacat per les seves iniciatives tecnològiques. De fet, la regió pràcticament mancava d'indústria de programari.

En aquest context, gnuLinEx ha suposat una aportació molt interessant en el panorama del programari lliure a escala mundial. Molt més enllà de ser una nova distribució de GNU/Linux basada en Debian (cosa que no deixa de ser una cosa relativament anecdòtica), i més enllà del seu enorme impacte en els mitjans de comunicació (és la primera vegada que Extremadura ha estat portada de *The Washington Post* i una de les primeres que ho ha estat un producte de programari lliure), l'extraordinari és la (almenys aparentment) sòlida aposta d'una administració pública pel programari lliure. La Junta d'Extremadura va decidir provar un model diferent quant al programari usat per a l'ensenyament i més endavant a tot l'ús de la informàtica dins de les seves competències. Això l'ha convertit en la primera Administració pública d'un país desenvolupat que pren decididament aquest camí. Entorn de la iniciativa de la Junta s'ha produït també molt moviment, tant dins com fora d'Extremadura: hi ha acadèmies que ensenyen informàtica amb gnuLinEx; s'han escrit llibres per a defensar aquest ensenyament; es venen ordinadors amb gnuLinEx preinstal·lat. En general, s'intenta crear entorn d'aquesta experiència tot un teixit docent i empresarial que li proporcioni suport. I l'experiència s'ha exportat. A començaments del segle XXI diverses comunitats autònomes a Espanya han apostat (d'una o una altra forma) pel programari lliure per a l'ensenyament, i en general, la seva importància per a les administracions públiques és reconeguda àmpliament.

Knoppix

Des de finals dels anys noranta hi ha distribucions de GNU/Linux que s'instal·len fàcilment, però probablement Knoppix, la primera versió del qual va aparèixer el 2002, ha portat aquest concepte a la seva màxima expressió. Es tracta d'un CD que arrenca pràcticament a qualsevol PC i el converteix (sense haver de formatar ni tan sols el disc, ja que permet el seu ús "en viu") en una

màquina GNU/Linux completament funcional, amb una selecció de les eines més habituals. Knoppix uneix una bona detecció automàtica de maquinari amb una bona selecció de programes i un funcionament "en viu". Permet, per exemple, una experiència ràpida i directa de què pot suposar treballar amb GNU/Linux. I està donant lloc a tota una família de distribucions del mateix tipus, especialitzades per a necessitats específiques d'un perfil d'usuaris.

OpenOffice.org

El 1999 Sun Microsystems va comprar una empresa alemanya anomenada Stardivision, el producte estrella de la qual era StarOffice, un joc d'eines ofimàtic similar en funcionalitat a Office, el joc d'eines de Microsoft. Un any més tard, Sun va distribuir gran part del codi de StarOffice sota una llicència lliure (la GPL) que donà lloc al projecte OpenOffice.org. Aquest projecte va alliberar la versió 1.0 d'OpenOffice.org el maig de 2002. OpenOffice.org s'ha convertit en un joc d'aplicacions ofimàtiques de qualitat i funcionalitat similar a la de qualsevol altre producte ofimàtic i, el que és més important, interopera molt bé amb els formats de dades de Microsoft Office. Aquestes característiques han fet de la l'aplicació de referència del programari lliure al món de l'ofimàtica.

La importància d'OpenOffice.org, des del punt de vista d'extensió del programari lliure a un gran nombre d'usuaris, és enorme. Per fi ja és possible canviar, pràcticament sense traumes, dels entorns privatius habituals en ofimàtica (sens dubte l'aplicació estrella al món empresarial) a entorns completament lliures (per exemple, GNU/Linux més GNOME i/o KDE més OpenOffice.org). A més, la transició pot fer-se de forma molt suau: com que OpenOffice.org funciona també sobre Microsoft Windows, no cal canviar de sistema operatiu per a experimentar en profunditat l'ús de programari lliure.

Mozilla, Firefox i els altres

Pràcticament des de la seva aparició el 1994 fins a 1996, Netscape Navigator va ser el líder indiscutible del mercat de navegadors web, amb quotes de fins al 80%. La situació va començar a canviar quan Microsoft va incloure Internet Explorer en el seu Windows 95, fet que va suposar que a poc a poc Netscape Navigator anés perdent quota de mercat. A començaments de 1998 Netscape va anunciar que distribuiria gran part del codi del seu navegador com a programari lliure, cosa que efectivament va fer el març del mateix any, llançant el projecte Mozilla. Durant bastant temps l'esmentat projecte va estar envoltat d'incertesa, i fins i tot de pessimisme (per exemple quan el seu líder, Jamie Zawinski, el va abandonar), ja que passava el temps i no apareixia cap producte com a resultat del seu llançament.

El gener del 2000, el projecte va alliberar Mozilla M13, que va ser considerat la primera versió raonablement estable. Però només el maig del 2002 es va publicar finalment la versió 1.0, la primera oficialment estable, més de quatre anys després de l'alliberament del primer codi de Netscape Navigator.

Bibliografia

A "Netscape Navigator", de Brian Wilson [234], pot consultar-se una ressenya detallada de les principals versions de Netscape Navigator i Mozilla, així com de les seves principals característiques.

Per fi Mozilla era una realitat, encara que potser massa tard, si tenim en compte les quotes de mercat que va tenir Internet Explorer durant el 2002 o 2003 (en els quals va ser líder indiscutible relegant Mozilla i altres a una posició completament marginal). Però el projecte Mozilla, malgrat trigar tant, va donar els seus fruits; no només els esperats (el navegador Mozilla), sinó molts altres que podrien considerar-se "col·laterals", com per exemple Firefox, un altre navegador basat en el mateix motor d'HTML, que amb el temps s'ha convertit en el producte principal, i que des de la seva aparició el 2005 està aconseguint erosionar a poc a poc les quotes de mercat d'altres navegadors.

El projecte Mozilla ha ajudat a completar un gran buit al món del programari lliure. Abans de l'aparició de Konqueror (el navegador del projecte KDE), no hi havia gaires navegadors lliures amb interfície gràfica. A partir de la publicació de Mozilla, han anat apareixent gran quantitat de projectes que s'hi han basat i que han produït una bona quantitat de navegadors. D'altra banda, la combinació de Moxilla Firefox i OpenOffice.org permet fer servir programari lliure per a les tasques més quotidianes, fins i tot en un entorn Microsoft Windows (ambdós funcionen no només sobre GNU/Linux, *BSD i altres sistemes de tipus Unix, sinó que també ho fan sobre Windows). Això permet, per primera vegada en la història del programari lliure, que la transició de programari privatiu a programari lliure en entorns d'oficina sigui simple: es pot començar utilitzant aquestes dues aplicacions sobre Windows, sense canviar de sistema operatiu (en el cas dels que l'usen habitualment), i amb el temps eliminar l'única peça no lliure i passar a GNU/Linux o a FreeBSD.

El cas SCO

A començaments del 2003 la corporació SCO (anteriorment Caldera Systems i Caldera International) va interposar una demanda contra IBM per presumpta infracció dels seus drets de propietat intel·lectual. Encara que la demanda era complexa, estava centrada en l'acusació que IBM havia contribuït al nucli de Linux amb codi que era d'SCO. El maig del 2007 l'assumpte encara no estava resolt i s'havia complicat encara més amb més demandes (d'IBM i Red Hat contra SCO, d'SCO contra AutoZone i DaimlerChrysler, dos grans usuaris informàtics) i amb campanyes de SCO en les quals amenaçava de demandar grans empreses que usen Linux, etc.

Encara que el guanyador d'aquesta gran batalla legal encara no es coneix, l'assumpte ha posat en relleu certs aspectes legals del programari lliure. En particular, moltes empreses s'han plantejat els problemes a què es poden en-

frontar en fer servir Linux i altres programes lliures i quines garanties tenen que en fer-ho no estan infringint drets de propietat intel·lectual o industrial de tercers.

D'alguna manera, aquest cas i alguns altres (com els relacionats amb la validesa de la GPL que s'han resolt a Alemanya el 2005) poden interpretar-se també com un símptoma de la maduresa del programari lliure. Ja ha deixat de ser un element estrany al món empresarial i ha entrat a formar part de moltes de les seves activitats (incloses les que tenen a veure amb estratègies legals).

Ubuntu, Canonical, Fedora i Red Hat

Encara que Canonical (l'empresa que produeix i comercialitza Ubuntu) podria considerar-se gairebé com una nouvinguda al negoci de les distribucions GNU/Linux, les seves activitats mereixen que li dediquem atenció. En relativament poc temps, Ubuntu s'ha establert com una de les distribucions més conegudes i utilitzades, amb fama de bona qualitat i molta simplicitat d'instal·lació i ús. Ubuntu també es caracteritza pel fet de tenir molta més cura a incloure fonamentalment programari lliure que la majoria de les altres distribucions produïdes per empreses.

Tanmateix, la característica probablement fonamental d'Ubuntu (i de l'estratègia de Canonical) ha estat basar-se en Debian, una distribució creada i mantinguda per voluntaris. De fet, Ubuntu no ha estat el primer cas de distribució basada en Debian (un altre cas ben conegut és gnuLinEx), però potser sí que ha estat, entre totes ells, el que més recursos ha rebut. Per exemple, Canonical ha contractat una gran quantitat d'experts en Debian (molts dels quals participen en el projecte) i ha seguit una estratègia que sembla buscar la col·laboració amb el projecte voluntari. D'alguna manera, Canonical ha tractat de completar el que considera que falta a Debian per a tenir una acceptació per l'usuari mitjà.

Red Hat, per la seva part, ha seguit un camí diferent per arribar a una situació bastant similar. Partint d'una distribució realitzada completament amb els seus propis recursos, va decidir col·laborar amb Fedora, un grup de voluntaris que ja estava treballant amb distribucions basades en Red Hat, per produir Fedora Core, la seva distribució "popular". Red Hat manté la seva versió per a empreses, però aquesta col·laboració amb voluntaris és, en el fons, similar a la que ha donat lloc a Ubuntu.

Potser tots aquests moviments no són més que el fruit de la ferotge competència que té lloc al mercat de distribucions GNU/Linux i d'una tendència de més calat: la col·laboració d'empreses amb voluntaris (amb *la comunitat*) per a produir programari lliure.

Les distribucions particularitzades

Des que Linux va entrar en escena, molts grups i empreses han creat la seva pròpia distribució basada en aquest. Però durant aquests anys, el fenomen s'ha estès a moltes organitzacions i empreses que volen tenir una distribució particularitzada per a les seves pròpies necessitats. L'abaratiment del procés de particularització d'una distribució i l'àmplia disposició del coneixement tècnic per a fer-ho han permès l'expansió d'aquesta activitat, que s'ha convertit fins i tot en una veta de negoci per a algunes empreses.

Potser un dels casos més coneguts de distribucions particularitzades és el de les distribucions autonòmiques a Espanya. La Junta d'Extremadura va començar amb el seu gnuLinEx una tendència que han seguit moltes altres comunitats autònomes. El procés és tan comú que n'hi ha diverses que convoquen de forma regular concursos públics per a crear i mantenir les noves versions de les seves distribucions.

La creació de distribucions particularitzades fa real una tendència de la qual es parlava des de feia temps al món del programari lliure: l'adaptació dels programes a les necessitats específiques dels usuaris, sense que faci falta que els productors originals necessàriament hagin de realitzar aquesta adaptació.

Bibliografia

Algunes de les distribucions autonòmiques més conegudes de GNU/Linux són les següents:

- gnuLinEx: http://linex.org (Extremadura)
- Guadalinex: http://guadalinex.org (Andalusia)
- Lliurex: http://lliurex.net (Comunitat Valenciana)
- Augustux: http://www.zaralinux.org/proy/augustux/ (Aragó)
- MAX: http://www.educa.madrid.org/web/madrid_linux/ (Madrid)
- MoLinux: http://molinux.info (Castella - la Manxa)

Col·laboració d'empreses amb empreses i de voluntaris amb empreses

Des de pràcticament el principi del programari lliure hi ha hagut empreses que col·laboraven amb voluntaris en el desenvolupament d'aplicacions. Tanmateix, durant aquests anys en què sembla que s'està arribant a la maduresa, cada vegada són més les empreses que usen el programari lliure com a part de la seva estratègia per a col·laborar amb altres empreses, quan això els resulta interessant. Dos dels casos més significatius, organitzats específicament amb aquesta finalitat, són ObjectWeb (aliança nascuda a França que amb el temps va passar a ser clarament internacional) i Morfeo (a Espanya). En ambdós ca-

sos, un grup d'empreses s'ha posat d'acord per desenvolupar un conjunt de sistemes lliures que els resulten interessants i que decideixen distribuir com a programari lliure.

En altres casos, les empreses busquen activament o bé col·laborar en projectes lliures promoguts per voluntaris, o bé tractar que siguin els voluntaris els que col·laborarin en els seus propis projectes lliures. Exemples de la primera situació són la GNOME Foundation o el ja esmentat Ubuntu respecte a Debian. Entre els segons, es pot destacar el cas de Sun i OpenOffice.org i OpenSolaris, o el de Red Hat amb Fedora Core.

Extensió a altres àmbits

El programari lliure ha demostrat que, en el camp de la producció de programes, és possible una altra forma de fer les coses. S'ha vist a la pràctica com atorgant les llibertats de distribució, modificació i ús es pot aconseguir la sostenibilitat, o bé mitjançant treball voluntari, o bé fins i tot mitjançant una generació de negoci que permeti la supervivència de les empreses.

Amb el temps, aquesta mateixa idea s'està traslladant a altres camps de producció d'obra intel·lectual. Les llicències Creative Commons han permès simplificar el procés d'alliberament en camps com la literatura, la música o el vídeo. Wikipedia està mostrant que en una àrea tan particular com la producció d'enciclopèdies es pot recórrer un camí molt interessant. I cada vegada hi ha més autors literaris, grups musicals i fins i tot productores de pel·lícules interessats en models lliures de producció i distribució.

En tots aquests dominis queda molt camí per recórrer, i en gairebé tots la pràctica encara no ha demostrat completament que és possible la creació sostenible amb models lliures. Però no es pot negar que l'experimentació sobre això està entrant en estat d'ebullició.

El programari lliure com a objecte d'estudi

Encara que alguns treballs, com el conegut "La catedral y el bazar", van començar a obrir el camí de l'estudi del programari lliure com a tal, no va ser fins a l'any 2001 i següents quan la comunitat acadèmica va començar a considerar el programari lliure com un objecte digne d'estudi. Amb el temps, la gran disponibilitat de dades (gairebé tot en el món del programari lliure és públic i està disponible en magatzems d'informació públics) i les novetats que s'hi observen han anat centrant l'atenció de molts grups. A mitjan dècada del 2000 hi ha ja diversos congressos internacionals que es dediquen específicament al programari lliure, les revistes més prestigioses li dediquen amb certa regularitat monogràfics, i les agències que financen la investigació estan obrint línies orientades específicament a ell.

2.5. El futur: una cursa d'obstacles?

Sens dubte, és difícil predir el futur. I sens dubte, no és una cosa a què ens hàgim de dedicar aquí. Per tant, més que tractar d'explicar com serà el futur del programari lliure, intentarem mostrar els problemes que previsiblement haurà d'afrontar (i de fet ja fa temps que afronta). De com sigui el món del programari lliure capaç de superar aquests obstacles dependrà, indubtablement, la seva situació d'aquí a uns anys.

- Tècnica FUD (*fear, uncertainity, doubt*, o en català, 'por, desconeixement, dubte'). Es tracta d'una tècnica bastant habitual al món de les tecnologies de la informació i que ha estat utilitzada pels competidors de productes de programari lliure per a tractar de desacreditar-los, amb més o menys raó i amb èxit variable. En línies generals, el programari lliure, potser a causa de la seva complexitat i de diversos mètodes de penetració en les empreses, ha resultat bastant immune a aquestes tècniques.

- Dissolució. Moltes empreses estan provant els límits del programari lliure com a model, i en particular tracten d'oferir als seus clients models que presenten algunes característiques similars al programari lliure. El principal problema que pot presentar aquest tipus de models és la confusió que genera en els clients i els desenvolupadors, que han d'estudiar amb molt detall la lletra petita per adonar-se que el que se'ls està oferint no té els avantatges que per a ells suposa el programari lliure. El cas més conegut de models d'aquest tipus és el programa Shared Source, de Microsoft.

- Desconeixement. En molts casos els usuaris arriben al programari lliure simplement perquè creuen que és gratis; o perquè consideren que està "de moda". Si no aprofundeixen més i no estudien amb cert deteniment els avantatges que els pot oferir el programari lliure com a model, corren el risc de no aprofitar-se'n. En molts casos, les suposicions de partida al món del programari lliure són tan diferents de les habituals al món del programari privatiu que és indispensable una mínima anàlisi per a comprendre que el que en un cas és habitual en l'altre pot ser impossible, i viceversa. El desconeixement, per tant, no pot sinó generar insatisfaccions i pèrdua d'oportunitats en qualsevol persona o organització que s'aproximi al programari lliure.

- Impediments legals. Sens dubte aquest és el principal problema amb què es trobarà el programari lliure en els pròxims anys. Encara que l'entorn legal en el qual es va desenvolupar el programari lliure durant la dècada de 1980 i la primera meitat de la de 1990 no era ideal, almenys deixava suficient espai perquè creixés en llibertat. Des de llavors, l'extensió de l'àmbit de la patentabilitat al programari (que s'ha produït a molts països desenvolupats) i les noves legislacions sobre drets d'autor (que limiten la llibertat de creació del desenvolupador de programari) suposen cada vegada bar-

reres més altes a l'entrada del programari lliure en segments importants d'aplicacions.

2.6. Resum

En aquest capítol es presenta la història del programari lliure. Els anys seixanta van ser una etapa dominada pels grans ordinadors i IBM en què el programari es distribuïa amb el maquinari, i habitualment amb el codi font. En la dècada dels setanta es va començar a vendre el programari separadament, i ràpidament la distribució privativa, que no inclou el codi font i que no atorga permís de modificació o redistribució, es va convertir gairebé en l'única opció.

Vegeu també

El lector interessat podrà trobar a l'apèndix B una llista d'algunes de les dates més rellevants de la història del programari lliure.

En la dècada de 1970 va començar el desenvolupament del sistema operatiu Unix en els Bell Labs d'AT&T, que va donar lloc més endavant a Unix BSD. La seva evolució, paral·lela al naixement d'Internet, va servir de camp de proves per a noves formes de desenvolupament en col·laboració que després van ser habituals al món del programari lliure.

El 1984 Richard Stallman va començar a treballar en el projecte GNU, va fundar la Free Software Foundation (FSF), va escriure la llicència GPL i, en general, va establir els fonaments del programari lliure tal com ha estat conegut més tard.

En la dècada de 1990 Internet va anar madurant i proporcionant a les comunitats de programari lliure nous canals de comunicació i distribució. El 1991, Linus Torvalds va començar a desenvolupar un nucli lliure (Linux) que va permetre completar el sistema GNU, que comptava ja amb gairebé totes les peces per convertir-se en un sistema complet similar a Unix: compilador de C (GCC), editor (Emacs), sistema de finestres (X Window), etc. Van néixer, doncs, els sistemes operatius GNU/Linux, que van fructificar en una multitud de distribucions, com Red Hat Linux i Debian GNU/Linux. A finals de la dècada de 1990 aquests sistemes es completaven amb dos entorns d'escriptori: KDE i GNOME.

En la dècada del 2000 el programari lliure arriba a liderar alguns sectors (com el de servidors web, dominat per Apache), i apareixen noves eines que cobreixen una gran quantitat de necessitats informàtiques.

3. Aspectes legals

> "The licenses for most software are designed to take away your freedom to share and change it."
>
> "Les llicències de la majoria dels programes estan dissenyades per a treure't la llibertat de compartir-los i canviar-los. "
>
> GNU General Public License, versió 2

En aquest capítol es presenten els principals aspectes legals relacionats amb el programari lliure. Per posar-los en context, es comença per una petita introducció als conceptes més bàsics de la propietat intel·lectual i industrial, abans d'exposar la definició detallada de *programari lliure*, *programari de font oberta* i altres conceptes relacionats. S'analitzen també amb cert detall les llicències de programari lliure més habituals i el seu impacte sobre els models de negoci (tema que es tractarà amb més detall al capítol 5) i els models de desenvolupament.

3.1. Breu introducció a la propietat intel·lectual

El terme *propietat intel·lectual* té diverses accepcions segons el context i qui l'utilitza. Avui en dia s'utilitza a molts fòrums per a agrupar diferents privilegis que s'atorguen sobre béns intangibles amb valor econòmic. Entre ells podem destacar els de *copyright* (drets d'autor) i similars, que protegeixen de la còpia no autoritzada treballs literaris o artístics, programes d'ordinador, recopilacions de dades, dissenys industrials, etc.; les marques, que protegeixen símbols; les indicacions geogràfiques, que protegeixen denominacions d'origen; els secrets industrials, que defensen l'ocultació d'informació, i les patents, que atorguen monopolis temporals sobre invencions a canvi de revelar-les. Tanmateix, en moltes tradicions legals, entre elles la hispana, es distingeix entre la *propietat intel·lectual*, que es refereix exclusivament als drets d'autor, i la *propietat industrial*, que inclou les figures restants.

En qualsevol cas, la legislació que s'aplica en tots aquests aspectes és una de les més coordinades a pràcticament tot el món. D'una banda, l'OMPI (Organització Mundial de la Propietat Intel·lectual, WIPO segons les seves sigles en anglès) promou ambdós tipus de propietat en tots els seus aspectes. Per una altra, l'acord TRIPS (aspectes comercials de la propietat intel·lectual) estableix uns mínims de protecció i obliga tots els països membres de l'OMC (Organització Mundial del Comerç, WTO) a desenvolupar-los en uns certs terminis, que depenen del nivell de desenvolupament del país[3].

(3) L'acord TRIPS va ser firmat per la pressió dels països industrialitzats (especialment els Estats Units i el Japó).

La Declaració Universal dels Drets Humans reconeix, al seu article 27, el dret que es protegeixin els interessos morals i materials que corresponguin a qualsevol persona per raó de les produccions científiques, literàries o artístiques de

les quals siguin autors. Tanmateix, en molts casos (i de forma habitual en el cas del programari), aquest dret sol ser transferit a la pràctica a les empreses que ocupen els creadors o que comercialitzen les seves creacions. No obstant això, la propietat intel·lectual es justifica no solament per raons morals, sinó també per raons pràctiques, per donar compliment a un altre dret: el de la societat a beneficiar-se de les creacions, incentivant-les amb beneficis i protegint les inversions per a la creació, la investigació i el desenvolupament. Per harmonitzar ambdós drets, la propietat intel·lectual és temporal, i caduca quan ha complert la seva funció de promoció.

Però la caducitat no és l'única característica que diferencia la propietat intel·lectual de l'ordinària. Avui en dia, els seus objectes poden copiar-se fàcilment i econòmicament, sense pèrdua de qualitat. La còpia no perjudica a qui ja gaudeix del del que s'ha copiat, al contrari del robatori, que sí que priva de l'objecte el posseïdor original. La còpia sí que pot perjudicar el propietari, ja que el priva potencialment dels ingressos d'una venda. El control de la còpia d'intangibles és molt més complicat que el del robatori de béns tangibles i pot portar-nos a una societat policial, que necessiti controlar totes les còpies d'informació, i a una gran inseguretat jurídica, perquè augmenten les possibilitats de violació accidental de drets. A més la creativitat és incremental: en crear sempre es copia alguna cosa, i la línia divisòria entre la còpia basta i la inspiració és subtil.

Per aprofundir més en tot això, en els següents apartats es repassen algunes de les categories de la propietat intel·lectual. En qualsevol cas, es pot avançar ja que el programari lliure proposa un nou punt d'equilibri en aquest àmbit, en què prevalen els beneficis de la còpia i la innovació incremental davant el control exclusiu d'una obra per part del seu autor.

3.1.1. Drets d'autor

Els drets d'autor (*copyright*) protegeixen l'expressió d'un contingut, no el contingut en si mateix. Es van desenvolupar per a recompensar els autors de llibres o d'art. Les obres protegides poden expressar idees, coneixements o mètodes lliurement utilitzables, però es prohibeix reproduir-les sense permís, de manera total o parcialment, amb modificacions o sense. Aquesta protecció és molt senzilla, ja que entra automàticament en vigor amb àmbit gairebé universal en el moment de la publicació de l'obra. Modernament s'ha estès als programes d'ordinador i (en algunes àrees geogràfiques) a recopilacions de dades.

La Llei de propietat intel·lectual (LPI) a Espanya, i lleis similars en altres països, desplegades sobre la base del Conveni de Berna per a la protecció de treballs literaris i artístics de 1886, regulen els drets d'autor. Aquests drets es divideixen en drets morals i drets patrimonials. Els primers garanteixen a l'autor el control sobre la divulgació de la seva obra, amb nom o pseudònim, el reconeixement d'autoria, el respecte a la integritat de l'obra i el dret de modificació i retirada. Els segons li donen dret a explotar-la econòmicament i poden ser cedits de

manera total o parcial, de forma exclusiva o no, a un tercer. Els drets morals són vitalicis o indefinits, mentre que els patrimonials tenen una durada bastant llarga (setanta anys després de la mort de l'autor, si és una persona física, en el cas de la llei espanyola).

La cessió de drets s'especifica mitjançant un contracte denominat *llicència*. En el cas de programes privatius, aquests generalment es distribueixen per mitjà de llicències d'ús "no exclusiu", que s'entén que accepten automàticament en obrir o instal·lar el producte. No és necessari, doncs, firmar el contracte, ja que, en el cas que no l'accepti el receptor, en regeixen automàticament els drets per omissió de la llei, és a dir, cap. Les llicències no poden restringir alguns drets que atorga la legislació vigent, com el de fer còpies privades d'art o música, la qual cosa permet regalar una còpia d'un enregistrament a un amic, però aquest dret no és aplicable als programes. Segons l'LPI de 1996 (Llei de propietat intel·lectual, Reial decret legislatiu 1/1996, de 12 d'abril) [77], modificada el 2006 (Llei de propietat intel·lectual, Llei 23/2006, de 7 de juliol) [79], respecte dels programes sempre es pot fer una còpia de seguretat, es poden estudiar per a fer programes interoperables i es poden corregir i adaptar a les nostres necessitats (cosa difícil, perquè no solem disposar dels codis font). Aquests drets no poden ser restringits per llicències, encara que les lleis estan en procés de revisió, en una tendència aparentment imparable a reduir els drets dels usuaris. Les recopilacions organitzades d'obres o dades alienes també estan sotmeses a drets d'autor, si bé els termes són diferents i la durada menor.

Les noves tecnologies de la informació, i en especial la Xarxa, han trastocat profundament la protecció dels drets d'autor, ja que les expressions de continguts són molt més fàcils de copiar que els continguts mateixos. I en el cas dels programes i algunes obres d'art (música, imatges, pel·lícules, i fins i tot literatura), "funcionen" automàticament a l'ordinador, sense necessitat d'un esforç humà apreciable. En canvi, els dissenys o invents s'han de construir i possiblement posar en producció. Aquesta possibilitat de crear riquesa sense cost ha portat gran part de la societat, en particular els països pobres, a duplicar programes sense pagar llicència, sense que hi hagi una consciència social que això sigui una "mala acció" (com sí que n'hi sol haver respecte del robatori de béns físics, per exemple). D'altra banda, els fabricants de programes, sols o en coalició (per exemple la BSA, Business Software Alliance), pressionen fortament perquè les llicències es paguin i els governs persegueixin el que s'ha convingut a anomenar *pirateria*.

Nota

La paraula *pirateria* s'ha popularitzat com a sinònim de 'violació de qualsevol forma de propietat intel·lectual, especialment en el cas de la còpia il·legal de programes, música i pel·lícules.' El terme sembla exagerat, i al diccionari de la Reial Acadèmia Espanyola de la Llengua apareix com una accepció en sentit figurat, ja que el terme original es refereix a 'robatori amb violència al mar'. És per això que Richard Stallman recomana evitar-lo ("Some confusing or loaded words and phrases that are worth avoiding", 2003) [212].

Precisament per protegir els drets d'autor d'aquells continguts amb llicències privatives, neixen els anomenats sistemes DRM (*digital rights management*, 'gestió de drets digitals'), a fi de controlar l'accés i la utilització de dades en suport digital o de restringir-ne l'ús a certs dispositius. L'ús de sistemes DRM s'ha criticat fortament en molts sectors, ja que tracta de protegir drets d'autor imposant restriccions més enllà de les suficients, per la qual cosa alguns, com la Free Software Foundation, recomanen interpretar les sigles com a *digital restrictions management* ('gestió de restriccions digitals'), intentant evitar la utilització de la paraula *drets* (en anglès, *rights*), en considerar que es priven excessius drets dels usuaris per aconseguir satisfer els drets dels autors.

3.1.2. Secret comercial

Un altre dels recursos que tenen les empreses per a rendibilitzar les seves inversions és el secret comercial, protegit per les lleis de propietat industrial, sempre que les empreses prenguin les mesures suficients per a ocultar la informació que no volen revelar. En el cas de productes químics o farmacèutics que requereixin aprovació governamental, l'Estat es compromet a no revelar les dades lliurades que no sigui obligatori fer públiques.

Una de les aplicacions més conegudes del secret comercial es troba en la indústria del programari propietari, que generalment comercialitza programes compilats sense donar accés al codi font, per així impedir el desenvolupament de programes derivats.

A primera vista sembla que la protecció del secret comercial és perversa, ja que pot privar indefinidament la societat de coneixements útils. En certa manera així ho entenen algunes legislacions, que permeten l'enginyeria inversa per a desenvolupar productes substituts, encara que la pressió de les indústries ha aconseguit que a molts països aquesta sigui una activitat prohibida i en d'altres només estigui permesa en nom de la compatibilitat.

Sigui pervers o no el secret comercial, en molts casos és millor que una patent, ja que dóna un avantatge competitiu al que posa un producte al mercat mentre la competència tracta d'imitar-lo amb enginyeria inversa. Com més sofisticat sigui el producte, més costarà a la competència reproduir-lo, mentre que si és trivial, el copiarà ràpidament. La imitació amb millores ha estat fonamental per al desenvolupament de les que avui són superpotències (Els Estats Units i el Japó) i és molt important per a la independència econòmica dels països en vies de desenvolupament.

3.1.3. Patents i models d'utilitat

L'alternativa al secret comercial és la patent. A canvi d'un monopoli de disset a vint-i-cinc anys i un determinat cost econòmic, un *invent* és revelat públicament, de manera que sigui fàcilment reproduïble. Amb ella es pretén promou-

re la investigació privada, sense cost per al contribuent i sense que el resultat es perdi. El posseïdor d'una patent pot decidir si permet a d'altres utilitzar-la i el preu que ha de pagar per la llicència.

La doctrina oficial és que el sistema de patents promou la innovació, però cada vegada més es fan sentir veus que afirmen que la dificulta, o bé perquè opinen que el sistema està mal implementat, o bé perquè creuen que és pervers en si mateix (François-René Rideau,"Patents are an economic absurdity", 2000) [194].

El que es considera un invent ha anat variant amb el temps, i existeixen grans pressions per a ampliar la cobertura del sistema, que inclouen algoritmes, programes, models de negoci, substàncies naturals, gens i formes de vida, incloses plantes i animals. TRIPS exigeix que el sistema de patents no discrimini cap àmbit del saber. Les pressions de l'Organització Mundial de la Propietat Intel·lectual (OMPI o WIPO) pretenen eliminar la necessitat que l'invent tingui aplicació industrial, i també rebaixar els estàndards d'inventiva exigibles en una patent. Els Estats Units està al capdavant dels països amb un mínim d'exigències sobre el que és patentable, i és també el més bel·ligerant perquè altres països adoptin els seus estàndards, sense recordar que ell mateix es va negar a acceptar les patents estrangeres quan era un país subdesenvolupat.

Una vegada obtinguda una patent, els drets del posseïdor són independents de la qualitat de l'invent i de l'esforç invertit a obtenir-lo. Atès el cost de manteniment d'una patent i els costos de litigació, només les grans empreses poden mantenir i mantenen una àmplia cartera de patents que les situen en una posició que els permet ofegar qualsevol competència. Atesa la facilitat per a col·locar patents sobre solucions trivials o de gran aplicabilitat, poden monopolitzar per a elles mateixes un espai molt ampli d'activitat econòmica.

Amb patents, moltes activitats, especialment la programació, es fan extremadament arriscades, ja que és molt fàcil que en el desenvolupament d'un programa complicat es violi accidentalment alguna patent. Quan dues o més empreses estan investigant per resoldre un problema, és molt probable que arribin a una solució similar gairebé alhora, però només una (generalment la que tingui més recursos) aconseguirà patentar el seu invent, de manera que les altres perdran tota possibilitat de rendibilitzar la seva inversió. Tot desenvolupament tècnic complex pot convertir-se en un malson si per a cada una de les solucions de les seves parts és necessari investigar si la solució trobada està patentada (o en tràmit), per a intentar obtenir la llicència o per a buscar una solució alternativa. Aquest problema és especialment greu en el programari lliure, on les violacions de patents d'algoritmes són evidents per simple inspecció del codi.

Encara que a Europa encara és il·legal patentar un algoritme, es podrà fer en un termini molt breu, potser quan el lector llegeixi aquestes línies.

3.1.4. Marques i logotips registrats

Les marques i logotips són noms i símbols que representen un patrimoni de qualitat (o una gran inversió en publicitat). No tenen gran importància dins del programari lliure, possiblement perquè registrar-los té un cost. Així, només alguns noms importants, com Open Source (per Open Source Foundation), Debian (per Software in the Public Interest), GNOME (per GNOME Foundation), GNU (per Free Software Foundation) o OpenOffice.org (per SUN Microsystems) estan registrats, i només en alguns països. Tanmateix, el no-registre de noms ha provocat problemes. Per exemple, als Estats Units (1996) i a Corea (1997) hi ha hagut persones que han registrat el nom Linux i han demandat diners per al seu ús. La resolució d'aquestes disputes suposa costos legals i la necessitat de demostrar un ús del nom anterior a la data del registre.

3.2. Llicències en el programari lliure

Legalment parlant, la situació dels programes lliures respecte dels privatius no és gaire diferent: també es distribueixen sota llicència. El que els diferencia és precisament el que permet aquesta llicència. En el cas de les llicències de programes lliures, que no restringeixen precisament l'ús, la redistribució i la modificació, el que poden imposar són condicions que cal satisfer precisament en cas que es vulgui redistribuir el programa. Per exemple, poden exigir que es respectin les indicacions d'autoria o que s'inclogui el codi font si es vol redistribuir el programa llest per executar.

Encara que en essència *programari lliure* i *programari propietari* es diferenciïn en la llicència amb què els autors publiquen els seus programes, és important posar l'accent en el fet que aquesta diferència es reflecteix en condicions d'ús i redistribució totalment diferents. Com s'ha vist al llarg dels últims anys, això ha originat no només mètodes de desenvolupament totalment diferents, sinó fins i tot formes pràcticament oposades (en molts sentits) d'entendre la informàtica.

Les lleis sobre propietat intel·lectual asseguren que en absència de permís explícit no es pot fer gairebé res amb una obra (en el nostre cas, un programa) que es rebi o es compri. Només l'autor (o qui tingui els drets de l'obra) ens pot donar aquest permís. En qualsevol cas, la propietat de l'obra no canvia per atorgar una llicència, ja que aquesta no suposa transferència de propietat, sinó només de dret d'ús i, en alguns casos (obligats en el programari lliure), de distribució i modificació. Les llicències de programari lliure es diferencien de les privatives precisament que en lloc de restringir acuradament el que es permet, atorguen certs permisos explícits. Quan un rep un programa lliure pot redistribuir-lo o no, però si el redistribueix, només pot fer-ho perquè la llicència li ho permet. Però per a això cal complir la llicència. En definitiva, la llicència conté les normes d'ús a què han d'atenir-se usuaris, distribuïdors, integradors i altres parts implicades en el món de la informàtica.

Per comprendre plenament totes le dificultats legals que es presentaran en aquest capítol (i que, sens dubte, són molt importants per a entendre la naturalesa del programari lliure), també és necessari saber que cada nova versió d'un programa és considerada com una nova obra. L'autor té, una altra vegada, plena potestat per a fer amb ella el que li vingui de gust, fins i tot distribuir-la en termes i condicions totalment diferents (és a dir, amb una llicència diferent de l'anterior). Així, si el lector és autor únic d'un programa podrà publicar una versió sota una llicència de programari lliure i, si li vingués de gust, una altra de posterior sota una llicència propietària. En cas que hi hagi més autors i que la nova versió contingui codi l'autoria del qual els correspongui, si es publica sota altres condicions, tots ells hauran de donar el vistiplau al canvi de llicència.

Un tema encara relativament obert és la llicència que s'aplica a les contribucions externes. Generalment se suposa que una persona que contribueixi al projecte accepta *de facto* que la seva contribució s'ajusti a les condicions especificades per la seva llicència, encara que això podria tenir poc fonament jurídic. La iniciativa de la Free Software Foundation de demanar mitjançant carta (física) la cessió de tots els drets de *copyright* a qualsevol que contribueixi amb més de deu línies de codi a un subprojecte de GNU és una bona mostra que al món del programari lliure hi ha polítiques més estrictes respecte a aquestes contribucions.

Partint de tot el que s'ha dit, ens centrarem ja en la resta d'aquest capítol en l'anàlisi de diverses llicències. Per posar en context aquest estudi, cal recordar que d'ara endavant, quan diem que una llicència és de programari lliure, ho diem en el sentit que compleix les definicions de *programari lliure* presentades a l'apartat 1.1.1.

3.2.1. Tipus de llicències

La varietat de llicències lliures és gran, encara que per raons pràctiques la majoria dels projectes utilitzen un petit conjunt de quatre o cinc. D'una banda, molts projectes no volen o no poden dedicar recursos a dissenyar una llicència pròpia; per una altra, la majoria dels usuaris prefereixen referir-se a una llicència àmpliament coneguda que llegir-se llicències completes i analitzar-les.

Bibliografia

Es poden veure recopilades i comentades tant llicències considerades lliures com llicències considerades no lliures o lliures però incompatibles amb la GPL des del punt de vista de la FSF Free Software Foundation, "Licencias libres" [121]. El punt de vista filosòficament diferent de l'Open Source Initiative es reflecteix al seu llistat (Open Source Initiative, "Licencias de fuente abierta") [181]. Poden veure's discrepàncies en algunes llicències, com l'Apple Public Source License versió 1.2, considerada no lliure per l'FSF per l'obligació de publicar tots els canvis (encara que siguin privats), de notificar a Apple les redistribucions, o per la possibilitat de revocació unilateral. No obstant això, la pressió d'aquesta classificació va fer que Apple publiqués la versió 2.0 l'agost del 2003, ja considerada lliure per l'FSF.

És possible dividir les llicències de programari lliure en dues grans famílies. La primera està composta per les llicències que no imposen condicions especials en la *segona redistribució* (això és, que només especifiquen que el programari es pot redistribuir o modificar, però que no imposen condicions especials si es fa, la qual cosa permet, per exemple, que algú que rebi el programa pugui després redistribuir-lo com a programari propietari): són les que anomenarem *llicències permissives*. La segona família, que denominarem *llicències robustes* (o *llicències* copyleft), inclou les que, a l'estil de la GNU GPL, imposen condicions en cas que es vulgui redistribuir el programari, condicions que van en la línia de forçar que es continuïn complint les condicions de la llicència després de la primera *redistribució*. Mentre que el primer grup fa èmfasi en la llibertat de qui rep un programa, que li permet fer gairebé tot el que vulgui amb ell (en termes de condicions de futures redistribucions), el segon fa èmfasi en la llibertat de qualsevol que potencialment pugui rebre algun dia un treball derivat del programa, que obliga que les successives modificacions i redistribucions respectin els termes de la llicència original.

Nota

El terme *copyleft* aplicat a una llicència, emprat sobretot per la Free Software Foundation per definir-ne les seves, té implicacions similars a les de l'expressió *llicència robusta* tal com la usem en aquest text.

La diferència entre aquests dos tipus de llicències ha estat (i és) tema de debat en la comunitat del programari lliure. En qualsevol cas, és convenient recordar que totes són llicències lliures.

3.2.2. Llicències permissives

Les llicències permissives, de vegades també anomenades *llicències liberals* o *minimalistes*, no imposen pràcticament cap condició sobre qui rep el programari, i tanmateix, li donen permís d'ús, redistribució i modificació. Des d'un cert punt de vista, aquest enfocament pot entendre's com la garantia de les màximes llibertats per a qui rep un programa. Però des d'un altre, pot entendre's també com la màxima despreocupació respecte al fet que, una vegada rebut un programa per part d'algú, es continuïn garantint les mateixes llibertats quan aquest programa es redistribueixi. De fet, típicament aquestes llicències permeten que es redistribueixi amb llicència privativa un programari l'autor del qual distribueix amb llicència permissiva.

Entre aquestes llicències, una de les més conegudes és la llicència BSD, fins al punt que moltes vegades s'anomena les llicències permissives *llicències de tipus BSD*. La llicència BSD (Berkeley Software Distribution) té el seu origen en la publicació de versions de Unix realitzades per la universitat californiana de Berkeley, als EUA. L'única obligació que exigeix és donar crèdit als autors, mentre que permet tant la redistribució binària com la dels codis font, encara que en cap cas no obliga cap de les dues. Així mateix, dóna permís per realitzar-hi modificacions i ser integrada amb altres programes gairebé sense restriccions.

Nota

Una de les conseqüències pràctiques de les llicències de tipus BSD ha estat la difusió d'estàndards, ja que els desenvolupadors no troben cap obstacle per realitzar programes compatibles amb una implementació de referència sota aquest tipus de llicències. De fet, aquesta és una de les raons de l'extraordinària i ràpida difusió dels protocols d'Internet i de la interfície de programació basada en sòcols (*sockets*), perquè la majoria dels desenvolupadors comercials va derivar la seva realització de la realització de la Universitat de Berkeley.

Les llicències permissives són bastant populars, i existeix tota una família amb característiques similars a la BSD: X Window, Tcl/Tk, Apache, etc. Històricament aquestes llicències van aparèixer a causa que el programari corresponent va ser creat en universitats amb projectes d'investigació finançats pel Govern dels Estats Units. Aquestes universitats prescindien de la comercialització d'aquests programes, assumint que ja havien estat pagats prèviament pel Govern, i per tant, amb els impostos de tots els contribuents, per la qual cosa qualsevol empresa o particular podia utilitzar el programari gairebé sense restriccions.

Com ja s'ha comentat, a partir d'un programa distribuït sota una llicència permissiva pot crear-se'n un altre (en realitat, una nova versió) que sigui privatiu. Els crítics de les llicències BSD veuen en aquesta característica un perill, ja que no es garanteix la llibertat de versions futures dels programes. Els seus partidaris, al contrari, la consideren la màxima expressió de la llibertat i argumenten que, al cap i a la fi, es pot fer (gairebé) tot el que es vulgui amb el programari.

La majoria de les llicències permissives són una còpia calcada de l'original de Berkeley en la qual es modifica tot el que fa referència a l'autoria. En alguns casos, com la llicència del projecte Apache, inclouen alguna clàusula addicional, com la impossibilitat d'anomenar les versions redistribuïdes de la mateixa manera que a l'original. Totes aquestes llicències solen incloure, com la BSD, la prohibició d'usar el nom del propietari dels drets per a promocionar productes derivats.

Així mateix, totes les llicències, siguin de tipus BSD o no, inclouen una *limitació de garantia* que és en realitat una *negació de garantia*, necessària per a evitar demandes legals per garanties implícites. Encara que s'ha criticat molt aquesta negació de garantia en el programari lliure, és pràctica habitual en el programari propietari, que generalment només garanteix que el suport és correcte i que el programa en qüestió s'executa.

Esquema resum de la llicència BSD

Copyright © *el propietari*. Tots els drets reservats.

Es permet la redistribució en font i en binari, amb modificació o sense, sempre que es compleixin les condicions següents:

1) Les redistribucions en font han de retenir la nota de *copyright* i fer una llista d'aquestes condicions i la limitació de garantia.

2) Les redistribucions en binari han de reproduir la nota de *copyright* i fer una llista d'aquestes condicions i la limitació de garantia en la documentació.

3) Ni el nom del *propietari* ni el dels que hi han contribuït no poden utilitzar-se sense permís per a promocionar productes derivats d'aquest programa.

Aquest programa es proporciona "tal qual", sense garanties expresses ni implícites, com ara la seva aplicabilitat comercial o la seva adequació per a un propòsit determinat. En cap cas *el propietari* serà responsable de cap dany causat pel seu ús (inclosa la pèrdua de dades, la pèrdua de beneficis o la interrupció de negoci).

A continuació descrivim breument algunes llicències permissives:

- Llicència d'X Window, versió 11 (X11) (http://www.x.org/Downloads_terms.html) [73].
  És la llicència usada per a la distribució del sistema X Window, el sistema de finestres més àmpliament utilitzat al món Unix, i també en entorns GNU/Linux. És molt similar a la llicència BSD, que permet redistribució, ús i modificació pràcticament sense restriccions. De vegades se l'anomena *llicència MIT* (amb perillosa poca precisió, perquè el MIT ha fet servir altres tipus de llicències). Sota aquesta llicència es distribueixen també treballs derivats d'X Windows, com XFree86.

- Zope Public License 2.0 (http://www.zope.org/Resources/ZPL) [76].
  Aquesta llicència (habitualment anomenada ZPL) s'usa per a la distribució de Zope (un servidor d'aplicacions) i altres productes relacionats. És similar a la BSD, amb l'interessant detall que prohibeix expressament l'ús de marques registrades per Zope Corporation.

- Llicència d'Apache.
  És una llicència sota la qual es distribueixen la major part dels programes produïts pel projecte Apache. És similar a la llicència BSD.

Hi ha alguns programes lliures que no es distribueixen amb una llicència específica, sinó que el seu autor els declara explícitament *public domain* ('de domini públic o del comú'). La principal conseqüència d'aquesta declaració és que l'autor renuncia a tots els seus drets sobre el programa, que per tant, es pot modificar, redistribuir, utilitzar, etc. de qualsevol manera. A efectes pràctics, és molt similar a què el programa estigui sota una llicència de tipus BSD.

3.2.3. Llicències robustes

La Llicència Pública General de GNU (GNU GPL)

La Llicència Pública General del projecte GNU (Free Software Foundation, 1991) [118] (més coneguda pel seu acrònim en anglès, GPL), que mostrem traduïda en l'apèndix C, és amb diferència la més popular i coneguda de totes les del món del programari lliure. La seva autoria correspon a la Free Software Foundation (promotora del projecte GNU), i al principi va ser creada per a ser la llicència de tot el programari generat per l'FSF. Tanmateix, la seva utilització ha anat més enllà fins a convertir-se en la llicència més utilitzada (per exemple, més del 70% dels projectes anunciats a Freshmeat estan llicenciats sota la GPL), fins i tot per projectes bandera del món del programari lliure, com el nucli Linux.

La llicència GPL és interessant des del punt de vista legal perquè fa un ús molt creatiu de la legislació de *copyright*, i aconsegueix efectes pràcticament contraris als que se suposen de l'aplicació d'aquesta legislació: en lloc de limitar els drets dels usuaris, els garanteix. Per aquest motiu, en molts casos es denomina aquesta "maniobra" *copyleft* (joc de paraules en anglès que es pot traduir com a 'esquerres d'autor'). Algú amb una mica d'humor va arribar fins i tot a llançar l'eslògan "copyleft, all rights reversed".

En línies bàsiques, la llicència GPL permet la redistribució binària i la del codi font, encara que en el cas que redistribueixi de manera binària obliga que també es pugui accedir als codis font. Així mateix, està permès realitzar modificacions sense restriccions. Tanmateix, només es pot redistribuir codi llicenciat sota GPL de forma integrada amb un altre codi (per exemple, mitjançant enllaços o *links*) si aquest té una llicència compatible. Això s'ha anomenat *efecte viral* (encara que molts consideren despectiva aquesta denominació) de la GPL, ja que un codi publicat una vegada amb aquestes condicions mai no pot canviar-les.

Nota

Una llicència és incompatible amb la GPL quan restringeix algun dels drets que la GPL garanteix, o bé explícitament, contradient alguna clàusula, o bé implícitament, imposant-ne alguna de nova. Per exemple, la llicència BSD actual és compatible, però la d'Apache, que exigeix que s'esmenti explícitament en els materials de publicitat que el treball combinat conté codi de tots i cada un dels titulars de drets, és incompatible. Això no implica que no es puguin emprar simultàniament programes amb ambdues llicències, o fins i tot integrar-los. Només suposa que aquests programes integrats no es poden distribuir, ja que és impossible complir simultàniament les condicions de redistribució d'ambdós.

La llicència GPL està pensada per a assegurar la llibertat del codi tothora, ja que un programa publicat i llicenciat sota les seves condicions mai no podrà ser privatiu. És més, ni aquest programa ni modificacions d'aquest poden ser publicats amb una llicència diferent de la mateixa GPL. Com ja s'ha dit, els partidaris de les llicències de tipus BSD veuen en aquesta clàusula un retall de la llibertat, mentre que els seus seguidors creuen que és una forma que d'assegurar que aquest programari sempre serà lliure. D'altra banda, es pot considerar que

la llicència GPL maximitza les llibertats dels usuaris, mentre que les de tipus BSD maximitzen les dels desenvolupadors. Cal observar, tanmateix, que en el segon cas estem parlant dels desenvolupadors en general i no dels autors, ja que molts autors consideren que la llicència GPL és més beneficiosa per als seus interessos, ja que obliga els seus competidors a publicar les seves modificacions (millores, correccions, etc.) en cas que redistribueixin el seu programari, mentre que amb una llicència de tipus BSD aquest no ha de ser pas el cas.

Quant a la naturalesa contrària al *copyright* d'aquesta llicència, es deu al fet que la seva filosofia (i la de la Free Software Foundation) és que el programari no ha de tenir propietaris (Richard Stallman, "Why software should not have owners", 1998) [207]. Encara que és cert que el programari llicenciat amb la GPL té un autor, que és el que al cap i a la fi permet l'aplicació de la legislació de *copyright* sobre aquest programari, les condicions sota les quals el publica li confereixen tal caràcter que podem considerar que la propietat del programari correspon a qui el té i no a qui l'ha creat.

Sens dubte, aquesta llicència també inclou *negacions de garantia* per protegir els autors. Així mateix, i per preservar la bona fama dels autors originals, tota modificació d'un fitxer font ha d'incloure una nota en la qual s'especifiqui la data i l'autor de la modificació.

La GPL té en compte també les patents de programari, i exigeix que si el codi porta algoritmes patentats (com hem dit, una cosa legal i usual als Estats Units i pràctica irregular a Europa), o es concedeix llicència d'ús de la patent lliure de taxes, o no es pot distribuir sota la GPL.

L'última versió de la llicència GPL, la segona, es va publicar el 1991 (encara que en el moment d'escriure aquest text està en procés de preparació avançat la tercera). Precisament tenint en compte futures versions, la llicència recomana llicenciar sota les condicions de la segona o de qualsevol altra de posterior publicada per la Free Software Foundation, cosa que fan molts autors. Tanmateix, altres, entre els quals destaca Linus Torvalds (creador de Linux), publiquen el seu programari només sota les condicions de la segona versió de la GPL, buscant desmarcar-se de les possibles evolucions futures de la Free Software Foundation.

La tercera versió de la GPL (http://gplv3.fsf.org) [115] pretén portar-la a l'escenari actual del programari, principalment en aspectes com patents, sistemes DRM (*digital rights management*, 'gestió de drets digitals') i altres limitacions de la llibertat del programari. Per exemple, a l'esborrany disponible en el moment d'escriure aquest text (maig del 2007), no permet que un fabricant de maquinari bloquegi la utilització de certs mòduls de programari si no presenten una firma digital que acrediti una determinada autoria. Un exemple d'aquestes pràctiques es dóna en els gravadors digitals de vídeo TiVo, que pro-

porcionen el codi font de tot el seu programari (llicenciat amb GPLv2), mentre que no permeten que s'utilitzin modificacions del codi en l'esmentat maquinari[4].

La llicència tampoc no permet que el programari obligui a l'execució en entorn prefixat, com passa quan es prohibeix la utilització de nuclis no firmats en distribucions la política de seguretat de les quals el consideri oportú.

(4)Aquest cas ha arribat fins i tot a suggerir la denominació de *tivoisation* per a altres gravadors similars que han sorgit.

Nota

Hi ha diversos punts en la llicència GPLv3 que han despertat una certa oposició. Un dels grups d'opositors està compost per desenvolupadors del nucli Linux (entre ells el mateix Linus Torvalds). Consideren que el requisit d'utilització de components de programari firmats permet atorgar certes característiques de seguretat impossibles d'una altra manera, mentre la seva prohibició explícita estendria la llicència al terreny del maquinari. A més, la limitació establerta pel mecanisme de firmes es donaria únicament a les plataformes de maquinari i programari així dissenyades, de manera que es podria modificar el programari per a la seva utilització en un altre maquinari. Respecte a aquest punt, l'FSF està a favor de l'ús de mecanismes de firmes que recomanin la no-utilització de components no firmats per motius de seguretat, però creu que la no-prohibició d'aquells mecanismes de firmes que impossibiliten la utilització de components no firmats podrien donar lloc a escenaris en els quals no hi hagués plataformes de maquinari o programari en les quals executar les esmentades modificacions del programari, per la qual cosa en aquest cas quedarien totalment limitades les llibertats del programari lliure pel que fa a la modificació del codi.

La Llicència Pública General Menor de GNU (GNU LGPL)

La Llicència Pública General Menor del projecte GNU (Free Software Foundation, GNU Lesser General Public License, versió 2.1, febrer 1999) [119], comunament coneguda per les seves inicials en anglès, LGPL, és l'altra llicència de la Free Software Foundation. Pensada al principi per al seu ús en biblioteques (la *L*, en els seus inicis, venia de *library*, 'biblioteca'), va ser modificada recentment perquè fos considerada la germana menor (*lesser*, 'menor') de la GPL.

La LGPL permet l'ús de programes lliures amb programari propietari. El programa per si mateix es redistribueix com si fos sota la llicència GPL, però se'n permet la integració amb qualsevol altre paquet de programari sense pràcticament limitacions.

Com es pot veure, al principi aquesta llicència estava orientada a les biblioteques, perquè se'n pogués potenciar l'ús i desenvolupament sense tenir els problemes d'integració que implica la GPL. Tanmateix, quan es va veure que l'efecte buscat de popularitzar les biblioteques lliures no es veia compensat per la generació de programes lliures, la Free Software Foundation va decidir el canvi de *library* a *lesser* i va desaconsellar-ne l'ús, llevat per a condicions molt puntuals i especials. Avui en dia, hi ha molts programes que no són biblioteques llicenciats sota les condicions de l'LGPL. Per exemple, el navegador Mozilla o el paquet ofimàtic (*suite*) OpenOffice.org estan llicenciats, entre altres, també sota l'LGPL.

Nota

Igual que passa amb la GPL, l'última versió publicada de l'LGPL és la segona, encara que ja hi ha un esborrany de la tercera versió (http://gplv3.fsf.org/pipermail/info-gplv3/2006-July/000008.html) [116]. Aquesta nova versió és més curta que l'anterior, ja que refereix tot el seu text a la GPLv3 i destaca únicament les seves diferències.

Altres llicències robustes

Altres llicències robustes que pot resultar interessant comentar són les següents:

- Llicència de Sleepycat (www.sleepycat.com/download/oslicense.html) [59].
  És la llicència sota la qual l'empresa Sleepycat (www.sleepycat.com/) [60] distribueix els seus programes (entre els quals destaca el conegut Berkeley DB). Obliga a certes condicions sempre que es redistribueixi el programa o treballs que se'n derivin. En particular, obliga a oferir el codi font (incloses les modificacions si es tracta d'un treball derivat) i que la redistribució imposi al receptor les mateixes condicions. Encara que molt més curta que la GNU GPL, és molt similar a aquesta en els seus efectes principals.

- eCos License 2.0 (http://www.gnu.org/licenses/ecos-license.html) [25].
  És la llicència sota la qual es distribueix eCos (http://sources.redhat.com/ecos/) [24], un sistema operatiu en temps real. És una modificació de la GNU GPL que no considera que el codi que s'enllaci amb programes protegits per ella quedin subjectes a les clàusules de la GNU GPL si es redistribueixen. Des d'aquest punt de vista, els seus efectes són similars als de la GNU LGPL.

- Affero General Public License (http://www.affero.org/oagpl.html) [78].
  És una interessant modificació de la GNU GPL que considera el cas dels programes que ofereixen serveis via web, o en general, via xarxes d'ordinadors. Aquest tipus de programes plantegen un problema des del punt de vista de les llicències robustes. Com que l'ús del programa no implica haver-lo rebut mitjançant una redistribució, encara que estigui llicenciat, per exemple, sota la GNU GPL, algú pot modificar-lo i oferir un servei a la Xarxa usant-lo, sense redistribuir-lo de cap manera, i per tant, estar obligat, per exemple, a distribuir el seu codi font. L'Affero GPL té una clàusula que obliga que, si el programa té un mitjà per a proporcionar el seu codi font via web a qui l'usi, no es pugui desactivar aquesta característica. Això significa que si l'autor original inclou aquesta capacitat en el codi font, qualsevol usuari pot obtenir-lo, i a més aquesta *redistribució* està sotmesa a les condicions de la llicència. La Free Software Foundation es planteja incloure provisions similars en la versió 3 del seu GNU GPL.

- IBM Public License 1.0 (http://oss.software.ibm.com/developerworks/opensource/license10.html) [40].

És una llicència que permet la redistribució binària de treballs derivats només si (entre altres condicions) es preveu algun mecanisme perquè qui rebi el programa pugui rebre el seu codi font. La redistribució del codi font s'ha de fer sota la mateixa llicència. A més, aquesta llicència és interessant perquè obliga al qui redistribueix el programa amb modificacions a llicenciar automàticament i gratuïtament les patents que puguin afectar aquestes modificacions, i que siguin propietat del redistribuïdor, a qui rebi el programa.

- Mozilla Public License 1.1 (http://www.mozilla.org/MPL/MPL-1.1.html) [49].
  Es tracta d'un exemple de llicència lliure amb origen en una empresa. És una evolució de la primera llicència lliure que va tenir Netscape Navigator, que en el seu moment va ser molt important per ser la primera vegada que una empresa molt coneguda decidia distribuir un programa sota la seva pròpia llicència lliure.

3.2.4. Distribució sota diverses llicències

Fins ara s'ha donat per fet que cada programa es distribueix sota una única llicència en la qual s'especifiquen les condicions d'ús i redistribució. Tanmateix, un autor pot distribuir obres amb diferents llicències. Per entendre-ho, hem de tenir en compte que cada publicació és una nova obra, i que es pot donar el cas que es distribueixin versions que només difereixin en la llicència. Com veurem, la majoria de les vegades això es tradueix en el fet que en funció del que l'usuari vulgui fer amb el programari es trobarà que ha d'obeir una llicència o una altra.

Un dels exemples més coneguts de doble llicència és el de la biblioteca Qt, sobre la qual es basa l'entorn d'escriptori KDE. Trolltech, una empresa establerta a Noruega, distribuïa Qt amb una llicència propietària, encara que eximia del pagament als programes que en fessin ús sense ànim de lucre. Per aquesta causa i per les seves característiques tècniques, va ser elegida a mitjan dècada dels noranta pel projecte KDE. Això va suposar una àrdua polèmica amb la Free Software Foundation, ja que KDE deixava de ser llavors programari lliure en el seu conjunt, en dependre d'una biblioteca propietària. Després d'un llarg debat (durant el qual va aparèixer GNOME com a competidor lliure de KDE a l'escriptori), Trolltech va decidir utilitzar el sistema de doble llicència per al seu producte estrella: els programes sota la GPL podien fer ús d'una versió de Qt GPL, mentre que si es volien integrar amb programes amb llicències incompatibles amb la GPL (per exemple, llicències privatives), els havien de comprar una llicència especial. Aquesta solució va satisfer totes les parts, i avui dia KDE es considera programari lliure.

Altres exemples coneguts de llicència dual són StarOffice i OpenOffice.org, o Netscape Communicator i Mozilla. En ambdós casos el primer producte és propietari, mentre que el segon és una versió lliure (generalment sota les condicions de diverses llicències lliures). Encara que al principi els projectes lliures eren versions limitades dels seus germans propietaris, amb el temps han anat prenent el seu propi camí, per la qual cosa avui en dia tenen un grau d'independència bastant gran.

3.2.5. Documentació de programes

La documentació que ve amb un programa és part integrant d'aquest, igual que els comentaris del codi font, com reconeix, per exemple a Espanya, la Llei de propietat intel·lectual. Atès aquest nivell d'integració, sembla lògic que a la documentació s'apliquin les mateixes llibertats i que evolucioni de la mateixa manera que el programa: tota modificació que es faci d'un programa requereix un canvi simultani i consistent en la seva documentació.

La major part d'aquesta documentació sol estar codificada com a fitxers de text sense format, ja que es pretén que sigui universalment accessible amb un entorn d'eines mínim, i (en el cas dels programes lliures) sol incloure una petita introducció al programa (`README` o `LEEME`), instruccions d'instal·lació (`INSTALL`), alguna història sobre l'evolució passada i el futur del programa (`CHANGELOG` i `TODO`), autoria i condicions de còpia (`AUTHORS` i `COPYRIGHT` o `COPYING`), així com les instruccions d'ús. Tots ells, menys l'autoria i les condicions de còpia, haurien de ser lliurement modificables a mesura que el programa evoluciona. A l'autoria només se li haurien d'afegir noms i crèdits, però sense esborrar res, i les condicions de còpia només haurien de modificar-se si aquestes ho permeten.

Les instruccions d'ús acostumen a estar codificades en formats més complexos, ja que solen ser documents més llargs i més rics. El programari lliure exigeix que aquesta documentació pugui ser modificada fàcilment, que al seu torn obliga a usar formats denominats *transparents*, d'especificació coneguda i processables per eines lliures, com són, a més del text pur i net, els formats de pàgines de manual de Unix, TexInfo, LaTeX o DocBook, sense perjudici de distribuir també el resultat de la transformació d'aquests documents font en formats més aptes per a la visualització o impressió, com HTML, PDF o RTF (formats en general més *opacs*).

Tanmateix, moltes vegades es fa documentació sobre programes per part de tercers que no han intervingut en el desenvolupament. De vegades és documentació de caràcter didàctic que facilita la instal·lació i l'ús d'un programa concret (HOWTO, CÓMO o receptaris); de vegades és documentació més àmplia, que inclou diversos programes i la seva integració, que compara solucions, etc., o bé en forma de tutorial o bé en forma de referència; de vegades és una mera recopilació de preguntes freqüents amb les seves respostes (FAQ o PUF). És un exemple notable el Projecte de documentació Linux (http://

www.tldp.org) [44]. En aquesta categoria podem incloure també altres documents tècnics, no necessàriament sobre programes, ja siguin les instruccions per a cablar una xarxa local, per a construir una cuina solar, per a reparar un motor o per a seleccionar un proveïdor de cargols.

Aquests documents són una cosa intermèdia entre la mera documentació de programes i els articles o llibres molt tècnics i pràctics. Sens detriment de la llibertat de lectura, còpia, modificació i redistribució, l'autor pot voler abocarhi opinions que no desitja que es tergiversin, o almenys pot voler que aquestes tergiversacions no se li atribueixin; o pot voler que es conservin paràgrafs, com ara agraïments; o que necessàriament se'n modifiquin d'altres, com el títol. Encara que aquestes inquietuds poden manifestar-se també amb els programes en si mateixos, no s'han expressat amb tanta força al món del programari lliure com en el de la documentació lliure.

3.3. Resum

En aquest capítol s'han revisat els aspectes legals que regeixen o que tenen impacte sobre el programari lliure. Aquests formen part del dret de propietat intel·lectual o industrial i han estat concebuts, en principi, per a estimular la creativitat recompensant els creadors per un temps determinat. De tots ells, l'anomenat *copyright* és el que més afecta el programari lliure, i convenientment emprat, serveix per a assegurar-ne l'existència en forma de llicències lliures.

Hem pogut veure la importància que tenen les llicències dins del món del programari lliure. Així mateix, hem presentat la gran varietat de llicències existents, la seva motivació, les seves repercussions i els seus avantatges i inconvenients. En definitiva, podem dir que la GPL tracta de maximitzar les llibertats que té l'usuari del programari, tant si el rep directament del seu autor com si no, mentre que les llicències de tipus BSD el que fan és maximitzar les llibertats del modificador o redistribuïdor.

En vista del que s'ha comentat en aquest capítol, es dedueix que és molt important decidir aviat quina llicència tindrà un projecte i conèixer detalladament els seus avantatges i inconvenients, ja que una modificació posterior sol ser molt difícil, sobretot si el nombre de contribucions externes és molt gran.

Per finalitzar, volem posar èmfasi en el fet que el programari lliure i el programari propietari es diferencien de manera estricta únicament i exclusivament en la llicència amb què es publiquen els programes. En pròxims capítols veurem, tanmateix, que aquesta puntualització merament legal pot tenir conseqüències –o no– en la manera com es desenvolupa el programari i donar lloc a un nou model de desenvolupament que es diferenciï poc o molt, segons el cas, dels mètodes de desenvolupament "tradicionals" utilitzats en la indústria del programari.

4. El desenvolupador i les seves motivacions

"Being a hacker is lots of fun, but it's a kind of fun that takes lots of effort. The effort takes motivation. Successful athletes get their motivation from a kind of physical delight in making their bodies perform, in pushing themselves past their own physical limits. Similarly, to be a hacker you have to get a basic thrill from solving problems, sharpening your skills and exercising your intelligence. "

"Ser un *hacker* és molt divertit, però és un tipus de diversió que suposa molt esforç. L'esforç suposa motivació. Els esportistes d'èxit obtenen la seva motivació d'un tipus de plaer físic en fer que els seus cossos funcionin, en portar-se a si mateixos més enllà dels seus propis límits físics. De forma similar, per ser un *hacker* has d'experimentar una emoció bàsica a resoldre problemes, a afilar les teves aptituds i a exercir la teva intel·ligència".

Eric Steven Raymond,"How to become a hacker".

4.1. Introducció

El desenvolupament parcialment anònim i distribuït del programari lliure ha permès que durant molts anys els recursos humans amb què compta siguin desconeguts. Conseqüència d'aquest desconeixement ha estat la mitificació, almenys parcial, del món del programari lliure i de la vida dels que són darrere d'ell, que s'empara en tòpics més o menys estesos sobre la cultura *hacker* i els ordinadors. Des de fa uns quants anys, s'ha anat realitzant un gran esforç per part de la comunitat científica per conèixer millor a les persones que participen en projectes de programari lliure, la seva procedència, les seves motivacions, la seva preparació i altres aspectes que poguessin semblar interessants. Des del punt de vista purament pragmàtic, conèixer qui s'implica i per què en aquest tipus de projectes pot ser de gran utilitat per a la generació de programari lliure. Alguns científics, principalment economistes, psicòlegs i sociòlegs, han volgut anar més enllà i han vist en aquesta comunitat el germen de futures comunitats virtuals amb regles i jerarquies pròpies, en molts casos totalment diferents de les que coneixem en la societat "tradicional". Entre les incògnites més importants hi ha la de conèixer els motius que porten els desenvolupadors a ser partícips en una comunitat d'aquestes característiques, tenint en compte que els beneficis econòmics, almenys els directes, són pràcticament inexistents, mentre que els indirectes són difícilment quantificables.

4.2. Qui són els desenvolupadors?

Aquest apartat pretén donar una visió global de les persones que dediquen el seu temps i el seu esforç a participar en projectes de programari lliure. Les dades que es mostraran provenen majoritàriament d'estudis científics realitzats en els últims anys, entre els quals els més significatius –encara que, sens dubte, no els únics– han estat *Free/libre and open source software. Survey and study*, part IV: "Survey of developers", 2002 [126], i "Who is doing it? Knowing more about libre software developers", 2001 [197].

Els desenvolupadors de programari lliure són generalment persones joves. La mitjana d'edat està situada entorn dels vint-i-set anys. La variància d'edat és molt gran, ja que el grup predominant es troba en una forquilla que va dels vint-i-un als vint-i-quatre anys, i la moda –el valor que apareix amb major freqüència– se situa en els vint-i-tres anys. És interessant observar com l'edat d'incorporació al moviment de programari lliure té els seus màxims entre els divuit i els vint-i-cinc anys –i és especialment pronunciada entre els vint-i-un i els vint-i-tres– la qual cosa equivaldria a l'edat universitària. Aquesta evidència contrasta amb l'afirmació que el programari lliure és cosa principalment d'adolescents, encara que la presència d'aquests és evident (al voltant d'un 20% dels desenvolupadors té menys de vint anys). En definitiva, podem veure que els desenvolupadors solen ser majoritàriament de vint anys, en un 60%, mentre que els menors de vint i els majors de trenta es reparteixen a parts iguals el 40% restant.

De l'edat d'incorporació es pot deduir que existeix una gran influència universitària en el programari lliure. Això no és d'estranyar, ja que com s'ha pogut veure al capítol d'història, el programari lliure –abans fins i tot de denominar-se així– ha estat íntimament lligat a les institucions educatives superiors. Encara avui, el verdader motor de l'ús i l'expansió del programari lliure continuen sent les universitats i els grups d'usuaris estudiantils. Per tant, no és d'estranyar que més d'un 70% dels desenvolupadors compti amb una preparació universitària. La dada té major importància si tenim en compte que del 30% restant molts no són universitaris perquè encara estan en la seva fase escolar. Tot i així, també hi tenen cabuda –i no per això són menys apreciats– desenvolupadors que no han accedit mai a estudis superiors, però que són amants de la informàtica.

El desenvolupador de programari lliure és generalment home. Les xifres en les diferents enquestes sobre la presència de dones en la comunitat varien entre un 1% i un 3%, i competeixen amb el mateix marge d'error d'aquestes mateixes. D'altra banda, una majoria (60%) afirma tenir parella, mentre que el nombre de desenvolupadors amb fills només és d'un 16%. Donats els marges d'edats en els quals estan compresos els desenvolupadors de programari lliure, aquestes dades concorden bastant bé amb una mostra aleatòria, per la qual cosa es poden considerar "normals". El mite del desenvolupador solitari l'afició del qual per la informàtica és l'única cosa en la seva vida es mostra, com es pot veure, com una excepció a la regla.

4.3. Què fan els desenvolupadors?

Professionalment, els desenvolupadors de programari lliure es defineixen com a enginyers de programari (33%), estudiants (21%), programadors (11%), consultors (10%), professors d'universitat (7%), etc. En el costat contrari, podem veure que no solen integrar ni els departaments comercials ni de màrqueting (al voltant d'un 1%). És interessant observar com molts d'ells es defineixen a si mateixos com a enginyers de programari abans que programadors –gairebé tres

vegades més–, tenint en compte, com es veurà al capítol dedicat a l'enginyeria del programari, que l'aplicació de les tècniques clàssiques d'enginyeria del programari (i fins i tot algunes modernes) no sol estar molt arrelada al món del programari lliure.

El vincle universitari, que ja ha estat mostrat anteriorment, torna a aparèixer en aquest apartat. Al voltant d'un de cada tres desenvolupadors és estudiant o professor d'universitat, la qual cosa demostra que existeix una gran col·laboració entre gent provinent principalment de la indústria del programari (els dos terços restants) i l'àmbit acadèmic.

D'altra banda, també s'ha pogut constatar una gran interdisciplinarietat: un de cada cinc desenvolupadors prové de camps diferents del de les tecnologies de la informació. Això, unit al fet que existeix també un nombre similar de desenvolupadors no universitaris, reflecteix l'existència d'una gran riquesa quant a interessos, procedències i, en definitiva, composició dels equips de desenvolupament. És molt difícil trobar una indústria moderna, si és que existeix, on el grau d'heterogeneïtat sigui tan gran com el que es pot veure en el programari lliure.

A més de l'aproximadament 20% d'estudiants, els desenvolupadors solen ser en la seva gran majoria assalariats, en un 64%, mentre que el percentatge d'autònoms és del 14%. Finalment, només un 3% diu trobar-se a l'atur, dada significativa, ja que l'enquesta va ser feta després del començament de la crisi de les puntcom.

Nota

El fet que el finançament del programari lliure, al contrari del que passa amb el programari propietari, no es pugui fer mitjançant la venda de llicències ha propiciat des de sempre encesos debats al voltant de com s'han de guanyar la vida els programadors. En les enquestes que s'estan comentant en aquest capítol més d'un 50% dels desenvolupadors deien haver-se beneficiat econòmicament de manera directa o indirecta de la seva implicació en el programari lliure. Tanmateix, n'hi ha molts que no ho veuen tan clar. El mateix Richard Stallman, fundador del projecte GNU, davant de la pregunta de què és el que ha de fer un desenvolupador de programari lliure per guanyar diners, sol respondre que pot treballar com a cambrer.

4.4. Distribució geogràfica

L'obtenció de dades geogràfiques dels desenvolupadors és una qüestió que encara ha de ser abordada de manera més científica. El problema que presenten els estudis els resultats dels quals es mostren en aquest capítol és que, en tractar-se d'enquestes a Internet obertes a tot aquell que volgués participar-hi, la participació depenia molt dels llocs on s'haguessin anunciat, així com de la forma en la qual s'haguessin anunciat. Per ser estrictes, cal esmentar que les enquestes no buscaven representativitat en aquest sentit, sinó més aviat obtenir la resposta i/o l'opinió del major nombre possible de desenvolupadors de programari lliure.

Tanmateix, podem aventurar-nos a fer unes quantes afirmacions sobre això, malgrat saber que les dades no són tan fiables com les exposades anteriorment i de les quals el marge d'error és, per tant, molt més gran. El que sembla un fet constatable és que la gran majoria dels desenvolupadors de programari lliure provenen de països industrialitzats, i que és escassa la presència de desenvolupadors de països de l'anomenat Tercer Món. No és d'estranyar, per tant, que el mapa de desenvolupadors del projecte Debian (http://www.debian.org/devel/developers.loc) [187], per posar un exemple, concordi amb les fotografies de la Terra de nit: allà on hi ha llum es –llegiu "on hi ha civilització industrialitzada"– és on solen concentrar-se més els desenvolupadors de programari lliure. Això que al principi podria semblar lògic, contrasta amb les possibilitats potencials que el programari lliure ofereix per a països del Tercer Món.

En podem veure un clar exemple a la taula següent, que conté els països d'origen més freqüents per als desenvolupadors del projecte Debian al llarg dels últims quatre anys. Es pot observar una tendència a la descentralització del projecte, una cosa que es constata en el fet que el creixement dels desenvolupadors als Estats Units –el país que més aporta– és inferior a la mitjana. I és que, en general, els països han aconseguit doblar el nombre de voluntaris en els últims quatre anys, i França, que ha aconseguit multiplicar per cinc la seva presència, és l'exemple més clar en aquest sentit. Considerant que els primers passos de Debian van tenir lloc al continent americà (en particular en els Estats Units i el Canadà), podem veure que en els últims quatre anys el projecte ha sofert una *europeïtzació*. Suposem que el següent pas serà la mundialització anhelada, amb la incorporació de països sud-americans, africans i asiàtics (exceptuant Corea i el Japó, ja ben representats), encara que les dades que fem servir (dos desenvolupadors a Egipte, la Xina o l'Índia, i un a Mèxic, Turquia o Colòmbia el juny del 2003) no són gaire falagueres en aquest sentit.

Taula 1. Països amb major nombre de desenvolupadors de Debian

País	01/07/1999	01/07/2000	01/07/2001	01/07/2002	20/06/2003
Estats Units	162	169	256	278	297
Alemanya	54	58	101	121	136
Regne Unit	34	34	55	63	75
Austràlia	23	26	41	49	52
França	11	11	24	44	51
Canadà	20	22	41	47	49
Espanya	10	11	25	31	34
Japó	15	15	27	33	33
Itàlia	9	9	22	26	31
Països Baixos	14	14	27	29	29

País	01/07/1999	01/07/2000	01/07/2001	01/07/2002	20/06/2003
Suècia	13	13	20	24	27

Dins del món del programari lliure (i no només en el cas de Debian), hi ha una àmplia discussió sobre la supremacia entre Europa i els Estats Units. Gairebé tots els estudis que s'han anat realitzant mostren que la presència de desenvolupadors europeus és lleugerament superior a la nord-americana, efecte que queda mitigat pel fet que la població europea és major que la nord-americana. Ens trobem, llavors, davant d'una situació de guerra de xifres, ja que el nombre de desenvolupadors per càpita afavoreix els nord-americans, però torna a ser favorable als europeus si considerem, en comptes de les xifres de població absolutes, només aquelles persones que disposen d'accés a Internet.

Quant a països, les zones amb major implantació (en nombre de desenvolupadors dividit per població) són les del nord d'Europa (Finlàndia, Suècia, Noruega, Dinamarca i Islàndia) i les centreeuropees (Benelux, Alemanya i Txèquia), seguides d'Austràlia, el Canadà, Nova Zelanda i els Estats Units. La zona mediterrània, malgrat que és important en magnituds absolutes (a causa de la gran població que tenen França, Itàlia i Espanya), es troba, tanmateix, per sota de la mitjana.

4.5. Dedicació

El nombre d'hores que els desenvolupadors de programari lliure dediquen a aquest programari és un dels aspectes més desconeguts. Cal assenyalar, a més, que aquesta és una de les grans diferències amb el programari generat per una empresa, on tant l'equip com la dedicació de cada membre de l'equip al desenvolupament són coneguts. El temps que els desenvolupadors de programari lliure dediquen es pot prendre com una mesura indirecta del grau de professionalització. Abans de mostrar les dades de què es disposa en l'actualitat, és important fer notar que aquestes han estat obtingudes de les estimacions que els mateixos desenvolupadors han donat en diverses enquestes, per la qual cosa a la inexactitud inherent a aquest tipus de collita de dades s'hi ha d'afegir un marge d'error a causa principalment del que cada desenvolupador entengui com a temps de desenvolupament. D'aquesta forma, és segur que molts desenvolupadors no compten el temps que dediquen a llegir el correu (o potser sí) i que indiquen només el que dediquen a programar i a depurar. Per això, totes les xifres que es presenten a continuació han de prendre's amb la cura deguda.

Els estudis que s'han realitzat fins ara mostren que cada desenvolupador de programari lliure dedica de mitjana al voltant d'onze hores setmanals ("Motivation of software developers in open source projects: an internet-based survey of contributors to the Linux kernel", 2003) [143]. Tanmateix, aquesta xifra pot portar ràpidament a l'engany, ja que hi ha una gran variància en la dedicació dels desenvolupadors de programari. En l'estudi *Free/libre and open source software. Survey and study*, part IV: "Survey of developers", 2002 [126], un

22,5% dels enquestats va indicar que la seva aportació era inferior a les dues hores setmanals, xifra que pujava al 26,5% per als qui dedicaven entre dues i cinc hores setmanals; entre sis i deu hores és el temps que dedica un 21,0%, mentre que el 14,1% ho fa entre onze i vint hores setmanals; el 9,2% afirmava que el temps que dedicaven a desenvolupar programari lliure eren d'entre vint i quaranta hores setmanals, i el 7,1%, més de quaranta hores setmanals.

Taula 2. Dedicació en hores setmanals

Hores setmanals	Percentatge
Menys de 2 hores	22,5%
Entre 2 i 5 hores	26,1%
Entre 5 i 10 hores	21,0%
Entre 10 i 20 hores	14,1%
Entre 20 i 40 hores	9,2%
Més de 40 hores	7,1%

Nota

A més de poder veure la professionalització dels equips de desenvolupament de programari lliure, la dedicació en hores és un paràmetre de gran importància a l'hora de realitzar estimacions de cost i fer comparacions amb els models de desenvolupament propietaris que se segueixen en la indústria. En el programari lliure, per ara, només comptem amb productes acabats (nous lliuraments del programari, sincronització de codi nou en els sistemes de versions...) que no ens permeten conèixer quant temps ha necessitat el desenvolupador per a aconseguir-los.

L'anàlisi d'aquestes xifres ens mostra que al voltant d'un 80% dels desenvolupadors realitzen aquestes tasques en el seu temps lliure, mentre que només un de cada cinc podria considerar-se que dedica tant temps a aquesta activitat com un professional. Més endavant, al capítol d'enginyeria del programari, podrem veure com aquesta dada concorda amb la contribució dels desenvolupadors, ja que ambdós semblen seguir la llei de Pareto (*vid.* apartat 7.6).

4.6. Motivacions

S'ha especulat i es continua especulant molt sobre les motivacions que hi ha darrere del desenvolupament de programari lliure, sobretot en el seu vessant d'activitat de temps lliure (que, com hem vist, correspon a prop d'un 80% dels desenvolupadors). Com en els apartats anteriors, només comptem amb les dades de les enquestes, per la qual cosa és important entendre que es tracta del que els desenvolupadors responen, que pot ser més o menys coherent amb la realitat. Els percentatges que es mostraran a continuació superen en suma el 100% perquè es donava la possibilitat als enquestats d'elegir diverses respostes.

En qualsevol cas, de les seves respostes sembla desprendre's que la majoria vol aprendre i desenvolupar noves habilitats (prop d'un 80%) i que molts ho fan per compartir coneixements i habilitats (50%) o per participar en una nova forma de cooperació (al voltant d'un terç). La primera dada no sembla gens sorprenent, tenint en compte que un professional amb majors coneixements es troba més cotitzat que un que no en té. La segona dada, però, no és tan fàcil d'explicar, i fins i tot sembla anar en contra de l'afirmació de Nikolai Bezroukov a "A second look at the cathedral and the bazaar" (desembre, 1998) [91], que ve a dir que els líders dels projectes de programari lliure tenen bona cura de no compartir tota la informació que posseeixen per perpetuar el seu poder. Mentrestant, la tercera opció més freqüent és, sense cap dubte, fidel reflex que els mateixos desenvolupadors es mostren entusiasmats per la forma en la qual generalment es crea el programari lliure; és difícil trobar una indústria en la qual un grup de voluntaris lleument organitzats pugui plantar cara tecnològicament a les grans companyies del sector.

Encara que la teoria "clàssica" per a explicar els motius pels quals els desenvolupadors de programari lliure es dediquen a fer aportacions a aquest tipus de projectes gira entorn de la reputació i de beneficis econòmics indirectes a mitjà i llarg termini, sembla que els mateixos desenvolupadors no estan d'acord amb aquestes afirmacions. Només un 5% dels enquestats respon que desenvolupa programari lliure per guanyar diners, mentre que el nombre d'ells que ho fan per obtenir reputació puja a un 9%, lluny de les respostes que s'han presentat en el paràgraf anterior. En qualsevol cas, sembla que l'estudi de les motivacions que tenen els desenvolupadors per a entrar a formar part de la comunitat del programari lliure és una de les tasques primordials amb què s'hauran d'enfrontar sociòlegs i psicòlegs en els pròxims temps.

4.7. Lideratge

Reputació i lideratge són dues característiques amb què s'ha tractat d'explicar l'èxit del programari lliure, i en especial, el del model de basar, tal com es veurà al capítol dedicat a l'enginyeria del programari. Com hem pogut veure en un altre capítol, el dedicat a les llicències del programari, hi ha certes diferències entre les llicències de programari lliure i les seves homòlogues en el camp de la documentació. Aquestes diferències rauen en la forma en la qual es preserven l'autoria i l'opinió de l'autor –més accentuada en textos que en programes.

A *Free/lliure and open source software. Survey and study*, part IV: "Survey of developers", (2002) [126] es va incloure una pregunta en la qual s'instava els desenvolupadors a indicar quines persones d'una llista donada els eren conegudes, encara que no fos necessàriament de manera personal. Els resultats, que s'exposen a la taula 3, mostren que aquestes persones es poden aglutinar en tres grups clarament diferenciats:

Taula 3. Grau de coneixement de desenvolupadors importants

Desenvolupador	Conegut per
Linus Torvalds	96,5%
Richard Stallman	93,3%
Miguel de Icaza	82,1%
Eric Raymond	81,1%
Bruce Perens	57,7%
Jamie Zawinski	35,8%
Mathias Ettrich	34,2%
Jörg Schilling	21,5%
Marc Pesenti Gritti	5,7%
Bryan Andrews	5,6%
Guenter Bartsch	3,5%
Arpad Gereoffy	3,3%
Martin Hoffstede	2,9%
Angelo Roulini	2,6%
Sal Valliger	1,2%

- Un primer grup de gent amb clares connotacions filosoficohistòriques dins del món del programari lliure (encara que, com es pot veure, comptin també amb notables aptituds tècniques):

  1) Linus Torvalds. Creador del nucli Linux, el sistema operatiu lliure més utilitzat, i coautor de *Just for fun: the story of an accidental revolutionary* [217].

  2) Richard Stallman. Ideòleg i fundador de la Free Software Foundation i desenvolupador en diversos projectes GNU. Autor de diversos escrits molt importants dins del món del programari lliure ("Why *free software* is better than *open source*", 1998 [206], "Copyleft: pragmATIC idealism", 1998 [205], "The GNU Project" [208] i The GNU Manifesto", 1985 [117]).

  3) Miguel de Icaza. Cofundador del projecte GNOME i de Ximian Inc., i desenvolupador de part de GNOME i de MONO.

  4) Eric Raymond. Impulsor de l'Open Source Initiative, autor de "La catedral y el bazar" [192] i desenvolupador principal de Fetchmail.

5) Bruce Perens. Antic líder del projecte Debian, impulsor (convers) de l'Open Source Initiative i desenvolupador de l'eina e-fence.

6) Jamie Zawinsky. Exdesenvolupador de Mozilla i famós per una carta de 1999 en la qual deixava el projecte Mozilla argumentant que el model utilitzat no donaria mai fruits ("Resignation and postmortem", 1999) [237].

7) Mathias Ettrich. Fundador de KDE i desenvolupador de LyX i d'altres.

- Un segon grup format per desenvolupadors. Per a aquesta enquesta es van agafar els noms dels desenvolupadors principals dels sis projectes més populars a l'índex d'aplicacions de programari lliure FreshMeat. Es pot veure que (a excepció de Linus Torvalds, per motius obvis, i de Jörg Schilling) el grau de coneixement d'aquests desenvolupadors és petit:
  1) Jörg Schilling, creador de cdrecord, entre altres aplicacions.

  2) Marc Pesenti Gritti, desenvolupador principal de Galeon.

  3) Bryan Andrews, desenvolupador d'Apache Toolbox.

  4) Guenther Bartsch, creador de Xine.

  5) Arpad Gereoffy, desenvolupador d'MPEGPlayer.

- Un tercer grup compost pels noms de les tres últimes "persones" de la taula. Aquests noms van ser inventats per l'equip de l'enquesta per a poder comprovar el marge d'error de les respostes.

Dels resultats es desprenen dues coses: la primera és que el marge d'error de les respostes es pot considerar petit (menor d'un 3%), i la segona és que la majoria dels desenvolupadors de les aplicacions de programari lliure més populars són tan coneguts com persones que no existeixen. Aquesta dada pot fer pensar als qui addueixen com una de les primeres causes per desenvolupar programari lliure el fet de buscar la fama.

4.8. Resum i conclusions

Aquest capítol ha pretès donar una mica de llum sobre l'àmpliament desconegut tema de la gent que dedica el seu temps al programari lliure. En línies generals es pot afirmar que el desenvolupador de programari lliure és un home jove amb estudis universitaris (o en via d'aconseguir-los). La relació del món del programari lliure amb la universitat (estudiants i professors) és molt estreta, encara que continua predominant el desenvolupador que no té res a veure amb l'àmbit acadèmic.

Quant a la dedicació en nombre d'hores, s'ha mostrat com existeix una gran desigualtat a l'estil de la postulada en la llei de Pareto. Les motivacions dels desenvolupadors –segons ells mateixos–, lluny de ser monetàries i egocèntriques, tal com solen suggerir economistes i psicòlegs, estan més aviat centrades a compartir i aprendre. Finalment, s'ha exposat una taula dels personatges del món del programari lliure més significants (que n'incloïa altres que no ho eren tant, com hem pogut veure) i s'ha demostrat que la reputació en la gran comunitat del programari lliure sol dependre de més raons que la simple codificació d'una aplicació lliure reeixida.

5. Economia

"Res publica non dominetur."

"Les coses públiques no tenen amo." (traducció lliure)

Aparegut en un anunci d'IBM sobre Linux (2003)

En aquest capítol es tracten alguns aspectes econòmics relacionats amb el programari lliure. Començarem mostrant com es financen els projectes de programari lliure (quan efectivament es financen, ja que en molts casos es desenvolupen únicament amb treball i recursos aportats voluntàriament). A continuació exposarem els principals models de negoci que estan posant en pràctica les empreses relacionades directament amb el programari lliure. El capítol acaba amb un petit estudi sobre la relació entre el programari lliure i els monopolis en la indústria del programari.

5.1. Finançament de projectes de programari lliure

El programari lliure es desenvolupa de moltes formes diferents i amb mecanismes per a aconseguir recursos que varien molt d'un cas a l'altre. Cada projecte lliure té la seva pròpia forma de finançar-se, des del que està format completament per desenvolupadors voluntaris i utilitza només recursos cedits de manera altruista, fins al que és dut a terme per una empresa que factura el 100% dels seus costos a una entitat interessada en el desenvolupament corresponent.

En aquest apartat ens centrarem en els projectes en els quals hi ha finançament extern i no tot el treball realitzat és voluntari. En aquests casos, hi ha algun tipus de flux de capital amb origen extern al projecte que s'encarrega d'aportar recursos per al seu desenvolupament. Així, el programari lliure construït pot considerar-se, d'alguna forma, com un producte d'aquest finançament extern. És per això que és comú que sigui aquesta font externa la que decideixi (almenys parcialment) com i en què es gasten els recursos.

En cert sentit, aquest finançament extern per a projectes lliures pot considerar-se com un tipus de patrocini, encara que aquest patrocini no ha de ser pas desinteressat (i habitualment no ho és). En els següents apartats comentarem els tipus de finançament extern més habituals. Mentre el lector s'hi dedica, convé, però, no oblidar que aquesta és només una de les maneres com els projectes que construeixen programari lliure aconsegueixen recursos. Però n'hi ha d'altres, i entre aquestes la més important és (com s'ha vist al capítol 4) el treball de molts desenvolupadors voluntaris.

5.1.1. Finançament públic

Un tipus molt especial de finançament de projectes lliures és el públic. L'entitat finançadora pot ser directament un govern (local, regional, nacional o fins i tot supranacional) o una institució pública (per exemple, una fundació). En aquests casos, el finançament sol ser similar al dels projectes d'investigació i desenvolupament, i de fet és molt habitual que provingui d'entitats públiques promotores d'R+D. Normalment, l'entitat finançadora no busca recuperar la inversió (o almenys no de forma directa), encara que sol tenir objectius clars (com afavorir la creació de teixit industrial i investigador, promoure certa tecnologia o cert tipus d'aplicacions, etc.).

En la major part d'aquests casos, no es troba explícitament el finançament de productes o serveis relacionats amb programari lliure, sinó que aquesta sol ser el subproducte d'un contracte amb altres finalitats més generals. Per exemple, la Comissió Europea, dins dels seus programes d'investigació, finança projectes orientats a millorar la competitivitat europea en determinades àrees. Alguns d'aquests projectes tenen com a part dels seus objectius usar, millorar i crear programari lliure en el seu àmbit d'investigació (com a eina per a la investigació o com a producte derivat d'aquesta).

Les motivacions per a aquest tipus de finançament són molt variades, però es poden destacar les següents:

1) Científica. Aquest és el cas més habitual en projectes d'investigació finançats amb fons públics. Encara que el seu objectiu no és produir programari sinó investigar en un determinat camp (relacionat amb la informàtica o no), és molt possible que per a això calgui desenvolupar programes que s'usin com a eines necessàries per a assolir les metes del projecte. Normalment el projecte no està interessat a comercialitzar aquestes eines, o fins i tot està activament interessat que altres grups les utilitzin i les millorin. En aquests casos, és bastant habitual distribuir-les com a programari lliure. D'aquesta manera, els recursos aconseguits pel grup que realitza la investigació s'han dedicat en part a la producció d'aquest programari, per la qual cosa es pot dir que ha estat desenvolupat amb finançament públic.

2) De promoció d'estàndards. Tenir una implementació de referència és una de les millors formes de promoure un estàndard. En molts casos això suposa tenir programes que formin part d'aquesta implementació (o si l'estàndard es refereix al camp del programari, que siguin la implementació ells mateixos). Perquè la implementació de referència sigui d'utilitat en la promoció de l'estàndard, cal que estigui disponible, almenys per a comprovar la interoperabilitat per a tots els que vulguin desenvolupar productes que s'acullin a aquest estàndard. I en molts casos és convenient també que els fabricants puguin adaptar directament la implementació de referència per usar-la als seus productes si així ho desitgen. D'aquesta manera es van desenvolupar, per exemple, molts dels protocols d'Internet que avui s'han

convertit en norma universal. En aquests casos, l'alliberament d'aquestes implementacions de referència com a programari lliure pot ajudar molt a aquesta promoció. De nou, també aquí el programari lliure és un subproducte, en aquest cas de la promoció de l'estàndard. I habitualment qui s'encarrega d'aquesta promoció és una entitat pública (encara que de vegades és un consorci privat).

3) Social. El programari lliure és una eina de gran interès en la creació de la infraestructura bàsica per a la societat de la informació. Les entitats interessades a utilitzar programari lliure per a ajudar l'accés universal a aquesta societat de la informació poden finançar projectes que hi esiguin relacionats (normalment projectes de desenvolupament de noves aplicacions o d'adaptació d'altres de ja existents).

Nota

Un exemple de finançament públic amb finalitat fonamentalment social és el cas de LinEx, promogut per la Junta d'Extremadura (Extremadura, Espanya) per a promoure la societat de la informació fonamentalment en alfabetització informàtica. La Junta ha finançat el desenvolupament d'una distribució basada en Debian per aconseguir aquest objectiu. Un altre cas similar és el del finançament per part del Govern alemany de desenvolupaments de GnuPG orientats a facilitar-ne l'ús per als usuaris no experimentats, amb la idea d'afavorir la utilització del correu segur entre els seus ciutadans.

El desenvolupament de GNAT

Un cas notori de finançament públic per al desenvolupament de programari lliure va ser el del compilador GNAT. GNAT, compilador d'Ada, va ser finançat pel projecte Ada 9X del Departament de Defensa dels EUA amb la idea de disposar d'un compilador de la nova versió del llenguatge de programació Ada (la que després seria Ada 95), l'ús del qual tractava de promoure en aquella època. Una de les causes que s'havien identificat quant a l'adopció de la primera versió d'Ada (Ada 83) per les empreses de programari havia estat la tardana disposició d'un compilador del llenguatge, i el seu alt preu quan va estar finalment disponible. És per això que van tractar que no passés el mateix amb Ada 95, assegurant-se que hi hagués un compilador de manera pràcticament simultània a la publicació del nou estàndard del llenguatge.

Per aconseguir-ho, Ada 9X va contractar un projecte amb un equip de la Universitat de Nova York (NYU) per un import aproximat d'un milió d'USD per a la realització d'una "prova de concepte" de compilador d'Ada 95. Amb aquests fons, i aprofitant l'existència de GCC (el compilador de C de GNU, del qual es va aprofitar la major part del dorsal), l'equip de la NYU va construir efectivament el primer compilador d'Ada 95, que va alliberar sota la GNU GPL. El compilador va tenir tant èxit que en acabar el projecte part dels seus constructors van fundar una empresa (Ada Core Technologies), que des de llavors s'ha convertit en líder al mercat de compiladors i eines d'ajuda en la construcció de programes en Ada.

En aquest projecte és notable observar la combinació d'elements d'investigació (de fet, aquest projecte va avançar en el coneixement sobre la construcció de frontals i sistemes de temps d'execució per a compiladors de llenguatges de tipus Ada) i de promoció d'estàndards (que era l'objectiu més clar del seu finançador).

5.1.2. Finançament privat sense ànim de lucre

Aquest és un tipus de finançament, amb moltes característiques similars a les del cas anterior, que realitzen normalment fundacions o organitzacions no governamentals. La motivació directa en aquests casos sol ser produir programari lliure per al seu ús en algun àmbit que l'entitat finançadora consideri especialment rellevant, però també pot trobar-se la motivació indirecta de contri-

buir a resoldre un problema (per exemple, una fundació dedicada a promoure la investigació sobre una malaltia pot finançar la construcció d'un programa estadístic que ajudi a l'anàlisi de grups d'experimentació en els quals s'estudia aquesta malaltia).

En general, tant els motius per a realitzar aquest tipus de finançament com els seus mecanismes són molt similars als del finançament públic, encara que naturalment, estan sempre marcats pels objectius de l'entitat finançadora.

Nota

Probablement, el cas paradigmàtic de fundació que promou el desenvolupament de programari lliure és la Free Software Foundation (FSF). Des de mitjan dècada de 1980 aquesta fundació es dedica a la promoció del projecte GNU i a fomentar en general el desenvolupament del programari lliure.

Un altre cas interessant, encara que en un altre àmbit bastant diferent, és l'Open Bioinformatics Foundation. Entre els objectius d'aquesta fundació es troba promoure el desenvolupament dels programes informàtics bàsics per a la investigació a qualsevol de les branques de la bioinformàtica. I en general, realitza aquesta promoció finançant i ajudant a la construcció de programes lliures.

5.1.3. Finançament per part de qui necessita millores

Un altre tipus de finançament per al desenvolupament de programari lliure, ja no tan altruista, és el que té lloc quan algú necessita millores en un producte lliure. Per exemple, per a ús intern, una empresa pot necessitar, en un programa donat, certa funcionalitat o que certs errors siguin corregits. En aquests casos, és habitual que l'empresa en qüestió contracti el desenvolupament que necessita. Aquest desenvolupament, molt habitualment (o bé perquè l'imposa la llicència del programa modificat, o bé perquè l'empresa ho decideix així), és programari lliure.

El cas de Corel i Wine

A finals de la dècada de 1990, Corel va decidir migrar els seus productes a GNU/Linux. En aquest procés, va descobrir que un programa lliure dissenyat per a facilitar l'execució de binaris per a Windows en entorns Linux podria permetre-li molts estalvis de desenvolupament. Però per a això calia millorar-lo, fonamentalment afegint-hi l'emulació de certa funcionalitat de Windows que usaven els programes de Corel.

Per a això, Corel va contractar Macadamian, que va contribuir amb les seves millores al projecte Wine. Així, tant Corel com Wine en van sortir beneficiats.

5.1.4. Finançament amb beneficis relacionats

En aquest tipus de finançament, el que busca l'entitat finançadora és aconseguir beneficis en productes relacionats amb el programa al desenvolupament del qual aporta recursos. Normalment, en aquests casos els beneficis que obté l'empresa finançadora no són exclusius, ja que d'altres poden entrar també al mercat de la venda de productes relacionats, però o bé la quota de mercat que té és suficient perquè no li preocupi gaire repartir el pastís amb d'altres, o bé té algun avantatge competitiu clar.

Alguns exemples de productes relacionats amb un programari donat són els següents:

- Llibres. L'empresa en qüestió ven manuals, guies d'ús, textos per a cursos, etc., relacionats amb el programa lliure que ajuda a finançar. Sens dubte, altres empreses poden vendre també llibres relacionats, però normalment finançar el projecte li donarà accés abans que als seus competidors a desenvolupadors clau, o simplement li facilitarà una bona imatge de cara a la comunitat d'usuaris del programa en qüestió.

- Maquinari. Si una empresa finança el desenvolupament de sistemes lliures per a cert tipus de maquinari, pot dedicar-se a vendre amb més facilitat aquest tipus de maquinari. De nou, com que el programari desenvolupat és lliure, poden aparèixer competidors que venguin aparells del mateix tipus, usant aquests desenvolupaments però sense col·laborar en el seu finançament. Però fins i tot així, l'empresa en qüestió té diversos avantatges sobre els seus competidors, un dels quals pot ser que la seva posició com a aportadora de recursos per al projecte li permet influir per aconseguir que els desenvolupaments que es realitzin amb prioritat siguin els que més li interessin.

- CD amb programes. Probablement, el model més conegut d'aquest tipus és el de les empreses que financen certs desenvolupaments que després apliquen a la seva distribució de programari. Per exemple, tenir un bon entorn d'escriptori pot ajudar molt a vendre CD amb una certa distribució de GNU/Linux, i per tant, finançar el seu desenvolupament pot ser un bon negoci per a qui els vengui.

Cal tenir en compte que per estar en aquest apartat el finançament en qüestió ha de fer-se amb ànim de lucre, i per a això l'entitat finançadora ha de percebre algun benefici possible en aquest finançament. En els casos reals, tanmateix, és habitual que hi hagi sempre una combinació d'ànim de lucre i altruisme quan una empresa aporta recursos perquè es realitzi un programa lliure del qual espera beneficiar-se indirectament.

Nota

Un cas molt conegut d'aportació de recursos a un projecte, si bé de forma relativament indirecta, és l'ajuda que l'editorial O'Reilly presta al desenvolupament de Perl. Naturalment, no és casualitat que aquesta editorial sigui també una de les principals editores de temes relacionats amb Perl. En qualsevol cas, és obvi que O'Reilly no té l'exclusiva de l'edició de llibres d'aquest tipus, i altres editorials competeixen en aquest segment de mercat, amb èxit divers.

VA Software (als seus començaments VA Research i més tard VA Linux) ha col·laborat activament en el desenvolupament del nucli de Linux. Amb això ha aconseguit, entre altres coses, assegurar la seva continuïtat, la qual cosa era especialment crític per a ella, de cara als seus clients, quan el seu principal negoci era vendre equips amb GNU/Linux preinstal·lat.

Red Hat ha finançat el desenvolupament de molts components de GNOME, amb la qual cosa ha aconseguit fonamentalment tenir un entorn d'escriptori per a la seva distribució, cosa que ha contribuït a augmentar les seves vendes. Com en altres casos, altres fabricants de distribucions s'han beneficiat d'aquest desenvolupament, encara que n'hi hagi molts que no hagin col·laborat amb el projecte GNOME en la mateixa mesura que Red Hat (i no són pocs els que no hi han col·laborat en absolut). Malgrat això, Red Hat es beneficia de la seva contribució a GNOME.

5.1.5. Finançament com a inversió interna

Hi ha empreses que, com a part del seu model de negoci, desenvolupen directament programari lliure. Per exemple, una empresa pot decidir iniciar un nou projecte lliure en un àmbit on percebi que hi pot haver oportunitats de negoci amb la idea de rendibilitzar posteriorment aquesta inversió. Aquest model podria considerar-se una variant de l'anterior (finançament indirecte), i els "beneficis relacionats" serien els avantatges que obtindria l'empresa de la producció del programa lliure. Però en ser en aquest cas el producte lliure en si mateix el que s'espera que produeixi els beneficis, sembla convenient obrir una classificació específica.

Aquest tipus de finançament dóna lloc a diversos models de negoci. Quan s'analitzin (apartat 5.2) s'explicaran també els avantatges que normalment obté una empresa d'aquesta inversió en un projecte i quins mètodes solen utilitzar-se per a rendibilitzar-la. Però en qualsevol cas, cal destacar que de vegades pot ser que el programari en qüestió es desenvolupi simplement per a satisfer les necessitats de la mateixa empresa, i que només després es decideixi alliberar, i potser obrir, una línia de negoci que hi estigui relacionada.

Nota

Digital Creations (avui Zope Corporation) és un dels casos més coneguts d'empresa que es dedica al desenvolupament de programari lliure amb l'esperança de rendibilitzar la seva inversió. El projecte lliure en què més està invertint és Zope, un servidor d'aplicacions que està tenint un cert èxit. La seva història amb el programari lliure va començar quan la llavors Digital Creations buscava capital risc per desenvolupar el seu servidor d'aplicacions propietari, cap a 1998. Un dels grups interessats a invertir-hi (Opticality Ventures) els va posar com a condició que el producte resultant havia de ser lliure, perquè en cas contrari no veien com podria aconseguir una quota de mercat significativa. Digital Creations es va decidir per aquest camí, i pocs mesos després anunciava la primera versió de Zope (uns anys després va canviar el seu nom). Avui dia Zope Corporation està especialitzada a oferir serveis de consultoria, formació i suport per a sistemes de gestió de continguts basats en Zope, i altres productes en els quals sens dubte Zope és la pedra angular.

Ximian (abans Helix Code) és un cas ben conegut de desenvolupament d'aplicacions lliures en l'entorn empresarial. Molt lligada des dels seus orígens al projecte GNOME, Ximian ha produït sistemes de programari com Evolution (un gestor d'informació per-

sonal que té una funcionalitat relativament similar a l'oferta per Miscrosoft Outlook), Red Carpet (un sistema fàcil d'usar per a la gestió de paqueteria en un sistema operatiu) i MONO (una implementació de gran part de .NET). L'empresa va ser fundada l'octubre de 1999 i va atreure molts desenvolupadors de GNOME, que van passar a formar part del seu grup de desenvolupadors (en molts casos continuant la seva col·laboració amb el projecte GNOME). Ximian es va posicionar com una empresa d'enginyeria experta en adaptacions de GNOME, en la construcció d'aplicacions basades en GNOME, i en general, a proporcionar serveis de desenvolupament basats en programari lliure, especialment d'eines relacionades amb l'entorn d'escriptori. L'agost del 2003, Ximian va ser adquirida per Novell.

Cisco Enterprise Print System (CEPS) (http://ceps.sourceforge.net/) [17] és un sistema de gestió d'impressió per a organitzacions amb gran quantitat d'impressores. Va ser desenvolupat internament a Cisco per a satisfer les seves pròpies necessitats i va ser alliberat l'any 2000 sota la GNU GPL. És difícil saber amb seguretat els motius que va tenir Cisco per fer això, però és possible que tinguessin a veure amb la recerca de contribucions externes (informes d'error, nous controladors, pedaços, etc.). En qualsevol cas el que és clar és que, com que Cisco no tenia cap pla per a comercialitzar el producte i no se sabia gaire bé quin era el seu mercat potencial, no tenia gaire a perdre amb aquesta decisió.

5.1.6. Altres modes de finançament

Hi ha altres maneres de finançament difícils de classificar entre els anteriors. A tall d'exemple, poden destacar-se els següents:

- Utilització de mercats per a posar en contacte desenvolupadors i clients. La idea que sosté aquest mode de finançament és que, sobretot per a petits desenvolupaments, és difícil que un client que desitgi un desenvolupament concret pugui entrar en contacte amb un desenvolupador capaç d'emprendre'l de forma eficient. Per millorar aquesta situació, es postulen els mercats de desenvolupament de programari lliure, on els desenvolupadors publicarien les seves habilitats i els clients els desenvolupaments que necessiten. Un desenvolupador i un client es posen d'acord; tenim una situació similar a la ja descrita com a "finançament per part de qui necessita millores" (apartat 5.1.3).

  SourceXchange

  SourceXchange va ser un exemple de mercat que posava en contacte desenvolupadors amb els seus clients potencials. Per oferir un projecte, un client escrivia una RFP (*request for proposal*, 'petició de proposta') en la qual especificava el desenvolupament que necessitava i els recursos que estava disposat a proporcionar per a aquest desenvolupament. Aquestes RFP es publicaven al lloc. Quan un desenvolupador en veia una que li interessava, feia una oferta per a ella. Si un desenvolupador i un client es posaven d'acord en els termes del desenvolupament, començava un projecte. Normalment cada projecte estava supervisat per un *peer reviewer*, un revisor que s'encarregava d'assegurar-se que el desenvolupador complia les especificacions i que efectivament aquestes tenien sentit, que aconsellava sobre com portar endavant el projecte, etc. SourceXchange (propietat de l'empresa CollabNet) s'encarregava d'oferir el lloc, de garantir la competència dels revisors, d'assegurar-se el pagament en cas que els projectes es completessin i d'oferir eines per a fer-ne el seguiment (serveis que facturava el client). El primer projecte mitjançant per SourceXchange va acabar el març del 2000, però poc més d'un any després, l'abril del 2001, el lloc va tancar.

- Venda de bons per a finançar un projecte. Aquesta idea de finançament és similar a la dels mercats de bons als quals acudeixen les empreses, però orientada al desenvolupament de programari lliure. Té unes quantes variants, però una de les més conegudes funciona com segueix. Quan un desenvolupador (individual o empresa) té idea d'un nou programa o de la

millora d'un d'existent, ho escriu en forma d'especificació, estima el cost que tindria el seu desenvolupament i emet bons per fer-ne la construcció. Aquests bons tenen un valor que s'executa només si el projecte s'acaba finalment. Quan el desenvolupador ha venut prou bons, comença el desenvolupament, que va finançant amb préstecs basats en aquests. Quan acaba el desenvolupament, i alguna tercera part independent certifica que efectivament el que s'ha realitzat compleix les especificacions, el desenvolupador "executa" els bons que havia venut, paga els seus deutes, i el que li queda són els beneficis que obté pel desenvolupament.
Qui estaria interessat a adquirir els bons esmentats? Òbviament, els usuaris que desitgessin que aquest nou programa o aquesta millora d'un de ja existent es realitzessin. D'alguna manera, aquest sistema de bons permetria que les parts interessades fixessin (si més no parcialment) les prioritats dels desenvolupadors mitjançant la compra de bons. Això permetria també que no fes falta que una sola entitat assumís els costos de desenvolupament, sinó que aquests podrien repartir-se entre moltes (incloent-hi individus), que a més només haurien de pagar si finalment el projecte acabés amb èxit. Un mecanisme molt similar a aquest es proposa, amb molt més detall a "The Wall Street performer protocol. Using software completion bonds to fund open source software development", de Chris Rasch (1991) [191].

Bibliografia

El sistema de bons descrit està basat en l'*street performer protocol* ('protocol de l'artista de carrer') ("The street performer protocol" a: *Third USENIX Workshop on Electronic Commerce Proceedings*, 1998 [152], i "The street performer protocol and digital copyrights", 1999 [153]), un mecanisme basat en el comerç electrònic dissenyat per a facilitar el finançament privat de treballs de creació lliures. Resumint, qui estigui interessat que es realitzi un determinat treball prometria formalment pagar una certa quantitat si el treball es fa i és publicat lliurement. Les seves intencions són buscar una nova manera de finançar treballs relativament petits que quedin a disposició de tothom, però poden estendre's de moltes formes (els bons per a la construcció de programari lliure en són una). Pot veure's un petit cas de posada en pràctica d'un derivat d'aquest protocol, el *rational street performer protocol* ('protocol racional de l'artista de carrer', Paul Harrison, 2002, [137]) a http://www.csse.monash.edu.au/~pfh/circle/funding_results.html, on s'aplica a la consecució de fons per al finançament de part de The Circle, un projecte de programari lliure.

- Cooperatives de desenvolupadors. En aquest cas, els desenvolupadors de programari lliure, en lloc de treballar individualment o per a una empresa, es reuneixen en algun tipus d'associació (normalment similar a una cooperativa). Fora d'això, el seu funcionament és similar al d'una empresa, matisat potser pel seu compromís ètic amb el programari lliure, que pot ser part dels seus estatuts (encara que això també ho pot fer una empresa). En aquest tipus d'organitzacions poden donar-se combinacions variades de treball voluntari amb treball remunerat. N'és un exemple Free Developers.

- Sistema de donacions. Consisteix a habilitar un mecanisme de pagament a l'autor d'un determinat programari a la pàgina web que acull el projecte. D'aquesta forma, els usuaris interessats que l'esmentat projecte continuï publicant noves versions poden donar-li suport econòmicament, mit-

jançant la realització de donacions voluntàries a tall de finançament per al desenvolupador.

5.2. Models de negoci basats en programari lliure

A més dels mecanismes de finançament dels projectes, dels quals ja hem parlat, un altre aspecte molt relacionat amb l'economia que val la pena tractar és el dels models de negoci. En parlar d'aquests mecanismes de finançament, ja se n'han esmentat de passada uns quants. Ara, en aquest apartat, els descriurem de forma una mica més metòdica.

En general, pot dir-se que són molts els models de negoci que s'estan explorant entorn del programari lliure, alguns de més clàssics i uns altres de més innovadors. Cal tenir en compte que entre els models més habituals en la indústria del programari no és fàcil fer-ne servir aquells basats en la venda de llicències d'ús de programes produïts, ja que al món del programari lliure aquest és un mecanisme de finançament molt difícil d'explotar. Tanmateix, sí que es poden utilitzar els basats en el servei a tercers, amb l'avantatge que sense ser necessàriament el productor d'un programa s'hi pot donar suport complet.

Venda de programari lliure a tant per còpia

Al món del programari lliure és difícil cobrar llicències d'ús, però no impossible. En general, no hi ha res en les definicions de programari lliure que impedeixi que una empresa creï un producte i només el distribueixi a qui pagui una certa quantitat. Per exemple, un determinat productor podria decidir distribuir el seu producte amb una llicència lliure, però només a qui li pagui 1.000 euros per còpia (de manera similar a com es fa al món clàssic del programari propietari).

Tanmateix, encara que això és teòricament possible, a la pràctica és bastant difícil que succeeixi. Perquè una vegada que el productor ha *venut* la primera còpia, qui la rep pot estar motivat a tractar de recuperar la seva inversió venent més còpies a un preu més baix (una cosa que no pot prohibir la llicència del programa, si aquest és lliure). En l'exemple anterior, podria tractar de vendre deu còpies a 100 euros cada una, amb la qual cosa el producte li sortiria gratis (a més, això dificultaria molt que el productor original vengués una altra còpia a 1.000 euros, ja que el producte es podria obtenir legalment per la desena part). És fàcil deduir com aquest procés continuaria en cascada fins a la venda de còpies a un preu proper al cost marginal de còpia, que amb les tecnologies actuals és pràcticament zero.

Tot i així, i tenint en compte que el mecanisme descrit farà que normalment el productor no pugui posar un preu (especialment un preu alt) al simple fet de la redistribució del programa, hi ha models de negoci que implícitament fan justament això. Un exemple és el de les distribucions de GNU/Linux, que es ven per un preu molt baix comparat amb el dels seus competidors propietaris, però superior (i normalment clarament superior) al cost de còpia (fins i tot quan es poden descarregar lliurement d'Internet). Sens dubte, en aquests casos entren en joc altres factors, com la imatge de marca o la comoditat per al consumidor. Però no és aquest l'únic cas. Per tant, més que indicar que amb programari lliure "no es pot vendre a tant per còpia", cal tenir en compte que és més difícil fer-ho, i que probablement s'obtindrà menys benefici, però que hi pot haver models basats justament en això.

Ateses les limitacions (i els avantatges), des de fa uns anys s'estan provant variants dels models de negoci habituals en la indústria del programari, alhora que se'n busquen d'altres de més innovadores per a explotar les possibilitats

que ofereix el programari lliure. Sens dubte, en els pròxims anys veurem encara més experimentació en aquest camp, i també tindrem més informació sobre quins models poden funcionar bé i en quines circumstàncies.

En aquest apartat oferirem un panorama dels models que més habitualment trobem avui en dia, agrupats amb la intenció de mostrar al lector el que tenen de comú i el que els diferencia, centrant-nos en aquells basats en el desenvolupament i els serveis entorn d'un producte de programari lliure. Els ingressos, en aquest cas, provenen directament d'aquestes activitats de desenvolupament i serveis per al producte, però no necessàriament impliquen desenvolupament de nous productes. Quan sí que es du a terme aquest desenvolupament, aquests models tenen com a *subproducte* el finançament de productes de programari lliure, per la qual cosa són models especialment interessants l'impacte dels quals pot ser gran al món del programari lliure en general.

En qualsevol cas, i encara que aquí s'ofereix una classificació relativament clara, no s'ha d'oblidar que gairebé totes les empreses usen en realitat combinacions dels models que descrivim, entre ells i amb altres de més tradicionals.

5.2.1. Millor coneixement

L'empresa que utilitza aquest model de negoci tracta de rendibilitzar el seu coneixement d'un producte (o un conjunt de productes) lliure. Els seus ingressos provindran de clients a qui vendrà serveis relacionats amb aquest coneixement: desenvolupaments basats en el producte, modificacions, adaptacions, instal·lacions i integracions amb d'altres. L'avantatge competitiu de l'empresa estarà en gran manera lligat al millor coneixement del producte: és per això que l'empresa estarà especialment ben situada si és la productora o si participa activament en el projecte que el produeix.

Aquesta és una de les raons per les quals les empreses que utilitzen aquest model solen participar activament en els projectes relacionats amb el programari sobre el qual tracten de vendre serveis: és una forma molt eficient d'obtenir coneixement sobre ell, i el que és més important, que aquest coneixement sigui reconegut. Sens dubte, explicar-li a un client que entre els empleats hi ha diversos desenvolupadors del projecte que produeix el programari que, per exemple, es vol modificar, pot ser una bona garantia.

Relació amb els projectes de desenvolupament

Per tant, aquest tipus d'empreses tenen un gran interès a donar la imatge que tenen un bon coneixement de determinats productes lliures. Una interessant conseqüència d'això és que el seu suport a projectes de programari lliure (per exemple, participant activament en ells o permetent que els seus empleats ho facin durant la seva jornada laboral) no és per tant una cosa merament filantròpica. Al contrari, pot ser un dels actius més rendibles de l'empresa, ja que els seus clients el valoraran molt positivament com una mostra clara que aquesta coneix el producte en qüestió. A més, d'aquesta forma podrà seguir molt de prop el desenvolupament, tractant d'assegurar-se, per exemple, de quines millores demanades pels seus clients passen a formar part del producte desenvolupat pel projecte.

Analitzant-ho des d'un punt de vista més general, aquesta és una situació en què ambdues parts, l'empresa i el projecte de desenvolupament, guanyen de la col·laboració. El projecte guanya pel desenvolupament realitzat per l'empresa, o perquè alguns dels seus desenvolupadors passen a estar remunerats (si més no parcialment) pel seu treball en el projecte. L'empresa guanya en coneixement del producte, en imatge cap als seus clients i en una certa influència sobre el projecte.

Els serveis que proporcionen aquest tipus d'empreses poden ser molt amplis, però normalment consisteixen en desenvolupaments a mida, adaptacions o integracions dels productes en els quals són expertes, o bé serveis de consultoria en els quals aconsellen als seus clients com utilitzar millor el producte en qüestió (especialment si és complex o si el seu correcte funcionament és crític per al mateix client).

Exemples

Exemples d'empreses que fins a cert punt utilitzen aquest model de negoci són les següents:

- LinuxCare (http://www.linuxcare.com) [45]. Fundada el 1996, proporcionava en els seus orígens serveis de consultoria i suport per a GNU/Linux i programari lliure als EUA i la seva plantilla estava composta fonamentalment per experts en GNU/Linux. Tanmateix, el 1992 va canviar els seus objectius, i des de llavors s'ha especialitzat a proporcionar serveis gairebé exclusivament a GNU/Linux executant sobre màquines virtuals z/VM a grans ordinadors d'IBM. El seu model de negoci ha canviat també el de "millor coneixement amb limitacions", ja que ofereix com a part fonamental dels seus serveis una aplicació no lliure, Levanta.

- Alcôve (http://www.alcove.com) [3]. Fundada el 1997 a França, proporciona principalment serveis de consultoria, consultaria estratègica, suport i desenvolupament per a programari lliure. Des de la seva fundació, Alcôve ha mantingut en plantilla desenvolupadors de diversos projectes lliures, la qual cosa ha tractat de rendibilitzar en termes d'imatge. També ha intentat oferir una imatge, en general, d'empresa vinculada a la comunitat de programari lliure, per exemple, col·laborant amb associacions d'usuaris i donant publicitat a les seves col·laboracions amb projectes lliures (per exemple, des d'Alcôve-Labs –http://www.alcove-labs.org– [4]).

5.2.2. Millor coneixement amb limitacions

Aquests models són similars als exposats en l'apartat anterior, però intenten limitar la competència a què poden veure's sotmesos. Mentre que en els models *purs* basats en el millor coneixement qualsevol pot entrar, en principi, en competència, ja que el programari utilitzat és el mateix (i lliure), en aquest cas es tracta d'evitar aquesta situació posant barreres a aquesta competència. Aquestes barreres solen consistir en patents o llicències propietàries, que nor-

malment afecten una part petita (però fonamental) del producte desenvolupat. Per això aquests models poden considerar-se en realitat mixtos, en el sentit que estan a cavall del programari lliure i el programari propietari.

En molts casos, la comunitat del programari lliure desenvolupa la seva pròpia versió per a aquest component, amb la qual cosa l'avantatge competitiu pot desaparèixer, o fins i tot tornar en contra de l'empresa en qüestió si el seu *competidor* lliure es converteix en l'estàndard del mercat i passa a ser demanat pels seus propis clients.

Exemples

Hi ha molts casos en els quals s'usa aquest model de negoci, ja que és comú considerar-lo menys arriscat que el de coneixement *pur*. Tanmateix, les empreses que l'han utilitzat han tingut evolucions variades. Algunes són les següents:

- Caldera (http://www.sco.com) [16]. La història de Caldera és complicada. En els seus inicis, va crear la seva pròpia distribució de GNU/Linux, orientada a les empreses: Caldera OpenLinux. El 2001 va comprar la divisió de Unix de SCO, i el 2002 va canviar el seu nom a SCO Group. La seva estratègia empresarial ha fet tants tombs com el seu nom, des del seu total suport a GNU/Linux fins a les seves demandes contra IBM i Red Hat el 2003 i l'abandonament de la seva pròpia distribució. Però referent a aquest apartat, el negoci de Caldera, almenys fins al 2002, és un clar exponent del millor coneixement amb limitacions. Caldera tractava d'explotar el seu coneixement de la plataforma GNU/Linux, però limitant la competència a què podia veure's sotmesa mitjançant la inclusió de programari propietari en la seva distribució. Això feia difícil als seus clients canviar de distribució una vegada que l'havien adoptat, ja que encara que les altres distribucions de GNU/Linux incloïen la part lliure de Caldera OpenLinux, no es trobava en aquestes la part propietària.

- Ximian (http://ximian.com/) [74]. Fundada el 1999 amb el nom d'Helix Code per desenvolupadors molt vinculats al projecte GNOME, va ser adquirida l'agost de 2003 per Novell. La major part del programari que ha desenvolupat ha estat lliure (en general, parteix de GNOME). Tanmateix, en un àmbit molt concret Ximian va decidir llicenciar un component com a programari propietari: el Connector for Exchange. Aquest mòdul permet a un dels seus productes estrella, Evolution (un gestor d'informació personal que inclou correu electrònic, agenda, calendari, etc.), interactuar amb servidors Microsoft Exchange, molt utilitzats en grans organitzacions. D'aquesta manera tractava de competir avantatjosament amb altres empreses que proporcionaven serveis basats en GNOME, potser amb els productes desenvolupats per la mateixa Ximian que no podien interaccionar tan fàcilment amb Exchange. Excepte per aquest producte, el model de Ximian ha estat de "millor coneixement", i també s'ha basat en el fet de ser la font d'un programa (com veurem més endavant). En qualsevol cas, aquest component va ser alliberat el 2005.

5.2.3. Font d'un producte lliure

Aquest model és similar al basat en el millor coneixement, però l'especialitza, amb la qual cosa l'empresa que l'utilitza és productora, de forma pràcticament íntegra, d'un producte lliure. Naturalment, l'avantatge competitiu augmenta en ser els desenvolupadors del producte en qüestió, controlar la seva evolució i tenir-lo abans que la competència. Tot això posiciona l'empresa desenvolupadora en un lloc molt bo de cara als clients que vulguin serveis sobre aquest programa. A més, és un model molt interessant en termes d'imatge, ja que l'empresa ha demostrat el seu potencial desenvolupador amb la creació i el manteniment de l'aplicació en qüestió, la qual cosa pot ser molt interessant a l'hora de convèncer possibles clients de les seves capacitats. Igualment, pro-

porciona molt bona imatge de cara a la comunitat del programari lliure en general, ja que aquesta rep de l'empresa un nou producte lliure que passa a formar part del patrimoni comú.

Exemples

Són molts els productes lliures que van començar el desenvolupament dins d'una empresa, i molt habitualment ha continuat essent aquesta empresa la que ha guiat el seu desenvolupament posterior. A tall d'exemple, podem esmentar els casos següents:

- Ximian. Ja s'ha esmentat com en part ha usat el model de millor coneixement amb limitacions. Però en general, Ximian ha seguit un clar model basat a ser la font de programes lliures. Els seus productes principals, com Evolution o Red Carpet, s'han distribuït sota llicències GPL. Tanmateix, altres també importants, com ara MONO, es distribueixen en gran part sota llicència MIT X11 o LGPL. En tots aquests casos, Ximian ha desenvolupat els productes gairebé en exclusiva des del començament. L'empresa ha tractat de rendibilitzar aquest desenvolupament aconseguint contractes per a fer-los evolucionar en certs sentits, per a adaptar-los a les necessitats dels seus clients, i oferint personalització i manteniment.

- Zope Corporation (http://www.zope.com/) [75]. El 1995 es funda Digital Creations, que desenvolupa un producte propietari per a la gestió d'anuncis classificats via web. El 1997 va rebre una injecció de capital per part, entre altres, d'una empresa de capital risc, Opticality Ventures. El més estrany (en aquella època) d'aquesta inversió va ser que com a condició li van posar que distribuís com a programari lliure l'evolució del seu producte, el que més endavant va ser Zope, un dels gestors de continguts més populars a Internet. Des de llavors, el model de negoci de l'empresa va ser produir Zope i productes relacionats amb aquest, i oferir serveis d'adaptació i manteniment per a tots ells. Zope Corporation ha sabut, a més, crear una dinàmica comunitat de desenvolupadors de programari lliure entorn dels seus productes i col·laborar-hi activament.

5.2.4. Font d'un producte amb limitacions

Aquest model és similar a l'anterior, però pren mesures per limitar la competència o maximitzar els ingressos. Entre les limitacions més habituals, podem considerar les següents:

- Distribució propietària durant un temps, després lliure. Amb promesa de distribució lliure posterior o sense, cada nova versió del producte es ven com a programari propietari. Passat un temps (normalment, quan es comença a comercialitzar una nova versió, també com a programari propietari), aquesta versió passa a distribuir-se amb una llicència lliure. D'aquesta manera, l'empresa productora obté ingressos dels clients interessats a disposar tan aviat com es pugui de noves versions, i alhora minimitza la competència, ja que qualsevol empresa que vulgui competir usant aquest producte només podrà fer-ho amb la versió lliure (disponible només quan ja hi ha una nova versió propietària, suposadament millor i més completa).

- Distribució limitada durant un temps. En aquest cas, el programari és lliure des que es comença a distribuir. Però com que no hi ha res en una llicència lliure que obligui a distribuir el programa a tothom qui el vulgui (això és una cosa que qui té el programari pot fer o no), el productor el distribueix durant un temps només als seus clients, que li paguen per això (normalment en forma de contracte de manteniment). Al cap d'un temps, el distribueix a qualsevol, per exemple posant-lo en un arxiu d'accés públic.

D'aquesta manera, el productor obté ingressos dels seus clients, que perceben aquesta disposició preferent del programari com un valor afegit. Naturalment, el model només funciona si els clients al seu torn no fan públic el programa quan el reben. Per a cert tipus de clients, això pot no ser habitual.

En general, en aquests casos les empreses desenvolupadores obtenen els beneficis esmentats, però no a cost zero. A causa del retard amb què el producte està disponible per a la comunitat del programari lliure, és pràcticament impossible que aquesta pugui col·laborar en el seu desenvolupament, per la qual cosa el productor es beneficiarà molt poc de contribucions externes.

Exemples

Algunes empreses que utilitzen aquest model de negoci són les següents:

- artofcode LLC (http://artofcode.com/) [9]. Des de l'any 2000, artofcode comercialitza Ghostscript en tres versions (anteriorment ho havia fet Alladin Enterprises amb un model similar). La versió més actual la distribueix com a AFPL Ghostscript, sota una llicència propietària (que permet l'ús i la distribució no comercial). La següent (amb un retard d'un any, més o menys) la distribueix com a GNU Ghostscript, sota la GNU GPL. Per exemple, l'estiu del 2003, la versió AFPL és la 8.11 (alliberada el 16 d'agost), mentre que la versió GNU és la 7.07 (distribuïda com a tal el 17 de maig, però que té com a versió AFPL equivalent la del 2002). A més, artofcode ofereix una tercera versió, amb una llicència propietària que permet la integració en productes no compatibles amb la GNU GPL (en aquest cas usa un model dual, que serà descrit més endavant).

- Ada Core Technologies (http://www.gnat.com/) [2]. Va ser fundada el 1994 pels autors del primer compilador d'Ada 95, el desenvolupament del qual va ser finançat en part pel Govern dels EUA, que estava basat en GCC, el compilador de GNU. Des del principi els seus productes han estat programari lliure. Però la majoria d'ells els ofereixen primer als seus clients, com a part d'un contracte de manteniment. Per exemple, el seu compilador, que continua basant-se en GCC i es distribueix sota la GNU GPL, s'ofereix als seus clients com a GNAT Pro. Ada Core Technologies no ofereix aquest compilador al públic en general de cap manera, i normalment no se'n troben versions a la Xarxa. Tanmateix, amb un retard variable (entorn d'un any), Ada Core Technologies ofereix les versions *públiques* del seu compilador, molt similars però sense cap tipus de suport, en un arxiu d'FTP anònim.

5.2.5. Llicències especials

En aquests models, l'empresa produeix un producte que distribueix sota dues o més llicències. Almenys una és de programari lliure, però les altres típicament són propietàries i li permeten vendre el producte d'una forma més o menys tradicional. Normalment, aquestes vendes es complementen amb l'oferta de consultoria i desenvolupaments relacionats amb el producte. Per exemple, una empresa pot distribuir un producte com a programari lliure sota la GNU GPL, però oferir també una versió propietària (simultàniament, i sense retard per a cap de les dues) per a qui no vulgui les condicions de la GPL, per exemple, perquè vulgui integrar el producte amb un de propietari (una cosa que la GPL no permet).

Exemple

Sleepycat Software (http://www.sleepycat.com/download/oslicense.html) [60]. Aquesta empresa va ser fundada el 1996 i anuncia que des d'aquell moment ha tingut beneficis (la qual cosa sens dubte és notable en una empresa relacionada amb el programari). Els seus productes, incloent-hi Berkeley DB (un gestor de dades molt popular que pot encastar-se fàcilment en altres aplicacions), es distribueixen sota una llicència lliure que especifica que en cas d'encastar-se en un altre producte, ha d'oferir-se el codi font d'ambdós. Sleepycat ofereix serveis de consultoria i desenvolupament per als seus productes, però a més els ofereix sota llicències que permeten encastar-los sense haver de distribuir el codi font. Naturalment, això ho fa sota contracte específic, i en general, en règim de venda com a programari propietari. El 2005, Sleepycat Software va ser adquirida per Oracle.

5.2.6. Venda de marca

Encara que puguin aconseguir-se productes molt similars per menys diners, molts clients estan disposats a pagar l'extra per comprar una *marca*. Aquest principi és utilitzat per empreses que inverteixen a establir una marca amb bona imatge i ben reconeguda que els permeti després vendre amb marge suficient productes lliures. En molts casos no solament venen aquests productes, sinó que els acompanyen de serveis que els clients acceptaran també com a valor afegit.

Els casos més coneguts d'aquest model de negoci són les empreses que comercialitzen distribucions GNU/Linux. Aquestes empreses tracten de vendre una cosa que en general es pot obtenir a un cost bastant més baix a la Xarxa (o en altres fonts amb menys imatge de marca). És per això que han d'aconseguir que el consumidor reconegui la seva marca i estigui disposat a pagar el sobrepreu. Per a això no solament inverteixen en publicitat, sinó que també ofereixen avantatges objectius (per exemple, una distribució ben conjuntada o un canal de distribució que arribi a les proximitats del client). A més, solen oferir al seu voltant una gran quantitat de serveis (des de formació fins a programes de certificació per a terceres parts), tractant de rendibilitzar al màxim aquesta imatge de marca.

Exemple

Red Hat (http://www.redhat.com) [56]. Red Hat Linux va començar a distribuir-se el 1994 (l'empresa va començar a conèixer-se amb el nom actual el 1995). Durant molt temps Red Hat va aconseguir col·locar el seu nom com el de la distribució de GNU/Linux per excel·lència (encara que a mitjan dècada del 2000 comparteix aquesta posició amb altres empreses, com OpenSUSE, Ubuntu, i potser Debian). Diversos anys després Red Hat comercialitza tot tipus de serveis relacionats amb la distribució, amb GNU/Linux i amb programari lliure en general.

5.3. Altres classificacions de models de negoci

En la literatura sobre programari lliure, hi ha altres classificacions de models de negoci clàssiques. A continuació, i a tall d'exemple, n'oferim una.

5.3.1. Classificació de Hecker

La classificació que s'ofereix a "Setting up shop: the business of open-source software" (Frank Hecker, 1998) [141] va ser la més emprada per la publicitat de l'Open Source Initiative, i també una de les primeres a tractar de categoritzar els negocis que estaven sorgint per aquella època. Tanmateix, inclou diversos models poc centrats en programari lliure (en els quals aquest és poc més que un acompanyant del model principal). En qualsevol cas, els models que descriu són els següents:

- *Support seller* (venda de serveis relacionats amb el producte). L'empresa promou un producte lliure (que ha desenvolupat o en el desenvolupament del qual participa activament) i ven serveis, com consultoria o adaptació a necessitats concretes per a aquest.

- *Loss leader* (venda d'altres productes propietaris). En aquest cas, el programa lliure s'utilitza per a promoure d'alguna forma la venda d'altres productes propietaris relacionats amb ell.

- *Widget frosting* (venda de maquinari). El negoci fonamental és la venda de maquinari i el programari lliure es considera un complement d'aquest que pot ajudar l'empresa a obtenir un avantatge competitiu.

- *Accessorizing* (venda d'accessoris). Es comercialitzen productes relacionats amb el programari lliure, com ara llibres, dispositius informàtics, etc.

- *Service enabler* (venda de serveis). El programari lliure serveix per a crear un servei (normalment accessible en línia) del qual l'empresa obté algun benefici.

- *Brand licensing* (venda de marca). Una empresa registra marques que aconsegueix associar amb programes lliures, probablement desenvolupats per ella. Després obté ingressos quan ven drets de l'ús d'aquestes marques.

- *Sell it, free it* (ven, allibera). És un model similar al de *loss leader*, però realitzat de forma cíclica. Primer es comercialitza un producte com a programari lliure. Si s'aconsegueix que tingui cert èxit, la següent versió es distribueix com a programari propietari durant un temps, al cap del qual s'allibera també. Per llavors, es comença a distribuir una nova versió propietària, i així successivament.

- *Software franchising* (franquícia de programari). Una empresa franquicia l'ús de les seves marques relacionades amb un programa lliure determinat.

Nota

El lector haurà pogut observar que aquesta classificació és bastant diferent de la que hem ofert nosaltres, però tot i així algunes de les seves categories coincideixen gairebé exactament amb algunes de les nostres.

5.4. Impacte sobre les situacions de monopoli

El mercat informàtic tendeix a la dominació d'un producte en cada un dels seus segments. Els usuaris volen rendibilitzar l'esforç realitzat en aprendre com funciona un programa, les empreses volen trobar gent formada en l'ús del seu programari, i tots volen que les dades que gestionen puguin ser enteses pels programes de les empreses i les persones amb qui es relacionen. Per això, qualsevol iniciativa dedicada a trencar una situació *de facto* en la qual un producte domina clarament el mercat està destinada a produir més del mateix: si té èxit, en vindrà un altre a ocupar aquest buit, i en breu tindrem un nou producte dominant. Només els canvis tecnològics produeixen, durant un temps, la inestabilitat suficient perquè ningú no domini clarament.

Però el fet que hi hagi un producte dominant no ha de portar necessàriament a la constitució d'un monopoli empresarial. Per exemple, la gasolina és un producte que gairebé domina el mercat de combustibles per a turismes, però (en un mercat de la gasolina lliure) hi ha moltes empreses productores i distribuïdores d'aquest producte. En realitat, en parlar de programari, el preocupant és el que ocorre quan un producte arriba a dominar el mercat perquè aquest producte té una sola empresa proveïdora possible. El programari lliure ofereix una alternativa a aquesta situació: els productes lliures poden estar promoguts per una empresa en concret, però aquesta empresa no els controla, o almenys no fins als extrems a què ens té acostumats el programari propietari. Al món del programari lliure, un producte dominant no comporta necessàriament un monopoli d'empresa. Al contrari, sigui el que sigui el producte que domini el mercat, moltes empreses poden competir en el fet de proporcionar-lo, millorar-lo, adaptar-lo a les necessitats dels seus clients i oferir serveis entorn.

5.4.1. Elements que afavoreixen els productes dominants

En informàtica és molt comú que hi hagi un producte clarament dominant en cada segment de mercat. I això és normal per diversos motius, entre els quals es poden destacar els següents:

- Formats de dades. En molts casos el format de dades està fortament lligat a una aplicació. Quan un nombre prou alt de gent la utilitza, el seu format de dades es converteix en estàndard *de facto*, i les pressions per a usar-lo (i, per tant, l'aplicació) són formidables.

- Cadenes de distribució. Normalment un dels problemes per a començar a usar un programa és obtenir-ne una còpia. I sol ser difícil trobar els programes que no són líders al seu mercat. Les cadenes de distribució són costoses

de mantenir, de manera que els competidors minoritaris ho tenen difícil per a arribar a la botiga d'informàtica, on l'usuari final els pugui comprar. El producte dominant, tanmateix, ho té fàcil: el primer interessat a tenir-lo serà la mateixa botiga d'informàtica.

- Màrqueting. El màrqueting "gratuït" que obté un producte una vegada que l'usa una fracció significativa d'una població determinada és enorme. El "boca a boca" funciona molt, també el fet de preguntar i intercanviar informació amb els coneguts. Però sobretot l'impacte en els mitjans és molt gran: les revistes d'informàtica parlaran una i altra vegada d'un producte si sembla que és el que més s'utilitza; hi haurà cursos de formació entorn seu, llibres que el descriguin, entrevistes als seus usuaris, etc.

- Inversió en formació. Una vegada que s'ha invertit temps i recursos a aprendre com funciona una eina, s'està molt motivat per a no canviar-la. A més, usualment aquesta eina és la que ja domina el mercat, perquè és més fàcil trobar personal i material que ajudin a aprendre a usar-la.

- Programari preinstal·lat. Rebre una màquina amb programari ja instal·lat és, sens dubte, un gran incentiu per a usar-lo, fins i tot si cal, pagar per ell a part. I normalment, el tipus de programari que el venedor de la màquina estarà disposat a preinstal·lar serà només el més utilitzat.

5.4.2. El món del programari propietari

Al món del programari propietari l'aparició d'un producte dominant en un segment qualsevol equival a un monopoli per part de l'empresa que el produeix. Per exemple, tenim aquestes situacions monopolístiques *de facto* (o gairebé) de producte i empresa als mercats de sistemes operatius, autoedició, bases de dades, disseny gràfic, processadors de textos, fulls de càlcul, etc.

I això és així perquè l'empresa en qüestió té un gran control sobre el producte líder, tan gran que només ells poden marcar la seva evolució, les línies fonamentals en les quals es desenvoluparà, la seva qualitat, etc. Els usuaris tenen molt poc control, ja que estaran molt poc motivats per a provar amb altres productes (per les raons que s'han comentat en l'apartat anterior). Davant d'això, poc podran fer, llevat d'intentar desafiar la posició dominant del producte millorant excepcionalment els seus (per tractar de contrarestar aquests mateixos motius), normalment amb poc èxit.

Aquesta situació posa tot el sector a les mans de l'estratègia de l'empresa dominant. Tots els actors depenen d'ella, i fins i tot el desenvolupament de la tecnologia programari en aquest camp estarà mediatitzat per les millores que faci al seu producte. En el cas general, aquesta és una situació en la qual apa-

reixen els pitjors efectes econòmics del monopoli i, en particular, la falta de motivació de l'empresa líder per a apropar el producte a les necessitats (sempre en evolució) dels seus clients. Aquests s'han convertit en un mercat captiu.

5.4.3. La situació amb programari lliure

Tanmateix, en el cas del programari lliure un producte dominant no es tradueix automàticament en un monopoli d'empresa. Si el producte és lliure, qualsevol empresa pot treballar-hi, millorar-lo, adaptar-lo a les necessitats d'un client, i en general, ajudar en la seva evolució. A més, precisament per la seva posició dominant, hi haurà moltes empreses interessades a treballar-hi. Si "el productor original" (l'empresa que va desenvolupar originalment el producte) vol romandre en el negoci, ha de competir amb totes elles, i per això estarà molt motivada per a fer evolucionar el producte precisament en la línia que els seus usuaris vulguin. Naturalment, tindrà l'avantatge d'un millor coneixement del programa, però això és tot. Han de competir per cada client.

Per tant, l'aparició de productes dominants es tradueix, en el món del programari lliure, en una major competència entre empreses. I amb això els usuaris reprenen el control: les empreses en competència no poden fer cap altra cosa que fer-los cas si volen sobreviure. I precisament això és el que assegurarà que el producte millori.

Productes lliures que són dominants al seu sector

Apache és des de fa temps líder al mercat de servidors web. Però hi ha moltes empreses darrere d'Apache, des d'algunes de molt grans (com IBM) fins a altres de molt petites. I totes elles no tenen més remei que competir millorant-lo i normalment contribuint al projecte amb les seves millores. Malgrat que Apache és gairebé un monopoli en molts àmbits (per exemple, és gairebé l'únic servidor web que es considera sobre la plataforma GNU/Linux o *BSD), no depèn d'una sola empresa, sinó de literalment desenes d'aquestes.

Les distribucions de GNU/Linux són també un cas interessant. GNU/Linux no és, sens dubte, un monopoli, però és possiblement la segona opció al mercat de sistemes operatius. I això no ha forçat una situació en la qual una empresa tingui el seu control. Al contrari, hi ha desenes de distribucions, realitzades per empreses diferents, que competeixen lliurement al mercat. Cada una tracta d'oferir millores que els seus competidors han d'adoptar a risc de quedar-hi fora. Però a més, no poden separar-se gaire del que és "GNU/Linux estàndard", ja que això seria rebutjat pels usuaris com una "sortida de la norma". La situació després de diversos anys de creixement de la quota de mercat de GNU/Linux ens mostra desenes d'empreses que competeixen i fan evolucionar el sistema. I de nou, totes aquestes van darrere de la satisfacció de les necessitats dels seus usuaris. Només així poden mantenir-se en el mercat.

GCC és un producte dominant al món de compiladors C i C++ per al mercat GNU/Linux. I tanmateix, això no ha portat a cap situació de monopoli d'empresa, fins i tot quan Cygnus (avui Red Hat) es va encarregar durant molt temps de coordinar-ne el desenvolupament. Hi ha moltes empreses que fan millores en el sistema i totes competeixen, cada una a la seva veta específica, per satisfer les demandes dels seus usuaris. De fet, quan alguna empresa o organització específica ha fallat en el treball de coordinació (o així ho han percebut una part dels usuaris) hi ha hagut espai per a un *fork* (bifurcació) del projecte, amb dos productes en paral·lel durant un temps, fins i tot que eventualment han tornat a unir-se (com està passant ara amb GCC 3.x).

5.4.4. Estratègies per a constituir-se en monopoli amb programari lliure

Malgrat que el món del programari lliure és molt més hostil als monopolis d'empresa que el món del programari propietari, hi ha estratègies que una empresa pot utilitzar per a tractar d'aproximar-se a una situació de dominació monopolística d'un mercat. Aquestes són pràctiques comunes a molts altres sectors econòmics, i per evitar-les treballen les entitats de regulació de la competència, per la qual cosa no en parlarem en detall. Tanmateix, sí que n'esmentarem una que és fins a cert punt específica del mercat del programari, i que ja s'està experimentant en algunes situacions: l'acceptació de productes certificats per tercers.

Quan una empresa vol distribuir un producte programari (lliure o propietari) que funcioni en combinació amb d'altres, és comú "certificar" aquest producte per a una certa combinació. El fabricant es compromet a oferir serveis (actualització, suport, resolució de problemes, etc.) només si el client li assegura que està fent servir el producte en un entorn certificat per ell. Per exemple, un fabricant de gestors de bases de dades pot certificar el seu producte per a una determinada distribució de GNU/Linux, i per a cap altra. Això implicarà que els seus clients hauran d'emprar aquesta distribució de GNU/Linux o oblidar-se del suport per part del fabricant (cosa que, si el producte és propietari, pot ser impossible a la pràctica). Si un cert fabricant aconsegueix una posició clarament dominant com a producte certificat de terceres parts, els usuaris no tindran gaires més possibilitats per a utilitzar aquest producte. Si en aquest segment la certificació és important, estarem de nou davant d'una situació d'empresa monopolística.

Nota

Fins a cert punt, al mercat de distribucions de GNU/Linux es comencen a observar certs casos de situacions tendents al monopoli *de facto* en la certificació. Per exemple, hi ha molts fabricants de productes propietaris que només certifiquen els seus productes sobre una distribució de GNU/Linux donada (molt habitualment, Red Hat Linux). De moment això no sembla redundar en una situació monopolística per part de cap empresa, la qual cosa podria ser deguda al fet que el mercat de distribucions de GNU/Linux la certificació ha de tenir poca importància per als usuaris. Però només el futur dirà si en algun moment aquesta situació s'apropa a una de monopoli *de facto*.

Tanmateix, és important tenir en compte dos comentaris respecte al que s'ha exposat. El primer és que aquestes posicions monopolístiques no seran fàcils d'aconseguir, i en qualsevol cas s'aconseguiran per mecanismes en general "no informàtics" (a diferència de la situació de producte dominant, que com ja hem vist és relativament normal, a la qual s'arriba per mecanismes purament relacionats amb la informàtica i els seus patrons d'ús). El segon és que si tot el programari utilitzat és lliure, aquesta estratègia té poques possibilitats d'èxit (si és que en té cap). Un fabricant podrà aconseguir que moltes empreses cer-

tifiquin per als seus productes, però els clients sempre podran buscar fonts de servei i suport diferents d'aquestes empreses que han certificat per a ell si així ho consideren convenient.

6. Programari lliure i administracions públiques

> "[...] el programari, per a ser acceptable per a l'Estat, no n'hi ha prou que sigui tècnicament suficient per a dur a terme una tasca, sinó que a més les condicions de contractació han de satisfer una sèrie de requisits en matèria de llicència, sense els quals l'Estat no pot garantir al ciutadà el processament adequat de les seves dades, i vetllar per la seva integritat, confidencialitat i accessibilitat al llarg del temps, perquè són aspectes molt crítics per al seu normal acompliment."
>
> Edgar David Villanueva Núñez (carta de resposta al gerent general de Microsoft del Perú, 2001)

Les institucions públiques, tant les que tenen capacitat legislativa com les que es dediquen a administrar l'estat (les "administracions públiques"), tenen un paper molt rellevant referent a adopció i promoció de tecnologies. Encara que pràcticament l'any 2000 no hi va haver (llevat de casos anecdòtics) interès d'aquestes institucions pel fenomen del programari lliure, la situació va començar a canviar a partir d'aquesta data. D'una banda, moltes administracions públiques van començar a emprar programari lliure com a part de la seva infraestructura informàtica. Per una altra, en el seu paper de promotores de la societat de la informació, algunes van començar a promoure directament o indirectament el desenvolupament i l'ús de programari lliure. A més, alguns cossos legislatius s'han anat ocupant (a poc a poc) del programari lliure, de vegades afavorint el seu desenvolupament, de vegades dificultant-lo, i de vegades simplement tenint-lo en compte.

Abans d'entrar en detall, és convenient recordar que el programari lliure es va desenvolupar durant molt temps al marge del suport explícit (i fins i tot de l'interès) de les institucions públiques. És per això que la recent atenció que està despertant en moltes d'aquestes no està exempta de polèmiques, confusions i problemes. A més, durant els últims anys les iniciatives relacionades amb els estàndards oberts també van cobrant força, i es tradueixen en molts casos en mesures (més o menys directes) relacionades amb el programari lliure.

En aquest capítol tractarem d'exposar quina és la situació actual i quines peculiaritats té el programari lliure quan es relaciona amb "el públic".

6.1. Impacte en les administracions públiques

Hi ha hagut diversos estudis que s'han centrat en les implicacions de l'ús del programari lliure en les administracions públiques (per exemple, "Open source software for the public administration", 2004 [159]; "Open-source software in e-Government, analysis and recommendations drawn up by a working group under the danish board of technology", 2002 [180]; "Free software / open sour-

ce: information society opportunities for Europe?", 1999 [132], i "The case for government promotion of open source software", 1999 [213]). A continuació en comentem algunes de les més destacables (tant positives com negatives).

6.1.1. Avantatges i implicacions positives

Alguns dels avantatges de l'ús de programari lliure en les administracions públiques i de les principals noves perspectives que permet són les següents:

1) Foment de la indústria local

Un dels majors avantatges del programari lliure és la possibilitat de desenvolupar indústria local de programari. Quan s'utilitza programari propietari, tot el que s'ha gastat en llicències va directament al fabricant del producte, i a més aquesta compra redunda en l'enfortiment de la seva posició, la qual cosa no és necessàriament perjudicial, però sí que és poc eficient per a la regió vinculada a l'administració si analitzem l'alternativa de fer servir un programa lliure.

En aquest cas, les empreses locals podran competir proporcionant serveis (i el mateix programa) a l'Administració, en condicions molt similars a qualsevol altra empresa. Diguem que, d'alguna manera, l'Administració està aplanant el camp de joc i fent més fàcil que qualsevol pugui competir-hi. I naturalment, entre aquests "qualssevol" hi haurà les empreses locals, que tindran la possibilitat d'aprofitar els seus avantatges competitius (millor coneixement de les necessitats del client, proximitat geogràfica, etc.).

2) Independència de proveïdor i competència al mercat

És obvi que qualsevol organització preferirà dependre d'un mercat en règim de competència que d'un sol proveïdor, que pot imposar les condicions en les quals proporciona el seu producte. Tanmateix, al món de l'administració, aquesta preferència es converteix en requisit fonamental, i fins i tot en obligació legal en alguns casos. L'administració no pot elegir, en general, contractar un subministrador donat, sinó que ha d'especificar les seves necessitats de manera que qualsevol empresa interessada que compleixi unes certes característiques tècniques i que proporcioni el servei o el producte demanat amb una certa qualitat, pugui optar a un contracte.

De nou, en el cas del programari propietari, per a cada producte no hi ha més que un proveïdor (encara que empri una varietat d'intermediaris). Si s'especifica un producte donat, s'està decidint també quin proveïdor contractarà l'administració. I en molts casos és pràcticament impossible evitar d'especificar un cert producte, quan estem parlant de programes d'ordinador. Raons de compatibilitat dins de l'organització o d'estalvis en formació i administració, o moltes d'altres, fan habitual que una administració decideixi fer servir un cert producte.

L'única sortida a aquesta situació és que el producte especificat sigui lliure. En aquest cas, qualsevol empresa interessada podrà proporcionar-lo, i també podrà proporcionar qualsevol tipus de servei sobre ell (subjecte únicament a les seves capacitats i coneixements del producte). A més, en cas de contractar d'aquesta manera, l'administració podrà canviar en el futur de proveïdor si així ho desitja, immediatament i sense cap problema tècnic, ja que encara que canviï d'empresa, el producte que utilitzarà serà el mateix.

3) Flexibilitat i adaptació a les necessitats exactes

Encara que l'adaptació a les seves necessitats exactes és una cosa necessària en qualsevol organització a la qual cal la informàtica, les peculiaritats de l'administració fan que aquest sigui un factor molt important per a l'èxit de la implantació d'un sistema informàtic. En el cas de fer servir programari lliure, l'adaptació pot fer-se amb molta major facilitat, i el que és més important, servint-se d'un mercat amb competència si fa falta contractar-la.

Quan l'administració compra un producte propietari, modificar-lo passa normalment per assolir un acord amb el seu productor, que és l'únic que legalment (i moltes vegades tècnicament) pot fer-ho. En aquestes condicions, és difícil realitzar bones negociacions, sobretot si el productor no està excessivament interessat en el mercat que li ofereix l'administració en qüestió. Tanmateix, emprant un producte lliure, l'Administració pot modificar-lo al seu gust si disposa de personal per a això, o contractar externament la modificació. Aquesta contractació la pot realitzar en principi qualsevol empresa que tingui els coneixements i les capacitats per a això, per la qual cosa és d'esperar que n'hi concorrin diverses. Això, necessàriament, tendeix a abaratir-ne els costos i a millorar-ne la qualitat.

El cas de les distribucions de GNU/Linux

A Espanya, durant els últims anys, s'ha fet comuna la creació de distribucions de GNU/Linux pròpies per part de certes comunitats autònomes. Aquesta tendència va començar amb GNU/LinEx, però avui en dia són moltes més. Encara que l'existència de totes aquestes distribucions ha estat criticada per alguns experts, suposa un clar exponent de la flexibilitat que permet el programari lliure. Cada administració pública, gastant relativament pocs recursos, pot contractar l'adaptació de GNU/Linux fent-ho d'acord amb les seves necessitats i preferències, pràcticament sense límit. Per exemple, pot canviar l'aspecte de l'escriptori, escollir el conjunt d'aplicacions i l'idioma per defecte, millorar la localització de les aplicacions, etc. En altres paraules: si es vol, l'escriptori (i qualsevol altre element del programari que funcioni a l'ordinador) pot adaptar-se a unes necessitats exactes.

Sens dubte, aquesta adaptació suposarà uns costos, però l'experiència mostra que pot aconseguir-se molt amb relativament pocs recursos, i la tendència sembla indicar que cada vegada serà més fàcil (i més barat) realitzar distribucions personalitzades.

4) Adopció més fàcil d'estàndards oberts

Per la seva pròpia naturalesa, és habitual que els programes lliures utilitzin estàndards oberts o estàndards no privatius. De fet, gairebé per definició, qualsevol aspecte d'un programa lliure que es vulgui considerar pot reproduir-se amb facilitat, i per tant, no és privatiu. Per exemple, els protocols que un pro-

grama lliure utilitza per a interactuar amb altres programes poden estudiar-se i reproduir-se, per la qual cosa no són privatius. Però a més, molt habitualment i pel mateix interès dels projectes, s'intenta utilitzar estàndards oberts.

En qualsevol cas, i sigui quina sigui la raó, és un fet que els programes lliures utilitzen habitualment estàndards no privatius per a l'intercanvi de dades. Els avantatges que això suposa per a les administracions públiques van molt més enllà de les que pot trobar qualsevol organització, ja que la promoció d'estàndards privatius (fins i tot de forma indirecta, en fer-los servir) en aquest cas seria molt més preocupant. I en un aspecte almenys l'ús d'estàndards no privatius és fonamental, el de la interacció amb el ciutadà, que no ha de veure's obligat a comprar cap producte a una empresa en particular per poder relacionar-se amb la seva administració.

5) Escrutini públic de seguretat

Per a una administració pública, poder garantir que els seus sistemes informàtics fan només el que està previst que facin és un requisit fonamental, i a molts països, un requisit legal. No són poques les vegades que aquests sistemes fan servir dades privades, en què poden estar interessats tercers (pensem en dades fiscals, penals, censals, electorals, etc.). Difícilment si s'empra una aplicació propietària, sense codi font disponible, pot assegurar-se que efectivament aquesta aplicació tractarà aquestes dades com cal. Però fins i tot si s'ofereix el seu codi font, les possibilitats que tindrà una institució pública per a assegurar que no conté codi *estrany* seran molt limitades. Només si es pot encarregar aquest treball de forma habitual i rutinària a tercers, i a més qualsevol part interessada pot escrutar-los, l'administració podrà estar raonablement segura de complir amb aquest deure fonamental, o almenys, prendre totes les mesures a la seva mà per fer-ho.

6) Disponibilitat a llarg termini

Moltes dades que empren les administracions, i els programes que serveixen per a calcular-los, han d'estar disponibles dins de desenes d'anys. És molt difícil assegurar que un programa propietari qualsevol estarà disponible quan hagin passat aquells períodes de temps, i més si el que es vol és que funcioni a la plataforma habitual en aquell moment futur. Al contrari, és molt possible que el seu productor hagi perdut interès en el producte i que no l'hagi portat a noves plataformes, o que només estigui disposat a fer-ho davant de grans contraprestacions econòmiques. De nou, cal recordar que només ell pot fer aquesta migració, i per tant, serà difícil negociar amb ell. En el cas del programari lliure, en canvi, l'aplicació està disponible, amb seguretat, que qualsevol la porti i la deixi funcionant segons les necessitats de l'administració. Si això no passa de forma espontània, l'administració sempre pot dirigir-se a diverses empreses buscant la millor oferta per fer la feina. Així es pot garantir que l'aplicació, i les dades que maneja, estaran disponibles quan faci falta.

7) Impacte més enllà de l'ús per part de l'Administració

Moltes aplicacions utilitzades o promogudes per les administracions públiques són utilitzades també per molts altres sectors de la societat. És per això que qualsevol inversió pública en el desenvolupament d'un producte lliure que li interessi beneficia no solament la mateixa administració, sinó també tots els ciutadans, que podran utilitzar aquest producte per a les seves tasques informàtiques, potser amb les millores aportades per l'administració.

Bibliografia

El lector interessat en un informe sobre els avantatges del programari lliure per a l'administració, escrit en el context nord-americà de 1999, pot consultar "The case for government promotion of open source software" (Mitch Stoltz, 1999) [213].

Nota

Un cas molt especial, però de gran impacte, que mostra aquest millor aprofitament dels recursos públics és la localització (adaptació als usos i costums d'una comunitat) d'un programa. Encara que l'aspecte més visible de la localització és la traducció del programa i la seva documentació, n'hi ha altres que es veuen també afectats per ella (des de la utilització de símbols de la moneda local o la presentació de la data i l'hora en formats habituals en la comunitat en qüestió, fins a l'ús d'exemples i formes d'expressió adequats als costums locals en la documentació).

En qualsevol cas, és clar que si una administració pública dedica recursos que una determinada aplicació sigui localitzada per a les seves necessitats, és més que probable que aquestes necessitats coincideixin amb les dels seus ciutadans, de manera que no solament aconseguirà tenir un programa informàtic que satisfaci les seves necessitats, sinó que també podrà posar-lo, sense despesa extra, a disposició de qualsevol ciutadà que pugui utilitzar-lo amb profit. Per exemple, quan una administració pública finança l'adaptació d'un programa ofimàtic a una llengua que es parla en el seu àmbit d'actuació, no solament podrà usar-lo a les seves pròpies dependències, sinó que també podrà oferir-lo als seus ciutadans, amb el que això pot suposar de foment de la societat de la informació.

6.1.2. Dificultats d'adopció i altres problemes

Tanmateix, encara que els avantatges d'ús del programari lliure en les administracions siguin molts, també són moltes les dificultats a què s'han d'afrontar a l'hora de posar-lo en pràctica. Entre aquestes poden destacar-se les següents:

1) Desconeixement i falta de decisió política

El primer problema amb què es troba el programari lliure per a la seva introducció en les administracions és un que, sens dubte, és compartit per altres organitzacions: el programari lliure és encara molt desconegut entre els qui prenen les decisions.

Afortunadament, aquest és un problema que es va solucionant gradualment, però encara en molts àmbits de les administracions el programari lliure és percebut com una cosa *estranya*, de manera que prendre decisions en la línia d'utilitzar-lo implica assumir certs riscos.

Unit a això, sol trobar-se un problema de decisió política. El principal avantatge del programari lliure en l'administració no és el cost (encara que aquest és important, sobretot quan es parla de desplegaments per a gran quantitat de llocs), sinó com ja hem vist, sobretot avantatges *de fons* estratègics. I per tant, queden en gran manera en l'àmbit de les decisions polítiques, no tècniques. Sense voluntat política de canviar els sistemes informàtics i la filosofia amb què es contracten, és difícil avançar en la implantació de programari lliure en l'administració.

2) Poca adequació dels mecanismes de contractació

Els mecanismes de contractació que es fan servir avui en dia en l'administració, des dels models de concurs públic habituals fins a la divisió de la despesa en partides, estan dissenyats fonamentalment per a la compra de productes informàtics, i no tant per a l'adquisició de serveis relacionats amb els programes. Tanmateix, quan s'utilitza programari lliure, habitualment no hi ha producte a comprar o el seu preu és gairebé menyspreable. Al contrari, per aprofitar les possibilitats que ofereix el programari lliure, és convenient poder contractar serveis al seu voltant. Això demana que, abans d'utilitzar seriosament programari lliure, s'hagin dissenyat mecanismes burocràtics adequats que facilitin la contractació en aquests casos.

3) Falta d'estratègia d'implantació

Moltes vegades el programari lliure comença a utilitzar-se en una administració simplement perquè el cost d'adquisició és més baix. És molt habitual en aquests casos que el producte en qüestió s'incorpori al sistema informàtic sense una planificació especial, i en general, sense una estratègia global d'ús i aprofitament de programari lliure. Això causa que la major part dels seus avantatges es perdin pel camí, ja que tot queda en l'ús d'un *producte més barat*, quan ja hem vist que, en general, els més grans avantatges són d'un altre tipus.

Si a això hi unim que si no es fa un disseny adequat del canvi, l'ús de programari lliure pot suposar uns costos de transició no menyspreables, veurem que certes experiències aïllades, i fora d'un marc clar, d'ús de programari lliure en l'administració poden resultar fallides i frustrants.

4) Escassetat o absència de productes lliures en certs segments

La implantació de programari lliure en qualsevol organització pot xocar amb la falta d'alternatives lliures de la qualitat adequada per a cert tipus d'aplicacions. En aquests casos, la solució és complicada: l'únic que es pot fer és tractar de promoure l'aparició del producte lliure que es necessita.

Afortunadament, les administracions públiques estan en una bona posició per a estudiar seriosament si els convé fomentar, o fins i tot finançar o cofinançar, el desenvolupament d'aquest producte. Cal recordar que entre els seus objectius sol haver-hi, per exemple, que els seus administrats puguin accedir millor a la societat de la informació o el foment del teixit industrial local. Sens dubte, la creació de molts programes lliures incidirà positivament en ambdós objectius, per la qual cosa al mer càlcul de cost/benefici directe caldria sumar-hi els beneficis indirectes que aquesta decisió causaria.

5) Interoperabilitat amb sistemes existents

No és habitual que s'emprengui una migració a programari lliure amb tot el sistema informàtic complet, alhora. És per això que és important que la part que es vol migrar continuï funcionant correctament en el marc de la resta del programari amb què haurà d'interoperar. Aquest és un problema

conegut en qualsevol migració (encara que sigui a un producte privatiu), però pot tenir una especial incidència en el cas del programari lliure. En qualsevol cas, serà una cosa que caldrà tenir en compte en estudiar la transició. Afortunadament, moltes vegades es poden fer adaptacions del programari lliure que s'ha d'instal·lar perquè interoperi adequadament amb altres sistemes, però si és el cas, caldrà considerar aquest punt en pressupostar les despeses de la migració.

6) Migració de les dades
Aquest és un problema genèric de qualsevol migració a noves aplicacions que utilitzin un format de dades diferent, fins i tot si són privatives. De fet, en el cas del programari lliure aquest problema de vegades està molt mitigat, ja que és habitual fer un esforç especial per preveure quants formats i estàndards d'intercanvi de dades sigui possible. Però habitualment és necessari migrar les dades. I el cost de fer-ho és alt. És per això que, en fer el càlcul del que pot suposar una migració a programari lliure, aquest és un factor que caldrà tenir molt en compte.

6.2. Actuacions de les administracions públiques al món del programari

Les administracions públiques actuen sobre el món del programari almenys de tres formes:

- Comprant programes i serveis relacionats amb ells. Les administracions, com a grans usuaris d'informàtica, són un actor fonamental al mercat del programari.

- Promovent de diverses formes l'ús (i l'adquisició) de certs programes en la societat. Aquesta promoció es fa de vegades oferint incentius econòmics (desgravacions fiscals, incentius directes, etc.), de vegades amb informació i recomanacions, de vegades per "efecte exemple"...

- Finançant (directament o indirectament) projectes d'investigació i desenvolupament que dissenyin el futur de la informàtica.

En cada un d'aquests àmbits el programari lliure pot presentar avantatges específics (a més dels ja descrits en l'apartat anterior) interessants tant per a l'administració com per a la societat en general.

6.2.1. Com satisfer millor les necessitats de les administracions públiques?

Les administracions públiques són grans consumidores d'informàtica. Pel que fa al programari, comprèn habitualment tant productes de consum massiu (*off-the-shelf*) com sistemes a mida. Des d'aquest punt de vista, són fonamen-

talment grans centres de compres, similars a les grans empreses, encara que amb les seves pròpies peculiaritats. Per exemple, en molts àmbits se suposa que les decisions de compra de les administracions públiques no han de tenir en compte simplement paràmetres de cost davant funcionalitat, sinó altres, com impacte de la compra en el teixit industrial i social o consideracions estratègiques a llarg termini, que poden ser també importants.

En qualsevol cas, el que és habitual avui en dia en programari de consum massiu és utilitzar els productes propietaris líders del mercat. La quantitat de recursos públics que es gasten els ajuntaments, les comunitats autònomes, l'Administració central i les administracions europees a comprar llicències de Windows, Office o altres productes similars és certament considerable. Però progressivament les solucions lliures van penetrant en aquest mercat. Cada vegada més, s'estan considerant solucions basades en programari lliure per als servidors, i productes com ara OpenOffice.org, i GNU/Linux amb GNOME o KDE són cada vegada més considerats a l'escriptori.

Què es pot guanyar amb aquesta migració a programari lliure? Per il·lustrar-ho, considerem l'escenari següent. Suposem que amb una fracció del que han gastat en dos o tres productes propietaris "estrella" totes les administracions europees (o probablement les de qualsevol país desenvolupat de mida mitjana), es podria promoure un concurs públic que una empresa (o dues, o tres, o quatre) millorés i adaptés els programes lliures ara disponibles perquè en el termini d'un o dos anys estiguessin llestos per al seu ús massiu almenys per a certes tasques típiques (si no ho estiguessin ja). Imaginem per exemple un esforç coordinat, a escala nacional o europea, perquè totes les administracions participessin en un consorci que s'encarregués de la gestió d'aquests concursos. En poc temps hi hauria una indústria "local" especialitzada a realitzar aquestes millores i aquestes adaptacions. I les administracions podrien escollir entre les tres o quatre distribucions lliures produïdes per aquesta indústria. Per fomentar la competència, es podria recompensar econòmicament cada empresa segons la quantitat d'administracions que elegissin usar la seva distribució. I tot el resultat d'aquesta operació, en ser programari lliure, estaria també a disposició d'empreses i usuaris individuals, que en molts casos tindrien necessitats similars a les de les administracions.

En el cas del programari fet a mida, el procés habitual de moment passa normalment per contractar amb una empresa els programes necessaris sota un model propietari. Tot el desenvolupament realitzat a petició de l'administració és propietat de l'empresa que el desenvolupa. I normalment l'administració contractant queda lligada al seu proveïdor per a tot el que tingui a veure amb millores, actualitzacions i suport, en un cercle viciós que dificulta molt la competència i alenteix el procés de modernització de les administracions públiques. El que és pitjor, moltes vegades el mateix programa és venut una i altra vegada a administracions similars, aplicant en cada nou cas els costos que hauria suposat fer el desenvolupament des de zero.

Considerem de nou un escenari que mostri com podrien ser les coses d'una altra forma. Un consorci d'administracions públiques amb necessitats d'un cert programari a mida podria exigir que el resultat obtingut fos programari lliure. Això permetria que altres administracions es beneficiessin també del treball i que a mitjà termini estiguessin interessades a col·laborar en el consorci perquè es tinguessin en compte les seves necessitats peculiars. En ser lliure el programari resultant, no hi hauria obligació de contractar les millores i les adaptacions al mateix proveïdor, de manera que s'introduiria competència en aquest mercat (que ara com ara és gairebé captiu). I en qualsevol cas, el cost final per a qualsevol de les administracions implicades no seria mai major que si s'hagués utilitzat un model propietari.

Són aquests escenaris ciència-ficció? Com es veurà més endavant, ja hi ha tímides iniciatives en direccions similars a les que s'hi exposen. A més d'ajudar a crear i mantenir una indústria en l'àmbit de l'administració pública que realitza la compra, el programari lliure té més avantatges específics en l'entorn públic. Per exemple, és la forma més efectiva de tenir programari desenvolupat en llengües minoritàries (preocupació essencial de moltes administracions públiques). També pot ajudar molt en el manteniment de la independència estratègica a llarg termini i a assegurar l'accés a les dades que custodien les administracions públiques d'aquí a molt temps. Per tot això, les entitats públiques estan cada vegada més interessades en el programari lliure com a usuàries.

Alguns casos relacionats amb administracions alemanyes

El juliol del 2003 es va alliberar la primera versió estable de Kolab, un producte del projecte Kroupware. Kolab és un sistema lliure d'ajuda informàtica al treball en grup (*groupware*) basat en KDE. El motiu pel qual esmentem aquest projecte en aquest punt és perquè el seu origen va ser un concurs del Bundesamt für Sicherheit in der Informationstechnik (BSI) del Govern alemany (que podria traduir-se 'Agència Federal de Seguretat en Tecnologies de la Informació). Aquest concurs demanava la provisió d'una solució que interoperés amb Windows i l'Outlook d'una banda i GNU/Linux i KDE per una altra. Entre les ofertes presentades, va guanyar la proposada conjuntament per tres empreses, Erfrakon, Intevation i Klarälvdalens Datakonsult, que proposaven una solució lliure basada parcialment en programari ja desenvolupat pel projecte KDE, completat amb desenvolupaments propis lliures, que van donar lloc a Kolab.

El maig del 2003 l'Ajuntament de Munic (Alemanya) va aprovar la migració a GNU/Linux i aplicacions d'ofimàtica lliures de tots els ordinadors emprats com a escriptoris, uns catorze mil. La decisió de fer-ho no va ser només econòmica: també es van tenir en compte aspectes estratègics i qualitatius, segons van declarar els seus responsables. En l'exhaustiu estudi que es va realitzar prèviament a la decisió, la solució finalment escollida (GNU/Linux més OpenOffice.org, fonamentalment) va obtenir 6.218 punts (d'un màxim de deu mil) davant els poc més de cinc mil que va obtenir la solució "tradicional" basada en programari de Microsoft.

El juliol del 2003 va fer públic el Koordinierungs-und Beratungsstelle der Bundesregierung für Informationstechnik in der Bundesverwaltung (KBSt), del Ministeri de l'Interior alemany, el document *Leitfaden für die Migration von Basissoftwarekomponenten auf Server- und Arbeitsplatzsystemen* [107] ('Guia de migració per a components programari bàsics en servidors i estacions de treball'), que ofereix un conjunt d'orientacions sobre com poden migrar a solucions basades en programari lliure les entitats públiques alemanyes. Aquestes orientacions estan dissenyades perquè qui estigui a càrrec de la presa de decisions pugui avaluar si convé una migració a programari lliure i en quines condicions l'ha de fer si pren aquesta decisió.

6.2.2. Promoció de la societat de la informació

Les entitats públiques dediquen molts recursos a la promoció de plans que incentiven la despesa en informàtica. Aquesta és una eina formidable, que pot ajudar molt a l'extensió de noves tecnologies per la societat. Però també és una eina perillosa. Per exemple, pot no ser molt bona idea promoure l'ús d'Internet en la societat recomanant un determinat navegador que fomenti la posició de monopoli *de facto* d'una empresa, la qual cosa a la llarga podria ser perjudicial per a la societat que s'està tractant de beneficiar.

De nou, el programari lliure pot ajudar en aquestes situacions. En primer lloc, és neutre davant fabricants, ja que ningú no té l'exclusiva de cap programa lliure. Si una administració vol promoure l'ús d'una família de programes lliures, pot convocar concursos, als quals es pot presentar qualsevol empresa del sector, per gestionar el seu lliurament als ciutadans, la seva millora o la seva ampliació amb les funcionalitats que es desitgi, etc. En segon lloc, pot ajudar molt en els aspectes econòmics. Per exemple, en molts casos pot utilitzar-se la mateixa quantitat de recursos en adquirir una certa quantitat de llicències d'un programa propietari perquè en facin ús els ciutadans, que en adquirir una còpia d'un de lliure i contractar suport o adaptacions per a aquest; o fins i tot en negociar amb un productor de programari propietari la compra dels drets del seu producte per a convertir-lo en programari lliure.

En un altre àmbit, podem imaginar que part de la quantitat destinada a un programa d'informatització d'escoles es dedica a crear una distribució de GNU/Linux adaptada a les necessitats de la docència en ensenyament primari. I amb la resta dels recursos es contracta suport perquè el programari sigui mantingut en aquestes escoles, de manera que no sigui un simple "programari gerro", sinó que realment hi hagi gent encarregada del seu funcionament correcte. D'aquesta manera no solament es cobreixen les necessitats del sistema educatiu, sinó que a més es genera un mercat per a empreses, habitualment d'àmbit local, capaces d'oferir serveis de manteniment. I sens dubte, es deixa completament obert el camí del futur: el programari no quedarà obsolet en pocs anys i això obligarà a recomençar des de zero, sinó que es podrà anar actualitzant incrementadament, any rere any, per a mantenir els beneficis del programa amb una inversió similar.

Nota

El lector coneixedor de les iniciatives públiques respecte al programari lliure reconeixerà en aquest exemple el cas de LinEx. A finals del 2001 la Junta d'Extremadura (Espanya) va decidir utilitzar una distribució de GNU/Linux per a informatitzar tots els col·legis públics de la regió. Per a això, va finançar la construcció de LinEx, una distribució basada en Debian GNU/Linux que va ser anunciada la primavera del 2002, i es va encarregar que fos un requisit en tots els concursos d'adquisició d'equipament informàtic per a centres educatius. A més, va iniciar programes de formació i capacitació de professors, de creació de materials docents i d'extensió de l'experiència a altres àmbits. A mitjan 2003, l'experiència sembla un èxit, que ja s'està estenent institucionalment a altres regions (per exemple, a Andalusia, també a Espanya, mitjançant el projecte Guadalinex).

6.2.3. Foment de la investigació

També en política d'R+D el programari lliure té interessants avantatges que val la pena analitzar. Amb diners públics s'estan finançant els desenvolupaments d'una gran quantitat de programari de la qual la societat no acaba beneficiant-se ni tan sols indirectament. Habitualment, els programes públics de foment de la investigació i el desenvolupament financen de manera totalment o parcialment projectes que creen programes sense atendre quins drets tindrà el públic sobre ells. En molts casos els resultats, sense un pla adequat de comercialització, simplement queden en alguna caixa, coberts de pols. En d'altres, les mateixes persones que van finançar un programa mitjançant impostos acaben pagant-lo de nou si el volen utilitzar (ja que han d'adquirir llicències d'ús).

El programari lliure ofereix una opció interessant, que a poc a poc les autoritats encarregades de la política d'innovació en moltes administracions estan considerant amb atenció. Especialment quan la investigació és precompetitiva (el més habitual en els casos de finançament públic), el fet que els programes resultants siguin lliures permet que la indústria en el seu conjunt (i per tant la societat) es beneficiï molt dels diners públics gastats en R+D en el camp del programari. On una empresa veu un resultat d'impossible comercialització, una altra pot veure una oportunitat de negoci. Així, d'una banda, es maximitzen els resultats dels programes d'investigació, i per una altra, s'afavoreix la competència entre les empreses que vulguin utilitzar els resultats d'un projecte, ja que totes elles competiran a partir dels mateixos programes resultat del projecte.

Aquest model no és nou. En gran manera és el que ha permès el desenvolupament d'Internet. Si les administracions públiques exigeixen que els resultats de la investigació realitzada amb els seus fons siguin distribuïts en forma de programes lliures, no seria estrany que apareguessin, a diverses escales, casos similars. O bé els resultats d'aquestes investigacions són dolents o inútils (i en aquest cas hauria de replantejar-se la forma de selecció dels projectes), o bé la dinamització que suposaria deixar-los llestos perquè qualsevol empresa pogués convertir-los en producte permetria desenvolupaments senzillament impredictibles.

6.3. Exemples d'iniciatives legislatives

En els apartats següents es repassen algunes iniciatives legislatives concretes relacionades amb l'ús i la promoció del programari lliure per part de les administracions públiques. Sens dubte, la llista oferta no pretén ser exhaustiva, i s'ha centrat en les iniciatives que han estat pioneres des d'algun punt de vista (fins i tot si no han estat aprovades finalment). El lector interessat a completar-la pot consultar "GrULIC. Legislación sobre el uso de software libre en el

Estado" [133], on s'esmenen molts més casos similars. A més, en un apèndix (apèndix D) s'inclou amb finalitats il·lustratives el text íntegre, o les parts més rellevants, de diverses d'aquestes iniciatives.

6.3.1. Projectes de llei a França

El 1999 i el 2000 es van presentar a França dos projectes de llei relacionats amb el programari lliure que van ser els pioners d'una llarga sèrie de debats legislatius sobre la matèria:

- El Projecte de llei 1999-495, proposat per Laffitte, Trégouet i Cabanel, va ser exposat al servidor web del Senat de la República Francesa a partir d'octubre de 1999. Després d'un procés de discussió pública per Internet (http://www.senat.fr/consult/loglibre/index.htm) [102] que es va prolongar durant dos mesos, el Projecte va sofrir algunes modificacions. El resultat és el Projecte de Llei número 2000-117 (Laffitte, Trégouet i Cabanel, Proposition de Loi numéro 117, Senat de la República Francesa, 2000) [162], que advocava per l'ús obligatori de programari lliure en l'Administració, i preveia excepcions i mesures transitòries per a aquells casos en els quals això no fos encara tècnicament possible, en un marc més general, que intentava generalitzar en l'Administració francesa l'ús d'Internet i del programari lliure.

- L'abril del 2000 els diputats Jean-Yves Le Déaut, Christian Paul i Pierre Cohen van proposar una nova llei l'objectiu de la qual era similar al projecte de Laffitte, Trégouet i Cabanel: reforçar les llibertats i la seguretat del consumidor, així com millorar la igualtat de drets en la societat de la informació.
  Tanmateix, a diferència del projecte de llei de Laffitte, Trégouet i Cabanel, aquest altre no forçava a utilitzar programari lliure en l'Administració. Aquest projecte de llei se centrava en el fet que el programari utilitzat en l'Administració tingués el codi font disponible, si bé no obligava que aquest es distribuís amb llicències de programari lliure.
  Per aconseguir els seus objectius els legisladors pretenien garantir el "dret a la compatibilitat" del programari, proporcionant mecanismes que portessin a la pràctica el principi d'interoperabilitat plasmat en la Directiva del Consell Europeu relativa a la protecció jurídica de programes d'ordinador (Directiva 91/250/CEE del Consell, de 14 de maig de 1991, relativa a la protecció jurídica de programes d'ordinador, 1991) [111].

Cap dels dos projectes francesos no es va convertir en llei, però ambdós han servit d'inspiració a la majoria de les iniciatives posteriors a tot el món, i és per això que és especialment interessant el seu estudi. El segon (el proposat per Le Déaut, Paul i Cohen) perseguia la compatibilitat i la interoperabilitat del programari, posant èmfasi en la disponibilitat del codi font del programari utilitzat en l'Administració. Tanmateix, no requeria que les aplicacions

desenvolupades fossin programari lliure, entès com aquell que es distribueix amb llicències que garanteixen la llibertat de modificació, ús i redistribució del programa.

Més endavant (apartat D.1 i apartat D.2 dels apèndixs) reproduirem de manera gairebé íntegra l'articulat i l'exposició de motius d'ambdós projectes de llei. En particular, són d'especial interès les exposicions de motius, que destaquen quins són els problemes que acuiten avui dia les administracions públiques en relació amb l'ús de programari en general.

6.3.2. Projecte de llei al Brasil

El diputat Walter Pinheiro va presentar el desembre de 1999 un projecte de llei sobre el programari lliure a la Cambra Federal del Brasil (Proposição pl-2269/ 1999. Dispõe sobre a utilização de programas abertos pelos entes de direito público e de direito privado sob controle acionário da administração pública, Cambra dels Diputats del Brasil, desembre de 1999) [185]. Aquest projecte afectava la utilització de programari lliure en l'Administració pública i en les empreses privades controlades accionarialment per l'Estat.

S'hi recomana l'ús de programari lliure en aquestes entitats que no tingui restriccions quant al seu préstec, modificació o distribució. L'articulat de la llei descriu de manera detallada què s'entén per programari lliure i com han de ser les llicències que l'acompanyin. Les definicions coincideixen amb la definició clàssica de programari lliure del projecte GNU. En l'exposició de motius es repassa la història del projecte GNU, i s'analitza els seus avantatges i assoliments. Així mateix es fa referència a la situació actual del programari lliure, utilitzant com a exemple el sistema operatiu GNU/Linux.

Una de les parts més interessants, l'article primer, deixa ben clar l'àmbit en el qual es proposa l'ús de programari lliure (tenint en compte que la definició que s'ofereix als articles posteriors per a "programa obert" és, com ja s'ha dit, programari lliure):

> "L'Administració pública en tots els nivells, els poders de la República, les empreses estatals i d'economia mixta, les empreses públiques i tots els altres organismes públics o privats subjectes al control de la societat brasilera estan obligats a utilitzar preferentment, en els seus sistemes i equipaments d'informàtica, programes oberts lliures de restriccions propietàries quant a la seva cessió, modificació i distribució. "

6.3.3. Projectes de llei al Perú

Són diversos els projectes de llei relacionats amb l'ús del programari lliure en les administracions públiques que s'han proposat al Perú ("GNU Perú. Proyectos de ley sobre software libre en la Administración pública del gobierno peruano", Congrés de la República) [184]. El primer i el més conegut va ser proposat pel congressista Edgar Villanueva Núñez el desembre del 2001 (Projecte de Llei sobre programari lliure número 1609, desembre de 2001) [222]. S'hi defineix *programari lliure* segons la definició clàssica de les quatre llibertats (a la qual se

li dóna potser una major precisió legal, amb una definició que especifica sis característiques que ha de tenir un programa lliure) i se'n proposa la utilització exclusiva en l'Administració peruana:

> "Article 2. Els poders executiu, legislatiu i judicial, els organismes descentralitzats i les empreses on l'Estat tingui majoria accionària, empraran en els seus sistemes i equipaments d'informàtica exclusivament programes o programari lliures."

No obstant això, una mica més endavant, els articles 4 i 5 inclouen algunes excepcions a aquesta regla.

Al seu dia aquesta proposta de llei va tenir gran repercussió mundial. D'una banda, va ser la primera vegada que es va proposar l'ús exclusiu del programari lliure en una administració. Però fins i tot més important per a la seva repercussió que aquesta novetat va ser l'intercanvi epistolar entre el congressista Villanueva i la representació de Microsoft al Perú, que va realitzar al·legacions a aquesta proposta. Aquesta proposta de llei també és interessant per la posició que va prendre sobre això l'ambaixada dels EUA, que fins i tot va arribar a enviar per conducte oficial una notificació (adjuntant un informe elaborat per Microsoft) al Congrés peruà en la qual expressava la seva "preocupació sobre les recents propostes del Congrés de la República per restringir les compres per part del Govern peruà de programari de codi obert o programari lliure" ("Carta al presidente del Congreso de la República", 2002) [147]. Entre altres motius, tant les al·legacions de Microsoft com les de l'ambaixada dels EUA tractaven de mostrar com la proposta de llei discriminaria unes empreses davant d'altres i faria impossible les inversions necessàries per a la creació d'una indústria nacional de creació de programari. Davant d'això, Villanueva va argumentar que la seva proposició de llei no discriminava ni afavoria de cap manera cap empresa en particular, ja que no especificava qui podria ser proveïdor de l'Administració, sinó com (en quines condicions) hauria de realitzar-se la provisió de programari. Aquesta argumentació és molt clara per a entendre com la promoció del programari lliure en l'Administració no perjudica en cap cas la lliure competència entre les empreses subministradores.

Més endavant, els congressistes peruans Edgar Villanueva Núñez i Jacques Rodrich Ackerman van presentar un nou projecte de llei, el número 2485, de 8 d'abril de 2002 (Projecte de llei d'ús de programari lliure en l'Administració pública número 2485, 2002) [223], que l'agost del 2003 segueix el seu tràmit parlamentari. Aquest projecte és una evolució del Projecte de llei 1609 [222], del qual recull diversos comentaris i millores, i pot considerar-se un bon exemple de projecte de llei que proposa l'ús exclusiu de programari lliure en les administracions públiques, llevat de certs casos excepcionals. Pel seu interès, incloem el seu text íntegre (apartat D.3 dels apèndixs). En particular, la seva exposició de motius és un bon resum de les característiques que hauria de tenir el programari utilitzat per les administracions públiques i de com aquestes les compleix millor el programari lliure que el programari propietari.

6.3.4. Projectes de llei a Espanya

A Espanya hi ha hagut diverses iniciatives legislatives relacionades amb el programari lliure. Esmentem-ne unes quantes:

- Decret de mesures d'impuls de la societat del coneixement a Andalusia
  Una de les iniciatives legislatives més importants (per haver entrat en vigor) que han tingut lloc a Espanya ha estat sens dubte l'adoptada per Andalusia. Al Decret de mesures d'impuls de la societat del coneixement a Andalusia (Decret 72/2003, de 18 de març, de mesures d'impuls de la societat del coneixement a Andalusia, 2003) [99] es tracta de l'ús de programari lliure, fonamentalment (però no solament) en l'entorn educatiu.
  Entre altres detalls, fomenta l'ús de programari lliure de forma preferent en els centres docents públics, obliga que tot l'equipament adquirit per a aquests centres sigui compatible amb sistemes operatius lliures, i el mateix per als centres de la Junta que ofereixin accés públic a Internet.

- Proposició de llei de programari lliure en el marc de l'Administració pública de Catalunya
  En altres comunitats s'han discutit proposicions més ambicioses, però sense que hagin aconseguit la majoria necessària. Entre elles, la més coneguda és probablement la debatuda en el Parlament de Catalunya (Proposició de llei de programari lliure en el marc de l'Administració pública de Catalunya, 2002) [221], molt similar a la que el mateix partit (Esquerra Republicana de Catalunya) va presentar en el Congrés dels Diputats, de la qual parlarem a continuació. Aquesta proposta no va prosperar quan va ser sotmesa a votació.

- Projecte de llei de Puigcercós Boixassa en el Congrés dels Diputats
  També hi va haver una iniciativa en el Congrés dels Diputats proposada per Joan Puigcercós Boixassa (Esquerra Republicana de Catalunya) (Proposició de llei de mesures per a la implantació del programari lliure en l'Administració de l'Estat, 2002) [188]. Aquesta iniciativa proposa la preferència de l'ús de programari lliure en l'Administració de l'Estat, i en aquest sentit és similar a altres iniciatives amb aquesta finalitat. Tanmateix, té la peculiaritat interessant de fer èmfasi en la disposició dels programes lliures localitzat per als idiomes cooficials (en les comunitats autònomes que els tenen). La iniciativa no va aconseguir ser aprovada en el tràmit parlamentari.

7. Enginyeria del programari lliure

"The best way to have a good idea is to have many of them."

"La millor forma de tenir una bona idea és tenir-ne moltes."

Linus Pauling

Als capítols anteriors s'ha mostrat per què el desafiament del programari lliure no és el d'un nou competidor que sota les mateixes regles genera programari de manera més ràpida, més barata i de millor qualitat: el programari lliure es diferencia del programari "tradicional" en aspectes més fonamentals, començant per raons i motivacions filosòfiques, seguint per noves pautes econòmiques i de mercat, i finalitzant amb una forma diferent de generar programari. L'enginyeria del programari no pot ser aliena a això i des de fa poc més d'un lustre ha anat aprofundint en la investigació de tots aquests aspectes. Aquest capítol pretén mostrar els estudis més significatius i les evidències que aporten amb l'objectiu d'oferir al lector una visió de l'estat de l'art i de les perspectives de futur del que hem convingut a anomenar l'*enginyeria del programari lliure*.

7.1. Introducció

Encara que fa ja diverses dècades que es desenvolupa programari lliure, només des de fa uns quants anys s'ha començat a prestar atenció als seus models i processos de desenvolupament des del punt de vista de l'enginyeria del programari. Igual que no hi ha un únic model de desenvolupament de programari propietari, tampoc no hi ha un únic model de programari lliure[5], però tot i així poden trobar-se interessants característiques que comparteixen gran part dels projectes estudiats i que podrien està arrelades en les propietats dels programes lliures.

[(5)]A l'article "The Ecology of Open Source Software Development" (Kieran Healy i Alan Schussman, 2003) [140] es mostra la gran varietat de projectes, així com la seva diversitat quant al nombre de desenvolupadors, a l'ús d'eines i a les descàrregues.

El 1997 Eric S. Raymond va publicar el primer article àmpliament difós, "La catedral y el bazar" (*The cathedral and the bazaar. Musings on Linux and open source by an accidental revolutionary*, O'Reilly & Associates –http://www.ora.com–, 2001) [192], en el qual es descrivien algunes característiques dels models de desenvolupament de programari lliure, fent especial èmfasi en el que diferenciava aquests models dels de desenvolupament propietari. Aquest article s'ha convertit des de llavors en un dels més coneguts (i criticats) del món del programari lliure, i fins a cert punt, en el senyal de començament per al desenvolupament de l'enginyeria del programari lliure.

7.2. "La catedral y el bazar"

Raymond estableix una analogia entre la forma de construir les catedrals medievals i la manera clàssica de produir programari. Argumenta que en ambdós casos hi ha una clara distribució de tasques i funcions, en la qual destaca l'existència d'un dissenyador que està per damunt de tot i que ha de controlar tothora el desenvolupament de l'activitat. Així mateix, la planificació està estrictament controlada, la qual cosa dóna lloc a uns processos clarament detallats en els quals idealment cada participant en l'activitat té un paper específic molt delimitat.

Dins del que Raymond pren com el model de creació de catedrals no només tenen cabuda els processos pesats que podem trobar en la indústria del programari (el model en cascada clàssic, els diferents vessants del Rational Unified Process, etc.), sinó també projectes de programari lliure, com és el cas de GNU i NetBSD. Per a Raymond, aquests projectes es troben fortament centralitzats, ja que unes quantes persones són les que realitzen el disseny i la implementació del programari. Les tasques que exerceixen aquestes persones, així com les seves funcions, estan perfectament definides, i algú que volgués entrar a formar part de l'equip de desenvolupament necessitaria que se li assignés una tasca i una funció segons les necessitats del projecte. D'altra banda, els lliuraments d'aquest tipus de programes es troben espaiats en el temps a partir d'una planificació bastant estricta. Això suposa tenir pocs lliuraments del programari i cicles llargs, que consten de diverses etapes, entre els lliuraments.

El model antagònic al de la catedral és el basar. Segons Raymond, alguns dels programes de programari lliure, en especial el nucli Linux, s'han desenvolupat seguint un esquema similar al d'un basar oriental. En un basar no hi ha una màxima autoritat que controli els processos que es van desenvolupant ni que planifiqui estrictament el que ha de succeir. D'altra banda, els rols dels participants poden canviar de manera contínua (els venedors poden passar a ser clients) i sense indicació externa.

Però el més nou de "La catedral y el bazar" és la descripció del procés que ha fet de Linux un èxit dins del món del programari lliure; és una successió de "bones maneres" per a aprofitar al màxim les possibilitats que ofereix la disponibilitat de codi font i la interactivitat mitjançant l'ús de sistemes i eines telemàtics.

Un projecte de programari lliure sol sorgir arran d'una acció purament personal; la causa s'ha de buscar en un desenvolupador que vegi limitada la seva capacitat de resoldre un problema. El desenvolupador ha de tenir els coneixements necessaris per a, almenys, començar a resoldre'l. Una vegada que hagi aconseguit tenir una cosa utilitzable, amb una mica de funcionalitat, senzill i, si pot ser, ben dissenyat o escrit, el millor que pot fer és compartir aquesta solució amb la comunitat del programari lliure. És el que es denomina *publicació*

primerenca (release early en anglès), que permet cridar l'atenció d'altres persones (generalment desenvolupadors) que tinguin el mateix problema i que puguin estar interessats en la solució.

Un dels principis bàsics d'aquest model de desenvolupament és considerar els usuaris com a codesenvolupadors. Se'ls ha de tractar amb afecte, no solament perquè poden proporcionar publicitat mitjançant el "boca a boca", sinó també pel fet que realitzaran una de les tasques més costoses que hi ha en la generació de programari: les proves. Al contrari que el codesenvolupament, que és difícilment escalable, la depuració i les proves tenen la propietat de ser altament paral·lelitzables. L'usuari serà qui prengui el programari i el provi a la seva màquina sota unes condicions específiques (una arquitectura, unes eines, etc.), una tasca que multiplicada per un gran nombre d'arquitectures i entorns suposaria un gran esforç per a l'equip de desenvolupament.

Si es tracta els usuaris com a codesenvolupadors pot donar-se el cas que n'hi hagi algun que trobi un error i el resolgui enviant un pedaç al desenvolupador del projecte perquè el problema estigui solucionat en la següent versió. També pot succeir, per exemple, que una persona diferent de la que descobreix un error sigui la que finalment l'entengui i el corregeixi. En qualsevol cas, totes aquestes circumstàncies són molt profitoses per al desenvolupament del programari, per la qual cosa és molt beneficiós entrar en una dinàmica d'aquesta índole.

Aquesta situació es torna més efectiva amb lliuraments freqüents, ja que la motivació per trobar, notificar i corregir errors és alta perquè se suposa que s'atendrà immediatament. A més s'aconsegueixen avantatges secundaris, com el fet que en integrar freqüentment –idealment una o més vegades al dia– no faci falta una última fase d'integració dels mòduls que componen el programa. Això s'ha denominat *publicació freqüent* (que en la terminologia anglosaxona es coneix com a *release often*) i possibilita una gran modularitat (Alessandro Narduzzo i Alessandro Rossi, "Modularity in action: GNU/Linux and free/open source software development model unleashed", maig 2003) [176], alhora que maximitza l'efecte propagandístic que té la publicació d'una nova versió del programari.

Nota

La gestió de noves versions depèn, lògicament, de la mida del projecte, ja que els problemes a què cal enfrontar-se no són iguals quan l'equip de desenvolupament té dos components que quan en té centenars. Mentre que, en general, en els projectes petits aquest procés és més aviat informal, la gestió de lliuraments en els projectes grans sol seguir un procés definit no exempt de certa complexitat. Hi ha un article titulat "Release management within open source projects" (Justin R. Ehrenkrantz, 2003) [110] que descriu detalladament la seqüència que se segueix al servidor web Apache, el nucli Linux i el sistema de versions Subversion.

Per a evitar que la publicació freqüent espanti usuaris que prioritzin l'estabilitat sobre la rapidesa amb què el programari evoluciona, alguns projectes de programari lliure compten amb diverses branques de desenvolupament en

paral·lel. El cas més conegut és el del nucli Linux, que té una branca estable –dirigida a aquelles persones que estimen la seva fiabilitat– i una altra d'inestable –pensada per a desenvolupadors– amb les últimes innovacions i novetats.

7.3. Lideratge i presa de decisions al basar

Raymond suposa que tot projecte de programari lliure ha de comptar amb un *dictador benevolent*, una espècie de líder –que generalment coincideix amb el fundador del projecte– que ha de guiar el projecte i que es reserva sempre l'última paraula en la presa de decisions. Les habilitats que ha de tenir aquesta persona són principalment les de saber motivar i coordinar un projecte, entendre els usuaris i codesenvolupadors, buscar consensos i integrar tot aquell que pugui aportar alguna cosa. Com es pot observar, entre els requisits més importants no s'ha esmentat que sigui tècnicament competent, encara que mai no serà sobrer.

Amb el creixement en mida i en nombre de desenvolupadors d'alguns projectes de programari lliure, han anat apareixent noves formes d'organitzar la presa de decisions. Linux, per exemple, disposa d'una estructura jeràrquica basada en la delegació de responsabilitats per part de Linus Torvalds, el "dictador benevolent". Així, veiem com hi ha informes de Linux que compten amb els seus propis "dictadors benevolents", encara que aquests vegin limitat el seu "poder" pel fet que Linus Torvalds sigui qui decideixi en darrer terme. Aquest cas és un clar exemple de com l'alta modularitat existent en un projecte de programari lliure ha propiciat una forma d'organització i de presa de decisions específica (Alessandro Narduzzo i Alessandro Rossi, Modularity in action: "GNU/Linux and free/open source software development model unleashed" 2003) [176].

Nota

Hi ha qui diu que la forma d'organitzar-se dins dels projectes de programari lliure s'assembla a la d'un equip quirúrgic, com va proposar Harlan Mills (d'IBM) a començaments de la dècada dels setanta i va popularitzar Brooks al seu famós llibre *The mythical man-month* (Frederick Brooks Jr., 1975) [150]. Encara que es poden donar casos en els quals l'equip de desenvolupament d'alguna aplicació de programari lliure estigui format per un dissenyador/desenvolupador principal (el cirurgià) i per molts codesenvolupadors que realitzen tasques auxiliars (administració de sistemes, manteniment, tasques especialitzades, documentació...), no hi ha en cap cas una separació tan estricta i definida com la proposada per Mills i Brooks. De tota manera, com bé apunta Brooks en el cas de l'equip quirúrgic, en el programari lliure el nombre de desenvolupadors que han de comunicar-se per a crear un sistema gran i complex és més reduït que el nombre total de desenvolupadors.

En el cas de la Fundació Apache ens trobem amb una *meritocràcia*, ja que l'esmentada institució té un comitè de directors format per persones que han contribuït de manera notable al projecte. En realitat, no es tracta d'una meritocràcia rigorosa en la qual els que més hi contribueixen són els que governen, ja que el comitè de directors és elegit democràticament i periòdicament pels membres de la Fundació (que s'encarrega de gestionar diversos projectes de programari lliure, com Apache, Jakarta, etc.). Per arribar a ser membre de la

Fundació Apache, s'ha d'haver aportat de forma continuada i important un o diversos projectes de la Fundació. Aquest sistema també és utilitzat en altres projectes grans, com ara FreeBSD o GNOME.

Un altre cas interessant d'organització formal és el GCC Steering Committee. Va ser creat el 1998 per a evitar que ningú no aconseguís el control del projecte GCC (GNU Compiler Collection, el sistema de compilació de GNU) i ratificat per l'FSF (promotora del projecte GNU) pocs mesos després. En cert sentit és un comitè continuador de la tradició d'un de similar que hi havia hagut en el projecte EGCS (que durant un temps va ser paral·lel al projecte GCC, però que més tard es va unir a aquest). La seva missió fonamental és assegurar-se que el projecte GCC respecta la declaració d'objectius (*mission statement*) del projecte. Els membres del comitè ho són a títol personal, i són elegits pel mateix projecte de manera que representin, amb certa fidelitat, les diferents comunitats que col·laboren en el desenvolupament de GCC (desenvolupadors de suport per a diversos llenguatges, desenvolupadors relacionats amb el nucli, grups interessats en programació encastada, etc.).

El lideratge d'un projecte de programari lliure per part d'una mateixa persona no té per què ser etern. Es poden donar bàsicament dues circumstàncies per les quals un líder de projecte deixi de ser-ho. La primera d'aquestes és la falta d'interès, temps o motivació per a seguir endavant. En aquest cas, s'ha de passar el testimoni a un altre desenvolupador que assumeixi la funció de líder del projecte. Estudis recents (Jesús M. González Barahona i Gregorio Robles, 2003) [87] demostren que, en general, els projectes solen tenir un lideratge que canvia de mans amb freqüència, de manera que es poden observar diverses *generacions* de desenvolupadors en el temps. El segon cas és més problemàtic: es tracta d'una bifurcació. Les llicències de programari lliure permeten prendre el codi, modificar-lo i redistribuir-lo per qualsevol persona sense que faci falta el vistiplau del líder del projecte. Això no se sol donar generalment, excepte en els casos en els quals es faci a fi d'evitar de manera deliberada precisament el líder del projecte (i un possible vet seu a l'aportació). Estem certament davant d'una espècie de "cop d'estat", d'altra banda totalment lícit i legítim. És per això que un dels objectius que busca un líder de projecte, en mantenir satisfets els seus codesenvolupadors, és minimitzar la possibilitat que existeixi una bifurcació.

7.4. Processos en el programari lliure

Encara que el programari lliure no està necessàriament associat amb un procés de desenvolupament programari específic, hi ha un ampli consens sobre els processos més comuns que s'utilitzen. Això no vol dir que no hi hagi projectes de programari lliure que hagin estat creats utilitzant processos clàssics, com el model en cascada. Generalment, el model de desenvolupament en projectes

de programari lliure sol ser més informal, a causa que gran part de l'equip de desenvolupament realitza aquestes tasques de manera voluntària i sense recompensa econòmica, almenys directa, a canvi.

La forma en la qual es capturen requisits al món del programari lliure depèn tant de l'"edat" com de la mida del projecte. En les primeres etapes, el fundador del projecte i l'usuari solen coincidir en una mateixa persona. Més endavant, i si el projecte creix, la captura de requisits sol tenir lloc per mitjà de les llistes de correu electrònic i se sol arribar a una clara distinció entre l'equip de desenvolupament –o almenys, els desenvolupadors més actius– i els usuaris. Per a projectes grans, amb molts usuaris i molts desenvolupadors, la captura de requisits es fa mitjançant la mateixa eina que s'utilitza per a gestionar els errors del projecte. En aquest cas, en comptes de parlar d'errors, es refereixen a activitats, encara que el mecanisme utilitzat per a la seva gestió és idèntic al de la correcció d'errors (se les classificarà segons importància, dependència, etc., i es podrà monitoritzar si han estat resoltes o no). L'ús d'aquesta eina per a la planificació és més aviat recent, per la qual cosa es pot observar que al món del programari lliure hi ha una certa evolució des de la manca absoluta fins a un sistema centralitzat de gestió d'enginyeria d'activitats, encara que indubtablement aquest sigui bastant limitat. En qualsevol cas, no sol ser comú un document que reculli els requisits, tal com és normal en el model en cascada.

Quant al disseny global del sistema, només els grans projectes solen tenir-lo documentat de manera exhaustiva. En la resta, el més probable és que el desenvolupadors o els desenvolupadors principals siguin els únics que el posseeixin –de vegades només en la seva ment– o que es vagi forjant en versions posteriors del programari. La manca d'un disseny detallat no solament implica limitacions quant a la possible reutilització de mòduls, sinó que també és un gran obstacle a l'hora de permetre l'accés a nous desenvolupadors, ja que aquests s'hauran d'enfrontar a un procés d'aprenentatge lent i costós. El disseny detallat, per la seva part, tampoc no està molt generalitzat. La seva absència implica que es perdin moltes possibilitats de reutilització de codi.

La implementació és la fase en què es concentra el major esforç per part dels desenvolupadors de programari lliure, entre altres raons perquè als seus ulls és manifestament la més divertida. Per a això se sol seguir el paradigma de programació clàssic de prova i error fins que s'aconsegueixen els resultats desitjats des del punt de vista subjectiu del programador. Històricament, és estrany el cas en el qual s'han inclòs proves unitàries conjuntament amb el codi, encara que facilitarien la modificació o la inclusió de codi posterior per part d'altres desenvolupadors. En el cas d'alguns projectes grans, com per exemple Mozilla, existeixen màquines dedicades exclusivament a descarregar-se els dipòsits que contenen el codi més recent per a compilar-lo per a diferents arquitectures ("An overview of the software engineering process and tools in the Mozilla project" 2002) [193]. Els errors detectats es notifiquen a una llista de correu de desenvolupadors.

Tanmateix, les proves automàtiques no estan molt arrelades. En general, seran els mateixos usuaris, dins de la gran varietat d'usos, arquitectures i combinacions, els que les realitzaran. Això té l'avantatge que es paral·lelitzen a un cost mínim per a l'equip de desenvolupament. El problema que planteja aquest model és com organitzar-se perquè la realimentació per part dels usuaris existeixi i sigui tan eficient com es pugui.

Quant al manteniment del programari al món del programari lliure –entenentlo com el manteniment de versions antigues–, aquesta és una tasca l'existència de la qual depèn del projecte. En projectes on es requereix una gran estabilitat, com per exemple nuclis del sistema operatiu, es mantenen versions antigues, ja que un canvi a una nova versió pot resultar traumàtic. Però en general, en la majoria dels projectes de programari lliure, si es té una versió antiga i es troba un error, els desenvolupadors no solen posar-se a corregir-lo, sinó que aconsellen utilitzar la versió més moderna amb l'esperança que aquest error hagi desaparegut pel fet que el programari ha evolucionat.

7.5. Crítica a "La catedral y el bazar"

"La catedral y el bazar" pateix d'una falta de sistematicitat i rigor d'acord amb la seva naturalesa més aviat assagística i certament poc científica. Les crítiques més freqüents es refereixen que explica bàsicament una experiència puntual –el cas de Linux– i que es pretenen generalitzar les conclusions per a tots els projectes de programari lliure. En aquest sentit, a "Cave or community? An empirical examination of 100 mature open source projects" [160] es pot veure que l'existència d'una comunitat tan àmplia com la comunitat amb què compta el nucli Linux és més aviat una excepció.

Encara més crítics es mostren aquells que pensen que Linux és un exemple de desenvolupament que segueix el model de desenvolupament de les catedrals. Argumenten que sembla evident que existeix un cap pensant, o almenys una persona que té la màxima potestat, i un sistema jeràrquic mitjançant delegació de responsabilitats fins a arribar als obrers/programadors. Així mateix, existeix repartiment de tasques, encara que sigui de manera implícita. A "A second look at the cathedral and the bazaar" [91] es va més enllà i se sosté –no sense una certa acritud i arrogància en l'argumentació– que la metàfora del basar és internament contradictòria.

Un altre dels punts més criticats de "La catedral y el bazar" és la seva afirmació que la llei de Brooks, que diu que "agregar desenvolupadors a un projecte de programari retardat el retarda encara més" (*The mythical man-month. Essays on software engineering*) [150], no és vàlida al món del programari lliure. A [148] es pot llegir com en realitat el que passa és que les condicions d'entorn són diferents i el que al principi sembla una incongruència amb la llei de Brooks, després d'un estudi més exhaustiu, no deixa de ser un miratge.

7.6. Estudis quantitatius

El programari lliure permet endinsar-se en l'estudi quantitatiu del codi i de la resta de paràmetres que intervenen en la seva generació, atesa l'accessibilitat a aquests. Això permet que àrees de l'enginyeria del programari tradicional –com l'enginyeria del programari empírica– es puguin veure potenciades a causa de l'existència d'una ingent quantitat d'informació a què es pot accedir sense necessitat d'una forta intromissió en el desenvolupament de programari lliure. Els autors estem convençuts que aquesta visió pot ajudar enormement en l'anàlisi i en la comprensió dels fenòmens lligats a la generació de programari lliure (i de programari en general), i que es pot arribar fins i tot, entre altres possibilitats, a tenir models predictius del programari que es realimentin en temps real.

La idea que hi ha darrere és molt simple: "ja que tenim la possibilitat d'estudiar l'evolució d'un nombre immens de projectes de programari lliure, ho fem". I és que a més de l'estat actual d'un projecte, tota la seva evolució en el passat és pública, per la qual cosa tota aquesta informació, degudament extreta, analitzada i empaquetada, pot servir com una base de coneixement que permeti avaluar la *salut* d'un projecte, facilitar la presa de decisions i pronosticar complicacions actuals i futures.

El primer estudi quantitatiu d'una certa importància al món del software lliure data de 1998, encara que va ser publicat a començaments del 2000 ("The orbitin free programari survey") [127]. La seva finalitat era conèixer de manera empírica la participació dels desenvolupadors al món del programari lliure. Per a això es valien del tractament estadístic de l'assignació d'autoria que els autors solen situar a la capçalera dels fitxers de codi font. Es va demostrar que la participació segueix la llei de Pareto ("Course of Political Economy" Lausana, 1896) [182]: el 80% del codi correspon al 20% més actiu dels desenvolupadors, mentre que el 80% restant de desenvolupadors té una aportació d'un 20% del total. Molts estudis posteriors han confirmat i ampliat a altres formes de participació diferents de l'aportació de codi font (missatges a llista de correu, notificació d'errors o fins i tot el nombre de descàrregues, com es poden veure a http://www-mmd.eng.cam.ac.uk/people/fhh10/Sourceforge/Sourceforge%20paper.pdf [145]) la validesa d'aquest resultat.

Nota

El fet que apareguin molts termes de les ciències econòmiques en l'estudi d'enginyeria del programari lliure és conseqüència de l'interès que alguns economistes han mostrat en els últims temps a conèixer i entendre els processos que porten voluntaris a produir béns de gran valor sense obtenir generalment benefici directe a canvi. L'article més conegut és "Cooking pot markets: an economic model for the trade in free goods and services on the Internet" [125], en el qual s'introdueix l'*economia del regal* a Internet. A www.wikipedia.org/wiki/Pareto [232] es pot obtenir més informació sobre el principi de Pareto i la seva generalització en la distribució de Pareto. Així mateix, són interessants la corba de Lorenz (www.wikipedia.org/wiki/Lorenz_curve) [231], que representa de manera gràfica la participació en el projecte dels desenvolupadors, i el coeficient de Gini (www.wikipedia.org/wiki/Gini_coefficient) [230], que es calcula a partir de la corba de

Lorenz i del qual resulta un nombre a partir del qual es pot veure la desigualtat en el sistema.

L'eina que es va utilitzar per a la realització d'aquest estudi va ser publicada amb una llicència lliure pels autors, per la qual cosa s'ofereix tant la possibilitat de reproduir els seus resultats com d'efectuar nous estudis.

En un estudi posterior, Koch ("Results from software engineering research into open source development projects using public data", 2000) [158] va anar més enllà i també va analitzar les interaccions que es duen a terme en un projecte de programari lliure. La font d'informació eren les llistes de correu i el dipòsit de versions del projecte GNOME. Però l'aspecte més interessant de l'estudi de Koch és l'anàlisi econòmica. Koch se centra a comprovar la validesa de predicció de costos clàssics (punts de funció, model de COCOMO...) i mostra els problemes que la seva aplicació comporta, encara que certament admet que els resultats que obté –presos amb certes precaucions– s'ajusten parcialment a la realitat. Conclou que el programari lliure necessita mètodes d'estudi i models propis, ja que els coneguts no s'adapten a la seva naturalesa. Tanmateix, resulta evident que la possibilitat d'obtenir públicament moltes de les dades relacionades amb el desenvolupament del programari lliure permet ser optimistes de cara a la consecució d'aquests en un futur no molt llunyà. El de Koch es pot considerar el primer estudi quantitatiu complet, encara que certament pateix d'una metodologia clara i, sobretot, d'unes eines *ad hoc* que permetin comprovar els seus resultats i l'estudi d'altres projectes.

Mockus *et al.* van presentar l'any 2000 el primer estudi de projectes de programari lliure que englobava la descripció completa del procés de desenvolupament i de les estructures organitzatives, incloent-hi evidències tant qualitatives com quantitatives ("What is the context of charityware?") [172]. Utilitzen per a aquesta finalitat la història de canvis del programari, així com dels informes d'error, per a quantificar aspectes de la participació de desenvolupadors, la mida del nucli, l'autoria de codi, la productivitat, la densitat de defectes i els intervals de resolució d'errors. En cert sentit, aquest estudi no deixa de ser un estudi d'enginyeria de programari clàssic, amb l'excepció que l'obtenció de les dades ha estat realitzada íntegrament mitjançant la inspecció semiautomàtica de les dades que els projectes ofereixen públicament a la Xarxa. Igual com a "Results from software engineering research into open source development projects using public data", 2000 [158], amb aquest article no es va proporcionar cap eina ni procés automàtic que permetés ser reutilitzada en el futur per altres equips d'investigació.

A "Estimating Linux's size", 2000 [227], i "More than a gigabuck: estimating GNU/Linux's" [228] trobem l'anàlisi quantitativa de les línies de codi i els llenguatges de programació utilitzats en la distribució Red Hat. González Barahona *et al.* han seguit els seus passos en una sèrie d'articles dedicats a la distribució Debian (*vid.* per exemple "Anatomy of two GNU/Linux distributions" [88]). Tots ells ofereixen una espècie de *radiografia* d'aquestes dues populars distribucions de GNU/Linux realitzades a partir de les dades que aporta una eina que

compta el nombre de línies físiques (les línies de codi que no són ni línies en blanc ni comentaris) d'un programa. A part de l'espectacular resultat en línies de codi totals (la versió de Debian 3.0 –coneguda com a Woody– supera els cent milions de línies de codi), es pot veure la distribució del nombre de línies en cada llenguatge de programació. La possibilitat d'estudiar l'evolució de les diferents versions de Debian en el temps aporta algunes evidències interessants [88]. Es pot esmentar que la mida mitjana dels paquets roman pràcticament constant al llarg dels últims cinc anys, per la qual cosa la seva tendència natural a anar creixent es veu neutralitzada per la inclusió de paquets més petits. D'altra banda, es pot percebre com la importància del llenguatge de programació C, encara predominant, decreix amb el temps, mentre que llenguatges de guió (Python, PHP i Perl) i Java comptabilitzen un creixement explosiu. Els llenguatges compilats "clàssics" (Pascal, Ada, Modula...) estan en clara recessió. Finalment, aquests articles inclouen un apartat dedicat a mostrar els resultats obtinguts si s'aplica el model COCOMO –un model d'estimació d'esforços clàssic que data de principis de la dècada dels vuitanta (*Software Engineering Economics*, 1981) [93] i que s'utilitza en projectes de programari propietari– per a realitzar una estimació d'esforç, durada del projecte i costos.

Encara que precursors, la majoria dels estudis presentats en aquest apartat són bastant limitats als projectes que analitzen. La metodologia que s'ha emprat s'ajusta al projecte en estudi, és parcialment manual i en comptades ocasions la part automatitzada pot ser utilitzada de forma general amb la resta de projectes de programari lliure. Això suposa que l'esforç que cal realitzar per estudiar un nou projecte és molt gran, ja que s'ha de tornar a adaptar el mètode utilitzat i repetir les accions manuals que s'han dut a terme.

Per això, els últims esforços ("Studying the evolution of libre software projects using publicly available data": *Proceedings* del 3rd Workshop on Open Source Software Engineering, 25th International Conference on Software Engineering, Portland, EUA [196] o "Automating the measurement of open source projects", 2003 [124]) se centren a crear una infraestructura d'anàlisi que integri diverses eines de manera que s'automatitzi el procés al màxim. Hi ha dues motivacions bastant evidents per a fer això: la primera és que una vegada que s'ha invertit molt temps i esforç a crear una eina per a analitzar un projecte –subrallant especialment el fet que sigui genèrica–, utilitzar-la per a altres projectes de programari lliure implica un esforç mínim. D'altra banda, l'anàlisi mitjançant una sèrie d'eines que analitzen els programes des de diferents punts de vista –de vegades complementaris, altres vegades no– permet obtenir una major perspectiva del projecte. A Libre Software Engineering Web Site [86] es poden seguir amb major deteniment aquestes iniciatives.

7.7. Treball futur

Després de presentar la breu però intensa història de l'enginyeria del programari aplicada al programari lliure, podem afirmar que aquesta es troba encara fent els seus primers passos. Queden diversos aspectes de gran importància

pendents d'estudi i examen minuciosos per a donar amb un model que expliqui, almenys en part, com es genera programari lliure. Les qüestions que s'han d'afrontar en un futur pròxim són la classificació dels projectes de programari lliure, la creació d'una metodologia basada tant com sigui possible en elements d'anàlisis automàtiques i la utilització dels coneixements adquirits per a la realització de models que permetin entendre el desenvolupament de programari lliure, alhora que facilitin la presa de decisions a partir de l'experiència adquirida.

Un aspecte que tampoc no s'ha de deixar de costat i que ja s'està abordant en l'actualitat és l'avaluació de la validesa dels mètodes d'enginyeria clàssica en el camp del programari lliure en totes les intensificacions de l'enginyeria del programari. Així, per exemple, les lleis de l'evolució del programari postulades per Lehman ("Metrics and laws of software evolution - the nineties view" [165]) a començaments de la dècada dels setanta i actualitzades i augmentades als vuitanta i els noranta sembla que no es compleixen incondicionalment en el desenvolupament d'alguns projectes de programari lliure ("Understanding open source software evolution: applying, breaking and rethinking the laws of software evolution", 2003 [199]).

En l'actualitat, una de les deficiències més importants és la falta d'una classificació estricta del programari lliure que permeti catalogar els projectes en diferents categories. Avui dia, els criteris de classificació són massa bastos, de manera que es fiquen al mateix sac projectes amb unes característiques organitzatives, tècniques o de qualsevol altra índole molt dispars. L'argument que Linux, amb una àmplia comunitat i un gran nombre de desenvolupadors, té una naturalesa diferent i no es comporta d'idèntica manera que un projecte molt més limitat en nombre de desenvolupadors i usuaris, és molt encertat. En qualsevol cas, una classificació més exhaustiva permetria reutilitzar l'experiència adquirida en projectes similars (és a dir, amb característiques semblants), facilitaria les prediccions, possibilitaria pronosticar riscos, etc.

El segon aspecte important a què s'ha d'enfrontar l'enginyeria del programari lliure, molt lligat al punt anterior i a les tendències actuals, és la creació d'una metodologia i d'eines que la suportin. Una metodologia clara i concisa permetrà estudiar tots els projectes sense establir-hi diferències, esbrinar el seu estat actual, conèixer la seva evolució i, sens dubte, classificar-los. Les eines són essencials a l'hora d'abordar aquest problema, ja que, una vegada creades, permeten tenir a l'abast de la mà l'anàlisi de milers de projectes amb un mínim esforç addicional. Un dels objectius de l'enginyeria del programari lliure és que a partir d'un nombre limitat de paràmetres que indiquin on trobar a la Xarxa informació sobre el projecte (l'adreça del dipòsit de versions del programari, el lloc on s'emmagatzemen els arxius de les llistes de correu electrònic, la localització del sistema de gestió d'errors i una mínima enquesta) se'n pugui realitzar un estudi exhaustiu. Els gestors del projecte només els separaria un

botó d'una anàlisi completa, una espècie d'*anàlisi clínica* que permetés jutjar la *salut* del projecte i que alhora inclogués una indicació sobre els aspectes que necessiten ser millorats.

Una vegada que s'aconsegueixin mètodes, classificació i models, les possibilitats que ofereix la simulació i, per a ser més concrets, els agents intel·ligents, poden ser enormes. Tenint en compte que partim d'un sistema la complexitat del qual és notòria, resulta interessant la creació de models dinàmics en els quals els diferents ens que participen en la generació de programari puguin ser modelats. Evidentment com més sapiguem dels diferents elements, més ajustat a la realitat serà el model. Encara que es coneixen diverses propostes de simulacions per al programari lliure, aquestes són bastant simples i incompletes. En certa manera, això es deu al fet que encara hi ha una gran llacuna de coneixement respecte a les interaccions que tenen lloc en la generació de programari lliure. Si s'aconsegueix empaquetar i processar convenientment la informació dels projectes al llarg de tota la seva història, els agents poden ser crucials per a conèixer com serà l'evolució en el futur. Encara que existeixen moltes propostes sobre com s'ha d'enfocar aquest problema, una de les més avançades per ara es pot trobar a wwwai.wu-wien.ac.at/~koch/oss-book/ [82].

7.8. Resum

En resum, en aquest capítol hem pogut veure com l'enginyeria del programari lliure és un camp jove i encara per explorar. Els seus primers passos es deuen a escrits assagístics en els quals es proposava –no sense certa falta de rigor científic– un model de desenvolupament més eficient, però gradualment s'ha anat progressant en l'estudi sistemàtic del programari lliure des d'una òptica d'enginyeria. En l'actualitat, després de diversos anys d'informes i estudis de projectes lliures complets quantitatius i qualitatius, s'està duent a terme un enorme esforç en la consecució d'una infraestructura global que permeti la classificació, l'anàlisi i la modelització dels projectes en un espai de temps limitat i de manera parcialment automàtica. Quan l'anàlisi dels projectes de programari lliure deixi de ser tan costosa en temps i en esforç com ho és avui en dia, és molt probable que s'obri una nova etapa en l'enginyeria del programari lliure en la qual entrin en escena un altre tipus de tècniques el propòsit principal de les quals se situï entorn de la predicció de l'evolució del programari i de pronòstic de possibles complicacions.

8. Entorns i tecnologies de desenvolupament

> "The tools we use have a profound (and devious!) influence on our thinking habits, and, therefore, on our thinking abilities."
>
> "Les eines que usem tenen una influència profunda (i tortuosa!) sobre els nostres hàbits de pensament i, per tant, sobre les nostres habilitats de pensament".
>
> Edsger W. Dijkstra, "How do we tell truths that might hurt?"

Els projectes de programari lliure han anat creant al llarg dels anys les seves pròpies eines i sistemes (també lliures) per a l'ajuda en el procés de desenvolupament. Encara que cada projecte segueix les seves pròpies regles i usa el seu propi conjunt d'eines, hi ha certes pràctiques, entorns i tecnologies que poden considerar-se habituals en el món del desenvolupament de programari lliure. En aquest capítol tractarem els més comuns i comentarem el seu impacte en la gestió i l'evolució dels projectes.

8.1. Caracterització d'entorns, eines i sistemes

Abans d'explicar eines concretes, definirem les característiques i les propietats generals que aquestes tenen en funció del treball que s'ha de dur a terme i de la manera d'organització dels desenvolupadors.

En primer lloc, encara que no necessàriament determinant, és habitual que es tracti que l'entorn, les eines de desenvolupament (i fins i tot la màquina virtual objectiu, quan n'hi ha), sigui també *lliure*. Això no sempre ha estat així. Per exemple, el projecte GNU, amb l'objectiu de reemplaçar Unix, va haver de desenvolupar-se sobre i per a sistemes Unix propietaris fins a l'aparició dels Linux i els BSD lliures. Avui en dia, especialment quan el programari lliure es desenvolupa com a part d'un model de negoci, s'estén que la màquina objectiu pugui ser també un sistema propietari, sovint mitjançant màquines virtuals interposades (Java, Python, PHP, etc.). En qualsevol cas, l'entorn i la màquina virtual han d'estar prou difosos i ser prou econòmics per a poder reunir suficients codesenvolupadors que disposin d'aquestes eines.

En segon lloc, també per atreure el major nombre possible de codesenvolupadors, les eines han de ser *senzilles*, àmpliament *conegudes* i capaces de funcionar en màquines *econòmiques*. Possiblement per aquestes raons el món del programari lliure és relativament conservador en llenguatges, eines i entorns.

En tercer lloc, el model de desenvolupament de programari lliure sol ser eminentment distribuït, amb molts col·laboradors potencials repartits per tot el món. És per això que calen eines de col·laboració, generalment asíncrones,

que permetin a més que el desenvolupament avanci amb facilitat, independentment de la quantitat i el ritme de treball de cada col·laborador, sense fer endarrerir ningú.

Finalment, és convenient proporcionar als desenvolupadors recursos de què manquen, com ara màquines d'arquitectures diverses en les quals puguin compilar i provar els seus programes.

8.2. Llenguatges i eines associades

La majoria del programari lliure està realitzada en llenguatge C, no solament perquè C és el llenguatge natural de tota variant de Unix (plataforma habitual del programari lliure), sinó també per la seva àmplia difusió, tant en les ments com a les màquines (GCC és un compilador estàndard instal·lat per defecte en gairebé totes les distribucions). Precisament per aquestes raons i per la seva eficiència, el recomana Stallman en els projectes de GNU ("GNU coding standards") [203]. Altres llenguatges que se li apropen bastant són C++, també suportat per GCC per defecte, i Java, que presenta certa semblança i que és popular perquè permet desenvolupar per a màquines virtuals disponibles en gran varietat de plataformes. Generalment no es tenen en compte raons d'enginyeria programari: a SourceForge (*vid.* apartat 8.9.1), l'any 2004, per cada cent seixanta projectes en C n'hi havia un en Ada, llenguatge suposadament més apropiat per a desenvolupar programes de qualitat. De la mateixa manera, l'anglès és la *lingua franca* entre els desenvolupadors de programari lliure, malgrat que l'esperanto és un idioma molt més fàcil d'aprendre i amb una estructura més lògica. També són populars llenguatges interpretats dissenyats per a prototipatge ràpid d'aplicacions normals i serveis web, com ara Perl, Python i PHP.

Així com C és el llenguatge estàndard, *make* és l'eina estàndard de construcció de programes, donats els seus codis font. El programador lliure usarà normalment la versió de GNU ("GNU make") [36] molt més que la incompatible de BSD (Adam de Boor, "PMake - a tutorial") [100]. Amb elles pot especificar arbres de dependències entre fitxers i regles per generar els fitxers dependents a partir d'aquells de què depenen. Així, es pot especificar que un fitxer objecte `x.o` depèn dels fitxers font `x.c` i `x.h` i que per construir-lo cal executar **`gcc -c x.c`**. O que l'executable del nostre programa depèn d'una col·lecció d'objectes i es munta d'una determinada manera. Quan es modifica un codi font i després s'executa *make*, es recompilaran només els mòduls afectats i es tornarà a muntar l'objecte final. És una eina de nivell molt baix, ja que, per exemple, és incapaç d'esbrinar per si mateixa quan s'ha de recompilar un mòdul en C, malgrat que podria esbrinar-ho examinant les cadenes d'*includes*. També és molt potent, perquè pot combinar totes les eines disponibles de transformació de fitxers per construir objectes molt complexos d'un projecte multillenguatge. Però és molt complicada i molt dependent d'entorns de tipus Unix. Altres

alternatives suposadament millors, com *jam* ("Jam Product Information") [41], *aap* ("Aap Project") [1] o *ant* ("The Apache Ant Project") [7] són poc utilitzades (aquesta última està guanyant popularitat, sobretot en el món de Java).

Atesa l'heterogeneïtat de sistemes existents fins i tot al món de Unix, també es fan servir eines dedicades a ajudar-nos que els nostres programes siguin portàtils. Les eines de GNU *autoconf* (http://www.gnu.org/software/autoconf) [10], *automake* (http://www.gnu.org/software/automake) [32] i *libtool* (http://www.gnu.org/software/libtool) [35] faciliten aquestes tasques en entorns C i Unix. Atesa l'heterogeneïtat d'idiomes, jocs de caràcters i entorns culturals, el programador de C (i de molts altres llenguatges) recorre a *gettext* (http://www.gnu.org/software/gettext) [31] i a les facilitats d'internacionalització de la biblioteca estàndard de C (http://www.gnu.org/software/libc) [34] per realitzar programes que es puguin localitzar per a un entorn cultural fàcilment i en temps d'execució.

Així, quan rebem un paquet font, el més probable és que estigui escrit en C, empaquetat amb *tar*, comprimit amb *gzip*, fet portàtil amb *autoconf* i eines associades, i construïble i instal·lable amb *make*. La seva instal·lació es durà a terme per mitjà d'un procés molt similar al següent:

```
tar xzvf paquet-1.3.5.tar.gz
cd paquet-1.3.5
./configure
make
make install
```

8.3. Entorns integrats de desenvolupament

Un entorn integrat de desenvolupament (IDE) *integrated development environment* és un sistema que facilita el treball del desenvolupador de programari, integrant sòlidament l'edició orientada al llenguatge, la compilació o interpretació, la depuració, les mesures de rendiment, la incorporació dels codis font a un sistema de control de fonts, etc., normalment de forma modular.

No a tots els desenvolupadors de programari lliure els agraden aquestes eines, encara que el seu ús s'ha anat estenent progressivament. Al món del programari lliure, la primera que es va utilitzar àmpliament va ser GNU Emacs (http://www.gnu.org/software/emacs/) [33], obra estrella de Richard Stallman, escrita i extensible a Emacs Lisp, per al qual existeixen multitud de contribucions.

Eclipse (Eclipse - An Open Development Platform) [23] pot considerar-se actualment com l'IDE de referència al món del programari lliure, amb l'inconvenient que funciona millor (cap a maig del 2007) sobre una màquina virtual Java que no és lliure (la de Sun, que s'espera que ho sigui aviat, de tota manera). Altres entorns populars són Kdevelop (http://www.kdevelop.org)

[42] per a KDE, Anjuta (http://www.anjuta.org) [6] per a GNOME, Netbeans (http://www.netbeans.org) [51] de Sun per a Java i Code::Blocks (http://www.codeblocks.org) [18] per a aplicacions C++.

8.4. Mecanismes bàsics de col·laboració

El programari lliure és un fenomen que és possible a causa de la col·laboració de comunitats distribuïdes i que, per tant, necessiten eines que facin efectiva aquesta col·laboració. Encara que durant molt temps es va utilitzar correu postal de cintes magnètiques, el desenvolupament àgil del programari lliure va començar quan va ser possible comunicar-se ràpidament amb moltes persones i que els distribuïssin els codis font dels programes o respondre amb comentaris i pedaços. Per conveniència, millor que enviar codi, en els missatges podria difondre's informació sobre un lloc on recollir-lo. De fet, molt al principi dels anys setanta, el correu electrònic va ser una extensió del protocol de transferència de fitxers d'ARPANET.

Al món de Unix, a mitjan setanta, es desenvolupa *uucp* el protocol de transferència de fitxers de Unix, apte per a comunicar màquines per línies commutades, a més de dedicades, sobre el qual es va muntar correu electrònic, i el 1979, el primer enllaç USENET sobre UUCP. Les notícies (*news*) d'USENET, un sistema de fòrums temàtic estructurat jeràrquicament i distribuït per inundació als llocs subscrits a una jerarquia, van tenir un paper fonamental en el desenvolupament lliure de programari, en enviar-se programes complets en font als grups de la jerarquia `comp.sources`.

En paral·lel, es van desenvolupar les llistes de correu, entre les quals es poden esmentar els gestors de llistes de BITNET (1981). Avui en dia existeix la tendència a preferir les llistes de correu als grups de notícies del tipus USENET. La raó principal ha estat l'abús amb finalitats comercials i la intrusió de gent "despistada", que introdueix soroll en les discussions. A més, les llistes de correu ofereixen més control i poden arribar a més gent. Els destinataris han de subscriure-s'hi i qualsevol adreça de correu és vàlida, encara que no hi hagi accés directe a Internet. L'administrador de la llista pot elegir conèixer qui se subscriu o donar de baixa algú. Pot restringir les contribucions als membres o pot elegir moderar els articles abans que apareguin[6].

(6)També existeixen grups de notícies moderats.

Tradicionalment l'administració de les llistes s'ha fet per correu electrònic, per mitjà de missatges especials amb contrasenya, la qual cosa permet que l'administrador no tingui accés permanent a Internet, encara que cada vegada això és un fenomen més estrany, de manera que el gestor de llistes de correu més popular avui en dia (Mailman, the GNU Tramesa List Manager) [46] no pot ser administrat per correu electrònic, sinó necessàriament via web. Les llistes de correu tenen un paper crucial en el programari lliure, i arriben a ser en molts casos[7] l'únic mètode de contribució.

(7)Per exemple, les contribucions a Linux s'han de fer com a pedaços de text a la llista `linux-kernel@vger.kernel.org`.

Avui en dia, amb la popularitat del web, molts fòrums són purs fòrums web o *weblogs*, sense una altra interfície que la que s'ofereix per mitjà del navegador. Poden ser genèrics, com ara els populars SlashDot ("Slashdot: News for Nerds") [58] o BarraPunto (http://barrapunto.com) [11], on s'anuncia nou programari lliure o es discuteixen notícies relacionades, o especialitzats en un programa concret, que normalment estan integrats a llocs de col·laboració amb diverses eines addicionals (*vid.* apartat 8.6.2). També hi ha interfícies web a grups de notícies i llistes tradicionals.

Així mateix s'ha fet popular un mecanisme de col·laboració basat en *wikis*, sobretot quan es tracta d'elaborar un document conjunt, que pot ser l'especificació d'un programa, un mòdul o un sistema. En parlarem a l'apartat 8.6.2.

Finalment, cal esmentar els mecanismes d'interacció en els quals els desenvolupadors conversen en temps real. Per a programari lliure no sol ser un mecanisme pràctic, perquè amb desenvolupadors distribuïts per tot el món no és fàcil trobar una hora apropiada per a tots. No obstant això, hi ha diversos projectes que fan ús d'eines de xerrada textual, o bé regularment o bé en congressos virtuals amb dates delimitades. L'eina més usada és l'IRC (Internet Relay Chat, http://www.ietf.org/rfc/rfc2810.txt) [151], que normalment comunica gent per mitjà de "canals" temàtics establerts a partir d'una sèrie de servidors col·laboradors. No és comú que s'utilitzin eines multimèdia (so, imatge...), probablement perquè poden requerir connexions de qualitat no disponibles per a tots i presentar problemes de disponibilitat de programari lliure i la dificultat de registrar i editar els resultats de les converses, per tal de documentar.

8.5. Gestió de fonts

A tot projecte de desenvolupament de programes li convé tenir arxivada la seva història, per exemple, perquè alguna modificació pot produir un error ocult que es descobreixi tardanament i calgui recuperar l'original, almenys per a analitzar el problema. Si el projecte el desenvolupen diverses persones, és necessari també registrar l'autor de cada canvi, amb les raons d'aquest explicades. Si del projecte van fent-se lliuraments versionades, és necessari saber exactament quines versions de cada mòdul formen els esmentats lliuraments. Moltes vegades, un projecte manté una versió estable i una altra d'experimental; cal mantenir-les totes dues, corregir-ne els errors i transferir errors corregits d'una versió a l'altra. Tot això pot fer-se guardant i etiquetant convenientment totes i cada una de les versions dels fitxers, la qual cosa generalment s'ha considerat com un cost excessiu, encara que amb els discos actuals comenci a no ser tan cert. El que normalment fa un *sistema de control de fonts*, també anomenat *sistema de gestió de versions*, és registrar la història dels fitxers com un conjunt de diferències sobre un patró, normalment el més recent, per eficiència, etiquetant a més cada diferència amb les metadades necessàries.

Però també volem que un sistema d'aquestes característiques serveixi perquè molts programadors col·laborin efectivament, sense que es trepitgi el treball, però sense detenir l'avenç de cada un. Ens ha de permetre, doncs, que hi hagi diversos programadors treballant de manera concurrent, però amb un cert control. Aquest control pot ser optimista o pessimista. Amb un control pessimista, un programador pot reservar-se uns fitxers per a una millora durant un temps, durant el qual ningú no pot tocar aquests fitxers. Això és molt segur, però bloquejarà altres programadors i el projecte pot retardar-se, sobretot si el que ha bloquejat els fitxers està ocupat en altres coses o fins i tot se n'ha oblidat. Permetre a d'altres avançar és més dinàmic, però més perillós, ja que hi pot haver modificacions incompatibles. Un sistema optimista deixa avançar, però ens avisa quan hi ha hagut conflictes i ens proporciona eines per resoldre'ls.

8.5.1. CVS

CVS (Concurrent Version System) és un sistema de gestió de fonts optimista dissenyat a finals dels vuitanta i que és usat per aclaparadora majoria en els projectes lliures (Concurrent Version System [20], *Open source code development with CVS* 2a. edició) [113], "The Internet Standards process", 3a. revisió [95]). Utilitza un dipòsit central al qual s'accedeix segons un sistema client/servidor. L'administrador del lloc decideix qui tenen accés al repositori o a quines parts d'ell, encara que normalment, una vegada que un desenvolupador ha estat admès en el seu cercle de confiança, té accés a tots els fitxers. A més, es pot permetre un accés anònim, només en lectura, a tothom.

El col·laborador anònim

El *CVS anònim* és una eina fonamental per a realitzar el concepte de "lliurar aviat i freqüentment" defensat per Eric Raymond. Qualsevol usuari ansiós de provar l'última versió d'un programa la pot extreure del CVS, descobrir errors i comunicar-los, fins i tot en forma de pedaços amb la correcció. I pot examinar la història de tot el desenvolupament.

Vegem una mica de mecànica. Un usuari avançat vol obtenir l'última versió del mòdul `mod` d'un dipòsit accessible anònimament a `progs.org`, directori `/var/lib/cvs` i protocol `pserver`. La primera vegada declararà la seva intenció d'accedir:

```
cvs -d:pserver:anonymous@progs.org:/var/lib/cvs login
```

Es demana una contrasenya, que serà la de l'usuari anònim (normalment retorn de carro), que es registrarà en un fitxer local (realment aquesta operació no és necessària per a accés anònim, però el programa es queixa si no existeix el fitxer amb la contrasenya). Seguidament, l'important és obtenir la primera còpia del mòdul:

```
cvs -d:pserver:anonymous@progs.org:/var/lib/cvs co mod
```

Això crearà un directori `mod` amb tots els fitxers i els directoris del mòdul i certes metadades (continguts en subdirectoris anomenats `CVS`), que permetran, entre altres coses, no haver de repetir la informació ja donada. El nostre usuari avançat s'introdueix en el directori creat, genera el paquet i prova:

```
cd mod
./configure
make
make install
```

Quan vol una nova versió, simplement actualitza la seva còpia dins de `mod`.

```
cd mod
cvs update
./configure
make
make install
```

Si descobreix un error, pot corregir-lo al lloc i després enviar un pedaç per correu electrònic al mantenidor del programa (individual o llista de correu):

```
cvs diff -ubB | mail -s "Mis parches" mod-maint@progs.org
```

El desenvolupador normal

El desenvolupador normal tindrà un compte al servidor. Pot fer servir el mateix mecanisme i el mateix protocol que l'usuari anònim, canviant `anonymous` pel nom del seu compte.

Nota

Per seguretat, per a comptes amb dret a escriptura, se sol usar `ssh`, que proporciona un canal autenticat i xifrat.

Una vegada que té una còpia de treball del mòdul, pot fer les modificacions necessàries, i quan consideri que s'han estabilitzat, *comprometre* els canvis al dipòsit. Per exemple, si modifica els fitxers `parte.h` i `parte.c`, els comprometrà així:

```
cvs ci parte.h parte.c
```

Abans d'acabar l'operació, el CVS li demanarà una explicació del que ha fet, que s'adjuntarà a l'historial d'ambdós fitxers. Així mateix, incrementarà el *número de revisió* de cada fitxer en una unitat. Aquest número identifica cada moment important en la història d'un fitxer i pot emprar-se per a recuperar cada un d'aquells moments.

Quan ha de realitzar un desenvolupador una operació de compromís? Aquesta és una qüestió metodològica que han d'acordar els membres d'un projecte, però sembla obvi que no s'han de comprometre canvis que no es compilin. Però és preferible a més que passin una bateria de proves mínima. En molts projectes és, a més, necessari comptar amb l'aprovació d'un supervisor de projecte o subprojecte que examini la modificació.

Durant el desenvolupament de la modificació, algú pot haver alterat altres fitxers, o fins i tot els mateixos. És per això que convé que el desenvolupador faci, amb relativa freqüència, una operació d'actualització de la seva còpia (*cvs update*). Si s'han modificat altres fitxers, pot haver canviat l'entorn i poden fallar proves que abans passaven. Si s'han modificat els mateixos fitxers, pot ser que aquests canvis s'hagin produït en llocs o rutines que no hem tocat o en codi que sí que hem modificat. En el primer cas no hi ha conflicte (almenys aparent) i l'operació de modificació "barreja" la nostra versió amb la del dipòsit, de manera que es generen fitxers combinats, amb totes les modificacions. En cas contrari hi ha conflicte, cas en què cal posar-se d'acord amb el desenvolupador que ha fet els altres canvis i acordar una versió final.

Per identificar millor cada component d'un projecte, és convenient que porti associada directament informació de revisió. CVS pot marcar els codis font i els objectes automàticament, sempre que s'observi certa disciplina. Per exemple, si en un comentari del codi font escrivim la paraula clau `$Id$`, cada vegada que el fitxer es comprometi al dipòsit, l'esmentada paraula se substituirà per una cadena d'identificació en la qual podrem veure el nom del fitxer, el número de revisió[8], la data i l'hora del compromís i l'autor:

(8) A CVS els números de revisió normalment tenen dos components (major i menor), però poden tenir-ne quatre, sis, etc.

```
$Id: parte.c,v 1.7 2003/07/11 08:20:47 joaquin Exp $
```

Si aquesta paraula clau la incloem dins d'una cadena de caràcters del programa, en compilar-lo la cadena apareixerà a l'objecte i l'executable, amb la qual cosa serà possible identificar-lo amb una eina (*ident*).

L'administrador

Òbviament, els administradors són els que tenen i han de migrar la part més complicada del manteniment del dipòsit Per exemple, han de donar d'alta el projecte, donar permisos als desenvolupadors i coordinar-los, etiquetar versions lliurades, etc.

És pràctica comuna que tot projecte tingui una versió estable i una altra d'experimental. Per a això es creen branques. Mentre que els que es dediquen al manteniment corregeixen errors de la branca estable, els nous desenvolupaments es fan sobre la branca experimental. Quan aquesta s'estabilitza, cal passar-la a estable, no sense abans aplicar les correccions fetes sobre la branca estable anterior. Aquesta operació s'anomena *barrejar*, és delicada i està suportada en CVS, encara que de forma una mica primitiva. Aquesta idea pot estendre's al concepte de branques experimentals que evolucionen en diferents direccions, que arribaran a bon port o no, i que en tot cas, llevat que siguin vies mortes, caldrà integrar de manera total o parcialment en el producte estable, amb mescles apropiades.

Un dret que ens dóna el programari lliure és la modificació d'un programa per a ús privat. Encara que és desitjable contribuir amb tota millora al cabal comunitari, moltes vegades les modificacions que volem fer són massa específiques i no interessen al públic en general. Però sí que ens interessa incorporar l'evolució del programa original. Això pot realitzar-se amb un tipus especial de ramificació i mescla (*branques de venedor*).

L'administrador també pot facilitar la coordinació de l'equip per mitjà de mecanismes automàtics, com fer que es produeixin missatges de correu electrònic quan ocorrin certes coses, com els compromisos, o forçar que es realitzin certes accions automàtiques abans de realitzar un compromís, com ara comprovacions automàtiques d'estil, compilacions o proves.

8.5.2. Altres sistemes de gestió de fonts

Malgrat ser el sistema de control de versions més àmpliament usat, CVS té notables inconvenients:

Nota

El 2007 `Subversion` és ja clarament el successor de CVS, i molts desenvolupaments de programari lliure hi han migrat.

1) CVS no suporta reanomenaments o canvis de directori de fitxers, ni tampoc metadades (propietaris, permisos, etc.) ni enllaços simbòlics.

2) En ser una evolució d'un sistema de control de versions de fitxers individuals, no suporta, naturalment, el control de versions de grups complets.

3) CVS no suporta conjunts de canvis coherents. En efecte, per afegir una característica o corregir un error, pot caldre canviar diversos fitxers. Aquests canvis haurien de ser atòmics.

4) En CVS és bastant complicat l'ús de branques i mescles. En efecte, si creem una branca experimental d'un projecte i desitgem incloure les correccions fetes en la branca estable, cal conèixer en detall quines correccions s'han fet ja, per a no fer-les diverses vegades.

5) CVS depèn d'un servidor centralitzat, i encara que es pot treballar sense connexió, sí que la necessitem per a generar versions, comparar-les i barrejar-les.

6) CVS no genera, sense l'ajuda d'eines a part, el fitxer `changelog`, que mostra la història global de canvis d'un projecte.

7) CVS no suporta bé projectes amb un nombre gaire gran de fitxers, com és el cas del nucli Linux.

I tanmateix, hi ha altres sistemes lliures que solucionen diversos d'aquests problemes. Podem destacar el successor ja designat de CVS, Subversion (http://subversion.tigris.org) [62], (http://svnbook.red-bean.com/) [96], que resol estrictament els problemes bàsics de CVS i pot utilitzar extensions d'HTTP (WebDAV) per a defugir polítiques de seguretat agressives.

El model de desenvolupament basat en un dipòsit centralitzat, encara que apte per al treball cooperatiu, no satisfà totes les expectatives, ja que d'una banda depèn de l'accessibilitat i el bon funcionament del servidor, i per una altra dels administradors d'aquest servidor, el fet que puguem crear les nostres pròpies branques de desenvolupament. De vegades es desitgen dipòsits distribuïts que permetin a qualsevol tenir un dipòsit amb una branca privada o pública que després pugui barrejar o no amb l'oficial. Així funcionen *GNU arch* (Arch Revision Control System) [8] o *bazaar* (Bazaar GPL Distributed Version Control Software) [12], així com el sistema propietari BitKeeper (Bitkeeper Source Management) [14], elegit per Linus Torvalds per a mantenir Linux des de febrer del 2002, ja que segons ell no hi havia cap eina lliure apropiada. Es diu que l'ús de Bitkeeper va doblar el ritme de desenvolupament de Linux. No obstant això, la decisió va ser molt criticada perquè era propietari, amb una llicència que permetia als projectes lliures obtenir-lo gratuïtament sempre que totes les operacions de compromís de canvis amb les seves metadades es registressin en un servidor públic designat pels propietaris i accessible a tothom, i sempre que el llicenciatari no desenvolupés cap altre sistema de control de fonts que pogués competir amb ell. Va ser precisament l'intent de desenvolupar un producte lliure compatible per part d'un empleat de la mateixa empresa on treballava Linus Torvalds el detonador del canvi de sistema de gestió de fonts. Linus desenvolupa ràpidament un substitut provisional, *git* ("Git manual page") [218], que aviat es converteix en definitiu, on es condensa tota l'experiència en el desenvolupament cooperatiu i descentralitzat de Linux: suporta projectes de gran mida de forma descentralitzada, facilitant en gran manera el desenvolupament de branques temptatives i la seva mescla amb d'altres o amb la principal, amb mecanismes de seguretat criptogràfics que impedeixen alterar l'historial. A partir d'abril del 2005 Linux es manté amb *git* o embolcalls d'aquest (per exemple, *cogito* –"Cogito manual page"– [90]).

8.6. Documentació

Al món del programari lliure, amb prou feines s'utilitzen processadors de text WYSIWYG i altres eines ofimàtiques que tant d'èxit tenen en altres entorns, malgrat que hi ha ja eines lliures, com OpenOffice.org. Això és a causa de diversos factors importants:

- És convenient mantenir la documentació sota control de fonts, i els sistemes de control de fonts, com CVS, encara que admeten formats binaris, prefereixen formats textuals transparents, editables amb un editor de text normal i processables amb eines desenvolupades per a programes, que ens permetin veure fàcilment les diferències entre versions, generar i aplicar pedaços basats en aquestes diferències, i realitzar operacions de mescla.

Nota

En Unix les eines que fan aquestes operacions més comunes són, *diff*, *diff3*, *patch* i *merge*.

- Certes llicències de documentació lliure, especialment la GFDL (*vid.* apartat 10.2.1), exigeixen formats transparents, sobretot perquè faciliten el treball als qui realitzen documents derivats.

- Les eines WYSIWYG ("*what you see is what you get*") generalment no contenen més informació que l'estricta de visualització, per la qual cosa fan molt difícil, si no impossible, el processament automàtic, com identificar autors o títol, i la conversió a altres formats. Fins i tot encara que permetin conversió de formats, aquesta sol fer-se de forma interactiva, i és moltes vegades impossible automatitzar-la (amb *make*, per exemple).

- Generalment les eines ofimàtiques generen uns formats de fitxers molt voluminosos, assumpte molt desagradable per als desenvolupadors o els repositoris.

Per tot això, el programador lliure, acostumat també a programar i compilar, prefereix formats de documentació transparents, en molts casos simple text pur i en molts altres formats de documentació processables.

Els formats processables utilitzats no són gaires. Tradicionalment, al món de Unix els programes s'han documentat en els formats esperats per la família de processadors *roff*, amb versió lliure (*GNU troff*) [37] de Norman Walsh. No obstant això, aquesta pràctica s'ha anat abandonant, excepte per a les pàgines de manual tradicionals, ja que és gairebé obligat preparar pàgines de manual per a les eines més bàsiques del sistema. A causa que moltes pàgines de manual han crescut excessivament de manera que difícilment se les pot anomenar *pàgines*, va ser necessari elaborar un format alternatiu hipertextual que permetés seguir documents estructurats amb índexs i referències encreuades. El projecte GNU va dissenyar el format *texinfo* (Texinfo - The GNU Documentation System) [63] i el va convertir en el seu estàndard. Aquest format permet obtenir

documents navegables amb l'eina *info* o dins de l'editor *emacs*, i al seu torn, obtenir documents impresos de qualitat amb ajuda del processador de texts TeX, deDonald Knuth (The TeXbook) [156].

El format *texinfo* pot traduir-se a HTML multipàgina si es desitja, i molta gent prefereix veure la informació amb un navegador web, però es perd la capacitat de cerca de paraules en un document. Aquesta és una de les conseqüències indesitjables de la popularitat d'HTML, ja que els navegadors no implementen el concepte de *document multipàgina*, malgrat que existeixen elements `link` que permeten enllaçar les parts.

Existeix una imperiosa demanda que els documents complexos es puguin veure com a pàgines web multipàgina fàcilment navegables. Hi ha gent que escriu documentació en LaTeX (*LaTeX user's guide and reference manual*) [163], també aplicació de TeX, molt popular entre els científics, més expressiva que Texinfo i convertible a HTML multipàgina amb certes eines (The LaTeX Web Companion) [130], sempre que es guardi certa disciplina. En efecte, les aplicacions de TeX són conjunts de macros que combinen operadors tipogràfics de nivell molt baix per convertir-los en llenguatges abstractes que treballen amb conceptes d'alt nivell (autor, títol, resum, capítol, apartat, etc.). Si només s'utilitzen les macros bàsiques, la conversió és senzilla. Però com que ningú no impedeix usar operadors de baix nivell i, a més, hi ha quantitats enormes de paquets de macros fora de la capacitat de manteniment dels mantenidors dels convertidors, és difícil aconseguir que les conversions surtin bé.

8.6.1. DocBook

El problema rau en el fet que no existeix separació entre contingut i presentació, ni en les aplicacions de TeX ni en les de *nroff*, ja que l'abstracció es construeix per capes. Aquesta separació la tenen les aplicacions d'SGML (*standard generalized markup language*) [81] i XML (*extensible markup language*) [224], en què la presentació s'especifica amb *fulls d'estil* completament separats. Molt aviat van començar a usar-se aplicacions molt senzilles d'SGML, com `linuxdoc` i `debiandoc`, però a causa de la seva escassa capacitat expressiva, es va optar per `DocBook` (*DocBook: the definitive guide*) [225].

`DocBook` és una aplicació d'SGML originalment desenvolupada per a documentació tècnica informàtica i avui amb una variant XML. Actualment DocBook és l'estàndard de format de documentació lliure per a molts projectes (Linux Documentation Project, KDE, GNOME, Mandriva Linux, etc.) i un objectiu que s'ha d'assolir en altres (Linux, *BSD, Debian, etc.).

Tanmateix, DocBook és un llenguatge complex, ple d'etiquetes, per la qual cosa és convenient disposar d'eines d'ajuda a l'edició, encara escasses i elementals, entre les quals la més popular és la manera `psgml` d'*emacs*. També és pesat de processar i els processadors lliures encara generen una qualitat de documents poc atractiva.

8.6.2. *Wikis*

Molta gent troba massa complicat escriure documentació amb llenguatges tan complexos com DocBook i mecanismes de col·laboració com CVS. És per això que s'ha fet molt popular un nou mecanisme de col·laboració per a l'elaboració de documents en línia via web anomenat *wiki*, inventat per Ward Cunningham ("Wiki design principles") [97], posat per primera vegada en servei el 1995 i avui àmpliament utilitzat en molt diverses variants per a elaborar documents molt dinàmics, no destinats a ser impresos i moltes vegades amb una vida curta (per exemple, l'organització d'una conferència).

Al contrari que DocBook, un *wiki* té un llenguatge de marques molt simple i concís que recorda la presentació final, sense ser exactament com ella. Per exemple, els paràgrafs se separen per una línia en blanc, els elements de llistes comencen per un guió si no es numeren i per un zero si es numeren, i les cel·les de les taules se separen per barres verticals i horitzontals.

Tampoc no existeix el concepte de "document complet", sinó que un *wiki* és més aviat un conjunt de petits documents enllaçats que es van creant a mesura que és necessari explicar un nou concepte o tema. La creació dels documents és gairebé automàtica, ja que l'eina d'edició mostra molt clarament que hem introduït un concepte (per mitjà d'un `NomWiki`, gairebé sempre dues paraules juntes amb la primera lletra en majúscula). Gairebé cap *wiki* no admet hiperenllaços dins de la mateixa pàgina.

A diferència de CVS, qualsevol pot escriure en un *wiki*, encara que s'aconsella que l'autor s'identifiqui amb un registre previ. Quan es visita un *wiki* es veu que totes les pàgines tenen un botó que permet la seva edició. Si es prem, el navegador ens mostrarà en un formulari el codi font del document, que podrem canviar. No és una edició WYSIWYG, la qual cosa dissuadeix el qui vulgui molestar, però és tan senzill que qualsevol interessat pot modificar documents amb un esforç molt petit.

Els *wikis* porten el seu propi control de versions de documents, de manera que generalment són accessibles totes les seves versions, amb indicació de qui les va fer i quan. També es poden comparar amb facilitat. A més, solen portar mecanismes de cerca, almenys per nom de pàgina i per paraula continguda.

Normalment l'autor original d'una pàgina voldrà assabentar-se de les modificacions que s'hi facin. Per a això pot subscriure's als canvis i rebre'n notificacions per correu electrònic. De vegades, qui veu un document no s'atreveix

a canviar res, però pot fer un comentari. Normalment tota pàgina *wiki* té associat un fòrum de comentaris que s'enganxen al final del document i que o bé l'autor original o bé algú que assumeixi la funció d'editor pot emprar per a reformar el text original, possiblement movent frases dels comentaris als llocs oportuns.

Suggeriment

La millor forma d'entendre el concepte de *wiki* és accedint a un i experimentant en una pàgina destinada a això, denominada habitualment `SandBox`.

8.7. Gestió d'errors i altres temes

Un dels punts forts del model de desenvolupament lliure és que la comunitat contribueixi amb informes d'errors i que senti que aquests informes o solucions són tinguts en compte. És per això que és necessari un mecanisme senzill d'informe d'errors, de manera que els desenvolupadors rebin informació suficient, de forma sistemàtica i amb tots els detalls necessaris, o bé aportats pel col·laborador, com l'explicació del que passa, el nivell d'importància i la possible solució, o bé per algun mecanisme automàtic que determini, per exemple, la versió del programa i l'entorn en el qual funciona. Així mateix és necessari que els errors es guardin en una base de dades que pugui ser consultada, per a veure si un error ja ha estat comunicat, si ha estat corregit, el seu nivell d'importància, etc.

Hi ha diversos d'aquests sistemes, de diferent filosofia. Uns són via web, altres via correu electrònic, a partir d'algun programa intermediari. Tots tenen interfície web per a consulta. Uns permeten informes anònims, d'altres requereixen identificació (una adreça de correu vàlida) per a evitar el soroll. Encara que els procediments via web semblen els més senzills, amb ells no és possible obtenir fàcilment informació automàtica de l'entorn de l'error. Per exemple, el sistema de Debian proporciona programes com reportbug, que després de preguntar el nom del paquet sobre el qual volem informar, consulta al servidor d'errors quins errors ja hi ha comunicats sobre aquest. Si cap d'ells no es refereix al nostre problema, ens en demana una descripció, un nivell d'importància ("crític", "greu", "seriós", "important", "no es pot regenerar a partir dels codis font", "normal", "menor" o "suggeriment") i etiquetes sobre la seva categoria (per exemple, "seguretat"). Després d'això, si confirmem la petició, automàticament esbrina la versió del paquet i d'aquells dels quals depèn, així com la versió i l'arquitectura del nucli. Òbviament, l'adreça electrònica la sap, per la qual cosa s'envia al lloc correcte un informe similar al següent:

```
Package: w3m-ssl
Version: 0.2.1-4
Severity: important

After reloading a page containing complex tables several dozen times, w3m
```

```
had used all physical memory and thrashing commenced. This is an Alpha machine.

--System Information
Debian Release: testing/unstable
Kernel Version: Linux romana 2.2.19 #1 Fri Jun 1 18:20:08 PDT 2001 alpha Unknown

Versions of the packages w3m-ssl depends on:
ii  libc6.1     2.2.3-7        GNU C Library: Shared libraries and Timezone data
ii  libgc5      5.0.alpha4-8   Conservative garbage collector for C
ii  libgpmg1    1.19.3-6       General Purpose Mouse Library [libc6]
ii  libncurses5 5.2.20010318-3 Shared libraries for terminal handling
ii  libssl0.9.6 0.9.6a-3       SSL shared libraries
ii  w3m         0.2.1-2        WWW browsable pager with tables/frames support
```

Aquest missatge genera un número d'error que ens és tornat, és enviat al mantenidor i és guardat en la base de dades. Quan es resolgui l'error, rebrem també una notificació. Cada error té assignada una adreça electrònica que es pot utilitzar per a proporcionar informació addicional, per exemple. La base de dades d'errors la podrem consultar a http://bugs.debian.org en qualsevol moment.

De vegades els sistemes de seguiment d'errors tenen mecanismes per a assignar la correcció a algú i posar-li un termini. També hi ha altres temes, com ara treballs pendents, millores sol·licitades, traduccions, etc., que requereixen així mateix mecanismes de gestió similars. En programari lliure generalment no es poden emprar mecanismes gaire rígids de gestió dels treballs que ha de fer cada desenvolupador. Al cap i a la fi, molts col·laboradors són voluntaris i no se'ls pot obligar a res. No obstant això, sí que es poden definir tasques i esperar que algú es doni d'alta en el sistema i les assumeixi, i declari un termini per a això. Tant si es té control sobre el que poden fer determinades persones com si no, sempre és convenient tenir controlades les tasques que cal fer, les dependències que tenen, el seu nivell d'importància i qui hi està treballant. Molts dels projectes importants gestionen aquests temes amb Bugzilla (*The Bugzilla guide*) [89] o derivats d'aquest.

De vegades una persona, treballant en un projecte, pot descobrir errors en un altre de què depèn el seu treball, però que té un sistema de gestió d'errors diferent del sistema al qual està acostumat. Això és especialment cert per a usuaris de distribucions que volen utilitzar una única eina per a informar i seguir la correcció dels seus errors. Per a facilitar l'informe i el seguiment d'aquests errors, pot ser convenient *federar* diferents sistemes, com fa *Malone* (The Malone Bug Tracker) [47].

8.8. Suport per a altres arquitectures

El suport mínim per a treballar amb un programa portable és l'accés a *granges de compilació*, que permeten compilar-la en diverses arquitectures i sistemes operatius. Per exemple, SourceForge (*vid.* apartat 8.9.1) va oferir durant un temps entorns Debian GNU/Linux per a Intel x86, DEC Alpha, PowerPC i SPARC, a més d'entorns Solaris i Mac OS/X.

També és útil poder provar (no només compilar) el programa en aquests entorns. Però aquest servei requereix més recursos i més temps d'administrador. Ja el servei de compilació pot ser problemàtic, perquè habitualment cal proporcionar entorns de compilació per a diversos llenguatges, amb una gran quantitat de biblioteques. Si el que es vol és provar un programa qualsevol, les dificultats augmenten exponencialment, no només perquè és molt difícil disposar dels recursos requerits, sinó per raons de seguretat, que poden fer molt complicat administrar aquests sistemes. No obstant això, hi ha uns quants serveis de *granja de servidors*, amb instal·lacions estàndard d'arquitectures diverses, que poden permetre'ns provar algunes coses.

Les granges públiques esmentades anteriorment solen ser un servei d'ús manual. El desenvolupador convidat copia els seus fitxers en una d'aquestes màquines, els compila i prova el resultat. Probablement ho haurà de fer de tant en tant, en el moment previ a l'alliberament d'una versió important del programa. Pot ser molt més interessant que les compilacions i l'execució de les proves de regressió es facin sistemàticament, de forma automàtica, per exemple totes les nits, si hi ha hagut canvi en els codis font. Així funcionen alguns projectes importants, que proporcionen una infraestructura pròpia per a desenvolupadors externs, que sol anomenar-se *tinderbox* ('encenedor de pedra'). És el cas de Mozilla, finançat per Netscape, el *tinderbox* del qual (http://www.mozilla.org/tinderbox.html) [50] presenta una interfície web als resultats de la compilació i proves de components del seu navegador en totes les arquitectures on funciona. Aquesta interfície està íntimament relacionada amb el CVS i mostra aquests resultats per a diferents estats (entre compromisos), identifica el responsable dels errors, en facilita l'avenç i defuig el problema fins que se solucioni. També usen linderbox els projectes OpenOffice.org i FreeBSD, almenys.

8.9. Llocs de suport al desenvolupament

Els llocs de suport al desenvolupament proporcionen, de manera més o menys integrada, tots els serveis descrits anteriorment, juntament amb alguns d'addicionals que permeten la cerca de projectes per categories i la seva classificació segons paràmetres senzills d'activitat. Això allibera el desenvolupador d'haver-se de muntar i administrar tota una infraestructura de col·laboració per poder concentrar-se en el seu projecte.

8.9.1. SourceForge

Pel que fa a aquest tipus de serveis, un dels primers a establir-se, i el més popular, va ser SourceForge (http://sourceforge.net) [61], gestionat per l'OSDN (Open Software Development Network), subsidiària de VA Software, que acollia al març del 2007 més de 144.000 projectes. Està estructurat entorn d'un conjunt de programes del mateix nom, que fins a la versió 2 van ser programari lliure.

SourceForge, com a prototip d'aquests llocs, ofereix una interfície web o portal global d'entrada (http://sourceforge.net/) i un subportal per projecte (http://projecte.sourceforge.net). En la interfície global podem veure notícies, anuncis, enllaços i una invitació a fer-se membre o a entrar-hi si ja se n'és. Per a col·laborar en el lloc, convé fer-se'n membre, la qual cosa és imprescindible si es vol crear un projecte nou o participar en un de ja existent. Per a ser espectador no és necessari, i com a tals, podem veure quins són els projectes que experimenten un desenvolupament més actiu o els que són descarregats amb més freqüència, i podem buscar projectes per categories o donant una paraula de la seva descripció, i se'ns mostraran ordenats pel seu nivell d'activitat. Dins de cada projecte, podem veure'n la descripció, l'estat (alfa, beta, producció), els descriptors (llenguatge, sistema operatiu, temàtica, tipus d'usuaris, idioma, llicència...), errors i temes pendents o solucionats, els detalls de la seva activitat en el temps..., o descarregar-lo. També podem participar en fòrums o informar d'errors, fins i tot de forma anònima, la qual cosa no és gaire convenient (perquè, per exemple, no ens poden respondre).

Qualsevol usuari autenticat pot sol·licitar donar d'alta un projecte, que serà admès pels administradors del lloc si en compleix les polítiques, que en el cas de SourceForge són bastant liberals. Una vegada autoritzat, el creador pot donar d'alta altres usuaris registrats com a administradors addicionals o com a desenvolupadors, amb accés a la modificació de fonts. Després de l'autorització, no hi ha gaires més controls sobre el projecte, la qual cosa ocasiona l'existència de molts projectes morts. Això no confon massa els usuaris, perquè com en les cerques els projectes es mostren ordenats pel grau activitat, els que tenen una activitat baixa o nul·la amb prou feines són visibles. Aquests projectes corren el risc de ser eliminats pels propietaris del lloc. Els serveis que SourceForge proporciona a un projecte, i que podríem esperar de qualsevol servei similar, són els següents:

- Allotjament per a les pàgines web del portal del projecte, a l'adreça *projecte.sourceforge.net*, on es mostra al públic. Aquestes pàgines poden ser estàtiques o dinàmiques (amb CGI o PHP); en aquest cas poden fer ús d'una base de dades (MySQL). S'introdueixen directament mitjançant ordres de còpia remota i poden manipular-se per mitjà de sessions interactives de terminal remot (SSH).

- Opcionalment un servidor virtual que respongui a adreces d'un domini obtingut a part, com ara *www.projecte.org* o *cvs.projecte.org*.

- Tants fòrums web i/o llistes de correu com siguin necessaris, segons el criteri d'un administrador.

- Un servei de notícies on els administradors anunciïn les novetats sobre el projecte.

- Rastrejadors (*trackers*) per a informe i seguiment d'errors, peticions de suport, peticions de millores o integració de pedaços. Els administradors donen una prioritat a l'assumpte i assignen la seva solució a un desenvolupador.

- Gestors de tasques, similars als rastrejadors, que permeten definir subprojectes amb una sèrie de tasques. Aquestes tasques, a més d'una prioritat, tenen un termini. Els desenvolupadors als quals s'assignen poden manifestar de tant en tant un percentatge de realització de les tasques.

- Un CVS o un Subversion amb drets inicials d'accés per a tots els desenvolupadors.

- Servei de càrrega i baixada de paquets de programari. Utilitzant-lo es té un registre de les versions introduïdes i es possibilita que els interessats rebin un avís quan passa això. A més, la càrrega implica la creació de diverses rèpliques a tothom, la qual cosa facilita la distribució.

- Servei de publicació de documents en HTML. Qualsevol pot registrar-los, però només després de l'aprovació per part d'un administrador seran visibles.

- Còpia de seguretat per a recuperació de desastres com ara fallada de disc, no d'errors d'usuari com esborrar un fitxer accidentalment.

- Mecanisme integrat de donacions a usuaris, a projectes i al mateix SourceForge.

Un usuari autenticat té una pàgina personal on es reuneix tota la informació del seu interès, com ara projectes a què està associat, temes o tasques que té pendents, així com fòrums i fitxers que ha declarat que vol vigilar. A més, perquè no hagi d'estar pendent de la seva pàgina personal, l'usuari rebrà per correu electrònic notificacions del que vol vigilar.

8.9.2. Hereus de SourceForge

El 2001 VA Software va estar a punt de fer fallida, en plena crisi de les punt-com. Llavors va anunciar una nova versió del seu programari SourceForge amb llicència no lliure, en un intent de procurar-se una font d'ingressos venent-lo a empreses per als seus desenvolupaments interns. Així mateix, va eliminar mecanismes que permetien bolcar un projecte per a migrar-lo a un altre lloc. Ambdós fets van ser vistos com una amenaça per la qual els milers de projectes allotjats a SourceForge quedarien atrapats a les mans d'una sola empresa, que utilitzaria la plataforma per a mostrar programari no lliure. Davant d'això i la possibilitat que el lloc tanqués es van desenvolupar fills de la versió lliure i es van obrir portals que s'hi basaven, entre els quals destaquen Savannah (http://savannah.gnu.org) [57], dedicat al projecte GNU i a altres programes amb llicència de tipus *copyleft*, oBerliOS (BerliOS: The Open Source Mediator) [13], concebut com a punt de trobada entre desenvolupadors de programari lliure i empreses. Tanmateix, això és només un pas amb vista a desenvolupar una plataforma distribuïda i replicada, on ningú no tingui control absolut dels projectes (Savannah The Next Generation, 2001) [98].

Un altre exemple de sistema de gestió de projectes de programari lliure és Launchpad (https://launchpad.net) [43], utilitzat per Ubuntu per al desenvolupament de cada versió de la distribució. Launchpad no és un dipòsit de codi font, sinó que té com a propòsit oferir suport per a seguiment de codi, incidències i traduccions. Per a això utilitza la ja esmentada eina Malone, que permet redirigir les incidències a cada repositori de codi dels mòduls afectats.

8.9.3. Altres llocs i programes

Naturalment, s'han desenvolupat i continuen desenvolupant-se sistemes de col·laboració, i algunes empreses fan negoci del manteniment i del servei d'aquests llocs. Per exemple, el projecte Tigris (Tigris.org: Open Source Software Engineering Tools) [64], que només manté projectes d'enginyeria de programari lliure, utilitza un portal de col·laboració (SourceCast) mantingut per una empresa de serveis (CollabNet), que també manté llocs de projectes individuals, com ara OpenOffice.org. Nous llocs estan adoptant nou programari lliure, com GForce (http://gforge.org) [30], utilitzat pel projecte Debian (http://alioth.debian.org) [5]. Pot veure's una comparació exhaustiva de molts llocs a "Comparison of free/open source hosting (FOSPhost) sites available for hosting projects externally from project owners" [202].

9. Estudi de casos

> "GNU, which stands for 'Gnu's Not Unix', is the name for the complete Unix-compatible software system which I am writing so that I can give it away free to everyone who can use it. Several other volunteers are helping me. Contributions of time, money, programs and equipment are greatly needed."
>
> "GNU, que significa 'Gnu No és Unix', és el nom d'un sistema de programari completament compatible amb Unix que estic escrivint per poder lliurar-lo lliurement a qui pugui utilitzar-lo. Hi ha diversos voluntaris ajudant-me. Són molt necessàries les contribucions de temps, diners, programes i equip."
>
> Richard Stallman, "The GNU Manifesto" (1985)

Aquest capítol està dedicat íntegrament a estudiar més a fons alguns dels projectes de programari lliure més interessants quant a impacte al món del programari lliure, resultats obtinguts, model de gestió, evolució històrica, etc. Sens dubte, el nombre de projectes que es presentaran és molt petit en comparació amb el nombre de projectes lliures totals (desenes de milers), per la qual cosa aquest capítol no es pot considerar complet, ni es podria considerar mai complet. Tot i així, esperem que després de llegir-lo, el lector pugui tenir almenys una percepció de com s'ha portat a la pràctica la teoria que s'ha anat presentant al llarg d'aquest llibre.

Els projectes escollits inclouen des d'aplicacions de baix nivell –les que interactuen més aviat amb el sistema físic de l'ordinador en comptes de fer-ho amb l'usuari– fins a entorns de treball per a l'usuari final. Hem inclòs també projectes de programari lliure que, al principi, no són íntegrament de desenvolupament. Aquest és el cas de les distribucions, el treball de les quals és més aviat d'integració, ja que es dediquen principalment a prendre un conjunt ampli, però limitat, d'aplicacions independents i a crear un sistema en què tot interactuï de manera efectiva, incloent-hi també facilitats per a instal·lar, actualitzar i esborrar noves aplicacions segons ho desitgi l'usuari.

Els projectes de nivell més baix que veurem seran Linux, el nucli del sistema operatiu lliure més popular d'avui dia, i FreeBSD, que combina un nucli de la família BSD amb una sèrie d'aplicacions i utilitats realitzades per tercers projectes, al més pur estil de les distribucions. Els entorns de treball d'usuari final que estudiarem seran KDE i GNOME, certament els més estesos i populars. Els servidors –un dels punts forts dels sistemes lliures– estaran representats en aquest capítol per Apache, líder al mercat dels servidors WWW. Així mateix, presentarem Mozilla, un dels clients WWW (en realitat, molt més que això) amb els quals comptem en el món del programari lliure. L'últim projecte que es veurà en aquest capítol és OpenOffice.org, un paquet ofimàtic (*suite*) lliure.

Per acabar, hem cregut convenient aprofundir en dos de les distribucions més populars, Red Hat Linux i Debian GNU/Linux, i fer una comparació quant a mida amb altres sistemes àmpliament utilitzats, com poden ser els de Microsoft Windows o Solaris.

Després de les presentacions dels diferents casos d'estudi, mostrarem una taula amb les característiques més notables de cada aplicació o projecte. Un dels paràmetres que pot sorprendre més el lector són els resultats d'estimació de cost, de durada i de nombre mitjà de desenvolupadors. Aquests resultats els hem obtingut per mitjans tradicionals en l'enginyeria del programari, en especial, el model COCOMO. El model COCOMO (*Software Engineering Economics*) [93] pren com a mesura d'entrada el nombre de línies de codi font i genera estimacions de cost total, temps de desenvolupament i esforç dedicat. COCOMO és un model pensat per a processos de generació de programari "clàssics" (desenvolupament en cascada o en V) i per a projectes de mida mitjana o gran, per la qual cosa les xifres que ens ofereix en el nostre cas han de ser preses amb molta cura. De totes maneres, els resultats ens poden donar una idea de l'ordre de magnitud en el qual ens movem, informant-nos sobre els esforços òptims necessaris si s'hagués utilitzat un model de desenvolupament propietari.

En general, el que més sorprèn dels resultats de COCOMO és la seva estimació de costos. En l'esmentada estimació es tenen en compte dos factors: el salari mitjà d'un desenvolupador i el factor d'*overhead*. En el càlcul de l'estimació de costos, s'ha pres el salari mitjà per a un programador de sistemes a temps complet de l'enquesta de l'any 2000 d'acord amb "Salary survey 2000" [235]. L'*overhead* és el sobrecost que tota empresa ha d'assumir perquè el producte surti al carrer amb independència del salari dels programadors. En aquest apartat s'inclouen des del salari de les secretàries i l'equip de màrqueting fins als costos de les fotocòpies, la llum, els equips de maquinari, etc. En resum, el cost calculat per COCOMO és el cost total que li suposaria a una empresa crear una aplicació de la mida especificada, i no s'ha de veure simplement com els diners que percebrien els programadors per realitzar l'aplicació. Una vegada dit això, els càlculs de costos deixen de semblar tan voluminosos.

9.1. Linux

El nucli Linux és, sens dubte, l'aplicació estrella del programari lliure, fins al punt que, fins i tot essent una petita part del sistema, ha donat nom a la seva totalitat. És més, es podria afirmar fins i tot que el mateix programari lliure es confon en multitud d'ocasions amb Linux, la qual cosa no deixa de ser un gran desencert, ja que existeix programari lliure que s'executa sobre sistemes que no es basen en Linux (de fet, una de les grans metes del moviment i de molts projectes de programari lliure és que les aplicacions puguin executar-se en multitud d'entorns). D'altra banda, també hi ha aplicacions que funcionen en Linux i que no són programari lliure (Acrobat Reader, el lector de documents PDF, també té la seva versió per a Linux).

Nota

Per a evitar l'associació del programari lliure únicament amb sistemes Linux existeixen diversos projectes que es dediquen a integrar i a distribuir aplicacions lliures que s'executen en sistemes Windows. Un dels pioners en aquesta activitat (i probablement el que va arribar a ser més conegut i complet) va ser GNUWin, que distribuïa en un CD autoarrencable més d'un centenar d'aplicacions lliures per a sistemes Win32. La majoria d'aquestes aplicacions també es troben disponibles en les distribucions GNU/Linux comunes, per la qual cosa era una bona eina per a anar preparant el canvi de sistemes Windows en GNU/Linux de manera assossegada. A començaments del 2007 poden trobar-se altres sistemes semblants, com WinLibre.

9.1.1. Història de Linux

La història de Linux és una de les més conegudes dins del món del programari lliure, segurament perquè s'assembla més a un conte que a la història d'un programa d'ordinador. El 1991, un estudiant finès anomenat Linus Torvalds va decidir que volia aprendre a utilitzar el mode protegit 386 en una màquina que la seva limitada butxaca li havia permès adquirir. Llavors –i encara avui en dia– en els cursos de sistemes operatius universitaris hi havia un nucli de sistema operatiu gestat amb finalitats acadèmiques anomenat Minix. Al capdavant del grup de desenvolupament de Minix –basat en els sistemes Unix tradicionals– hi havia un dels professors d'universitat més prestigiosos, Andrew Tanenbaum. Minix era un sistema limitat, però bastant capaç i ben dissenyat que comptava amb una gran comunitat –acadèmica i d'enginyeria– al seu voltant.

Minix tenia una llicència de lliure distribució i es podia utilitzar sense problema per a finalitats acadèmiques, però tenia el gran desavantatge que persones alienes a la Universitat d'Amsterdam no hi podien integrar millores, sinó que aquestes millores les havien de fer de manera independent, generalment per mitjà de pedaços. Això suposava que a la pràctica existís una versió de Minix oficial que tothom emprava, i després una llarga sèrie de pedaços que calia aplicar posteriorment per a obtenir funcionalitats addicionals.

A mitjan 1991, Linus –l'encara anònim estudiant finès– va enviar un missatge al grup de notícies de Minix per anunciar que començaria des de zero un nucli de sistema operatiu basat en Minix, però sense incloure'n codi. Llavors, Linus –encara que no digués explícitament que el publicaria amb una llicència lliure– va apuntar que el sistema que crearia no tindria les *barreres* que tenia Minix, per la qual cosa –sense saber-ho, i probablement sense voler-ho– estava fent el primer pas per quedar-se amb la comunitat que llavors s'aglutinava al voltant de Minix.

La versió 0.02, que data d'octubre de 1991, encara que molt limitada, ja podia executar terminals *bash* i el compilador GCC. En els mesos següents el nombre d'aportacions externes va anar creixent fins al punt que ja el març de 1992 Linus publicava la versió 0.95, que va ser àmpliament reconeguda com a gairebé estable. El camí cap a la versió 1.0 –que sol associar-se amb l'estabilitat– va ser, tanmateix, encara llarg: el desembre de 1993 es publicaria, per exemple, la versió 0.99pl14 (que és com la catorzena versió corregida de la versió 0.99) i finalment, el març de 1994, Linux 1.0 va veure la llum. Ja aleshores, Linux

es publicava sota les condicions de la llicència GPL, segons el mateix Torvalds una de les millors decisions que ha pres, ja que va ajudar molt a la distribució i la popularització del seu nucli. A "Evolution in open source software: a case study" [128] es pot trobar una anàlisi exhaustiva de l'evolució de les diferents versions del nucli Linux quant a mida i modularitat.

Nota

Per als annals de la història del programari lliure queda també el debat que va tenir lloc a finals de gener de 1992 al grup de notícies de Minix entre Andrew Tanenbaum i Linus Torvalds. Tanenbaum, probablement ja una mica molest per l'èxit de Torvalds amb la seva "joguina", va atacar de manera una mica desproporcionada Linux i Linus. El punt essencial és que Linux era un sistema monolític (el nucli és una peça que integra tots els manejadors i altres) i no micronucli o *microkernel* (el nucli té un disseny modular, la qual cosa permet que sigui més petit i que es puguin carregar mòduls sota demanda). Es pot seguir la discussió original tal com va ocórrer al grup de notícies a "The Tanenbaum-Torvalds debate" [214].

9.1.2. La manera de treballar de Linux

La forma de treballar de Torvalds no era gaire comuna en aquells temps. El desenvolupament es basava principalment en una llista de correu[9]. En la llista de correu no solament es discutia, sinó que també es desenvolupava. I és que a Torvalds li agradava sobre manera que tota la vida del projecte es veiés reflectida en la llista, per la qual cosa demanava que s'hi enviessin els pedaços. En contra del que es podria esperar (que s'enviés el pedaç com a adjunt), Linus preferia que s'enviés el codi al cos del missatge perquè ell i els altres poguessin comentar el codi. En qualsevol cas, encara que molts opinessin i enviessin correccions o noves funcionalitats, el que era indiscutible és que l'última paraula, el que decidia el que entrava a Linux, corresponia a Linus Torvalds. Fins a cert punt, aquest mode de funcionament continua vigent el 2007.

(9)L'adreça de la llista de correu electrònic és linux-kernel@vger.kernel.org. Es pot veure l'històric de tots els missatges a http://www.uwsg.indiana.edu/hypermail/linux/kernel/.

Nota

La consolidació de Linus Torvalds com a "dictador benevolent" ha donat peu a un ampli anecdotari dins del projecte. Així, es comenta que si una idea agrada, s'ha d'implementar. Si no agrada, també s'ha d'implementar. El corol·lari, per tant, és que les bones idees no serveixen per a res (sense codi, sens dubte). D'altra banda, si no agrada la implementació, cal insistir-hi. És ben conegut el cas de Gooch, per a qui el sant Job era un aprenent. Gooch va realitzar fins a cent quaranta-sis pedaços paral·lels fins que Linus va decidir finalment integrar-lo a la branca oficial del nucli.

Una altra de les idees innovadores de Torvalds va ser el desenvolupament paral·lel de dues branques del nucli: l'estable (el segon número de versió del qual sol ser parell, com per exemple 2.4.18) i la inestable (el segon número de versió del qual és senar, com ara 2.5.12). Com no podia ser de cap altra forma, Torvalds és el que decideix què entra en un lloc i en un altre (moltes de les decisions més polèmiques les podem ubicar precisament aquí). En qualsevol cas, Linux no té una planificació de lliuraments amb dates fixes: estarà llest quan estigui llest, mentrestant, caldrà esperar. Segur que el lector deu haver assumit a hores d'ara que la decisió de si està llest o no correspon, com no podia ser de cap altra manera, únicament a Linus.

El mètode de desenvolupament utilitzat en Linux resulta, tal com s'ha demostrat, molt eficaç quant a resultats: Linux té una gran estabilitat i hi ha generalment un interval ínfim de correcció d'errors (de vegades de l'ordre dels minuts), ja que compta amb milers de desenvolupadors. En aquesta situació, quan hi ha un error, la probabilitat que un d'ells el trobi és molt alta, i en cas que no el pugui resoldre el descobridor, ja hi haurà algú que trobi la solució de manera ràpida. En definitiva, podem veure com Linux compta amb milers de persones al mes dedicades al seu desenvolupament, la qual cosa indubtablement fa que el seu èxit no sigui gens sorprenent.

S'ha d'esmentar que aquesta forma de treballar és, tanmateix, molt cara quant a recursos. No és inusual que hi hagi moltes propostes mútuament excloents per a una nova funcionalitat o que es rebin una dotzena de pedaços per al mateix error. En la gran majoria dels casos, només un d'ells serà inclòs en el nucli finalment, per la qual cosa es pot considerar que la resta del temps i esforç dedicat pels desenvolupadors ha estat debades. El model de desenvolupament de Linux és, per tant, un model que funciona molt bé en Linux, però que certament no tots els projectes es poden permetre.

9.1.3. Estat actual de Linux

A començaments del 2007 Linux es desenvolupa sobre la branca 2.6, que ha suposat (quant a millores sobre la branca 2.4) la inclusió de NUMA (accés a memòria no uniforme, utilitzada de manera comuna en multiprocessadors), nous sistemes de fitxers, millores per a la comunicació en xarxes sense fil i arquitectures de so (ALSA) i moltes altres millores (si s'està interessat en el detall dels canvis respecte a versions anteriors es pot consultar "The wonderful world of Linux 2.6" [186]).

El model de desenvolupament de Linux ha sofert alguns canvis en els últims anys. Encara que la llista de correu de desenvolupament continua essent l'*ànima* del projecte, el codi ja no ha de passar necessàriament per ella. Hi va contribuir en gran manera BitKeeper, un sistema propietari de control de versions desenvolupat per la companyia BitMover seguint les estrictes recomanacions de Linus Torvalds. L'ús d'una eina no lliure va generar una gran polèmica, en la qual es va poder constatar una altra vegada la posició pragmàtica de Linus, ja que per a ell, i per a molts més, el sistema lliure de control de versions CVS és molt antiquat. La polèmica va quedar conclosa amb el desenvolupament de *git*, un sistema de control de versions distribuït amb característiques similars a BitKeeper i utilitzat actualment en el desenvolupament de Linux. En concret, el procés de desenvolupament de Linux segueix una jerarquia piramidal, en la qual els desenvolupadors proposen pedaços, compartits via correu entre nivells, que han de ser acceptats pel nivell superior de mantenidors de controladors i de fitxers. En un nivell superior hi ha els mantenidors de subsistemes, i en el nivell més alt, Linus Torvalds i Andrew Morton, que té l'última paraula per a les admissions dels pedaços.

A tall de resum, a la taula següent s'ofereix una radiografia del projecte Linux en la qual es veu com ha superat els cinc milions de línies de codi, de manera que es pot incloure entre els projectes lliures més grans (juntament amb Mozilla i OpenOffice.org). Quant a l'estimació de temps d'execució i de nombre mitjà de desenvolupadors que el farien òptim, podem observar que el primer és certament menor que els anys d'història amb què compta Linux. Això, d'altra banda, queda compensat amb escreix tenint en compte el segon, ja que el nombre mitjà de desenvolupadors a temps complet és superior del nombre de què disposa Linux.

Nota

L'estimació de costos que ens ofereix COCOMO és de 215 milions de dòlars americans, xifra que si posem en el context de les que se solen manejar més assíduament, suposa el doble del que els grans clubs de futbol solen pagar per una gran estrella.

Taula 1. Anàlisi de Linux

Pàgina web	http://www.kernel.org
Inici del projecte	Primer missatge a news.comp.os.minix: agost 1991
Llicència	GNU GPL
Versió analitzada	2.6.20 (versió estable del 20/02/2007)
Línies de codi font	5.195.239
Estimació de cost (segons COCOMO bàsic)	$ 215.291.772
Estimació de temps d'execució (segons COCOMO bàsic)	8,83 anys (105,91 mesos)
Estimació de nombre mitjà de desenvolupadors (segons COCOMO bàsic)	180,57
Nombe aproximat de desenvolupadors	Es compten per milers (encara que en els crèdits apareixen només centenars [219])
Eines d'ajuda al desenvolupament	Llista de correu i *git*

La composició de Linux quant a llenguatges de programació mostra un clar predomini de C, considerat com un llenguatge ideal per a la realització de sistemes crítics quant a velocitat. Quan la velocitat és un requisit tan estricte que ni tan sols C pot assolir, es programa directament en llenguatge assemblador, un fet que com podem veure, ocorre amb certa freqüència. El llenguatge assemblador té el desavantatge, en comparació amb C, que no és tan portable: cada arquitectura té el seu joc d'instruccions particular, per la qual cosa molt codi escrit per a una arquitectura en llenguatge assemblador ha de ser portat a les altres arquitectures. La presència de la resta dels llenguatges, com es pot observar a la taula adjunta, és marginal i es limita a funcions d'instal·lació i

utilitats de desenvolupament. La versió analitzada per a aquest llibre ha estat Linux 2.6.20 tal com va ser publicada el 20 de febrer de 2007 (sense l'aplicació de cap pedaç posterior).

Taula 2. Llenguatges de programació utilitzats a Linux

Llenguatge de programació	Línies de codi	Percentatge
C	4.972.172	95,71%
Assemblador	210.693	4,06%
Perl	3.224	0,06%
Yacc	2.632	0,05%
Shell	2.203	0,04%

9.2. FreeBSD

Com ja s'ha comentat al capítol dedicat a la història del programari lliure, existeixen un altre tipus de sistemes operatius lliures, a més del popular GNU/Linux. Una família d'ells són els "hereus" de les distribucions de la Universitat de Berkeley, a Califòrnia (EUA): els sistemes de tipus BSD. El més antic i conegut dels sistemes BSD és FreeBSD, la història del qual es remunta a començaments de 1993, quan Bill Jolitz va deixar de publicar les actualitzacions no oficials a 386BSD. Amb el suport de l'empresa Walnut Creek CDROM, que més tard va passar a anomenar-se BSDi, un grup de voluntaris va decidir prosseguir amb l'esforç de realitzar aquest sistema operatiu lliure.

L'objectiu principal del projecte FreeBSD és la creació d'un sistema operatiu que pugui ser utilitzat sense cap tipus d'obligacions ni lligams, però amb tots els avantatges de la disponibilitat del codi i d'un acurat procés que asseguri la qualitat del producte. L'usuari té la llibertat de fer amb el programari el que desitgi, o bé modificant-lo al seu gust o bé redistribuint-lo –de forma oberta o fins i tot tancada i sota les condicions que desitgi– amb modificacions o sense. Tal com el seu propi nom indica, el projecte FreeBSD es guia, per tant, per la filosofia de les llicències BSD.

9.2.1. Història de FreeBSD

La versió 1.0 va aparèixer a finals de 1993 i estava basada en 4.3 BSD Net/2 i 386BSD. 4.3 BSD Net/2 disposava de codi procedent dels anys setanta, quan Unix era desenvolupat per AT&T, fet que a la fi va suposar una sèrie de problemes legals que no es van resoldre fins que el 1995 FreeBSD 2.0 va ser publicat sense incloure codi originari d'AT&T, aquesta vegada basant-se en 4.4BSD-Li-

te, una versió *light* de 4.4 BSD (en la qual s'havien suprimit molts mòduls per problemes legals, a banda que la migració –*port*– per a sistemes Intel encara era incomplet) alliberada per la Universitat de Califòrnia.

La història de FreeBSD no seria completa si no s'expliqués res sobre les seves distribucions "germanes", NetBSD i OpenBSD. NetBSD va aparèixer amb una versió 0.8 a mitjan 1993. El seu principal objectiu era ser molt portable (encara que als seus començaments només fos una adaptació per a i386), per la qual cosa el seu lema va ser: "Sens dubte que executa NetBSD." OpenBSD va sorgir d'una escissió de NetBSD fonamentada en diferències filosòfiques (i també personals) entre desenvolupadors a mitjan 1996. El principal focus d'atenció d'aquest sistema operatiu és la seguretat i la criptografia –diuen que és el sistema operatiu més segur que hi ha–, encara que també conserva una gran portabilitat per basar-se en NetBSD.

9.2.2. Desenvolupament en FreeBSD

El model de desenvolupament utilitzat pel projecte FreeBSD està fortament basat en dues eines: el sistema de versions CVS i el sistema d'informe d'error GNATS. Entorn d'aquestes dues eines gira tot el projecte, com es pot comprovar pel fet que s'ha creat una jerarquia a partir d'aquestes mateixes. Així, sobre els *commiters* (aquells desenvolupadors amb dret a escriptura al CVS) és sobre qui recau tota la sobirania del projecte, o bé directament o bé indirectament mitjançant l'elecció del *core group*, tal com es veurà en l'apartat següent.

Per a realitzar informes d'error en GNATS no fa falta ser *commiter*, per la qual cosa qualsevol que així ho desitgi podrà notificar una errata. Totes les contribucions obertes (*open*) en GNATS són avaluades per un *commiter*, que podrà assignar (*analysed*) la tasca a un altre *commiter* o demanar més informació a la persona que va realitzar l'informe original (*feedback*). Hi ha situacions en les quals l'errata ha estat solucionada per a algunes branques recents; en aquests casos la branca s'especifica com a "suspesa" *(suspended*). En qualsevol cas, la meta és que l'informe es tanqui (*closed*) després d'haver corregit l'error.

FreeBSD distribueix el seu programari de dues formes: d'una banda, les migracions, un sistema que descarrega els codis font, els compila i instal·la l'aplicació a l'ordinador local, i per una altra, els paquets, que no són més que els codis font de les migracions precompilades i, per tant, en binari. L'avantatge més important de les migracions sobre els paquets és que els primers permeten a l'usuari configurar i optimitzar el programari per al seu ordinador. En canvi, el sistema de paquets, en estar ja precompilat, permet instal·lar el programari d'una manera més ràpida.

9.2.3. Presa de decisions en FreeBSD

El consell de directors de FreeBSD, conegut popularment com a *core team*, s'encarrega de marcar la direcció del projecte i de vetllar perquè es compleixin els seus objectius, així com d'intervenir en cas que hi hagi conflictes entre *commiters*. Fins a l'octubre del 2000 era un grup tancat al qual només s'entrava a formar part per invitació expressa del mateix *core team*. A partir de llavors, els seus membres són elegits de manera periòdica i democràticament pels *commiters*. La normativa més important per a l'elecció del *core team* és la següent:

1) Podran votar els *commiters* que hagin realitzat almenys un *commit* l'últim any.
2) El consell de directors es renovarà cada dos anys.
3) Els membres del consell de directors podran ser "expulsats" amb el vot de dos terços dels *commiters*.
4) Si el nombre de membres del consell de directors és menor de set, se celebraran noves eleccions.
5) Se celebraran noves eleccions si així ho demana un terç dels *commiters*.
6) El canvi de normativa requereix un quòrum de dos terços dels *commiters*.

9.2.4. Empreses entorn de FreeBSD

Hi ha nombroses empreses que ofereixen serveis i productes basats en FreeBSD, que el projecte té en compte a la seva pàgina. En aquesta presentació de FreeBSD coneixerem una mica més a fons les més significatives: BSDi i Walnut Creek CDROM.

FreeBSD va néixer en part quan es va funda el 1991 la companyia BSDi per gent del CSRG (Computer Systems Research Group) de la Universitat de Berkeley, companyia que es dedicaria al suport comercial per al nou sistema operatiu. A més de la versió comercial del sistema operatiu FreeBSD, BSDi també va desenvolupar altres productes, com un servidor d'Internet i de passarel·la.

Walnut Creek CDROM va néixer amb l'objectiu de comercialitzar FreeBSD com a producte acabat, de manera que es pogués considerar com una distribució a l'estil de les que existeixen per a GNU/Linux, però amb FreeBSD. El novembre de 1998, Walnut Creek va ampliar els seus horitzons amb la creació del portal FreeBSD Mall, que es dedicaria a comercialitzar tot tipus de productes sobre FreeBSD (des de la distribució en si mateixa fins a samarretes, revistes, llibres, etc.), a anunciar productes de tercers a la seva pàgina web i a donar suport professional de FreeBSD.

El març del 2000, BSDi i Walnut Creek es van unir sota el nom BSDi per fer front de manera conjunta al fenomen Linux, que estava deixant clarament en l'ombra els sistemes BSD en general i FreeBSD en particular. Un any més tard, el maig del 2001, Wind River va adquirir la part dedicada a la generació de

programari de BSDi, amb la clara intenció de potenciar el desenvolupament de FreeBSD per al seu ús en sistemes encastats i dispositius intel·ligents connectats a la Xarxa.

9.2.5. Estat actual de FreeBSD

Segons les últimes dades de l'enquesta que realitza periòdicament Netcraft, el nombre de servidors web que executen FreeBSD s'apropa als dos milions d'unitats. Un nou usuari que volgués instal·lar FreeBSD podria elegir entre la versió 6.2 (que es podria considerar com la versió "estable") o la branca més avançada o "de desenvolupament". Mentre que la primera ofereix major estabilitat –sobretot en àrees com el multiprocessament simètric, que han estat totalment reelaborades en les noves versions–, la segona permet gaudir de les últimes novetats. També és important tenir en compte que les versions de desenvolupament solen incloure codi de proves, la qual cosa fa que la velocitat del sistema es vegi afectada sensiblement.

Una de les utilitats estrella de FreeBSD són les anomenades *presons* (*jail*, en anglès). Les presons permeten minimitzar el dany causat per un atac a serveis de xarxa bàsics, com podrien ser Sendmail per al correu electrònic o BIND (Berkeley Internet Name Domain) per a gestionar els noms. Els serveis són introduïts en una presó perquè s'executin en un entorn d'aïllament. La gestió de les presons es pot realitzar mitjançant un seguit d'utilitats incloses en FreeBSD.

9.2.6. Radiografia de FreeBSD

Com hem anat comentant al llarg d'aquest últim apartat, la tasca de BSD no es restringeix únicament al desenvolupament d'un nucli del sistema operatiu, sinó que també inclou la integració de multitud d'utilitats que es distribueixen conjuntament a l'estil de les distribucions de GNU/Linux. El fet que el procés de desenvolupament de FreeBSD estigui íntimament lligat al sistema de control de versions CVS fa que l'estudi d'aquest ens pugui donar una bona aproximació de tot el que conté. Les xifres que es mostren a continuació són les corresponents a l'anàlisi de FreeBSD efectuat el 21 d'agost de 2003.

Un dels aspectes més interessants de FreeBSD és que els seus números s'assemblen molt als que es presenten més endavant en relació a KDE i GNOME: la mida del programari supera àmpliament els cinc milions de línies de codi, el nombre de fitxers s'aproxima als 250.000 i el nombre total de *commits* se situa entorn dels dos milions. Tanmateix, és interessant observar que la principal diferència entre GNOME i KDE respecte a FreeBSD és l'*edat* del projecte. FreeBSD acaba de complir recentment la dècada d'existència i gairebé dobla en temps els entorns d'escriptori amb què l'estem comparant. Que la mida sigui similar, encara que el temps de desenvolupament ha estat superior, es deu en gran part al fet que el nombre de desenvolupadors que ha atret FreeBSD és més petit. Amb dret d'escriptura en el CVS (*commiter*) n'hi ha llistats uns quatre-cents, mentre que els col·laboradors que s'esmenten al manual de FreeBSD

són prop d'un miler. Per això l'activitat que registra el CVS de FreeBSD és menor de mitjana (cinc-cents *commits* diaris) que la que registraven tant GNOME (nou-cents) com KDE (mil set-cents comptant els *commits* automàtics).

Hem considerat com a sistema bàsic de FreeBSD tot allò que penja del directori *src/src* del mòdul *root* del CVS. L'activitat que ha anat registrant el sistema bàsic al llarg dels últims deu anys és de més de mig milió de *commits*. La seva mida supera els cinc milions de línies de codi, encara que cal comentar que no només inclou el nucli, sinó multitud d'utilitats addicionals, fins i tot jocs. Si tenim en compte només el nucli (que es troba sota el subdirectori *sys*), la seva mida és d'1,5 milions de línies de codi font, predominantment en C.

Resulta interessant veure com l'estimació de temps donada per COCOMO concorda a la perfecció amb el temps real del projecte FreeBSD, encara que l'estimació del nombre mitjà de desenvolupadors és, de llarg, més gran que la real. Cal dir que l'últim any només uns setanta-cinc *commiters* han estat actius, mentre que COCOMO suposa que durant els deu anys de desenvolupament el nombre de desenvolupadors hauria de ser de 235.

Finalment, es pot destacar, com s'ha dit anteriorment, que l'activitat principal de FreeBSD se situa entorn del CVS i de sistema de control d'errates i activitats GNATS.

Taula 6. Anàlisi de FreeBSD

Pàgina web	http://www.FreeBSD.org
Inici del projecte	1993
Llicència	De tipus BSD
Versió analitzada	4.8
Línies de codi font	7.750.000
Línies de codi font (només nucli)	1.500.000
Nombre de fitxers	250.000
Estimació de cost	$ 325.000.000
Estimació de temps d'execució	10,5 anys (126 mesos)
Estimació de nombre mitjà de desenvolupadors	235
Nombre aproximat de desenvolupadors	400 *commiters* (1.000 col·laboradors)
Nombre de *commiters* actius l'últim any	75 (menys 20% del total)
Nombre de *commiters* actius en els dos últims anys	165 (al voltant del 40% del total)
Nombre de *commits* en el CVS	2.000.000

Nombre mitjà de *commits* (totals) al dia	Uns 500
Eines d'ajuda al desenvolupament	CVS, GNATS, llistes de correu i lloc de notícies

C és el llenguatge predominant en FreeBSD, i guarda una distància respecte a C++ superior a la dels altres casos que hem estudiat en aquest capítol. És interessant observar que el nombre de línies de codi de llenguatge assemblador contingudes en FreeBSD concorda en l'ordre de magnitud amb les que té Linux, encara que les corresponents al nucli només són unes vint-i-cinc mil en total. En resum, es podria dir que en FreeBSD el que mana són els llenguatges més *clàssics* dins del programari lliure –C, Shell i Perl– i que la penetració dels llenguatges que hem observat en altres aplicacions i projectes –C++, Java, Python...– no ha tingut lloc.

Taula 7. Llenguatges de programació utilitzats en FreeBSD

Llenguatge de programació	Línies de codi	Percentatge
C	7.080.000	92,0%
Shell	205.000	2,7%
C++	131.500	1,7%
Assemblador	116.000	1,5%
Perl	90.900	1,20%
Yacc	5.800	0,75%

9.2.7. Estudis acadèmics sobre FreeBSD

Fins i tot essent certament un projecte molt interessant (podem obtenir tota la seva història –des de fa 10 anys!– mitjançant l'anàlisi del sistema de versions), l'atenció que ha despertat FreeBSD entre la comunitat científica ha estat més aviat petita. D'aquesta falta d'interès se salva un equip d'investigació que ha estudiat el projecte FreeBSD des de diversos punts de vista ("Incremental and decentralized integration in FreeBSD") [149], i que ha posat especial atenció en com es resolen els problemes de la integració de programari de manera incremental i descentralitzada.

9.3. KDE

Encara que amb probabilitat no va ser la primera solució quant a entorns d'escriptori *amigables* per a l'usuari, la difusió a mitjan 1995 del sistema operatiu Windows 95™ va suposar un canvi radical en la interacció dels usuaris normals amb els ordinadors. Dels sistemes unidimensionals de línia d'instruccions (els terminals), es va passar a la metàfora d'entorn de l'escriptori bidimensional, on el ratolí va guanyar terreny al teclat. Windows 95™, més que una

innovació tecnològica, ha de ser acreditat com el sistema que va aconseguir endinsar-se en tots els entorns personals i d'oficina, marcant les pautes a seguir (regles tècniques i socials que, a començaments del segle XXI, de vegades encara continuem patint).

Anteriorment als sistemes d'escriptori, cada aplicació gestionava l'aparença i la forma d'interactuar amb l'usuari de manera autònoma. Als escriptoris, al contrari, les aplicacions han de comptar amb propietats comunes i un aspecte compartit entre aplicacions de manera que això suposi un alleujament per a l'usuari, que pot *reutilitzar la interacció* apresa d'una aplicació a l'altra. També va resultar un alleujament per als desenvolupadors d'aplicacions, ja que no s'havien d'enfrontar amb el problema de crear els elements interactius des de zero (una tasca sempre complicada), sinó que podien partir d'un marc i unes regles predefinides.

9.3.1. Història de KDE

Els seguidors de Unix ràpidament es van fer ressò del notable èxit de Windows 95 i, en vista que els entorns Unix mancaven de sistemes tan intuïtius alhora que lliures, van decidir posar-se a treballar. Fruit d'aquesta preocupació va néixer el 1996 el projecte KDE –K Desktop Environment–, ideat per Matthias Ettrich (creador de LyX, un programa d'edició en mode gràfic de TeX) i altres *hackers*. El projecte KDE es va plantejar els objectius següents:

Nota

Originàriament el nom KDE significava Kool Desktop Environment, però amb el temps es va decidir que passés a anomenar-se simplement *K* Desktop Environment. L'explicació oficial va ser que la lletra K és la que precedeix en l'alfabet llatí la *L* de Linux.

- Dotar els sistemes Unix d'un entorn amigable que fos obert alhora, estable, de confiança i poderós.

- Desenvolupar un conjunt de biblioteques per a escriure aplicacions estàndard sobre el sistema gràfic per a Unix X11.

- Crear una sèrie d'aplicacions que permetessin a l'usuari emprendre els seus objectius de manera eficaç.

Quan els integrants del projecte KDE de nova creació van decidir utilitzar una biblioteca orientada a objectes anomenada Qt, propietat de la firma noruega Trolltech™, que no estava emparada sota una llicència de programari lliure, va sorgir una gran polèmica. Es donava la circumstància que, malgrat que les aplicacions de KDE estaven llicenciades sota la GPL, enllaçaven amb aquesta biblioteca, de manera que es feia impossible la seva redistribució. Conseqüentment, s'estava violant una de les quatre llibertats del programari lliure enunciades per Richard Stallman en el Manifest del Programari Lliure [117]. Des de la versió 2.0, Trolltech distribueix Qt sota una llicència dual que especifica que si l'aplicació que fa ús de la biblioteca és GPL, llavors la llicència vàlida per a Qt és la GPL. Gràcies a això, un dels debats més calents i irats dins del món del programari lliure va tenir, per sort, un final feliç.

9.3.2. Desenvolupament de KDE

KDE és dels pocs projectes de programari lliure que compleixen un calendari de llançament de noves versions de manera generalitzada (recordem, per exemple, que hi haurà una nova versió de Linux "quan estigui llesta", mentre que, com veurem més endavant, en el cas de GNOME s'han donat retards significatius a l'hora de publicar noves versions). La numeració de les versions segueix una política perfectament definida. Les versions de KDE consten de tres números de versió: un de més gran i dos de més petits. Per exemple, en KDE 3.1.2, el número més gran seria el 3, i els més petits l'1 i el 2. Les versions amb el mateix número més gran tenen compatibilitat binària, per la qual cosa no fa falta recompilar les aplicacions. Fins ara, el número més gran s'ha canviat en paral·lel als canvis que s'han introduït en la biblioteca Qt, de manera que es pot veure que els desenvolupadors van voler aprofitar les noves funcionalitats de la biblioteca Qt en la versió imminent de KDE.

Quant als números més petits, les versions amb un únic número més petit són versions en les quals s'han inclòs tant noves funcionalitats com correcció de les errates trobades. Les versions amb un segon número més petit no inclouen noves funcionalitats sobre les versions amb primer número més petit, i només contenen correcció d'errors. Per aclarir-ho amb un exemple: KDE 3.1 és una versió de la tercera generació de KDE (número més gran 3) a la qual s'han afegit noves funcionalitats, mentre que KDE 3.1.1 és la versió anterior –amb les mateixes funcionalitats–, però amb les errates que s'han trobat corregides.

KDE es va constituir, poc després de començar, en una associació registrada a Alemanya (KDE e.V.) i, com a tal, té uns estatuts que l'obliguen a comptar amb un consell directiu. La influència d'aquest consell directiu sobre el desenvolupament és nul·la, ja que la seva tasca és fonamentalment l'administració de l'associació, en especial de les donacions que percep el projecte. Per a la promoció i la difusió de KDE –incloent-hi empreses interessades– es va crear la Lliga KDE, de la qual parlarem a continuació.

9.3.3. La Lliga KDE

La Lliga KDE (KDE League, en la seva denominació original en anglès) és un grup d'empreses i de particulars de KDE que té l'objectiu de facilitar la promoció, la distribució i el desenvolupament de KDE. Les empreses i els particulars que participen en la Lliga KDE no han d'estar pas directament involucrats en el desenvolupament de KDE (encara que s'anima tots els membres a fer-ho), sinó que simplement representen un marc industrial i social amic de KDE. Els objectius de la Lliga KDE són els següents:

- Promoure, proveir i facilitar l'educació formal i informal de les funcionalitats, les capacitats i altres qualitats de KDE.

- Animar corporacions, governs, empreses i individus a usar KDE.

- Animar corporacions, governs, empreses i individus a participar en el desenvolupament de KDE.

- Proveir de coneixements, informació, direcció i posicionament entorn de KDE pel que fa al seu ús i el seu desenvolupament.

- Promoure la comunicació i la cooperació entre els desenvolupadors de KDE.

- Promoure la comunicació i la cooperació entre els desenvolupadors de KDE i el públic mitjançant publicacions, articles, llocs web, encontres, participació en congressos i exposicions, notes de premsa, entrevistes, material promocional i comitès.

Les empreses que participen en la KDE League són principalment distribucions (SuSE, ara part de Novell, Mandriva, TurboLinux, Lindows i Hancom, una distribució de programari lliure coreana), empreses de desenvolupament (Trolltech i Klarälvdalens Datakonsult AB), a més del gegant IBM i d'una empresa creada amb la finalitat de promoure KDE (KDE.com). Entre totes, es pot destacar per la implicació fonamental Trolltech, Novell i Mandriva Software, el model de negoci del qual està íntimament lligat al projecte KDE:

- Trolltech és una companyia noruega establerta a Oslo que desenvolupa Qt, la biblioteca que fa les vegades d'interfície gràfica d'usuari i API per al desenvolupament d'aplicacions, encara que també pot funcionar com a element encastat en PDA (com per exemple en els Sharp Zaurus). La importància del projecte KDE a Trolltech es pot constatar per dos elements bàsics de la seva estratègia comercial: d'una banda, reconeix KDE com la seva principal forma de promoció, encoratjant el desenvolupament de l'escriptori i acceptant i implementant les millores o les modificacions proposades; d'altra banda, alguns dels desenvolupadors més importants de KDE treballen professionalment per a Trolltech –el cas més conegut és el del mateix Mathias Ettrich, fundador del projecte–, cosa que sens dubte beneficia tant el projecte KDE com la companyia mateix. La implicació de Trolltech en el projecte KDE no es limita exclusivament a la biblioteca Qt, com es pot veure pel fet que un dels desenvolupadors principals de KOffice –el paquet ofimàtic de KDE– tingui en l'actualitat un contracte a temps parcial amb ells.

- SuSE (ara part de Novell) sempre ha demostrat una especial predilecció pel sistema d'escriptori KDE, en part perquè una gran majoria dels seus desenvolupadors són d'origen alemany o centreeuropeu, igual com la mateixa companyia. Coneixedora del fet que com millor i més fàcil sigui l'entorn d'escriptori que ofereixi la seva distribució, més gran en serà la implantació i, per tant, les vendes i la petició de suport, SuSE ha tingut sempre una

política molt activa quant a la dedicació de pressupost per a professionalitzar posicions clau dins del projecte KDE. D'aquesta manera, en l'actualitat, l'administrador del sistema de control de versions i un altre parell de desenvolupadors principals tenen una nòmina de SuSE. Així mateix, dins de la plantilla deSuSE hi ha una dotzena de desenvolupadors que poden dedicar part del seu temps laboral al desenvolupament de KDE.

- La distribució Mandriva és un altre dels grans benefactors de KDE, i compta en la plantilla amb diversos dels desenvolupadors principals. La seva situació econòmica el 2003 –amb suspensió de pagaments inclosa– ha fet que en els últims temps hagi perdut influència.

9.3.4. Estat actual de KDE

Després de la publicació de KDE 3 el maig de 2002, l'opinió generalitzada és que els escriptoris lliures es troben a l'altura dels seus competidors propietaris. Entre els seus grans èxits hi ha la incorporació d'un sistema de components (KParts) que permet encastar unes aplicacions en altres (un tros de full de càlcul de KSpread en el processador de textos KWord) i el desenvolupament de DCOP, un sistema de comunicació entre processos simple i amb autenticació. DCOP va ser l'aposta del projecte en detriment de les tecnologies CORBA, un tema d'ampli debat dins dels escriptoris lliures, en especial entre GNOME –que es va decidir a utilitzar tecnologies CORBA– i KDE. La història sembla haver posat cada tecnologia al seu lloc, com es pot veure amb la proposta de DBUS (una espècie de DCOP millorat) per part del FreeDesktop.org, un projecte interessat a fomentar la interoperabilitat i l'ús de tecnologies conjuntes als escriptoris lliures, que casualment està liderat per un dels *hackers* de GNOME més reconeguts.

La taula resum següent conté les característiques més importants del projecte KDE. Les llicències que accepta el projecte depenen de si es tracta d'una aplicació o d'una biblioteca. Les llicències de les biblioteques permeten una major "flexibilitat" a tercers; en altres paraules, possibiliten que tercers puguin crear aplicacions propietàries que enllacin amb les biblioteques.

L'última versió de KDE és, a començaments del 2007, la 3.5.6, i per a mitjan any està prevista la quarta generació, KDE 4, que es basarà en Qt4. Els canvis de generació suposen un gran esforç d'adaptació, una tasca tediosa i costosa en temps. Tanmateix, això no ha de suposar que les aplicacions "antigues" deixin de funcionar. Generalment, perquè continuïn funcionant s'inclouen també les antigues versions de les biblioteques sobre les quals es basen, encara que això signifiqui haver de carregar diverses versions de les biblioteques en memòria simultàniament, amb el consegüent malbaratament de recursos del sistema. Aquest fet és vist pels desenvolupadors de KDE com una cosa inherent a la mateixa evolució del projecte i, per tant, com un mal menor.

9.3.5. Radiografia de KDE

Pel que fa a la mida de KDE, les xifres que es mostraran a continuació corresponen a l'estat del CVS l'agost del 2003, per la qual cosa cal prendre les precaucions tradicionals que ja s'han comentat i encara una altra més: algun dels mòduls que s'han considerat en aquest estudi encara es troba en fase de desenvolupament i no compleix el requisit de ser un producte acabat. Això no ens ha de molestar gens ni mica per als nostres propòsits, ja que estem més interessats en l'ordre de magnitud dels resultats que en la xifra exacta.

El codi font inclòs en el CVS de KDE suma en total més de sis milions de línies de codi en diferents llenguatges de programació, tal com mostrarem més endavant. El temps que es necessitaria per a crear KDE s'aproximaria a nou anys i mig, una xifra superior als set anys que té el projecte, i el nombre mitjà estimat de desenvolupadors a temps complet s'aproximaria a dos-cents. Si tenim en compte que KDE disposa d'unes vuit-centes persones amb accés d'escriptura al CVS el 2003 (la meitat de les quals han estat inactives en els últims dos anys) i que el nombre de desenvolupadors de KDE contractats a temps complet no ha superat en cap moment la vintena, podem veure que la productivitat de KDE és, de llarg, superior a l'estimació que ofereix COCOMO.

Nota

Una empresa que volgués desenvolupar un producte d'aquesta mida des de zero necessitaria invertir més de 250 milions de dòlars, una xifra que a tall de comparació és el que va invertir una firma d'automòbils en la creació d'una nova planta de producció a l'est d'Europa o el que una coneguda petroliera planeja gastar per obrir dues-centes gasolineres a Espanya.

És interessant veure que una gran part de l'esforç –gairebé la meitat que el del desenvolupament– del projecte KDE el podem situar en la traducció de la interfície d'usuari i de la documentació. Encara que molt poques (uns quants milers) de les línies de programació es dediquen a aquesta tasca, el nombre de fitxers dedicats a aquesta comesa puja als setanta-cinc mil per a traduccions (xifra que s'eleva fins als cent mil si incloem la documentació en els seus diferents formats), la qual cosa representa gairebé la quarta (tercera) part dels 310.000 fitxers que hi ha en el CVS. L'activitat conjunta del CVS és de mil dos-cents *commits* diaris, per la qual cosa el temps mitjà entre *commits* és de prop d'un minut[10].

[(10)]Cal fer dues observacions a aquest resultat: la primera és que es considera que un *commit* que inclogui diversos fitxers és com si s'hagués fet un *commit* separadament per a cada fitxer; la segona és que la xifra de *commits* és una xifra estimada, ja que el projecte disposa d'una sèrie de *scripts* que realitza *commits* de manera automàtica.

Quant a les eines, els llocs d'informació i els esdeveniments d'ajuda al desenvolupament, veiem que el ventall de possibilitats que ofereix KDE és molt més ampli que l'utilitzat a Linux. A més del sistema de control de versions i de les llistes de correu, KDE compta amb una sèrie de llocs web on es pot trobar informació i documentació tècnica i no tècnica del projecte. Entre aquests llocs també hi ha un lloc de notícies on es presenten noves solucions i es debaten propostes. El lloc de notícies, tanmateix, no pot considerar-se com a substitut de les llistes de correu –que, igual com en Linux, és on es troben els verdaders debats i la presa de decisions i estratègies de futur–, sinó com un punt de tro-

bada amb els usuaris. Finalment, KDE organitza des de fa tres anys reunions periòdiques en les quals els desenvolupadors i els col·laboradors es reuneixen durant una setmana per presentar les últimes novetats, desenvolupar, debatre, conèixer-se i passar-ho bé (no necessàriament en aquest ordre).

Taula 8. Anàlisi de KDE

Pàgina web	http://www.kde.org
Inici del projecte	1996
Llicència (per a aplicacions)	GPL, QPL, MIT, Artistic
Llicència (per a biblioteques)	LGPL, BSD, X11
Versió analitzada	3.1.3
Línies de codi font	6.100.000
Nombre de fitxers (codi, documentació, etc.)	310.000 fitxers
Estimació de cost	$ 255.000.000
Estimació de temps d'execució	9,41 anys (112,98 mesos)
Estimació de nombre mitjà de desenvolupadors	200,64
Nombre aproximat de desenvolupadors	Uns 900 *commiters*
Nombre de *commiters* actius l'últim any	Al voltant de 450 (aproximadament el 50% del total)
Nombre de *commiters* actius en els dos últims anys	Uns 600 (aproximadament el 65% del total)
Nombre aproximat de traductors (actius)	Uns 300 traductors per a més de 50 llengües (incloses l'esperanto)
Nombre de *commits* (de desenvolupadors) en el CVS	Aproximadament 2.000.000 (xifra estimada sense *commits* automàtics)
Nombre de *commits* (de traductors) en el CVS	Aproximadament 1.000.000 (xifra estimada sense *commits* automàtics)
Nombre mitjà de *commits* (totals) al dia	1.700
Eines, documentació i esdeveniments d'ajuda al desenvolupament	CVS, llistes de correu, lloc web, lloc de notícies, reunions anuals

Quant als llenguatges de programació utilitzats en KDE, predomina l'ús de C++. Això es deu principalment que aquest és el llenguatge nadiu de Qt, encara que es duu a terme un gran esforç per a fer enllaços que permetin el desenvolupament en altres llenguatges de programació. Certament, el nombre de línies de codi en els llenguatges minoritaris correspon gairebé íntegrament

al mateix projecte de creació de l'enllaç, encara que això no vol dir que no s'utilitzin en absolut, ja que hi ha un gran nombre de projectes externs a KDE que els utilitza.

Taula 9. Llenguatges de programació utilitzats en KDE

Llenguatge de programació	Línies de codi	Percentatge
C++	5.011.288	82,05%
C	575.237	9,42%
Objective C	144.415	2,36%
Shell	103.132	1,69%
Java	87.974	1,44%
Perl	85.869	1,41%

9.4. GNOME

El projecte GNOME té com a principal objectiu crear un sistema d'escriptori per a l'usuari final que sigui complet, lliure i fàcil d'usar. Així mateix, pretén que GNOME sigui una plataforma molt potent de cara al desenvolupador. GNOME és l'acrònim en anglès de *GNU Network Object Model Environment*. Des dels inicis de GNOME s'han proposat diverses formes de traduir-lo al castellà, però no se n'ha trobat cap que hagi satisfet tothom. Tanmateix, del seu nom podem veure que GNOME és part del projecte GNU. En l'actualitat, tot el codi contingut en GNOME ha d'estar sota llicència GNU GPL o GNU LGPL. També veiem que les xarxes i el modelatge orientat a objectes tenen una importància capital.

9.4.1. Història de GNOME

Mentre es continuava discutint sobre la llibertat de KDE, la història va voler que l'estiu de 1997, Miguel de Icaza i Nat Friedman coincidissin a Redmond en unes jornades organitzades per Microsoft™. És probable que aquest encontre propiciés en tots dos un gir radical que va suposar tant la creació de GNOME per part de Miguel de Icaza a la tornada a Mèxic (juntament amb Federico Mena Quintero), com la seva admiració per les tecnologies d'objectes distribuïts. D'Icaza i Mena van decidir crear un entorn alternatiu a KDE, ja que van considerar que una reimplementació d'una biblioteca propietària hauria estat una tasca destinada a fracassar. Havia nascut GNOME.

Des d'aquells temps llunyans de 1997 fins a l'actualitat, GNOME ha anat creixent gradualment amb les seves publicacions reiterades. El novembre de 1998 es va llançar la versió 0.99, però la primera realment popular, distribuïda pràcticament per qualsevol distribució de GNU/Linux, seria GNOME 1.0, de març de 1999. Es pot destacar que l'experiència d'aquesta primera versió *estable* de

GNOME no va ser gaire satisfactòria, ja que molts la van considerar plena d'errates crítiques. Per això, GNOME October (GNOME 1.0.55) és tractada com la primera versió de l'entorn d'escriptori GNOME realment estable. Com es pot observar, amb GNOME October es va intentar evitar versions de publicació numerades per a no entrar en una "cursa" de versions amb KDE. La realització de la primera GUADEC, la conferència de desenvolupadors i usuaris europeus de GNOME, celebrada a París l'any 2000, no va coincidir en el temps per poc amb la publicació d'una nova versió de GNOME, anomenada GNOME April. Va ser l'última que va portar un mes com a nom de publicació, ja que es va mostrar que aquest sistema causava més confusió que una altra cosa (per exemple, GNOME April és posterior a GNOME October, encara que el sentit comú ens faria pensar el contrari). L'octubre d'aquell any, després de ser debatuda durant mesos en diferents llistes de correu, es va crear la Fundació GNOME, que serà presentada més endavant.

GNOME 1.2 va ser un pas endavant quant a l'arquitectura usada per GNOME, arquitectura que es va continuar usant en GNOME 1.4. Aquesta època va estar caracteritzada per la segona edició de la GUADEC, aquesta vegada a Copenhaguen. El que havia començat essent una reunió minoritària d'alguns *hackers*, va acabar convertint-se en un gran esdeveniment que va atreure mirades de tota la indústria del programari.

Mentrestant, el litigi sobre la llibertat de KDE es va resoldre amb el canvi d'actitud de Trolltech™, que va acabar llicenciant Qt sota una llicència dual, que era de programari lliure per a les aplicacions que són programari lliure. Avui en dia no hi ha pas cap dubte que tant GNOME com KDE són entorns d'escriptori lliures, per la qual cosa podem considerar que el desenvolupament de GNOME ha propiciat el fet de tenir no tan sols un entorn d'escriptori lliure, sinó dos.

9.4.2. La Fundació GNOME

El problema més difícil d'abordar quan se sent parlar de GNOME per primera vegada és l'organització dels més de mil contribuents al projecte. Resulta paradoxal que un projecte l'estructura del qual és més aviat anàrquica, arribi a fructificar i tiri endavant uns objectius complexos i a l'abast de poques multinacionals del sector de la informàtica.

Encara que GNOME va néixer amb una clara intenció de realitzar un entorn amigable i potent a què s'anaven afegint nous programes, aviat es va veure la necessitat de crear un òrgan que tingués certes competències que permetessin potenciar l'ús, el desenvolupament i la difusió de GNOME: d'aquesta forma, l'octubre del 2000 es va crear la Fundació GNOME, que té la seu a Boston (EUA).

La Fundació GNOME no és un consorci industrial, sinó una organització sense finalitat de lucre que té les funcions següents:

- Coordina les publicacions.
- Decideix quins projectes són part de GNOME.
- És la veu oficial (per a la premsa i per a organitzacions tant comercials com no comercials) del projecte GNOME.
- Patrocina conferències relacionades amb GNOME (com la GUADEC).
- Representa GNOME en altres conferències.
- Crea estàndards tècnics.
- Promou l'ús i el desenvolupament de GNOME.

A més, la Fundació GNOME permet la recaptació de fons econòmics per patrocinar i impulsar les funcions abans esmentades, fet que abans de crear-se era impossible realitzar de manera transparent.

En l'actualitat, la Fundació GNOME compta amb un empleat a temps complet que s'encarrega de resoldre tots els treballs burocràtics i organitzatius que es donen en una organització sense finalitat de lucre que realitza reunions i conferències de manera periòdica.

En termes generals, la Fundació GNOME s'estructura en dos grans consells: un consell directiu i un consell consultor.

El consell directiu (*Board of Directors*) està integrat, pel cap alt, per catorze membres elegits democràticament pels membres de la Fundació GNOME. Se segueix un model "meritocràtic", la qual cosa vol dir que per ser membre de la Fundació GNOME s'ha d'haver col·laborat d'alguna o una altra manera amb el projecte GNOME. L'aportació no ha de ser necessàriament codi font; també hi ha tasques de traducció, organització, difusió, etc., per les quals un pot demanar ser membre de la Fundació GNOME i tenir dret a vot. Per tant, són els membres de la Fundació els qui es poden presentar al consell directiu i els qui, democràticament, elegeixen els seus representants entre els qui s'hagin presentat. En l'actualitat, la votació es duu a terme per correu electrònic. La durada del càrrec com a conseller director és d'un any, període després del qual es tornen a convocar eleccions.

Hi ha unes normes bàsiques per a garantir la transparència del consell directiu. La que crida més l'atenció és la limitació de membres afiliats a una mateixa empresa, la qual no pot excedir de quatre empleats. És important posar èmfasi que els membres del consell directiu ho són sempre a nivell personal, i mai en representació d'una companyia. Tot i així, i després d'una llarga discussió, es va acceptar incloure aquesta clàusula per a evitar suspicàcies.

L'altre consell dins de la Fundació GNOME és el consell consultor, òrgan sense capacitat de decisió que serveix de vehicle de comunicació amb el consell directiu. Està compost per companyies comercials de la indústria del progra-

mari, així com per organitzacions no comercials. En l'actualitat els seus membres són Red Hat, Novell, Hewlett-Packard, Mandrake, SUN Microsystems, Red Flag Linux, Wipro, Debian i la Free Software Foundation. Per a formar part del consell de consultors s'exigeix una quota a totes les empreses amb més de deu empleats.

9.4.3. La indústria entorn de GNOME

GNOME ha aconseguit endinsar-se de manera substancial en la indústria, de manera que diverses empreses han participat molt activament en el desenvolupament. De totes, els casos més significatius són els de Ximian Inc., Eazel, els RHAD Labs de Red Hat i, més recentment, SUN Microsystems. A continuació, es descriuen, per a cada cas, tant les motivacions de les companyies, com les seves aportacions més importants a l'entorn d'escriptori GNOME:

- Ximian Inc. (al començament Helix Inc.) és el nom de l'empresa que van fundar el 1999 Miguel de Icaza, cofundador de GNOME, i Nat Friedman, un dels *hackers* de GNOME. La seva comesa principal era reunir sota un mateix paraigua els desenvolupadors més importants de GNOME per a potenciar-ne el desenvolupament, per la qual cosa no és estrany que compti o que hagi comptat entre els seus empleats amb una vintena llarga dels desenvolupadors més actius de GNOME. L'aplicació en la qual Ximian va insistir més des del començament va ser Evolution, un complet sistema de gestió d'informació personal a l'estil de Microsoft Outlook que incloïa client de correu electrònic, agenda i directori de contactes. Els productes que Ximian comercialitzava són el Ximian Desktop (una versió de GNOME amb finalitats més corporatives), Red Carpet (principalment, encara que no exclusivament, un sistema de distribució de programari de GNOME) i finalment MONO (una reimplementació de la plataforma de desenvolupament .NET), encara que aquest últim projecte, per ara, no tingui res a veure amb GNOME. Ximian també ha desenvolupat una aplicació que serveix perquè Evolution interactuï amb un servidor Exchange 2000. Aquesta aplicació, encara que bastant petita, va ser molt polèmica perquè es va publicar amb una llicència no lliure (posteriorment, el 2004, aquest component va passar també a llicenciar-se com a programari lliure). L'agost del 2003 Novell, com a part de la seva estratègia per a entrar a l'escriptori de GNU/Linux, va comprar Ximian.

- Eazel va ser fundada el 1999 per un grup de persones provinent d'Apple amb la finalitat de fer l'escriptori en GNU/Linux tan fàcil com ho és en Macintosh. L'aplicació en la qual van centrar l'esforç va rebre el nom de Nautilus i havia de ser el gestor de fitxers que jubilés el mític Midnight Commander, desenvolupat per Miguel de Icaza. La seva falta de model de negoci i la crisi de les puntcom –els inversors de risc van retirar el capital necessari perquè l'empresa pogués continuar funcionant– va provocar que el 15 de maig de 2001 Eazel es declarés en fallida i tanqués les portes. Encara va tenir temps per a publicar la versió 1.0 de Nautilus, tot i que la

seva numeració fos més aviat artificial, ja que l'estabilitat que es pressuposa a una versió 1.0 no apareixia per enlloc. Dos anys després de la fallida d'Eazel, es va veure com Nautilus havia evolucionat i s'havia convertit en un complet i manejable gestor de fitxers integrat a GNOME, per la qual cosa la història d'Eazel i Nautilus es pot considerar com un cas paradigmàtic d'un programa que sobreviu a la desaparició de l'empresa que el creava –una cosa gairebé només possible al món del programari lliure.

- Red Hat va crear els Red Had Advanced Development Labs ('laboratoris de desenvolupament avançat de Red Hat'), RHAD, amb la intenció que l'escriptori GNOME guanyés en usabilitat i potència. Per a això va contractar amb mitja dotzena dels *hackers* més importants de GNOME i els va donar llibertat perquè desenvolupessin el que creguessin més convenient. Dels RHAD Labs va sortir ORBit, la implementació de CORBA utilitzada pel projecte GNOME, coneguda com "la més ràpida de l'oest". També és destacable la tasca que es va fer en la nova versió de GTK+ i en el sistema de configuració de GNOME, GConf.

- SUN Microsystems es va involucrar tardanament en el desenvolupament de GNOME, ja que el setembre del 2000 GNOME era ja un producte relativament madur. La intenció de SUN era utilitzar GNOME com el sistema d'escriptori del sistema operatiu Solaris. Per a això, va crear un equip de col·laboració amb GNOME, els mèrits més importants del qual giren entorn de la usabilitat i l'accessibilitat de GNOME. El juny del 2003, SUN va anunciar que distribuiria GNOME 2.2 amb la versió 9 de Solaris.

9.4.4. Estat actual de GNOME

GNOME es troba, a començament del 2007, en la seva versió 2.18. La majoria de les tecnologies clau en què es basa es troben madures, com es pot desprendre del seu número de versió. Així, el *broker* CORBA utilitzat és ORBit2, mentre que l'entorn gràfic i API, GTK+, va acollir els canvis fruit de l'experiència acumulada en les versions anteriors de GNOME. Com a gran novetat apareix la inclusió d'una biblioteca d'accessibilitat, proposada per SUN, que permet que les persones que tinguin problemes d'accessibilitat puguin utilitzar l'entorn GNOME. Una menció especial també requereix Bonobo, el sistema de components de GNOME. Bonobo va marcar una època dins de GNOME, alhora que s'anava desenvolupant el gestor d'informació personal Evolution. Tanmateix, el temps ha demostrat que les expectatives creades per Bonobo van ser massa altes i que la reutilització d'esforç mitjançant l'ús de components no ha estat la que al principi s'esperava.

Nota

La biblioteca ATK és una biblioteca de classes abstractes que permet fer accessibles les aplicacions. Això vol dir que les persones amb alguna discapacitat (cecs, daltònics, gent amb problemes de vista... que no poden manejar el ratolí, el teclat, etc.) poden fer ús de GNOME. L'interès de SUN per l'accessibilitat neix del fet que per a poder oferir els seus productes al Govern dels Estats Units ha de complir una sèrie d'estàndards quant a accessibilitat. S'ho ha pres tan seriosament que en l'equip de desenvolupament de GNOME que treballa en SUN hi ha fins i tot un programador invident. L'arquitectura d'accessibilitat de GNOME va rebre el setembre del 2002 el premi Hellen Keller Achievement Award.

9.4.5. Radiografia de GNOME

Les dades i les xifres que es mostren a la taula 10 ens permetran tancar la presentació de GNOME. Les xifres corresponen a l'estat del CVS de GNOME el 14 d'agost del 2003. Aquell dia hi havia més de nou milions de línies de codi allotjades al repositori CVS que té el projecte GNOME. Encara que una comparació amb KDE seria el més natural, hem d'advertir el lector que les diferències quant a l'organització dels projectes la desaconsellen si es vol fer en igualtat de condicions. Per exemple, el CVS de GNOME inclou GIMP (un programa de creació i manipulació de gràfics), responsable ell sol de més de 660.000 línies de codi, o de la biblioteca GTK+, en la qual se centra el desenvolupament en GNOME, que té al seu torn 330.000 línies. Si a això s'hi afegeix que el repositori CVS de GNOME és més procliu a obrir nous mòduls per a programes (en total en té set-cents) que el de KDE (que en té menys de cent), podrem entendre per què GNOME té un nombre de línies superior al de KDE, malgrat ser un any i mig més jove. El repositori de GNOME acull més de 225.000 fitxers, que han estat afegits i han estat modificats gairebé dos milions de vegades (*vid.* el nombre de *commits* unes quantes files més avall, a la taula).

Nota

Una empresa el desig de la qual fos crear un programari de la mida de GNOME hauria de contractar de mitjana uns dos-cents cinquanta desenvolupadors durant més d'onze anys per a aconseguir un producte de mida similar, segons el model COCOMO utilitzat al llarg de tot aquest capítol. El cost associat pujaria a uns 400 milions de dòlars, una xifra semblant a la que una empresa de telefonia mòbil ja establerta invertirà el 2003 per reforçar la seva capacitat de xarxa o similar a la que desemborsarà una companyia d'automòbils per obrir una planta de producció a Barcelona.

GNOME compta amb uns recursos humans de gairebé mil desenvolupadors amb accés de lectura al sistema de control de versions CVS, entre els quals gairebé una vintena es dediquen a GNOME de manera professional (a temps complet o temps parcial). D'ells, només un 25% s'ha mostrat actiu l'últim any, xifra que puja al 40% si tenim en compte els dos últims anys. El nombre mitjà de *commits* diaris registrats des dels inicis del projecte gairebé arriba al miler. Les eines d'ajuda al desenvolupament que utilitza el projecte GNOME són bàsicament les mateixes que les que s'usen en KDE, per la qual cosa no s'hi incidirà en aquest apartat.

Taula 10. Anàlisi de GNOME

Pàgina web	http://www.gnome.org
Inici del projecte	Setembre 1997
Llicència	GNU GPL i GNU LGPL
Versió analitzada	2.2

Línies de codi font	9.200.000
Nombre de fitxers (codi, documentació, etc.)	228.000
Estimació de cost	$ 400.000.000
Estimació de temps d'execució	11,08 anys (133,02 mesos)
Estimació de nombre mitjà de desenvolupadors	250 aproximadament
Nombre de subprojectes	Més de 700 mòduls en el CVS
Nombre aproximat de desenvolupadors	Gairebé 1.000 amb accés d'escriptura al CVS
Nombre de *commiters* actius l'últim any	Al voltant de 500 (aproximadament el 55% del total)
Nombre de *commiters* actius en els dos últims anys	Uns 700 (el 75% del total)
Nombre de *commits* en el CVS	1.900.000
Nombre mitjà de *commits* (totals) al dia	Uns 900
Eines d'ajuda al desenvolupament	CVS, llistes de correu, lloc web, lloc de notícies, reunions anuals

Mentre que en KDE, C++ és indiscutiblement el llenguatge més utilitzat, en GNOME el lloc més alt correspon a C. En GNOME, igual com en KDE, això es deu al fet que la biblioteca principal està escrita en C, per la qual cosa el llenguatge *nadiu* és aquest, mentre que per a programar amb la resta de llenguatges s'ha d'esperar que apareguin els enllaços. L'enllaç més avançat de GNOME és el que s'inclou en GNOME, que no és cap altre que el de C++, raó per la qual no és sorprenent que el segon llenguatge en la classificació sigui aquest. Perl des de sempre ha tingut una àmplia acceptació dins de la comunitat GNOME i s'ha posat com a exemple del fet que en GNOME es pot programar en multitud de llenguatges. La seva posada en pràctica, però, no ha estat tan àmplia com es podria esperar i supera lleugerament Shell. D'altra banda, l'acceptació de Python i de Lisp en GNOME ha estat bastant gran, com es pot desprendre de la seva relativa importància en aquesta classificació, mentre que Java mai no s'ha arribat a enlairar –probablement a causa d'un enllaç incomplet.

Taula 11. Llenguatges de programació utilitzats en GNOME

Llenguatge de programació	Línies de codi	Percentatge
C	7.918.586	86,10%
C++	576.869	6,27%
Perl	199.448	2,17%
Shell	159.263	1,73%

Llenguatge de programació	Línies de codi	Percentatge
Python	137.380	1,49%
Lisp	88.546	0,96%

9.4.6. Estudis acadèmics sobre GNOME

Els estudis més significatius de GNOME en l'àmbit acadèmic són els dos següents: "Results from software engineering research into open source development projects using public data" [158] i "The evolution of GNOME" [132].

- [158] és un dels primers grans estudis d'enginyeria del programari en el camp del programari lliure. Els autors van aprofitar que les dades del desenvolupament solen ser públicament accessibles per a mesurar esforços i comparar-los amb els models d'estimació de costos, esforços i temps clàssics. Un dels models clàssics amb què es van comparar va ser el que s'ha utilitzat en aquest capítol, el model COCOMO.

- [132] fa un ràpid repàs dels objectius de GNOME i la seva breu història, així com de l'ús que fa el projecte GNOME de les tecnologies.

9.5. Apache

El servidor HTTP Apache és una de les aplicacions estrella del món del programari lliure, ja que és el servidor web de major implantació segons l'enquesta que realitza en temps real (http://news.netcraft.com/archives/2003/08/01/august_2003_web_server_survey.html) [167]. Així, el maig de 1999 el 57% dels servidors web corrien amb Apache, mentre que el maig del 2003 el percentatge havia augmentat fins al 68%. Apache està disponible per a tots els sabors de Unix (BSD, Solaris, GNU/Linux...), Microsoft Windows i altres plataformes minoritàries.

9.5.1. Història d'Apache

El març de 1989, Tim Berners Lee, un científic anglès que treballava al CERN (Suïssa), va proposar una nova forma per a gestionar la ingent quantitat d'informació dels projectes del CERN. Es tractava d'una xarxa de documents hiperenllaçats (hipertext, tal com Ted Nelson l'havia denominat ja el 1965); naixia el WWW. Va caldre esperar fins al novembre de 1990 perquè el primer programari WWW veiés la llum: en un paquet anomenat WorldWideWeb s'incloïa un navegador web d'interfície gràfica i un editor WYSIWYG ("what you see is what you get" és a dir, "el que veu a la pantalla és el que obté com a resultat"). Dos anys després, la llista de servidors WWW comptava amb una trentena d'entrades, entre les quals ja es trobava el NCSA HTTPd.

La verdadera història d'Apache comença quan el març de 1995, Rob McCool abandona l'NCSA. Apache 0.2 veuria la llum el 18 de març de 1995 basat en el servidor NCSA HTTPd 1.3, realitzat pel mateix Rob McCool durant la seva estada a NCSA. Durant aquells primers mesos, Apache era una col·lecció de pedaços aplicats al servidor NCSA, fins que Robert Thau va llançar Shambhala 0.1, una reimplementació gairebé completa que ja incloïa l'API per als mòduls que ha resultat tan reeixida.

Nota

El nom del projecte Apache es deu a la seva filosofia de desenvolupament i d'organització. Igual com la tribu dels Apatxes, els desenvolupadors d'Apache van decidir que la seva forma organitzativa s'havia de fonamentar en els mèrits personals dels desenvolupadors envers la resta de la comunitat Apache. S'ha estès, tanmateix, la llegenda que el nom Apache en realitat es deu al fet que en els primers temps no deixava de ser un servidor NCSA apedaçat, en anglès *a patchy server*.

Caldria esperar al gener de 1996 per a poder gaudir de la primera versió estable d'Apache, l'Apache 1.0, que incloïa la càrrega de mòduls en temps d'execució a tall de proves a més d'altres funcionalitats interessants. Els primers mesos d'aquell any van ser especialment fructífers per al projecte, ja que la versió 1.1, que comptava amb mòduls d'autenticació contra bases de dades (com MySQL), es va publicar amb prou feines dos mesos després. Des de llavors fins a l'actualitat, les fites més grans del projecte han estat la total conformitat amb l'estàndard HTTP 1.1 (inclòs l'abril de 1997 en Apache 1.2), la inclusió de la plataforma Windows NT (que va començar ja el juliol de 1997 amb les versions en proves d'Apache 1.3), la unificació dels arxius de configuració en un de sol (per a la qual caldria esperar a l'aparició d'Apache 1.3.3 a l'octubre de 1998) i el llançament, encara en proves, de la següent generació d'Apache, Apache 2.

Mentrestant, el juny de 1998, IBM va decidir que el motor del seu producte WebSphere fos Apache, en lloc de desenvolupar un servidor HTTP propi. Això es va interpretar com un gran suport per part del gegant blau al projecte Apache i al programari lliure en general, encara que per a facilitar aquest fet calgués canviar lleugerament la llicència Apache original.

9.5.2. Desenvolupament d'Apache

El servidor HTTP Apache és el projecte central dins dels molts que gestiona l'Apache Software Foundation. El disseny modular d'Apache ha permès que existeixin una sèrie de projectes satèl·lit –alguns fins i tot més grans quant a mida que el mateix Apache– entorn d'Apache. D'aquesta forma, el servidor HTTP Apache conté el nucli del sistema amb les funcionalitats bàsiques, mentre que les funcionalitats addicionals les aporten els diferents mòduls. Els mòduls més coneguts són *mod_perl* (un intèrpret del llenguatge de guió Perl encastat al servidor web) i Jakarta (un potent servidor d'aplicacions). En els següents paràgrafs es descriurà només el procés de desenvolupament seguit per al servidor HTTP, sense tenir en compte els altres mòduls, que poden tenir models semblants o no.

El desenvolupament del servidor HTTP Apache es fonamenta en el treball d'un reduït grup de desenvolupadors denominat Apache Group. L'Apache Group el constitueixen aquells desenvolupadors que han col·laborat durant un període de temps prolongat, generalment més de sis mesos. El desenvolupador, després de ser nomenat per un membre de l'Apache Group per a formar-ne part, és votat entre tots els membres de l'Apache Group. Al començament, l'Apache Group constava de vuit desenvolupadors, després de dotze, i en l'actualitat compta amb vint-i-cinc persones.

Sobre l'Apache Group recau la responsabilitat de l'evolució del servidor web i, per tant, de les decisions puntuals de desenvolupament en cada moment. Cal diferenciar l'Apache Group del nucli de desenvolupadors (*core group*) actiu en cada moment. El caràcter voluntari de la majoria dels desenvolupadors fa que sigui improbable que tots els que componen l'Apache Group puguin estar actius tot el temps, per la qual cosa el *core* es defineix com aquells que en un espai de temps poden ocupar-se de les tasques en Apache. En línies generals, les decisions que han de prendre els desenvolupadors pertanyents al nucli es limiten a la votació de la inclusió de codi –encara que això es reservi en realitat només per a grans canvis– i a qüestions de disseny. D'altra banda, en general solen tenir dret d'escriptura en el repositori CVS, de manera que serveixen com a porta d'entrada del codi i n'asseguren la correctesa i la qualitat.

9.5.3. Radiografia d'Apache

Les xifres que s'exposen a continuació corresponen a la versió del servidor HTTP Apache tal com es podia descarregar del servidor CVS del projecte Apache el 18 d'abril de 2003. No s'han tingut en compte cap dels nombrosos mòduls amb què compta el projecte Apache. Com es pot observar, Apache és un projecte relativament petit en comparació amb els altres casos d'estudi considerats en aquest capítol. Encara que ja s'ha comentat anteriorment, és important posar èmfasi en la modularitat d'Apache, que permet precisament això: que el nucli sigui petit i manejable. El repositori CVS del projecte Apache, que conté el nucli del servidor web i molts mòduls addicionals, acull en total més de quatre milions de línies de codi font, una xifra lleugerament inferior a projectes com KDE i GNOME.

La versió 1.3 d'Apache comptava amb poc més de 85.000 línies de codi font, una xifra que segons el model COCOMO requeriria un esforç de desenvolupament de vint desenvolupadors a temps complet de mitjana durant un any i mig. El cost total del projecte s'aproxima llavors als 4 milions de dòlars. En l'elaboració del servidor web d'Apache treballen fins a seixanta *commiters* diferents, mentre que el nombre de desenvolupadors que han aportat es calcula que són uns quatre-cents.

Taula 12. Anàlisi d'Apache

Pàgina web	http://www.apache.org
Inici del projecte	1995
Llicència	Apache Free Software License
Versió analitzada	2.2.4
Línies de codi font	225.065
Nombre de fitxers	2.807
Estimació de cost	$ 7.971.958
Estimació de temps d'execució	2,52 anys (30,27 mesos)
Estimació de nombre mitjà de desenvolupadors	23,4
Nombre aproximat de desenvolupadors	60 *commiters* (400 desenvolupadors)
Eines d'ajuda al desenvolupament	CVS, llistes de correu, sistema de notificació d'errors

Apache 1.3 està escrit gairebé íntegrament en llenguatge C, i la presència d'altres llenguatges de programació és escassa, sobretot si tenim en compte que la gran majoria de les línies escrites en el segon llenguatge, Shell, corresponen a fitxers de configuració i d'ajuda a la compilació.

Taula 13. Llenguatges de programació utilitzats en Apache

Llenguatge de programació	Línies de codi	Percentatge
C	208.866	92,8%
Shell	12.796	5,69%
Perl	1.649	0,73%
Awk	874	0,39%

9.6. Mozilla

El projecte Mozilla treballa un conjunt d'aplicacions integrat per a Internet, lliures i multiplataforma, els productes més destacats del qual són el navegador web Mozilla Firefox i el client de correu Mozilla Thunderbird. Aquest conjunt està pensat també com a plataforma de desenvolupament d'altres aplicacions, de manera que són molts els navegadors que utilitzen Gecko, el motor d'HTML de Mozilla (com Galeon).

El projecte està gestionat per la Fundació Mozilla, una organització sense ànim de lucre dedicada a la creació de programari lliure, amb la missió específica de "mantenir l'elecció i la innovació a Internet". És per això que els productes de Mozilla tenen en compte tres principis bàsics: ser programari lliure, respectar els estàndards i ser portables a diferents plataformes.

9.6.1. Història de Mozilla

La història de Mozilla és llarga i enrevessada, però al mateix temps molt interessant, ja que permet seguir la història del mateix WWW. I és que si tracem les persones i institucions que han estat involucrades en el desenvolupament de Mozilla, arribarem al punt de partida de la web, amb el llançament del primer navegador web complet.

Igual com amb l'antecessor d'Apache, va ser en l'NCSA on també "va néixer" el primer navegador web complet el 1993, Mosaic. Molts dels membres de l'equip de desenvolupament, amb Marc Andreessen i Jim Clark al capdavant, van crear una petita empresa per escriure, començant des de zero (ja que hi havia problemes amb els drets d'autor del codi de Mosaic i el disseny tècnic del programa tenia les seves limitacions –*vid. Speeding the Net: the inside story of Netscape and how it challenged Microsoft* [189]–), el que més tard seria el navegador Netscape Communicator, que va ser líder indiscutible del mercat dels navegadors web fins a l'arribada de Microsoft Internet Explorer. A més de la innovació purament tecnològica que va suposar el navegador Netscape, Netscape Inc. també va ser innovadora en la forma d'aconseguir copar el mercat. Contra tot el sentit comú de l'època, la seva aplicació estrella, el navegador WWW, es podia obtenir (i distribuir amb certes limitacions) de manera gratuïta. Aquest fet, fins llavors insòlit dins del món empresarial, va causar certa sorpresa, però va demostrar a la llarga ser un encert per a l'estratègia de Netscape Inc., que només un gegant com Microsoft va saber tallar amb una tàctica més agressiva (i probablement lesiva amb la lliure competència).

Cap a 1997, la quota de mercat de Netscape havia caigut en picat a causa de la implantació de Microsoft Explorer, i des de Netscape Inc. s'estudiaven noves fórmules per a tornar a guanyar-lo. Un informe tècnic de l'enginyer Frank Hecker ("Setting up shop: the business of open-source software", 1998) [142] va proposar que la millor solució al problema era alliberar el codi font del navegador i beneficiar-se dels efectes de la comunitat del programari lliure tal com Eric Raymond descrivia a "La catedral y el bazar". El gener de 1998 Netscape Inc. va anunciar oficialment que alliberaria el codi font del seu navegador, fet que va marcar una fita de gran importància dins de la curta història del món del programari lliure: una empresa publicaria tot el codi font d'una aplicació fins llavors privativa sota una llicència de programari lliure. La data indicada per al llançament era el 31 de març de 1998.

En els dos mesos que van de gener a març, l'activitat a Netscape perquè tot estigués a punt va ser frenètica. La llista de tasques era enorme i complexa ("Freeing the source: the story of Mozilla", 1999) [134]. En el pla tècnic, calia contactar amb les empreses que havien realitzat mòduls per demanar el seu consentiment en canvi de llicència; en cas de negativa, el mòdul havia de ser eliminat. A més, totes les parts escrites en Java havien de ser reimplementades, ja que es va considerar que Java no era lliure. Es va decidir d'anomenar el projecte lliure Mozilla, tal com els mateixos desenvolupadors de Netscape anomenaven el component principal de Netscape, i es va adquirir el domini Mozilla.org per a construir una comunitat de desenvolupadors i col·laboradors entorn d'aquest lloc web. Al final del procés, es van alliberar més d'un milió i mig de línies de codi font.

Nota

El nom de Mozilla és un joc de paraules amb un toc humorístic de l'equip de desenvolupament a Netscape Inc. Mozilla és el producte de l'adaptació de Godzilla, un monstre que causava pànic a les pel·lícules de terror japoneses des de la dècada dels cinquanta, perquè sonés com *Mosaic Killer*, ja que es pretenia que Mosaic quedés obsolet gràcies a aquest nou navegador, que presentava tecnologies molt més avançades.

D'altra banda, hi havia el pla legal. Les llicències lliures existents en aquell moment no convencien els executius de Netscape, que veia que no "congeniaven" amb el caràcter comercial d'una companyia. Netscape volia una llicència més *flexible* que permetés arribar a acords amb tercers per incloure-hi el seu codi indiferentment de la seva llicència o que altres desenvolupadors comercials hi poguessin contribuir i alhora defensar els seus interessos econòmics com volguessin. I encara que al principi no s'havia previst crear una nova llicència, es va arribar a la conclusió que era l'única forma d'aconseguir el que desitjaven. Així és com es va forjar la Netscape Public License (NPL), una llicència que es basava en els principis bàsics de les llicències de programari lliure, però que concedia uns drets addicionals a Netscape Inc. –cosa que la feia també una llicència *no lliure* sota el prisma de la Free Software Foundation. Quan es va publicar l'esborrany de l'NPL per a fer-ne la discussió pública, van ploure les crítiques sobre la clàusula de drets addicionals per a Netscape. Netscape Inc. va reaccionar ràpidament davant d'aquests comentaris i va crear una llicència addicional, la Mozilla Public License (MPL), que era idèntica a l'NPL però sense que Netscape hi tingués cap dret addicional.

La decisió final va ser alliberar el codi de Netscape sota la llicència NPL, que atorgava drets addicionals a Netscape, i que el codi nou que fos inclòs estigués sota l'MPL (o compatible). Les correccions al codi original (llicenciat amb l'NPL) havien de ser també sota aquesta llicència.

Nota

En l'actualitat, Mozilla accepta contribucions sota la llicència pròpia MPL, la GPL i l'LGPL. El canvi de llicència no va ser gens fàcil, ja que va caldre buscar tots els que havien contribuït amb codi alguna vegada perquè donessin el seu vistiplau al canvi d'NPL/MPL a MPL/GPL/LGPL. Per tal de poder rellicenciar tot el codi, es va crear una pàgina web que contenia una llista de tres-cents desenvolupadors "perduts" ("Have you seen these hackers?") [38]. El maig del 2007, encara es continua buscant dos d'aquests desenvolupadors.

El desenvolupament amb el codi original de Netscape Communicator va ser, sens cap dubte, més complicat del que inicialment s'esperava. Les condicions de partida ja eren dolentes per si mateixos, perquè el que va ser alliberat a vegades era incomplet (s'havien tret tots els mòduls de tercers que no havien donat el seu vistiplau a l'alliberament) i funcionava amb prou feines. Per si això no fos prou, al problema tècnic de pretendre que Mozilla funcionés sobre una gran quantitat de sistemes operatius i plataformes, calia afegir-hi els vicis adquirits de Netscape Inc., amb cicles d'alliberament llargs i ineficients per al món d'Internet i que no distingia entre els seus interessos i els d'una comunitat entorn de Mozilla. Tot això va comportar que, just un any després, un dels programadors més actius abans i després de l'alliberament, Jamie Zawinsky, decidís tirar la tovallola en una amarga carta ("Resignation and postmortem", 1999) [237] en la qual mostrava la seva desesperació i la seva desolació.

El 15 de juliol de 2003 Netscape Inc. (propietat d'America On Line) va anunciar que deixava de desenvolupar el navegador Netscape i, per tant, la tutela activa del projecte Mozilla. Com a liquidació "va aprovar la creació de la Fundació Mozilla, a la qual va donar suport amb una aportació de dos milions de dòlars. Així mateix, tot el codi que es trobava sota l'NPL (la llicència pública de Netscape) es va donar a la Fundació i es va redistribuir amb les llicències promulgades ja anteriorment pel projecte Mozilla: MPL, LGPL i GPL.

El 10 de març de 2005 la Fundació Mozilla va anunciar que no es publicarien més versions oficials de Mozilla SuiteApplication, que era substituïda per Mozilla SeaMonkey, que inclou navegador web, client de correu electrònic, llibreta de contactes, editor de pàgines i client d'IRC. D'altra banda, el projecte Mozilla allotja diverses aplicacions independents, entre les quals es poden destacar Mozilla Firefox (navegador web), sens dubte la més coneguda, Mozilla Thunderbird (client de correu i lector de notícies), Mozilla Sunbird (calendari), Mozilla Nvu (editor web), Camino (navegador per a Mac OS X) i Bugzilla (eina basada en web per al seguiment d'informes d'error).

Amb el temps, i malgrat molts dubtes i llargs períodes en què a molts els semblava que s'acostava al fracàs, el projecte sembla gaudir de bona salut. La versatilitat i la portabilitat de les seves aplicacions han fet que, tot i estar sovint molt necessitats de recursos d'execució, es considerin (en general, però molt especialment Firefox) la parella d'OpenOffice.org a l'escriptori de l'usuari final.

9.6.2. Radiografia de Mozilla

Les mesures que es presentaran en aquest apartat corresponen a l'estudi de Firefox, l'aplicació més coneguda del projecte. Segons les estimacions del model COCOMO, una companyia que volgués crear un programari de tals dimensions hauria d'invertir uns 111 milions de dòlars per obtenir-lo. El temps que hauria d'esperar se situaria entorn dels set anys i el nombre mitjà de programadors a temps complet que hauria d'emprar serien cent vint aproximadament.

Taula 14. Estat actual de Mozilla Firefox

Pàgina web	www.mozilla-europe.org/es/products/firefox/
Inici del projecte	2002
Llicència	MPL/LGPL/GPL
Versió	2.0
Línies de codi font	2.768.223
Estimació de cost	$ 111.161.078
Estimació de temps d'execució	6,87 anys (82,39 mesos)
Estimació de nombre mitjà de desenvolupadors	120
Nombre aproximat de desenvolupadors	50 *commiters*
Eines d'ajuda al desenvolupament	CVS, llistes de correu, IRC, Bugzilla...

Quant als llenguatges de programació, C++ i C són, per aquest ordre, els més utilitzats. La presència de Perl es deu en gran part al fet que les eines d'ajuda al desenvolupament dutes a terme pel projecte Mozilla, com ara BugZilla o Tinderbox, s'han fet en aquest llenguatge. El que sí que resulta una mica sorprenent és l'alt nombre de línies de codi en llenguatge assemblador en una aplicació d'usuari final. La inspecció del codi al repositori ha mostrat que, efectivament, hi ha bastants fitxers codificats en llenguatge assemblador.

Taula 15. Llenguatges de programació utilitzats en Mozilla Firefox

Llenguatge de programació	Línies de codi	Percentatge
C++	1.777.764	64,22%
C	896.551	32,39%
Assemblador	34.831	1,26%
Perl	26.768	0,97%
Shell	16.278	0,59%

Llenguatge de programació	Línies de codi	Percentatge
C#	6.232	0,23%
Java	5.352	0,19%
Python	3.077	0,11%
Pascal	459	0,02%

9.7. OpenOffice.org

OpenOffice.org és una de les aplicacions estrella del panorama actual del programari lliure. Es tracta d'un paquet ofimàtic (*suite*) multiplataforma que inclou les aplicacions clau en un entorn d'escriptori d'oficina, com ara processador de text (Writer), full de càlcul (Calc), gestor de presentacions (Impress), programa de dibuix (Draw), editor de fórmules matemàtiques (Math) i finalment, editor de llenguatge HTML (inclòs en Writer). La interfície que ofereix OpenOffice.org és homogènia i intuïtiva, similar en aspecte i funcionalitats a altres paquets ofimàtics, en especial a més arrelat en l'actualitat, Microsoft Office.

Escrit en C++, OpenOffice.org inclou l'API de Java i té el seu propi sistema de components encastables, que permet incloure, per exemple, taules del full de càlcul en el processador de textos d'una manera senzilla i intuïtiva. Entre els seus avantatges, podem dir que maneja una gran quantitat de formats de fitxer, inclosos els de Microsoft Office. Els seus formats de fitxer nadius, a diferència dels del paquet ofimàtic de Microsoft, estan basats en XML, per la qual cosa es mostra com una clara aposta per la versatilitat, la facilitat de transformació i la transparència. En l'actualitat, OpenOffice.org està traduït a més de vint-i-cinc llengües i s'executa en Solaris (el seu sistema nadiu), GNU/Linux i Windows. En un futur no gaire llunyà s'esperen versions per a FreeBSD, IRIX i Mac OS X.

OpenOffice.org va adoptar el seu nom definitiu (OpenOffice –tal com el coneix tothom– més l'afegitó *.org*) després d'un litigi en què va ser demandat per usurpació de nom de marca per una altra empresa.

9.7.1. Història d'OpenOffice.org

A mitjan dècada dels vuitanta, es va fundar a la República Federal d'Alemanya l'empresa StarDivision, que va tenir com a objectiu principal la creació d'un paquet ofimàtic: StarOffice. L'estiu de 1999, SUN Microsystems –amb la clara intenció d'arrabassar-li un mercat conquerit ja en aquell temps a Microsoft– va decidir adquirir l'empresa StarDivision i fer una forta aposta per StarOffice. Així, el juny del 2000 va llançar la versió 5.2 de StarOffice, que podia ser descarregat gratuïtament a la Xarxa.

Tanmateix, l'èxit de StarOffice va ser limitat, ja que el mercat estava fortament dominat pel paquet ofimàtic de Microsoft. SUN va decidir canviar la seva estratègia i, igual com Netscape amb el projecte Mozilla, va decidir aprofitar els avantatges del programari lliure per a guanyar en importància i implantació. D'aquesta forma, les futures versions de StarOffice (producte propietari de SUN) es crearien utilitzant OpenOffice.org (producte lliure) com a font, respectant les interfícies de programació (API) i els formats de fitxer, i servint com a implementació de referència.

9.7.2. Organització d'OpenOffice.org

OpenOffice.org pretén tenir una estructura decisòria en la qual tots els membres de la comunitat se sentin partícips. Per això, s'ha ideat un sistema perquè la presa de decisions tingui el major consens possible. El projecte OpenOffice.org es divideix en una sèrie de subprojectes que compten amb uns membres del projecte, els col·laboradors, i un únic líder. Sens dubte, els membres d'un projecte poden participar en més d'un projecte, igual com el líder. Tanmateix, no es pot liderar més d'un projecte alhora. Els projectes es divideixen en tres categories:

- Projectes acceptats. Poden ser tant de caràcter tècnic com no tècnic. Els líders de cada projecte acceptat tenen un vot a l'hora de la presa de decisions globals.

- Projectes *native-lang*. Són tots els projectes d'internacionalització i localització d'OpenOffice.org. En l'actualitat, com s'ha comentat anteriorment, hi ha més de vint-i-cinc equips que treballen per traduir les aplicacions d'OpenOffice.org a les diferents llengües i convencions. En conjunt, *native-lang* té un únic vot per a la presa de decisions globals.

- Projectes a la incubadora. Es tracta de projectes patrocinats per la comunitat (generalment experimentals o petits). Poden passar a ser acceptats després d'un període de sis mesos. D'aquesta forma, la comunitat OpenOffice.org pot garantir que els projectes acceptats tenen el suport d'un interès real, ja que la *mortalitat* de projectes nous al món del programari lliure és molt gran. En total, els projectes a la incubadora compten amb un vot a la presa de decisions.

9.7.3. Radiografia d'OpenOffice.org

El paquet ofimàtic OpenOffice.org està compost per prop de quatre milions de línies de codi font distribuïdes al llarg de quaranta-cinc mil fitxers.

El model COCOMO estima l'esforç necessari per a realitzar un "clon" d'OpenOffice.org en cent vuitanta programadors que treballin durant gairebé vuit anys a temps complet. El cost de desenvolupament pujaria, segons les estimacions de COCOMO, a uns 215 milions de dòlars.

Els resultats que es comenten en aquest apartat van ser obtinguts de l'estudi del codi font de la versió estable 2.1 d'OpenOffice.org.

Taula 16. Estat actual d'OpenOffice.org

Pàgina web	http://www.openoffice.org
Inici del projecte	Juny del 2000 (primera versió lliure)
Llicència	LGPL i SISSL
Versió	2.1
Línies de codi font	5.197.090
Estimació de cost	$ 215.372.314
Estimació de temps d'execució	8,83 anys (105,93 mesos)
Estimació de nombre mitjà de desenvolupadors	180
Nombre aproximat de desenvolupadors	200 *commiters*
Eines d'ajuda al desenvolupament	CVS, llistes de correu

Quant als llenguatges de programació utilitzats en OpenOffice.org, el primer lloc l'ocupa C++. És interessant observar com l'adquisició per part de SUN va implicar la integració de molt codi Java al paquet ofimàtic, que va superar fins i tot C.

Taula 17. Llenguatges de programació utilitzats en OpenOffice.org

Llenguatge de programació	Línies de codi	Percentatge
C++	4.615.623	88,81%
Java	385.075	7,41%
C	105.691	2,03%
Perl	54.063	1,04%
Shell	12.732	0,24%
Yacc	6.828	0,13%
C#	6.594	0,13%

9.8. Red Hat Linux

Red Hat Linux va ser una de les primeres distribucions comercials de GNU/Linux. Avui dia és probablement una de les més conegudes, i segurament la que es pot considerar la "canònica" d'entre les distribucions comercials. El treball dels distribuïdors està bàsicament relacionat amb tasques d'integració, i no tant amb el desenvolupament de programari. Sens dubte, tant Red Hat com altres distribucions poden tenir desenvolupadors entre els seus empleats, però la seva tasca és secundària per als objectius d'una distribució. En general, s'assumeix que el treball que realitzen les distribucions és simplement prendre els paquets font (generalment els arxius que publiquen els mateixos desenvolupadors) i empaquetar-los de manera que compleixin certs criteris (tant tècnics com organitzatius). El producte d'aquest procés és una distribució: un seguit de paquets convenientment organitzats que possibiliten a l'usuari que en faci la instal·lació, la desinstal·lació i l'actualització.

Les distribucions també són responsables de la qualitat del producte acabat, un aspecte molt important si tenim en compte que moltes de les aplicacions que inclouen han estat elaborades per voluntaris en el seu temps lliure. Aspectes de seguretat i estabilitat són, per tant, de capital importància per a una distribució.

9.8.1. Història de Red Hat

Red Hat Software Inc. va ser fundada el 1994 per Bob Young i Marc Ewing. El seu principal objectiu era compilar i comercialitzar una distribució GNU/Linux que va anomenar (i encara es continua anomenant) Red Hat Linux [236]. Bàsicament, es tractava d'una versió empaquetada del que hi havia a la Xarxa en aquells temps, que incloïa documentació i suport. Durant l'estiu de 1995, la versió 1.0 d'aquesta distribució va veure la llum. Uns mesos més tard, a la tardor, es va publicar la versió 2.0, que incloïa la tecnologia RPM (*RPM package mànager*, 'gestor de paquets RPM'). El sistema de paquets RPM s'ha convertit en un estàndard *de facto* per als paquets de sistemes GNU/Linux. El 1998, Red Hat va arribar al gran públic amb la versió 5.2. Per a una història completa dels noms de les diferents versions de Red Hat, es pot consultar "The truth behind Red Hat names" [201].

Nota

Des de la versió 1.1 del Linux Standar Base (una especificació la meta de la qual és aconseguir compatibilitat binària entre les distribucions GNU/Linux de què s'encarrega el Free Standards Group), RPM ha estat elegit com el sistema de paquets estàndard. El projecte Debian continua amb el seu format de paquet propi, així com moltes de les distribucions dependents del sistema de paquets Debian, i s'ajusten al format estandarditzat mitjançant una eina de conversió que es diu *alien*.

Abans de l'existència del sistema de gestió de paquets RPM, gairebé totes les distribucions de GNU/Linux oferien la possibilitat d'instal·lar el programari mitjançant un procediment dirigit per menús, però fer modificacions a una

instal·lació ja feta, especialment agregar paquets de programari nous després de la instal·lació, no era una tasca fàcil. RPM va permetre donar aquest pas més enllà de l'estat de l'art en proveir els usuaris de la possibilitat de gestionar els seus paquets ("Maximum RPM. Taking the Red Hat package manager to the limit", 1998) [83], fet que permetria esborrar, instal·lar o actualitzar qualsevol paquet programari existent en la distribució de manera molt més senzilla. El sistema de paquets RPM continua essent el sistema de paquets més utilitzat entre les diferents distribucions de GNU/Linux. Les estadístiques de Linux Distributions, "Facts and figures", 2003 [92], un lloc web que conté informació qualitativa i quantitativa sobre un gran nombre de distribucions, mostren que el maig del 2003 una gran majoria de les cent divuit distribucions computades utilitzen el gestor RPM, en total seixanta-cinc (que és aproximadament un 55% del total). En comparació, el sistema de paquets de Debian (conegut com a *deb*) l'utilitzen únicament setze distribucions (un 14% del total).

Tanmateix, Red Hat Inc. no solament és coneguda per la seva distribució de programari basada en Linux. L'agost de 1999, Red Hat va sortir a borsa i les seves accions van obtenir el vuitè guany de primer dia més gran en tota la història de Wall Street. Quatre anys més tard, el valor de les accions de Red Hat era al voltant d'una centèsima part del màxim valor que van arribar a assolir abans de la crisi de les puntcom. Tot i així, els seus començaments reeixits al mercat de valors van servir perquè Red Hat fos portada en diaris i revistes no directament relacionats amb temes informàtics. En qualsevol cas, sembla que Red Hat ha sabut superar els problemes d'altres companyies del món dels negocis entorn del programari lliure i va anunciar números negres per primera vegada en la seva història en l'últim quart de l'any 2002.

Un altre dels fets històrics més importants de Red Hat va ser l'adquisició el novembre de 1999 de Cygnus Solutions, una empresa fundada una dècada abans i que ja havia demostrat com amb una estratègia integral basada en programari lliure es poden guanyar diners ("Future of Cygnus Solutions. An entrepreneur's account") [216]. Cygnus va escollir l'exigent mercat dels compiladors per a fer-se un lloc. La seva estratègia comercial es basava en el desenvolupament i l'adaptació de les eines de desenvolupament de programari GNU (bàsicament GCC i GDB) a petició del client.

El setembre del 2003, Red Hat va decidir concentrar els seus esforços de desenvolupament en la versió corporativa de la seva distribució i va delegar la versió comuna a Fedora Core, un projecte obert independent de Red Hat.

El juny del 2006 Red Hat va adquirir la companyia JBoss Inc. i es va convertir en la responsable del desenvolupament del servidor més important d'aplicacions J2EE de programari lliure.

9.8.2. Estat actual de Red Hat

En l'actualitat, els productes estrella de Red Hat Inc. són Fedora Core i Red Hat Network, un servei d'actualització de programari per mitjà de la Xarxa. Aquest tipus de serveis estan més aviat orientats a l'usuari final i no tant a l'entorn empresarial, però serveixen a Red Hat com a bon reclam i per a assegurar la seva estratègia de marca.

La "verdadera" estratègia comercial de Red Hat es troba en els seus productes dirigits al món empresarial. Aquest tipus de productes són molt menys coneguts, però suposen una gran part de la facturació de Red Hat, molt superior a la que percep pels seus productes estrella més populars en el sentit literal.

Red Hat compta amb una distribució orientada a l'empresa, integrada entorn d'un servidor d'aplicacions i anomenada Red Hat Entreprise Linux AS. Amb l'adquisició d'aquest programari, el client té dret a suport. El servei anàleg a Red Hat Network per a usuaris comercials és Red Hat Enterprise Network, que inclou la gestió del sistema i la possibilitat d'actualitzacions. D'altra banda, Red Hat ofereix també serveis de consultoria informàtica i un programa de certificació similar al que hi ha al món Windows ofert per Microsoft.

9.8.3. Radiografia de Red Hat

Red Hat ha superat recentment la barrera dels cinquanta milions de línies de codi, que la converteixen en una de les majors distribucions de programari que han arribat a existir –fins a superar, com es veurà més endavant en aquest capítol, la mida de sistemes operatius propietaris. La versió 8.1 de Red Hat estava constituïda per 792 paquets, per la qual cosa podem assumir que també haurà franquejat el llistó dels vuit-cents paquets en la seva última versió, tenint en compte que aquest nombre se sol incrementar en cada versió.

Igual com en els casos anteriors, s'ha aplicat el model COCOMO per a estimar la inversió i l'esforç que caldria en la generació d'un programari de mida idèntica. Tanmateix, per al cas de Red Hat s'ha considerat que es tracta d'un producte realitzat a partir d'una sèrie d'aplicacions independents. Per això, s'ha fet una estimació mitjançant COCOMO independent per a cada un dels paquets de Red Hat, per a després sumar el cost i el personal total necessari. En el cas del temps d'execució òptim per a Red Hat, s'ha considerat el del paquet més gran, ja que idealment, com que tots els paquets són independents, podrien realitzar-se de manera paral·lela en el temps. Per això el temps d'execució òptim per a Red Hat és semblant al dels projectes que s'han presentat anteriorment en aquest capítol.

Segons COCOMO, caldrien al voltant de set anys i mig i un equip de programadors compost de mitjana per mil vuit-cents desenvolupadors per a realitzar la distribució Red Hat Linux 8.1 des de zero. El cost de desenvolupament final ascendiria a uns 1.800 milions de dòlars.

Nota

Mil vuit-cents milions de dòlars és el pressupost que el Ministeri de Defensa espanyol destinarà a renovar la seva flota d'helicòpters. De tot l'import la meitat es destinarà a l'adquisició de vint-i-quatre helicòpters, de manera que el preu de Red Hat seria l'equivalent a quaranta-vuit helicòpters de combat. Així mateix, 1.800 milions de dòlars és la xifra que va obtenir de beneficis a escala mundial la pel·lícula *Titanic*.

Taula 18. Estat de Red Hat Linux

Pàgina web	http://www.redhat.com
Inici del projecte	1993
Llicència	
Versió	9.0
Línies de codi font	Més de 50.000.000
Nombre de paquets	792
Estimació de cost	$ 1.800.000.000
Estimació de temps d'execució	7,35 anys (88,25 mesos)
Estimació de nombre mitjà de desenvolupadors	1.800
Nombre aproximat de desenvolupadors	Empleats de Red Hat (generalment només integració)
Eines d'ajuda al desenvolupament	CVS, llistes de correu

A causa de la presència d'un ampli nombre de paquets, la classificació de llenguatges en Red Hat té major diversitat que les que hem vist en les aplicacions més importants de programari lliure. En termes generals, es percep la gran importància de C, amb més d'un seixanta per cent de les línies de codi. En segon lloc, amb més de deu milions de línies de codi, es troba C++, seguit de lluny per Shell. És interessant observar que a Perl s'hi uneixen després Lisp (majoritàriament a causa del seu ús a Emacs), el codi en llenguatge assemblador (una quarta part del qual correspon al que ve amb Linux) i l'ús d'un llenguatge en franc retrocés, com és Fortran.

Taula 19. Llenguatges de programació utilitzats en Red Hat

Llenguatge de programació	Línies de codi	Percentatge
C	30.993.778	62,13%
C++	10.216.270	20,48%

Llenguatge de programació	Línies de codi	Percentatge
Shell	3.251.493	6,52%
Perl	1.106.082	2,22%
Lisp	958.037	1,92%
Assemblador	641.350	1,29%
Fortran	532.629	1,07%

9.9. Debian GNU/Linux

Debian és un sistema operatiu lliure que en l'actualitat utilitza el nucli de Linux per a dur a terme la seva distribució (encara que s'espera que hi hagi distribucions Debian basades en altres nuclis, com és el cas de "the HURD", en el futur). Actualment, està disponible per a diverses arquitectures diferents, que inclouen Intel x86, ARM, Motorola, 680 x 0, PowerPC, Alpha i SPARC.

Debian no és només la major distribució GNU/Linux de l'actualitat, sinó també una de les més estables, i gaudeix de diversos premis relacionats amb la preferència dels usuaris. Encara que la seva base d'usuaris sigui difícil d'estimar, ja que el projecte Debian no ven CD o altres mitjans amb el seu programari, i el programari que conté pot ser redistribuït per qualsevol que ho vulgui, no aniríem gaire errats si suposem que es tracta d'una distribució important dins del mercat de GNU/Linux.

En Debian hi ha una categorització segons la llicència i els requisits de distribució dels paquets. El nucli de la distribució Debian (la secció anomenada *main* aglutina una gran varietat de paquets) està format només per programari lliure d'acord amb les directrius de Debian (DFSG Debian Free Software Guidelines) [104]. Està disponible a Internet per a ser descarregat i molts redistribuïdors el venen en CD o altres mitjans.

Les distribucions de Debian són creades per prop d'un miler de voluntaris (generalment professionals de la informàtica). La tasca d'aquests voluntaris consisteix a prendre els programes font –en la majoria dels casos dels seus autors originals–, configurar-los, compilar-los i empaquetar-los de manera que un usuari típic d'una distribució Debian només hagi de seleccionar el paquet perquè el sistema l'afegeixi sense més problemes. Això que a cop d'ull pot semblar simple, es torna complex quan s'introdueixen factors com les dependències entre els diferents paquets (el paquet A necessita, per a poder funcionar, el paquet B) i les diferents versions de tots aquests paquets.

La tasca dels integrants del projecte Debian és la mateixa que la que es duu a terme en qualsevol altra distribució: la integració de programari per al correcte funcionament conjunt. A més del treball d'adaptació i d'empaquetament, els desenvolupadors de Debian s'encarreguen de mantenir una infraestructura de

serveis basats en Internet (lloc web, arxius en línia, sistema de gestió d'errors, llistes de correu d'ajuda, suport i desenvolupament, etc.), diversos projectes de traducció i internacionalització, el desenvolupament de diverses eines específiques de Debian i, en general, qualsevol element que faci possible la distribució Debian.

A part de la seva naturalesa voluntària, el projecte Debian té una característica que el fa especialment singular: el contracte social de Debian (http://www.debian.org/social_contract.html) [106]. Aquest document conté no solament els objectius principals del projecte Debian, sinó també els mitjans que s'utilitzaran per a dur-los a terme.

Debian també és coneguda per tenir una política de paquets i versions molt estricta a fi d'aconseguir una major qualitat del producte ("Debian policy manual") [105]. Així, en tot moment hi ha tres *sabors* diferents de Debian: una versió estable, una versió inestable i una altra en proves. Tal com indica el nom, la versió estable és la recomanada per a sistemes i persones no aptes a sobresalts. El seu programari ha de passar un període de congelació en què només es corregeixen errors. La norma és que en la versió estable de Debian no hi ha d'haver cap error crític conegut. En canvi, aquesta versió estable no sol tenir les últimes versions del programari (el més nou).

Per als qui vulguin tenir una versió amb el programari més actual hi ha dues versions més de Debian coetànies a l'estable. La versió inestable inclou paquets en via d'estabilització, mentre que la versió en proves, com el nom indica, és la més procliu a fallar i conté les últimes novetats en programari.

En el moment que es va realitzar un primer estudi, la versió estable de Debian era Debian 3.0 (també coneguda com a Woody), la inestable rebia el sobrenom de Sid i la que es trobava en proves era Sarge. Però en el passat, Woody va passar també per una etapa inestable i, abans d'això, per una altra en proves. Això és important, perquè el que considerarem en aquest article són les diferents versions estables de Debian, des que es va publicar la versió 2.0, cap al 1998. Així, tenim Debian 2.0 (àlies Hamm), Debian 2.1 (Slink), Debian 2.2 (Potato) i, finalment, Debian 3.0 (Woody).

Nota

Els sobrenoms de les versions de Debian corresponen als protagonistes de la pel·lícula de dibuixos animats *Toy story*, una tradició que es va implantar mig en broma quan es va publicar la versió 2.0 i Bruce Perens, llavors líder del projecte i després fundador de l'Open Source Initiative i del terme *open source*, treballava per a l'empresa que s'encarregava de realitzar aquesta pel·lícula. Es poden trobar més detalls sobre la història de Debian i la distribució Debian en general a "A brief history of Debian" [122].

9.9.1. Radiografia de Debian

Debian GNU/Linux és probablement la major compilació de programari lliure que funciona de forma coordinada, i sens dubte, un dels productes programari més grans mai no construïts. La versió 4.0, alliberada l'abril del 2007 (denominada Etch), consta de més de deu mil cent paquets font, que sumen més de 288 milions de línies de codi.

El nombre de línies de codi en Debian 3.0 és de 105 milions. Segons el model COCOMO caldria desemborsar una quantitat aproximada de 3.600 milions de dòlars per a obtenir un programari com el que està empaquetat amb aquesta distribució. Igual com en el cas de Red Hat, s'ha computat separadament l'esforç necessari per a cada paquet i després s'han sumat totes les quantitats. Per la mateixa raó, el temps de desenvolupament de Debian és de només set anys, ja que es considera que cada paquet pot realitzar-se paral·lelament als altres en el temps. Això sí, de mitjana caldria mobilitzar prop de quatre mil desenvolupadors durant aquells set anys per a aconseguir-ho.

Nota

Tres mil sis-cents milions de dòlars és el pressupost que té assignat el VI Programa marc de la Comissió Europea per a activitats d'investigació i desenvolupament relacionades amb la societat de la informació. També és la xifra que Telefónica preveu invertir a Alemanya per implantar UMTS.

Taula 20. Estat de Debian

Pàgina web	http://www.debian.org
Inici del projecte	16/08/1993
Llicència	Les que compleixin les DFSG
Versió estudiada	Debian 4.0 (àlies Etch)
Línies de codi font	288.500.000
Nombre de paquets	10.106
Estimació de cost	$ 10.140 milions
Estimació de temps d'execució	8,84 anys
Nombre aproximat de desenvolupadors (*maintainers*)	Uns 1.500
Eines d'ajuda al desenvolupament	Llistes de correu, sistema de notificació d'errors

El llenguatge de programació més utilitzat en Debian 4.0 és C, amb més d'un 51% de les línies de codi. Tanmateix, com es mostrarà una mica més endavant en aquest apartat, la importància de C remet amb el temps, ja que les primeres versions de Debian comptaven amb fins a un 80% del codi en C. Bona part de la "culpa" del retrocés de C el té el segon llenguatge, C++, però sobretot la

irrupció dels llenguatges de guió (*scripting languages*) com ara Perl, Python i PHP. Entremig s'hi escolen llenguatges com Lisp o Java (que en Debian està infrarepresentat per la seva política de no admetre codi que depengui de la màquina virtual privativa de Sun).

Taula 21. Llenguatges de programació utilitzats a Debian GNU/Linux 4.0

Llenguatge de programació	Línies de codi (en milions)	Percentatge
C	155	51%
C++	55	19%
Shell	30	10%
Perl	8,1	2,9%
Lisp	7,7	2,7%
Python	7,2	2,5%
Java	6,9	2,4%
PHP	3,5	1,24%

A la taula 21 es mostra l'evolució dels llenguatges més significatius en Debian.

Taula 22. Llenguatges més utilitzats en Debian

Llenguatge	Debian 2.0		Debian 2.1		Debian 2.2		Debian 3.0	
C	19.400.000	76,67%	27.800.00	74,89%	40.900.000	69,12%	66.500.000	63,08%
C++	1.600.000	6,16%	2.800.000	7,57%	5.980.000	10,11%	13.000.000	12,39%
Shell	645.000	2,55%	1.150.000	3,10%	2.710.000	4,59%	8.635.000	8,19%
Lisp	1.425.000	5,64%	1.890.000	5,10%	3.200.000	5,41%	4.090.000	3,87%
Perl	425.000	1,68%	774.000	2,09%	1.395.000	2,36%	3.199.000	3,03%
Fortran	494.000	1,96%	735.000	1,98%	1.182.000	1,99%	1.939.000	1,84%
Python	122.000	0,48%	211.000	0,57%	349.000	0,59%	1.459.000	1,38%
Tcl	311.000	1,23%	458.000	1,24%	557.000	0,94%	1.081.000	1,02%

Hi ha llenguatges que podríem considerar minoritaris que aconsegueixen un lloc bastant alt en la classificació. Això es deu al fet que fins i tot trobant-se presents en un nombre reduït de paquets, aquests són bastant grans. Tal és el cas d'Ada, que en tres paquets (GNAT, un compilador d'Ada; libgtkada, un enllaç a la biblioteca GTK, i ASIS, un sistema per a gestionar fonts en Ada) aglutina 430.000 d'un total de 576.000 línies de codi font que s'han comptabi-

litzat en Debian 3.0 per a Ada. Un altre cas semblant és el de Lisp, que compta només en GNU Emacs i XEmacs amb més d'1.200.000 línies de les prop de quatre milions en tota la distribució.

9.9.2. Comparació amb altres sistemes operatius

Si diuen que totes les comparacions són odioses, les de programari lliure amb programari propietari ho són encara més. Les radiografies tan detallades de Red Hat Linux i Debian han estat possibles per la seva condició de programari lliure. L'accés al codi (i a una altra informació que ha estat exposada en aquest capítol) és indispensable per a estudiar a fons les diferents versions quant a nombre de línies, paquets, llenguatges de programació utilitzats... Però els avantatges del programari lliure van més enllà, perquè a més, faciliten la revisió de terceres persones, o bé grups d'investigació o bé senzillament persones interessades.

En els sistemes propietaris, en general, realitzar un estudi així és tasca impossible. De fet, els comptes que s'ofereixen a continuació tenen les seves fonts en les mateixes companyies que hi ha darrere del desenvolupament de programari, per la qual cosa no podem avalar-ne la veracitat. Per acabar-ho d'adobar, en molts casos no sabem si s'està parlant de línies de codi font físiques tal com hem fet al llarg d'aquest capítol, o si també inclouen en els seus comptes les línies en blanc i les de comentaris. A això cal afegir-hi que tampoc no sabem del cert el que consideren en el seu programari, de manera que per a algunes versions de Microsoft Windows no sabem si inclouen el paquet de Microsoft Office o no.

En qualsevol cas, i tenint en compte tot el que s'ha comentat sobre això en paràgrafs anteriors, pensem que incloure aquesta comparativa és interessant, ja que ens ajuda a situar les diferents distribucions de Red Hat i de Debian dins d'un panorama més ampli. El que sembla estar fora de tot dubte és que tant Debian com Red Hat, però especialment el primer, són les majors col·leccions de programari vistes mai per la humanitat fins al moment.

Els números que s'esmenten a continuació procedeixen de Mark Lucovsky [168] per a Windows 2000, SUN Microsystems [171] per a StarOffice 5.2, Gary McGraw [169] per a Windows XP i Bruce Schneier [200] per a la resta de sistemes. A la taula 23 es mostra la comparativa en ordre creixent.

Taula 23. Comparació amb sistemes propietaris

Sistema	Data de publicació	Línies de codi (aproximadament)
Microsoft Windows 3.1	Abril 1992	3.000.000
SUNSolaris 7	Octubre 1998	7.500.000
SUN StarOffice 5.2	Juny 2000	7.600.000

Sistema	Data de publicació	Línies de codi (aproximadament)
Microsoft Windows 95	Agost 1995	15.000.000
Red Hat Linux 6.2	Març 2000	18.000.000
Debian 2.0	Juliol 1998	25.000.000
Microsoft Windows 2000	Febrer 2000	29.000.000
Red Hat Linux 7.1	Abril 2001	32.000.000
Debian 2.1	Març 1999	37.000.000
Windows NT 4.0	Juliol 1996	40.000.000
Red Hat Linux 8.0	Setembre 2002	50.000.000
Debian 2.2	Agost 2000	55.000.000
Debian 3.0	Juliol 2002	105.000.000

9.10. Eclipse

La plataforma Eclipse consisteix en un entorn de desenvolupament integrat (IDE, *integrated development environment*) obert i extensible. Un IDE és un programa compost per un conjunt d'eines útils per a un desenvolupador de programari. Com a elements bàsics, un IDE té un editor de codi, un compilador/intèrpret i un depurador. Eclipse serveix com a IDE Java i disposa de nombroses eines de desenvolupament de programari. També dóna suport a altres llenguatges de programació, com C/C++, Cobol, Fortran, PHP o Python. A la plataforma base d'Eclipse s'hi poden afegir extensions (*plug in*) per a estendre la funcionalitat.

El terme Eclipse a més identifica la comunitat de programari lliure per al desenvolupament de la plataforma Eclipse. Aquest treball es divideix en projectes que tenen l'objectiu de proporcionar una plataforma robusta, escalable i de qualitat per al desenvolupament de programari amb l'IDE Eclipse. Està coordinat per la Fundació Eclipse, que és una organització sense ànim de lucre creada per a la promoció i l'evolució de la plataforma Eclipse i que dóna suport tant a la comunitat com a l'ecosistema Eclipse.

9.10.1. Història d'Eclipse

Gran part de la programació d'Eclipse va ser realitzada per IBM abans que es creés el projecte Eclipse com a tal. L'antecessor d'Eclipse va ser VisualAge i es va construir usant Smalltalk en un entorn de desenvolupament anomenat Envy. Amb l'aparició de Java en la dècada dels noranta, IBM va desenvolupar una màquina virtual vàlida tant per a Smalltalk com per a Java. El ràpid creixement de Java i els seus avantatges amb vista a una Internet en plena expansió van obligar IBM a plantejar-se l'abandonament d'aquesta màquina virtual dual i la

construcció d'una nova plataforma basada en Java des del principi. El producte acabat resultant va ser Eclipse, que ja havia costat uns 40 milions de dòlars a IBM l'any 2001.

A finals del 2001 IBM, al costat de Borland, van crear la fundació sense ànim de lucre Eclipse, fet amb el qual s'obrien al món de codi obert. A aquest consorci s'hi han unit progressivament importants empreses del desenvolupament de programari a escala mundial: Oracle, Rational Software, Red Hat SuSE, HP, Serena, Ericsson i Novell, entre d'altres. Hi ha dues absències significatives: Microsoft i Sun Microsystems. Microsoft ha estat exclòs per la seva posició de monopoli del mercat, i Sun Microsystems compta amb el seu propi IDE, la principal competència d'Eclipse, NetBeans. De fet, el nom d'Eclipse va ser elegit perquè l'objectiu era crear un IDE capaç d'"eclipsar Visual Studio" (Microsoft) així com "eclipsar el sol" (Sun Microsystem).

L'última versió estable d'Eclipse es troba disponible per als sistemes operatius Windows, Linux Solaris, AIX, HP-UX i Mac OS X. Totes les versions d'Eclipse necessiten tenir instal·lat en el sistema una màquina virtual Java (JVM), preferiblement JRE (Java Runtime Environment) o JDK (Java Developer Kit) de Sun, que a principis del 2007 no eren lliures (encara que hi ha un anunci per part de Sun que ho seran).

9.10.2. Estat actual d'Eclipse

Tot el treball desenvolupat per al consorci Eclipse s'organitza en projectes. Aquests projectes, al seu torn, divideixen el treball en subprojectes, i els subprojectes en components. Els projectes d'alt nivell són gestionats per comitès de la Fundació Eclipse (PMC, *project management committees*). A continuació s'enumeren els projectes d'alt nivell:

- Eclipse. Plataforma base per a la resta de components. L'esmentada plataforma serà lliure, robusta, completa i de qualitat per al desenvolupament d'aplicacions riques de client (RCP, *rich client plaftorm*) i eines integrades (*plug in*). El nucli d'execució de la plataforma Eclipse s'anomena Equinox i és una implementació de l'especificació OSGi (Open Services Gateway Initiative), que descriu una arquitectura orientada a serveis (SOA) per a aplicacions.

- Eines (ETP, *Eclipse tools project*). Eines diverses i components comuns per a la plataforma Eclipse.

- Web (WTP, *web tools project*). Eines per al desenvolupament d'aplicacions web i JEE (Java Enterprise Edition).

- Proves i rendiment (TPTP, *test and performance tools project*). Eines de proves i mesura de rendiments perquè els desenvolupadors puguin monitoritzar les seves aplicacions i fer-les més productives.

- Informes web (BIRT, *business intelligence and reporting tools*). Sistema de generació d'informes web.

- Modelatge (EMP, *Eclipse modeling project*). Eines de desenvolupament basat en models.

- Dades (DTP, *data tools platform*). Suport per a tecnologies centrades en el maneig de dades.

- Dispositius encastats (DSDP, *device software development platform*). Eines per al desenvolupament d'aplicacions destinades a ser executades en dispositius limitats en maquinari, és a dir, dispositius encastats.

- Arquitectura orientada a serveis (*SOA, service oriented architecture*). Eines per al desenvolupament de projectes orientats a serveis.

- Tecnologia Eclipse. Investigació, divulgació i evolució de la plataforma Eclipse.

Els principis que guien el desenvolupament de la comunitat Eclipse segueixen les línies següents:

- Qualitat. El programari desenvolupat a Eclipse ha de seguir els patrons de qualitat de l'enginyeria del programari.

- Evolució. La plataforma Eclipse, així com les eines entorn d'aquesta, han d'evolucionar dinàmicament d'acord amb els requisits dels usuaris.

- Meritocràcia. Com més es contribueix, més responsabilitats es tenen.

- Ecosistema Eclipse. Hi haurà recursos donats per la comunitat de programari lliure al consorci Eclipse. Aquests recursos seran gestionats en benefici de la comunitat.

El procés de desenvolupament seguit a Eclipse segueix unes fases determinades. En primer lloc, hi ha una fase de preproposta, en la qual un individu o una empresa declaren el seu interès per establir un projecte. Si la proposta és acceptada, es decideix si serà un projecte d'alt nivell o bé un subprojecte. El pas següent és validar el projecte en termes d'aplicabilitat i qualitat. Després d'aquesta etapa d'incubació, tindrà lloc una revisió final. Si supera aquesta revisió, el projecte haurà demostrat la seva validesa de cara a la comunitat Eclipse, i es passarà a la fase d'implementació.

9.10.3. Radiografia d'Eclipse

Eclipse es distribueix sota llicència EPL (Eclipse Public License). Aquesta llicència és considerada lliure per l'FSF i per l'OSI. La llicència EPL permet usar, modificar, copiar i distribuir noves versions del producte llicenciat. L'antecessor d'EPL és CPL (Common Public License). CPL va ser escrita per IBM, mentre que EPL és obra del consorci Eclipse.

Estimar la inversió i l'esforç invertits en Eclipse no és una tasca senzilla. Això és a causa que el codi font que compon l'ecosistema Eclipse està distribuït en nombrosos projectes i dipòsits de programari.

A continuació, es mostra el resultat d'aplicar el mètode COCOMO a la plataforma Eclipse, que serveix de base a la resta de *plug in*.

Taula 24. Anàlisi d'Eclipse

Pàgina web	http://www.Eclipse.org
Inici del projecte	2001
Llicència	Eclipse Public License
Versió analitzada	3.2.2
Línies de codi font	2.163.932
Nombre de fitxers	15.426
Estimació de cost	$ 85.831.641
Estimació de temps d'execució	6,22 anys (74,68 mesos)
Estimació de nombre mitjà de desenvolupadors	102,10
Nombre aproximat de desenvolupadors	133 *commiters*
Eines d'ajuda al desenvolupament	CVS, llistes de correu, sistema de notificació d'errors (Bugzilla)

A la taula següent es mostren els llenguatges de programació usats en Eclipse 3.2.2:

Taula 25. Llenguatges de programació utilitzats a Eclipse

Llenguatge de programació	Línies de codi	Percentatge
Java	2.066.631	95,50%
C	85.829	3,97%
Perl	3.224	0,06%
C++	5.442	0,25%

Llenguatge de programació	Línies de codi	Percentatge
JSP	3.786	0,17%
Perl	1.325	0,06%
Lex	1.510	0,03%
Shell	849	0,04%
Python	46	0,00%
PHP	24	0,00%

10. Altres recursos lliures

"If you want to make an apple pie from scratch, you must first create the universe."

"si vols fer un pastís de poma des del principi, primer has de crear l'Univers."

Carl Sagan

Es poden estendre les idees dels programes lliures a altres recursos? Podem pensar que altres recursos d'informació fàcilment copiables electrònicament són de naturalesa similar als programes, per la qual cosa se'ls poden aplicar les mateixes llibertats, regles i models de desenvolupament i negoci. Tanmateix, hi ha diferències les implicacions de les quals han fet que no es desenvolupin amb la mateixa força que els programes. La principal és que n'hi ha prou de copiar els programes perquè funcionin, mentre que des que es copia un altre tipus d'informació fins que comença a ser útil s'ha de passar per un procés més o menys costós, que pot anar des de l'aprenentatge d'un document fins a la posada en producció d'un maquinari descrit en un llenguatge apropiat.

10.1. Recursos lliures més importants

De la documentació de programes i altres documents tècnics ja n'hem parlat a l'apartat 3.2.5. Parlarem aquí d'un altre tipus de creacions, que poden ser també textuals, però ja no relacionades amb el programari, tant en àmbits científics i tècnics com en l'artístic.

10.1.1. Articles científics

L'avenç de la ciència es deu en gran part al fet que els investigadors que la fan progressar per a benefici de la humanitat publiquen els resultats dels seus treballs en revistes d'àmplia difusió. Gràcies a aquesta difusió els investigadors desenvolupen un currículum que els permet progressar cap a llocs de major categoria i responsabilitat i alhora obtenir ingressos a partir de contractes d'investigació aconseguits gràcies al prestigi assolit.

Així doncs, aquesta difusió d'articles representa un *model de negoci* que s'ha demostrat molt fructífer. Perquè sigui possible, es necessita una àmplia difusió i qualitat garantida. La difusió es veu obstaculitzada per una gran quantitat de revistes existents, de cost no menyspreable, l'adquisició de les quals només és possible amb pressupostos generosos. La qualitat es garanteix per mitjà de la revisió per part d'especialistes.

És per això que han sorgit nombroses iniciatives de revistes a la Xarxa, entre les quals destaquen la veterana *First Monday* ("First Monday: peer rewiewed journal on the Internet") [26] o el projecte *Public Library Of Science* (PLOS –http://www.publiclibraryofscience.org– [55]). A "Directory of Open Access Journals"

[22] se n'esmenen bastants més. S'ha de permetre que persones diferents dels autors publiquin una modificació d'un article? Hi ha objeccions que al·leguen des d'una possible falta de qualitat o una tergiversació d'opinions o resultats, fins al perill de plagi fàcil que permet a alguns enfilar-se sense esforç i enfosquir els mèrits dels verdaders autors. Tanmateix, l'obligació d'esmenar l'autor original i de passar una revisió en una revista de prestigi pot contrarestar aquests problemes (*vid.* apartat 10.2.2).

S'ha volgut establir un paral·lelisme entre el programari lliure i la ciència, ja que el model de desenvolupament del primer implica la màxima difusió, la revisió per d'altres, presumiblement experts, i la reutilització de resultats ("Free software/free science", 2001) [154].

10.1.2. Lleis i estàndards

Hi ha documents el caràcter dels quals és normatiu que defineixen com s'han de fer les coses, o bé per facilitar la convivència entre les persones, o bé perquè programes o màquines interoperin entre si. Aquests documents requereixen la màxima difusió, per la qual cosa tot obstacle a aquesta és contraproduent. És per això que és comprensible que tinguin un tractament especial, com passa amb la Llei de la propietat intel·lectual espanyola:

> "No són objecte de propietat intel·lectual les disposicions legals o reglamentàries i els seus corresponents projectes, les resolucions dels òrgans jurisdiccionals i els actes, acords, deliberacions i dictàmens dels organismes públics, així com les traduccions oficials de tots els textos anteriors."

La variant tecnològica de les lleis són les normes o estàndards. En programació són especialment importants els protocols de comunicacions, o bé entre màquines remotes o bé entre mòduls de la mateixa màquina. És obvi que no se n'ha de limitar la difusió, especialment si volem que floreixin els programes lliures que interoperin amb d'altres, però malgrat això, tradicionalment, els organismes de normalització, com l'ISO[11] i ITU[12], venen les seves normes, fins i tot en format electrònic, i en prohibeixen la redistribució. Encara que això pugui intentar justificar-se per a cobrir parcialment les despeses, la lliure difusió del text dels estàndards ha resultat molt més productiva. Aquest és el cas de les recomanacions del W3C[13] i, sobretot, de les que governen Internet, disponibles des del principi en documents anomenats RFC, de *request for comments*, en formats electrònics llegibles en qualsevol editor de textos.

(11) International Organization for Standardization

(12) International Telecommunications Union

(13) World Wide Web Consortium

Però no és la disponibilitat l'única causa de l'èxit dels protocols d'Internet. També ho és el seu *model de desenvolupament*, molt similar al del programari lliure pel seu caràcter obert a la participació de qualsevol interessat i per la utilització de llistes de correu i de mitjans similars. A "The Internet standards process - revision 3" [94] i a "GNU make" [36] es descriu aquest procés.

S'ha de permetre la modificació del text de lleis i normes? Òbviament no si això dóna lloc a confusió. Per exemple, només s'admet que una RFC sigui modificada per a explicar-la o afegir-hi comentaris explicatius, mentre que ni tan sols això es permet sense autorització explícita per a les recomanacions del W3C (http://www.w3.org/Consortium/Legal/2002/copyright-documents-20021231) [65]. Les llicències són també documents legals no modificables. S'hauria de permetre la creació de noves normes derivades d'altres d'existents a partir dels documents originals? Probablement això portaria a la proliferació fàcil de normes similars i incompatibles que crearien confusió i podrien ajudar a empreses dominants al mercat a promoure la seva pròpia variant incompatible, com de fet ja està passant, especialment en l'àmbit del web. No obstant això, en el cas de les legislacions dels estats, moltes vegades s'han copiat literalment lleis d'altres països, adaptades amb petites modificacions a les particularitats locals.

Hi ha un model de negoci per a les lleis i normes? Al voltant de les lleis hi ha una multitud de professionals que s'encarreguen de fer-ne el disseny, interpretació i de forçar-ne l'aplicació (legisladors, advocats, procuradors, jutges, etc.). Entorn de les normes hi ha laboratoris que atorguen certificats de conformitat. Les organitzacions de normalització viuen, o haurien de viure, de les aportacions de membres interessats a promoure estàndards, per exemple, perquè el seu negoci estigui basat en productes que interoperin.

De la mateixa manera que és convenient tenir una definició de *programari lliure* o *obert*, també és necessària una definició d'*estàndards oberts*. Bruce Perens (http://perens.org/OpenStandards) [15] n'ha proposat una de basada en els principis següents:

1) Disponibilitat: si és possible, proporcionar fins i tot una implementació lliure de referència.

2) Maximització de les opcions de l'usuari final.

3) Sense taxes sobre la implementació (no així sobre la certificació, encara que aconsella la disponibilitat d'eines lliures d'*autocertificació*).

4) Sense discriminació d'implementador.

5) Permís d'extensió o restricció (no certificable).

6) Evitació de pràctiques predatòries per part fabricants dominants. Tota extensió propietària ha de tenir una implementació lliure de referència.

10.1.3. Enciclopèdies

El 1999 Richard Stallman va llançar la idea d'una enciclopèdia lliure ("The free universal encyclopedia and learning resource", 2001) [210] com un mecanisme per a evitar l'apropiació del coneixement i proporcionar accés universal a documentació formativa. Estaria formada per articles aportats per la comunitat sense un control centralitzat, on diferents actors assumirien diferents funcions, entre les quals s'aconsellava, però no s'obligava, la de revisor. Aquesta enciclopèdia no contindria només text, sinó també elements multimèdia i programari educatiu lliure.

Han sorgit diverses iniciatives per a realitzar aquesta visió. Per exemple, Nupedia (http://www.nupedia.com) [178] va tractar sense èxit de construir una enciclopèdia de qualitat, potser perquè requeria un format relativament difícil d'aprendre (TEI), encara que segurament en un major grau pel requisit que tots els articles necessitessin un editor, revisors científics i d'estil, etc.

Hereva de Nupedia però amb molt més èxit que aquesta, ha estat Wikipedia (http://www.wikipedia.org) [69]. Wikipedia és una enciclopèdia lliure plurilingüe basada en la tecnologia *wiki*. Wikipedia s'escriu de forma cooperativa per voluntaris i permet que la gran majoria dels articles siguin modificats per qualsevol persona amb accés mitjançant un navegador web. El seu èxit es basa en una estructura més flexible per a l'edició, que elimina els obstacles que imponia Nupedia i que s'aproxima més a la idea de llibertat de Stallman. La paraula *wiki* prové del hawaià *wiki wiki* ('ràpid'). La tecnologia *wiki* permet a qualsevol editar qualsevol document per mitjà del sistema de text estructurat, extraordinàriament simple, com s'ha vist a l'apartat 8.6.2. El febrer del 2007, el nombre d'articles en anglès a Wikipedia superava la xifra d'1.500.000, i en espanyol s'han assolit els 200.000 articles.

Nota

Wikipedia és un projecte de la fundació sense ànim de lucre Wikimedia, que manté a més, amb la mateixa fórmula que Wikipedia, els projectes següents:

- Viccionari (http://www.wiktionary.org) [66]. Es tracta d'un projecte cooperatiu per a produir un diccionari multilingüe gratuït, amb significats, etimologies i pronunciacions en aquelles llengües en les quals sigui necessari.
- Viquillibres (http://www.wikibooks.org/) [67]. És una col·lecció de llibres de contingut lliure que té com a objectiu posar a disposició de qualsevol persona llibres de text, manuals, tutorials o altres textos pedagògics de contingut lliure i d'accés gratuït.
- Viquicites (http://www.wikiquote.org) [70]. És una recopilació de frases cèlebres en tots els idiomes, que inclou les fonts quan aquestes es coneixen.
- Viquitext. És una biblioteca de textos originals que es troben en domini públic o que s'ha publicat amb llicència GFDL (llicència de documentació lliure de GNU).
- Wikispecies (http://species.wikimedia.org/) [71]. Són un repertori obert d'espècies animals, vegetals, fongs, bacteris i de totes les formes de vida conegudes.
- Wikinoticias (http://wikinews.org/) [68]. És una font de notícies de contingut lliure en la qual els usuaris són els redactors.
- Commons (http://commons.wikimedia.org/) [19]. És un dipòsit lliure d'imatges i contingut multimèdia.
- Wikiversidad (http://wikiversity.org/) [72]. És una plataforma educativa en línia lliure i gratuïta, basada en projectes d'aprenentatge a qualsevol nivell educatiu.
- Meta-Wiki (http://meta.wikimedia.org/) [48]. És el lloc web de suport als projectes de la Fundació Wikimedia.

Es pot destacar també la *Concise Encyclopedia of Mathem*, amb un concepte de llibertat més limitat (només es pot consultar a la Xarxa) i un model de desenvolupament que necessàriament fa passar totes les contribucions per un comitè editorial.

10.1.4. Cursos

Amb la mateixa finalitat que les enciclopèdies, es pot produir material docent lliure, com ara apunts, transparències, exercicis, llibres, planificacions o programari didàctic. Hi ha una tendència a veure les universitats com un negoci de producció i venda de coneixement que contradiu els seus principis. Els motius pels quals una universitat pot posar a disposició de tothom aquests materials són els següents:

- El compliment de la seva missió com a agent difusor del coneixement.
- El baix cost que suposa fer disponibles materials existents a tot el món.
- El fet que aquests materials no substitueixen l'ensenyament presencial.
- La concepció d'aquests materials com a propaganda que pot atreure alumnes i que contribueix al prestigi de la universitat.

- La possibilitat de creació d'una comunitat de docents que revisin els materials i els millorin.

La iniciativa més notable en aquest sentit és la del MIT (http://ocw.mit.edu) [174], que preveu fer accessibles de manera coherent i uniforme més de dos mil cursos ben catalogats.

10.1.5. Col·leccions i bases de dades

La mera recollida d'informació seguint determinats criteris, ordenant-la i facilitant el seu accés és per si mateixa un producte d'informació valuós, independent de la informació en si mateixa, subjecte per tant a autoria i, per això, a restriccions de les llibertats d'accés, modificació i redistribució. Per tant, si desitgem informació lliure, també podem desitjar col·leccions lliures.

Per exemple, podem voler classificar la informació rellevant a Internet, organitzant i comentant enllaços. Això és el que fa l'ODP (Open Directory Project –http://dmoz.org– [109]), que opera Netscape i que mantenen editors voluntaris organitzats segons un esquema jeràrquic. El directori complet pot copiar-se lliurement en format RDF i publicar-se modificat d'alguna manera, com fan Google i molts altres cercadors que l'aprofiten. Netscape, propietari del directori, garanteix un *contracte social* ("Open Directory Project social contract") [53] inspirat en el de la distribució Debian (http://www.debian.org/social_contract.html) [106], que facilita la col·laboració exterior assegurant que l'Open Directory Project sempre serà lliure, amb polítiques públiques, autoregulat per la comunitat i amb els usuaris com a primera prioritat.

Un altre exemple de col·leccions interessants per a nosaltres són les distribucions de programari lliure, amb els programes modificats perquè encaixin perfectament entre si i precompilats perquè es puguin executar fàcilment.

10.1.6. Maquinari

La llibertat al maquinari té dos aspectes. El primer és la necessitat que les interfícies i els jocs d'instruccions siguin oberts, de manera que qualsevol pugui realitzar un controlador de dispositiu o un compilador per a una arquitectura. El segon és disposar de la informació i el poder suficients per a reproduir un disseny maquinari, modificar-lo i combinar-lo amb d'altres. Els dissenys poden considerar-se programari en un llenguatge apropiat (VHDL, Verilog, etc.). Tanmateix, fer-los funcionar no és fàcil, ja que cal fabricar-los, la qual cosa és cara i lenta. Tanmateix, hi ha iniciatives en aquest sentit, entre les quals podem destacar OpenCores (http://www.opencores.org) [52], per a circuits integrats.

10.1.7. Literatura i art

Per acabar el nostre recorregut pels recursos lliures, no podem oblidar l'art i la literatura, que tenen com a objectiu últim no tant la utilitat com l'estètica. Quines raons pot tenir un artista per concedir llibertats de còpia, modificació i redistribució? D'una banda, donar-se a conèixer i afavorir la difusió de la seva obra, la qual cosa li pot permetre d'obtenir ingressos per altres activitats, com ara concerts i obres d'encàrrec, i de l'altra, afavorir l'experimentació i la creativitat. En l'art passa el mateix que en la tècnica: la innovació és incremental, i de vegades és difícil distingir el plagi de la pertinença a un mateix moviment o corrent artístic.

Òbviament no són el mateix la creació que la interpretació, ni la música que la literatura. La música, la pintura, la fotografia i el cinema són molt semblants als programes, en el sentit que se'ls fa "funcionar" immediatament en un ordinador, mentre que amb l'escultura, per exemple, no es pot. No hi ha moltes iniciatives en art i literatura lliures, i aquestes són molt diverses. Podem esmentar les novel·les del col·lectiu Wu Ming (http://www.wumingfoundation.com) [29].

10.2. Llicències d'altres recursos lliures

Les llicències de programari lliure han estat font d'inspiració per a altres recursos intel·lectuals, de manera que n'hi ha molts que les han adoptades directament, especialment en el cas de la documentació, i altres vegades les han adaptades lleugerament, com la pionera Open Audio License (http://www.eff.org/IP/Open_licenses/eff_oal.html) [114]. La majoria són de tipus *copyleft* si permeten treballs derivats.

Per a qualsevol tipus de textos s'ha utilitzat i s'utilitza bastant la llicència de documentació lliure de GNU (*vid.* apartat 10.2.1), encara que cada vegada s'estan acceptant més les llicències de Creative Commons (*vid.* apartat 10.2.2).

Fins i tot s'han utilitzat llicències de programes (GPL i LGPL) per a maquinari, encara que aquesta matèria és complexa i difícil de conciliar amb la legalitat vigent. En efecte, els dissenys i els diagrames poden ser utilitzats, sense ser copiats físicament, per a extreure idees que s'utilitzin per a nous dissenys tancats. Per exemple, 'OpenIPCore Hardware General Public License ("OpenIPCore hardware general public license") [155] recull la prohibició d'aquesta apropiació, però la seva legalitat és dubtosa [209]. L'única possibilitat de protegir aquestes idees és per mitjà d'algun tipus de patent lliure, una cosa encara no desenvolupada i fora de l'abast dels qui no tenen intenció o possibilitat de fer negoci amb aquestes.

10.2.1. Llicència de documentació lliure de GNU

Una de les llicències de tipus *copyleft* més conegudes per a la documentació tècnica, o bé de programes o bé de qualsevol altra cosa, és la de la Free Software Foundation. Després d'adonar-se que un document no és el mateix que un programa, Richard Stallman va promoure una llicència per als documents que acompanyessin els programes i per a altres documents de caràcter tècnic o didàctic.

Per facilitar el desenvolupament de versions derivades, cal posar a disposició de qui ho demani una còpia *transparent* del document, en el sentit del que s'ha explicat a l'apartat 3.2.5, a més de les còpies *opaques*, en una analogia entre els codis font i els objectes dels programes.

Una de les preocupacions de la llicència és reconèixer l'autoria i impedir que es tergiversin idees o opinions expressades per l'autor. Per a això exigeix que les obres derivades exhibeixin a la portada un títol diferent dels de les versions anteriors (llevat de permís exprés) i que es consigni expressament d'on es pot aconseguir l'original. També han de llistar-se com a autors els més importants dels originals, a més dels de les modificacions, i s'han de conservar totes les notes sobre drets d'autor. A més, s'han de conservar agraïments i dedicatòries, i s'ha de respectar l'apartat d'història, si el té, a l'hora d'afegir les modificacions noves. Fins i tot, i això és el més criticat de la llicència, poden anomenar-se seccions invariants i textos de cobertes, que ningú no pot modificar ni eliminar, si bé la llicència només permet considerar invariants textos *no tècnics*, que aquesta mateixa anomena *secundaris*.

Aquesta llicència ha generat una gran polèmica al món del programari lliure, fins al punt que en la distribució Debian s'està discutint (en el moment de publicar aquest llibre) si s'elimina o es passa a la secció extraoficial *no lliure* tot document amb aquesta mateixa. Fins i tot encara que no hi hagi seccions invariants, com que els treballs derivats s'han de regir per aquesta llicència, les poden afegir. S'argumenta, per exemple, que hi pot haver seccions invariants incorrectes o obsoletes, però que han de mantenir-se. En tot cas, la llicència és incompatible amb les directrius de programari lliure de Debian (http://www.debian.org/social_contract.html#guidelines) [104], però la qüestió potser es troba en el fet de si la documentació les ha de complir (per exemple, els textos de les llicències tampoc no es poden modificar).

Suggeriment

Les primeres versions d'aquest text van estar emparades per la llicència GFDL, però posteriorment els autors decidim fer servir una llicència de Creative Commons (*vid.* apartat 10.2.2), potser més adaptada a les característiques d'un llibre.

10.2.2. Llicències de Creative Commons

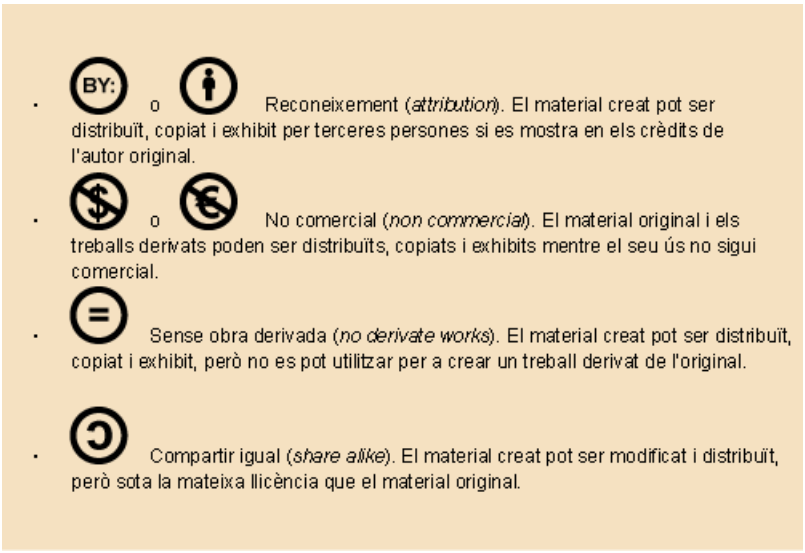
Creative Commons (http://creativecommons.org) [21] és una organització sense ànim de lucre fundada el 2001 per experts en propietat intel·lectual, dret en la societat de la informació i informàtica per tal de fomentar l'existència, la conservació i l'accessibilitat de recursos intel·lectuals cedits a la comunitat de

diverses maneres. Està basada en la idea que algunes persones poden no voler exercir sobre la seva obra tots els drets de propietat intel·lectual que els permet la llei, ja que això pot impedir una àmplia distribució.

A finals del 2002 van veure la llum les primeres llicències Creative Commons per a treballs creatius, que van passar per diverses versions. Han estat creades per a ser:

- robustes, per a resistir l'escrutini d'un tribunal en diferents països;
- senzilles, per a poder ser usades per persones que no són especialistes en assumptes legals;
- sofisticades, per a ser identificades per diverses aplicacions de la web.

Les diferents llicències permeten a l'autor seleccionar quin tipus de llibertats cedeix, a més de la de còpia, segons quatre dimensions:



En la versió 1.x de les llicències Creative Commons es recollien onze tipus de llicències, que combinaven les quatre característiques bàsiques damunt descrites. Un 98% dels autors elegia l'opció "atribució", amb la qual cosa en la versió 2.x de les llicències Creative Commons l'atribució és un requisit. D'aquesta manera, es redueixen els onze tipus de llicència a sis, que són els següents:

A la taula següent es mostren de forma esquemàtica les llicències segons la seva icona representativa. Aquesta icona sol ser un enllaç al resum de la llicència, allotjada a [21].

	Permet modificacions.	Permet modificacions si es comparteix de la mateixa manera.	No permet modificacions.
Permet ús comercial.	BY	BY SA	BY ND
No permet ús comercial.	BY NC	BY NC SA	BY NC ND

En lloc de la icona representativa de la llicència es pot usar la icona genèrica[14], encara que l'enllaç ha de ser a la llicència que l'autor elegeixi. El codi HTML de l'enllaç de la llicència pot obtenir-se de Creative Commons [21]. Una vegada escollida la llicència i afegida la icona corresponent, s'haurà llicenciat l'obra, de manera que s'obtindrà:

(14)

- *Commons deed*. És un resum del text legal amb les icones rellevants de la llicència. Aquest resum és la destinació de l'enllaç obtingut de Creative Commons [21].

- *Legal Code*. És el codi legal complet en què es basa la llicència. A aquest text es pot accedir des del resum anterior.

- *Digital code*. És la descripció RDF (*resource description framework*), que serveix perquè els motors de cerca i altres aplicacions identifiquin la llicència de l'obra i les seves condicions d'ús.

El febrer del 2007 es van publicar les llicències Creative Commons en la seva versió 3.0. Es tracta d'una actualització per a corregir moltes dificultats que se'ls atribuïen. El primer gran canvi és que la llicència genèrica deixa de ser la nord-americana per a ser una versió basada en la terminologia del Tractat de Berna. En segon lloc, s'esmenten específicament els drets morals i la societat

de gestió, ja que abans en cada jurisdicció s'havien pres decisions diferents. En tercer i últim lloc, s'ha modificat el text tant del *commons deed* com del *legal code* que acompanya cada llicència per a deixar clar que la clàusula de reconeixement d'autoria no permet al llicenciatari donar a entendre que té una relació o associació amb el llicenciador.

Creative Commons permet, a més, un altre tipus de llicències per a aplicacions específiques. Són els següents:

No totes les llicències Creative Commons són considerades lliures des de sectors afins al programari lliure, ja que per a això haurien de concedir les quatre llibertats essencials (*vid.* secció 1.1.1). Benjamin "Mako" Hill (desenvolupador de Debian i d'Ubuntu) ha creat la web Freedomdefined.org (http://freedomdefined.org/) [28], amb l'objectiu de definir millor què és cultura lliure i què no. Sobre la base d'això, de les sis llicències bàsiques Creative Commons només dues són estrictament lliures: reconeixement (BY) i reconeixement - compartir igual (BY-SA), aquesta última a més amb *copyleft*.

Bibliografia

[1] Aap Project: http://www.a-a-p.org

[2] Ada Core Technologies: http://www.gnat.com/

[3] Alcôve: http://www.alcove.com

[4] Alcôve-Labs: http://www.alcove-labs.org

[5] Alioth: http://alioth.debian.org

[6] Anjuta: http://www.anjuta.org

[7] The Apache Ant Project: http://ant.apache.org

[8] Arch Revision Control System: http://www.gnu.org/software/gnu-arch/

[9] artofcode LLC: http://artofcode.com/

[10] Autoconf: http://www.gnu.org/software/autoconf

[11] Barrapunto: http://barrapunto.com

[12] Bazaar GPL Distributed Version Control Software: http://bazaar-vcs.org/

[13] Berlios. The Open Source Mediator: http://berlios.de

[14] Bitkeeper Source Management: http://www.bitkeeper.com

[15] Bruce Perens: http://perens.com/OpenStandards/Definition.html

[16] Caldera: http://www.sco.com

[17] Cisco Enterprise Print System: http://ceps.sourceforge.net/

[18] Code::blocks: http://www.codeblocks.org

[19] Commons: http://commons.wikimedia.org/

[20] Concurrent Version System: http://ximbiot.com/cvs/

[21] Creative Commons. http://creativecommons.org

[22] Directory of Open Access Journals: http://www.doaj.org

[23] Eclipse - An Open Development Platform: http://www.Eclipse.org

[24] eCos: http://sources.redhat.com/ecos/

[25] eCos license 2.0: http://www.gnu.org/licenses/ecos-license.html

[26] *First Monday. Peer Rewiewed Journal on the Internet*: http://firstmonday.org/

[27] Free Software Foundation: http://www.fsf.org

[28] Freedom Defined (Free Cultural Works): http://freedomdefined.org/

[29] Fundación Wu Ming: http://www.wumingfoundation.com

[30] GForge: http://gforge.org

[31] Gettext: http://www.gnu.org/software/gettext

[32] GNU Automake: http://www.gnu.org/software/automake

[33] GNU Emacs: http://www.gnu.org/software/emacs/

[34] GNU Libc: http://www.gnu.org/software/libc

[35] GNU Libtool: http://www.gnu.org/software/libtool

[36] GNU Make: http://www.gnu.org/software/make/make.html

[37] GNU Troff: http://www.gnu.org/software/groff/groff.html

[38] "Have you seen these hackers?": http://www.mozilla.org/MPL/missing.html

[39] "History of TeX": http://www.math.utah.edu/software/plot79/tex/history.html

[40] IBM Public License Version 1.0: http://opensource.org/licenses/ibmpl.php

[41] Jam Product Information: http://www.perforce.com/jam/jam.html

[42] KDevelop: http://www.kdevelop.org

[43] Launchpad: https://launchpad.net

[44] The Linux Documentation Project: http://www.tldp.org

[45] LinuxCare: http://www.levanta.com

[46] Mailman, the GNU Mailing List Manager: http://www.list.org

[47] The Malone Bug Tracker: https://launchpad.net/products/malone

[48] Metawiki: http://meta.wikimedia.org/

[49] Mozilla Public License 1.1: http://www.mozilla.org/MPL/MPL-1.1.html

[50] Mozilla Tinderbox: http://www.mozilla.org/tinderbox.html

[51] NetBeans: http://www.netbeans.org

[52] Open Cores: http://www.opencores.org

[53] Open Directory Project Social Contract:

[54] Open Source Initiative: http://www.opensource.org

[55] Public Library of Science: http://www.publiclibraryofscience.org

[56] Red Hat: http://www.redhat.com

[57] Savannah: http://savannah.gnu.org i http://savannah.nongnu.org

[58] Slashdot: News for Nerds. http://slashdot.org

[59] Sleepycat License: http://www.sleepycat.com/download/oslicense.html

[60] Sleepycat Software: http://www.sleepycat.com/

[61] SourceForge: Open Source Software Development Website: http://sourceforge.net

[62] Subversion: http://subversion.tigris.org

[63] Texinfo - The GNU Documentation System:

[64] Tigris.org: Open Source Software Engineering: http://tigris.org

[65] W3c Document License:

http://www.w3.org/Consortium/Legal/2002/copyright-documents-20021231

[66] Wikcionari: http://www.wiktionary.org

[67] Wikillibres: http://www.wikibooks.org/

[68] Wikinoticias: http://wikinews.org/

[69] Wikipedia: http://www.wikipedia.org

[70] Wikiquote: http://www.wikiquote.org

[71] Wikispecies: http://species.wikimedia.org/

[72] Wikiversitat: http://wikiversity.org/

[73] X Window System Release 11 License: http://www.x.org/Downloads_terms.html

[74] Ximian: http://www.novell.com./linux/ximian.html

[75] Zope Corporation: http://www.zope.com/

[76] Zope Public License 2.0: http://www.zope.org/Resources/ZPL

[77] Llei de propietat intel·lectual. Reial decret legislatiu 1/1996, de 12 d'abril (abril 1996):

[78] Affero General Public License, 2002: http://www.affero.org/oagpl.html

[79] Llei de propietat intel·lectual. Llei 23/2006, de 7 de juliol (juliol 2006):

[80] Flossimpact Study. Technical Report, European Comission, 2007: http://flossimpact.eu

[81] ISO JTC 1/SC 34. Standard Generalized Markup Language (SGML, ISO 8879), 1986:

[82] **Antoniades, I.; Samoladas, I.; Stamelos, I.; Bleris, G. L.** "Dynamical simulation models of the open source development process" En: Koch [157].

http://wwwai.wu-wien.ac.at/~koch/oss-book/

[83] **Bailey, E. C.** (1998). *Maximum RPM. Taking the Red Hat package manager to the limit*. http://rikers.org/rpmbook/

[84] **González Barahona, J. M.** (2000). "Software libre, monopolios y otras yerbas". *Todo Linux* (3). http://sinetgy.org/~jgb/articulos/soft-libre-monopolios/

[85] **González Barahona, J. M.** (2002). "¿Qué se hace con mi dinero?". *Todo Linux* (17).

http://sinetgy.org/~jgb/articulos/sobre-administracion/

[86] **González Barahona, J. M.; Robles, G.** Libre Software Engineering Web Site.

http://libresoft.dat.escet.urjc.es/

[87] **González Barahona, J. M.; Robles, G.** (2003, maig). "Unmounting the *code god* assumption". En: *Proceedings of the Fourth International Conference on eXtreme Programming and Agile Processes in Software Engineering*. Gènova, Itàlia.

[88] **González Barahona, J. M.; Robles, G.; Ortuño Pérez, M. A.; Rodero Merino, L.; Centeno González, J.; Matellán Olivera, V.; Castro Barbero, E. M.; De las Heras Quirós; P.** "Anatomy of two GNU/Linux distributions". En: Koch [157].

http://wwwai.wu-wien.ac.at/~koch/oss-book/

[89] **Barnson, M. P.** *The Bugzilla guide*.

http://www.bugzilla.org/docs214/html/index.html

[90] **Baudis, P.** "Cogito manual page".

http://www.kernel.org/pub/software/scm/cogito/docs/

[91] **Bezroukov, N.** (1998, desembre). "A second look at the cathedral and the bazaar". *First Monday*, 4(12).

http://www.firstmonday.org/issues/issue4_12/bezroukov/index.html

[92] **Bodnar, L.** (2003). "Linux distributions. Facts and figures".

http://www.distrowatch.com/stats.php?section=packagemanagement

[93] **Boehm, B. W.** (1981). *Software Engineering Economics*. Prentice Hall.

[94] **Bradner, S.** (1996, octubre). "The Internet standards process. Revision 3 (rfc 2026, bcp 9)".

http://www.ietf.org/rfc/rfc2026.txt

[95] **Cederqvist, P.; GNU** (1993). "CVS - concurrent versions system". http://www.gnu.org/manual/cvs/index.html

[96] **Collins-Sussman, B.; Fitzpatrick; B. W.; Pilato, C. M.** (2004). *Version control with Subversion*. O'Reilly & Associates (http://www.ora.com).

http://svnbook.red-bean.com/

[97] **Cunningham, W.** "Wiki design principles".

[98] **Dachary, L.** (2001). "Savannah, the next generation".

http://savannah.gnu.org/docs/savannah-plan.html

[99] **Junta d'Andalusia** (2003, març). Decret 72/2003, de 18 de març, de mesures d'impuls de la societat del coneixement a Andalusia.

http://www.andaluciajunta.es/SP/AJ/CDA/Ficheros/ArchivosPdf/DecretoConocimiento.pdf

[100] **De Boor, A.** *Pmake. A tutorial*. http://docs.freebsd.org/44doc/psd/12.make/paper.html

[101] **De Icaza, M.** "The story of the GNOME Project".

http://primates.ximian.com/~miguel/gnome-history.html

[102] **Senat de la República Francesa**. Forum sur la proposition de loi tendant à généraliser dans.

l'administration l'usage d'Internet et de logiciels libres.

http://www.senat.fr/consult/loglibre/index.htm

[103] **De las Heras Quirós, P.; González Barahona, J. M.** (2000). "Iniciativas de las administraciones públicas en relación al software libre". *Bole. TIC, Revista de ASTIC* (14).

[104] **Debian.** "Debian Free Software Guidelines".

http://www.debian.org/social_contract.html#guidelines

[105] **Debian.** *Debian policy manual*.

http://www.debian.org/doc/debian-policy/

[106] **Debian.** "Debian social contract".

http://www.debian.org/social_contract.html

[107] **Schriftenreihe der KBSt** (2003, juliol). Leitfaden für die migration von basissoftwarekomponenten auf serverund arbeitsplatzsystemen. Technical report, Koordinierungs-und Beratungsstelle der Bundesregierung für Informationstechnik in der Bundesverwaltung (KBSt).

http://www.kbst.bund.de/download/mlf_v1_de.pdf

[108] **DiBona, C.; Ockman, S.; Stone, M.** (ed.) (1999). *Open sources. Voices from the open source revolution*. O'Reilly & Associates.

http://www.oreilly.com/catalog/opensources/

[109] **Open Directory Project**. http://dmoz.org

[110] **Ehrenkrantz, J. R.** (2003, maig). "Release management within open source projects". En: *Proceedings of the 3rd Workshop on Open Source Software Engineering at the 25th International Conference on Software Engineering*.Portland, EE.UU.

[111] **Consejo Europeo** (1991). Directiva 91/250/CEE del Consejo, de 14 de mayo de 1991, relativa a la protección jurídica de programas de ordenador.

http://europa.eu.int/scadplus/leg/es/lvb/l26027.htm

[112] **Feller, J.; Fitzgerald, B; Hissam, S.; Lakhani, K.** (ed.) (2003). *Making sense of the bazaar*. O'Reilly.

[113] **Fogel, K.; Bar, M.** (2001). *Open source code development with CVS* (2ª edición). Paragliph Press.

http://cvsbook.red-bean.com

[114] **Electronic Frontier Foundation.** Open Audio.

http://www.eff.org/IP/Open_licenses/eff_oal.html

[115] **Free Software Foundation.** GPLv3.

http://gplv3.fsf.org

[116] **Free Software Foundation.** LGPLv3. First discussion draft.

http://gplv3.fsf.org/pipermail/info-gplv3/2006-July/000008.html

[117] **Free Software Foundation** (1985): "The GNU Manifesto".

http://www.gnu.org/philosophy/

[118] **Free Software Foundation** (1991, juny). GNU General Public License, version 2. http://www.fsf.org/licenses/gpl.html

[119] **Free Software Foundation** (1999, febrer). GNU Lesser General Public License, version 2.1.

http://www.fsf.org/licenses/lgpl.html

[120] **Free Software Foundation.** "Free software definition".

http://www.gnu.org/philosophy/free-sw.html

[121] **Free Software Foundation.** "Licencias libres".

http://www.gnu.org/licenses/license-list.html

[122] **Garbee, B.; Koptein, H.; Lohner, N.; Lowe, W.; Mitchell, B.; Murdock, I.; Schulze, M.; Small, C.** "A brief history of Debian". En el paquete: *Debian-history*.

[123] **Germán, D.** (2002, maig). "The evolution of GNOME. En: *Proceedings of the 2nd Workshop on Open Source Software Engineering at the 24th International Conference on Software Engineering*. Florida, EE.UU.

[124] **Germán, D.; Mockus, A.** (2003, maig): "Automating the measurement of open source projects". En: *Proceedings of the 3rd Workshop on Open Source Software Engineering at the 25th International Conference on Software Engineering*. Portland, EE.UU.

[125] **Ghosh, R. A.** (1998, març). "Cooking pot markets: an economic model for the trade in free goods and services on the Internet. *First Monday*, 3(3).

http://www.firstmonday.dk/issues/issue3_3/ghosh/index.html

[126] **Ghosh, R. A.; Glott, R.; Krieger, B.; Robles, G.** (2002). *Free/libre and open source software: Survey and study*. Parte iv: "Survey of developers".

http://www.infonomics.nl/FLOSS/report/FLOSS_Final4.pdf

[127] **Ghosh, R. A.; Prakash, V. V.** (2000, juliol). "The orbiten free software survey". *First Monday*, 5(7).

http://www.firstmonday.dk/issues/issue5_7/ghosh/index.html

[128] **Godfrey, M. W.; Tu, Q.** (2000, agost). "Evolution in open source software. A case study". En: *Proceedings of the 2000 International Conference on Software Maintainance*.

[129] **González, J. A.** (2002, març). "Carta al congresista Villanueva".

http://www.gnu.org. pe/mscarta.html

[130] **Goosens, M.; Rahtz, S.** (1999). *The LaTeX Web Companion*. Addison Wesley.

[131] **Grad, B.** (2002, gener-març). "A personal recollection: IBM's unbundling of software and services". En: *IEEE Annals of the History of Computing*, 24(1):64-71.

[132] **Working Group on Libre Software** (1999). "Free software / open source. Information society opportunities for Europe?".

http://eu.conecta.it/paper.pdf

[133] **GrULIC.** "Legislación sobre el uso de software libre en el Estado".

http://proposicion.org.ar/doc/referencias/index.html.es

[134] **Hamerly, J; Paquin, T.; Walton, S.** (1999). "Freeing the source. The story of Mozilla". http://www.oreilly.com/catalog/opensources/book/netrev.html

[135] **Hammel, M. J.** (1991, desembre). "The history of xfree86". *Linux Magazine*.

http://www.linux-mag.com/2001-12/xfree86_01.html

[136] **Harris, S.** (2001, agost). *The Tao of IETF. A novice's guide to the Internet engineering task force* (RFC 3160, FYI 17).

http://www.ietf.org/rfc/rfc3160.txt

[137] **Harrison, P.** (2002). "The rational street performer protocol".

http://www.logarithmic.net/pfh/RSPP

[138] **Hasan, R.** "History of Linux".

http://ragib.hypermart.net/linux/

[139] **Hauben, M.; Hauben, R.** (1997). *Netizens. On the history and impact of Usenet and the Internet*. IEEE Computer Society Press.

[140] **Healy, K.; Schussman; A.** (2003, gener). "The ecology of open source software development". http://opensource.mit.edu/papers/healyschussman.pdf

[141] **Hecker, F.** (1998, maig). "Setting up shop. The business of open-source software".

http://www.hecker.org/writings/setting-up-shop.html

[142] **Hecker, F.** (1998). "Setting up shop. The business of open-source software".

http://www.hecker.org/writings/setting-up-shop.html

[143] **Hertel, G.; Niedner, S.; Herrmann, S.** (2003). "Motivation of software developers in open

source projects. An Internet-based survey of contributors to the Linux kernel".

http://opensource.mit.edu/papers/rp-hertelniednerherrmann.pdf

[144] **Himanen, P.** (2001). *The hacker ethic and the spirit of the information age*. Random House.

http://www.hackerethic.org

[145] **Hunt, F.; Johnson, P.** (2002). "On the Pareto distribution of SourceForge projects. Technical report". Centre for Technology Management, Cambridge University Engineering Department, Mill Lane, Cambridge CB2 1RX.

http://www-mmd.eng.cam.ac.uk/people/fhh10/Sourceforge/Sourceforge%20paper.pdf

[146] **Open Source Initiative.** "History of the OSI".

http://www.opensource.org/docs/history.php

[147] **Hamilton, J. R.** (ambaixador dels EUA a Perú) (2002, juny). "Carta al presidente del Congreso de la República".

http://www.gnu.org.pe/lobbyusa-congreso.html

[148] **Jones, P.** (2000, maig). "Brook's law and open source. The more the merrier?".

http://www-106.ibm.com/developerworks/opensource/library/os-merrier.html?dwzone=opensource

[149] **Jorgensen, N.** "Incremental and decentralized integration in FreeBSD". En: Feller *et al*. [112]. http://www.dat.ruc.dk/~nielsj/research/papers/bazaar-freebsd.pdf

[150] **Brooks, F. P.** (1975). *The mythical man-month. Essays on software engineering*. Addison-Wesley.

[151] **Kalt, C.** (2000, abril). "Internet relay chat: architecture (RFC 2810)".

http://www.ietf.org/rfc/rfc2810.txt

[152] **Kelsey, J.; Schneier, B.** (1998, novembre). "The street performer protocol". En: *Third USENIX Workshop on Electronic Commerce Proceedings*. USENIX Press.

http://www.counterpane.com/street_performer.html

[153] **Kelsey, J.; Schneier, B.** (1999, juny). "The street performer protocol and digital copyrights". *First Monday*, 4(6).

http://www.firstmonday.dk/issues/issue4_6/kelsey/

[154] **Kelty, C. M.** (2001, desembre). "Free software/free science". *First Monday*, 6(12).

http://firstmonday.org/issues/issue6_12/kelty/index.html

[155] **Khatib, J.** "OpenIPCore Hardware General Public License".

http://www.opencores.org/OIPC/OHGPL_17.shtml

[156] **Knuth, D.** (1989). *The TeXbook*. Addison Welsley.

[157] **Koch, S.** (ed.) (2003). *Free/open source software development*. Idea Group Inc.

http://wwwai.wu-wien.ac.at/~koch/oss-book/

[158] **Koch, S.; Schneider, G.** (2000). "Results from software engineering research into open source development projects using public data". En: *Diskussionspapiere zum Tätigkeitsfeld Informationsverarbeitung und Informationswirtschaft, H.R. Hansen und W.H. Janko (Hrsg.), Nr. 22*, WirtschaftsuniversitätWien.

[159] **Kovács, G. L.; Drozdik, S.; Succi, G.; Zuliani, P.** (2004). "Open source software for the public administration". En: *Proceedings of the 6th International Workshop on Computer Science and Information Technologies* (CIST 2004). Budapest, Hungría.

[160] **Krishnamurthy, S.** (2002, maig). "Cave or community? An empirical examination of 100 mature open source projects". *First Monday*, 7(6).

http://www.firstmonday.dk/issues/issue7_6/krishnamurthy/index.html

[161] **Laffitte; Trégouet; Cabanel** (1999). Proposition de loi numéro 495. Senado de la República Francesa.

http://www.senat.fr/consult/loglibre/texteloi.html

[162] **Laffitte; Trégouet; Cabanel** (2000). Proposition de loi numéro 117. Senado de la República Francesa.

http://www.senat.fr/consult/loglibre/texteloi.html

[163] **Lamport, L.** (1994). *LaTeX user's guide and reference manual* (2ª edición). Addison Welsley, Reading, Mass.

[164] **Lancashire, D.** (2001, desembre). "Code, culture and cash. The fading altruism of open source development". *First Monday*, 6(12).

http://www.firstmonday.dk/issues/issue6_12/lancashire/index.html

[165] **Lehman, M. M.; Ramil, J. F; Wernick, P. D.** (1997, novembre). "Metrics and laws of software evolution. The nineties view". En: *Proceedings of the 4th International Symposium on Software Metrics*.

http://www.ece.utexas.edu/~perry/work/papers/feast1.pdf

[166] **Leiner, B. M.; Cerf, V. G.; Kahn, R. E.; Clark, D. D.; Kleinrock, L.; Lynch, D. C.; Postel, J.; Roberts, L. G.; Wolff, S.** (1997). "A brief history of the Internet". En: *Communications of the ACM*.

http://www.isoc.org/internet/history/brief.shtml

[167] **Netcraft Ltd. August 2003 Web Server Survey, 2003.**

http://news.netcraft.com/archives/2003/08/01/august_2003_web_server_survey.html

[168] **Lucovsky, M.** (2000). "From NT OS/2 to Windows 2000 and beyond. A software-engineering odyssey".

http://www.usenix.org/events/usenix-win2000/invitedtalks/lucovsky_html/>

[169] **McGraw, G.** "Building secure software: how to avoid security problems the right way". Citat per: David A. Wheeler en http://www.dwheeler.com/sloc/

[170] **McKusick, M. K.** (1999). "Twenty years of Berkeley Unix. From AT&T owned to freely redistributable". En: DiBona *et al.* [108].

http://www.oreilly.com/catalog/opensources/

[171] **SUN Microsystems** (2000). "Sun microsystems announces availability of StarOffice™ source code on OpenOffice.org".

http://www.collab.net/news/press/2000/openoffice_live.html

[172] **Mockus, A.; Fielding, R. T.; Herbsleb, J. D.** (2000, juny). "A case study of open source software development: the Apache server". En: *Proceedings of the 22nd International Conference on Software Engineering (ICSE 2000)*, páginas 263–272. Limerick, Irlanda. ACM Press.

[173] **Molenaar, B.** "What is the context of charityware?".

http://www.moolenaar.net/Charityware.html

[174] **MT Open Course Ware.**

http://ocw.mit.edu

[175] **Nagel, L. W.** (1996, setembre). "The life of SPICE". En: *1996 Bipolar Circuits and Technology Meeting*. Minneapolis, MN, EE.UU.

http://www.icsl.ucla.edu/aagroup/Life%20of%20SPICE.html

[176] **Narduzzo, A.; Rossi, A.** (2003, maig). "Modularity in action: GNU/Linux and free/ open

source software development model unleashed".

http://opensource.mit.edu/papers/narduzzorossi.pdf

[177] **Newman, N.** (1999). "The origins and future of open source software".

http://www.netaction.org/opensrc/future/

[178] **Nupedia.**

http://www.nupedia.com

[179] **Villanueva Núñez, E.** (2002, abril). "Carta a Microsoft Perú".

http://www.gnu.org.pe/rescon.html

[180] **Danish Board of Technology** (2002, octubre). "Open-source software in e-Government, analysis and recommendations drawn up by a working group under the danish board of technology. Technical report".

[181] **Open Source Initiative.** "Licencias de fuente abierta".

http://www.opensource.org/licenses/index.html

[182] **Pareto, W.** (1896). "Course of Political Economy". Lausanne.

[183] **Perens, P.; The Open Source Initiative** (1998). "The open source definition". http://www.opensource.org/docs/definition_plain.html

[184] **GNU Perú.** "Proyectos ley de software libre en la Administración pública del Gobierno peruano, Congreso de la República".

http://www.gnu.org.pe/proleyap.html

[185] **Pinheiro, P.** (1999, desembre). Proposição pl-2269/1999: Dispõe sobre a utilização de programas abertos pelos entes de direito público e de direito privado sob controle acionário da administração pública. Câmara dos Deputados do Brasil.

http://www.camara.gov.br/Internet/sileg/Prop_Detalhe.asp?id=17879

http://www.fenadados.org.br/software.htm

[186] **Pranevich, J.** (2003). "The wonderful world of Linux 2.6".

http://www.kniggit.net/wwol26.html

[187] **The Debian Project.** "Debian developer map".

http://www.debian.org/devel/developers.loc

[188] **Puigcercós Boixassa, J.** (2002). Proposición de Ley de Medidas para la Implantación del Software Libre en la Administración del Estado.

http://www.congreso.es/public_oficiales/L7/CONG/BOCG/B/B_244-01.PDF

[189] **Quittner, J.; Slatalla, M.** (1998). *Speeding the net: the inside story of Netscape and how it challenged Microsoft*. Atlantic Monthly Pr.

[190] **Rasch, C.** "A brief history of free/open source software movement".

http://www.openknowledge.org/writing/open-source/scb/brief-open-source-history.html

[191] **Rasch, C.** (2001, maig). "The Wall Street performer protocol. Using software completion bonds to fund open source software development". *First Monday*, 6(6).

[192] **Raymond, E. R.** (2001, gener). *The cathedral and the bazaar. Musings on Linux and open source by an accidental revolutionary*. O'Reilly & Associates (http://www.ora.com).

http://catb.org/~esr/writings/cathedral-bazaar/

[193] **Reis, C R.; De Mattos Fortes, R. P.** (2002, febrer). "An overview of the software engineering process and tools in the Mozilla Project".

http://opensource.mit.edu/papers/reismozilla.pdf

[194] **Rideau, F. R.** (2000). "Patents are an economic absurdity".

http://fare.tunes.org/articles/patents.html

[195] **Roberts, L.** (1978, novembre). "The evolution of packet switching". *Proceedings of the IEEE*, (66).

[196] **Robles, G.; González Barahona, J. M.; Centeno González, J.; Matellán Olivera, V.; Rodero Merino, L.** (2003, maig). "Studying the evolution of libre software projects using publicly available data". En: *Proceedings of the 3rd Workshop on Open Source Software Engineering at the 25th International Conference on Software Engineering*. Portland, EE.UU.

[197] **Robles, G.; Scheider, H.; Tretkowski, I.; Weber, N.** (2001): "Who is doing it? Knowing more about libre software developers".

http://widi.berlios.de/paper/study.pdf

[198] **Rochkind, M.** (1986, maig). "Interview with Dick Haight". *Unix Review*.

[199] **Scacchi, W.** (2003). "Understanding open source software evolution. Applying, breaking and rethinking the laws of software evolution".

http://www.ics.uci.edu/~wscacchi/Papers/New/Understanding-OSS-Evolution.pdf

[200] **Schneier, B.** (2000). "Software complexity and security".

http://www.counterpane.com/crypto-gram-0003.html

[201] **Smoogen, S. J.** "The truth behind Red Hat names".

http://www.smoogespace.com/documents/behind_the_names.html

[202] **Haggen So.** "Comparison of free/open source hosting (FOSPhost) sites available for hosting projects externally from project owners".

http://www.ibiblio.org/fosphost/exhost.htm

[203] **Stallman, R.** "GNU coding standards".

http://www.gnu.org/prep/standards.html

[204] **Stallman, R.** "Why *free software* is better than *open source*".

http://www.fsf.org/philosophy/free-software-for-freedom.html

[205] **Stallman, R.** (1998). "Copyleft: pragmatic idealism".

http://www.gnu.org/philosophy/pragmatic.html

[206] **Stallman, R.** (1998). "Why *free software* is better than *open source*".

http://www.gnu.org/philosophy/free-software-for-freedom.html

[207] **Stallman, R.** (1998). "Why software should not have owners".

http://www.gnu.org/philosophy/why-free.html

[208] **Stallman, R.** "The GNU Project". En: DiBona *et al.* [108].

http://www.fsf.org/gnu/thegnuproject.html

[209] **Stallman, R.** (1999, juny). "On free hardware". *Linux Today*.

http://features.linuxtoday.com/news_story.php3?ltsn=1999-06-22-005-05-NW-LF

[210] **Stallman, R.** (2001). "The free universal encyclopedia and learning resource".

http://www.gnu.org/encyclopedia/free-encyclopedia.html

[211] **Stallman, R.** (2002). *Free software, free society. Selected essays of Richard M. Stallman*. Joshua Gay.

[212] **Stallman, R.** (2003). "Some confusing or loaded words and phrases that are worth avoiding".

http://www.gnu.org/philosophy/words-to-avoid.html

[213] **Stoltz, M.** (1999). "The case for government promotion of open source software".

http://www.netaction.org/opensrc/oss-report.html

[214] **Tanenbaum, A.; Torvalds, L.** (1999). "The Tanenbaum-Torvalds debate".

http://www.oreilly.com/catalog/opensources/book/appa.html

[215] **The Open Source Initiative.** "The open source definition".

http://www.opensource.org/docs/definition_plain.html

[216] **Tiemann, M.** "Future of Cygnus Solutions. An entrepreneur's account". En: DiBona *et al*. [108].

http://www.oreilly.com/catalog/opensources/book/tiemans.html

[217] **Torvalds, L; Diamond; D.** (2001). *Just for fun: the story of an accidental revolutionary*. Texere.

[218] **Linus Torvalds, Hamano, J. C.; Ericsson, A.** "Git manual page".

http://www.kernel.org/pub/software/scm/git/docs/

[219] **Tuomi, I.** (2002). "Evolution of the Linux credits file: methodological challenges and reference data for open source research".

http://www.jrc.es/~tuomiil/articles/EvolutionOfTheLinuxCreditsFile.pdf

[220] **Autors diversos** "Carta abierta al director de la WIPO".

http://www.cptech.org/ip/wipo/kamil-idris-7july2003.pdf

[221] **Vigo i Sallent, P.; Benach i Pascual, E.; Huguet i Biosca; J.** (2002, maig). Proposició de llei de programari lliure en el marc de l'Administració pública de Catalunya.

http://www.parlament-cat.es/pdf/06b296.pdf

http://www.hispalinux.es/
modules.php?op=modload&name=Sections&file=index&req=viewarticle&artid=49

[222] **Villanueva Núñez, E.** (2001, desembre). Proyecto de ley software libre, número 1609.

http://www.gnu.org.pe/proley1.html

[223] **Villanueva Núñez, E.; Rodrich Ackerman, J.** (2002, abril). Proyecto de ley de uso de software libre en la Administración pública, número 2485.

http://www.gnu.org.pe/proley4.html

[224] **W3C** (2000). *Extensible markup language (xml) 1.0* (2a. edició).

[225] **Walsh, N.; Muellner, L.; Stayton, B.** (2002). *DocBook: the definitive guide*. O'Reilly.
http://docbook.org/tdg/en/html/docbook.html

[226] **Welke, L; Johnson, L.** (1998). How the ICP Directory began.

http://www.softwarehistory.org/history/Welke1.html

[227] **Wheeler, D. A.** (2000, juliol). "Estimating Linux's size".

http://www.dwheeler.com/sloc

[228] **Wheeler, D. A.** (2001, juny). "More than a gigabuck: estimating GNU/Linux's".

http://www.dwheeler.com/sloc

[229] **Wiesstein, E.** "Concise encyclopedia of mathematics".

http://mathworld.wolfram.com/

[230] **Wikipedia.** "Gini coefficient".

http://www.wikipedia.org/wiki/Gini_coefficient

[231] **Wikipedia.** "Lorenz curve".

http://www.wikipedia.org/wiki/Lorenz_curve

[232] **Wikipedia.** "Pareto".

http://www.wikipedia.org/wiki/Pareto

[233] **Wikipedia.** "TeX".

http://www.wikipedia.org/wiki/TeX

[234] **Wilson, B.** "Netscape Navigator".

http://www.blooberry.com/indexdot/history/netscape.htm

[235] **Computer World** (2000). "Salary survey 2000".

http://www.computerworld.com/cwi/careers/surveysandreports

[236] **Young, R**. (1999). "Giving it away. how Red Hat software stumbled across a new economic model and helped improve an industry".

http://www.oreilly.com/catalog/opensources/book/young.html

[237] **Zawinsky, J. W.** (1999). "Resignation and postmortem".

http://www.jwz.org/gruntle/nomo.html