UNIVERZA V LJUBLJANI

FAKULTETA ZA UPRAVO

Diplomsko delo

univerzitetnega programa

ODPRTA KODA TER MOŽNOSTI NJENE UPORABE V JAVNEM SEKTORJU

Matija Koncilja

Ljubljana, september 2009

UNIVERZA V LJUBLJANI

FAKULTETA ZA UPRAVO

Diplomsko delo

univerzitetnega programa

ODPRTA KODA TER MOŽNOSTI NJENE UPORABE V JAVNEM SEKTORJU

Kandidat: Matija Koncilja

Številka indeksa: 04034413

Mentor: doc. dr. Ljupčo Todorovski

Ljubljana, september 2009

POVZETEK

Diplomsko delo govori o fenomenu odprte kode. Vsebina je razdeljena na tri osrednja poglavja. V prvem sta podrobno opisana zgodovina in razvoj odprte kode od začetka do danes. Predstavljeni so ključni dogodki ter osebe neposredno zaslužne za njen razvoj in uspeh. V drugem poglavju je opisan proces nastajanja odprtega programja, ključne razlike med procesi razvoja lastniške in odprte programske opreme ter načini sodelovanja in komuniciranja med razvijalci. Zadnje poglavje govori o uporabi odprtokodnih rešitev na območju Slovenije in Evrope tako v zasebnem kot tudi javnem sektorju. Predstavljene so vidnejše organizacije odgovorne za razmah in prepoznavnost tega pojava v Sloveniji. Podani so tudi primeri uporabe odprtih aplikacij na področju javnega sektorja iz nekaterih evropskih držav.

Ključne besede: odprta koda, odprto programje, prosto programje, brezplačno programje, lastniško programje, UNIX, Linux, Open Source Initiative, Free Software Foundation, licenciranje odprte kode, modeli trženja odprte kode, Center odprte kode Slovenije

SUMMARY

The thesis describes the phenomenon called open source. The content ot the thesis is organized in four main sections. The first one describes history and evolution of open source from its beginnings until nowadays. It presents the key events and persons directly related to its development and success. Second section is about the process of creating open source software and emphasizes main differences between the processes of development of proprietary and open source software. It also highlights methods of cooperation and communication among open source developers. The last sections is about usage of open source solutions in Slovenia, covering the public and private sector. It presents the most noticeable organizations responsible for its expansion and recognition in Slovenia. It also provides some examples of open source aplications in the public sectors domains in several European countries.

Key words: open source, open software, free software, freeware, proprietary software, UNIX, Linux, Open Source Initiative, Free Software Foundation, open source licensing, business models for open source, Open source center Slovenia.

KAZALO

1 UVOD

V diplomski nalogi sem se posvetil opisovanju fenomena odprte kode od njenega nastanka dalje. Sam izraz »odprta koda« je relativno mlad in se je uveljavil šele leta 1998, na srečanju vodilnih odprtokodnih razvijalcev v Kaliforniji. Pred tem datumom so bili v uporabi številni drugi izrazi. Ključna razlika med odprtim in zaprtim programjem je dostopnost izvorne kode (angl. *source code*) programa. Izvorna koda je tisti del programa, ki ga napiše človek, programer v enem izmed programskih jezikov (Java, C ++ ipd.). Torej je izvorna koda ljudem (programerjem) s primernim znanjem tudi berljiva. Izvorno kodo programa je pred uporabo potrebno pretvoriti v strojno kodo (angl. *machine code*), s katero lahko nato operira računalnik. Strojna koda je zapisana v strojnem jeziku in je človeku nerazumljivo zaporedje ničel in enk, ki ni dovzetno za spremembe oziroma nadaljnji razvoj. Proces pretvorbe oziroma prevoda iz izvorne v strojno kodo se opravi s pomočjo posebnih programov, znanih pod imenom prevajalniki (angl. *compilers*).

V diplomskem delu se pogosto pojavljata izraza odprtokodno programje oziroma odprta koda (angl. *open source*) in prosto programje (angl. *free software*). V tehničnem smislu bistvenih razlik med njima ni, obe vrsti programov morata biti izdani skupaj z izvorno kodo, katera omogoča spreminjanje (nadaljnji razvoj) programske opreme in posledično distribucijo. Razlika se pojavi predvsem v načelih, ki stojijo za obema idejama.

Za izrazom prosto programje stoji organizacija Free Software Foundation (FSF, 2009), ki zagovarja tezo, da je program predvsem odraz ustvarjalčeve kreativnosti in dojema lastniško (oziroma »zaprto-kodno«) programsko opremo za nemoralno saj omenjeno kreativnost omejuje. Izraz »prosto« je potrebno razumeti v smislu svobode posameznika, ne pa nujno brezplačnosti.

Odprta koda pa je izraz, ki je bil iznajden pod okriljem organizacije Open Source Initiative (OSI, 2009). Za razliko od FSF-ja organizacija deluje na bolj pragmatičnih temeljih. Lastniške programske opreme ne dojemajo kot moralno sporne temveč raje poudarjajo prednosti odprtokodnega programja v primerjavi z lastniškimi. Zavzemajo se tudi za širitev in uporabo odprtokodnih licenc ker verjamejo, da le-te pospešujejo razvoj odprtokodnih projektov.

Obe omenjeni gibanji sta torej ideološko ločeni vendar kljub temu sodelujeta pri nekaterih projektih. V nadaljevanju diplomske naloge sta izraza odrta koda in prosto programje uporabljana kot sopomenki.

Še en izraz, ki se v slovenščino lahko prevaja kot prosto programje, je freeware (Freeware definition, 2009). Izraz je v praksi nasproten free software-u. Ustreznejši prevod predstavlja besedna zveza brezplačno programje. V primeru freeware programja namreč avtor omogoči brezplačno uporabo programa, hkrati pa običajno obdrži pravice do spreminjanja in distribucije. Izvorna koda takšnemu programu običajno torej ni priložena. Licence, ki določajo možnost brezplačne uporabe pa so

lahko tudi v primeru brezplačnega programja različne, nekatere bolj omejujoče kot druge.

Nasproti odprtim oziroma prostim programom stoji klasična, lastniška, zaprto-kodna ali tudi komercialna programska oprema. Kot smo pojasnili zgoraj, poglavitna razlika med lastniško in odprto programsko opremo je torej v dostopnosti izvorne kode. Avtorji lastniških programov izvorno kodo varujejo in nam prodajo zgolj strojno kodo oziroma pravico za uporabo strojne kode. Po drugi avtorji odprtih programov poleg izvršljive strojne kode distribuirajo tudi izvorno in tako slehernemu posamezniku omogočijo spreminjanje in prilagajanje programa.

Jedro diplomske naloge je razdeljeno na tri poglavja. V prvem sta opisana zgodovina in razvoj odprte kode, vse od nastanka do njene dokončne uveljavitve. Poseben poudarek je podan pri razlagi vpliva interneta na njen razmah. Podrobno so predstavljeni najpomembnejši dogodki pri razvoju odprte kode ter njihovi nosilci. To so različne izvedbe operacijskih sistemov UNIX (angl. *Uniplexed Information and Computing System,* Zgodovina UNIX-a, 2009) in Linux (Zgodovina Linux, 2009) ter organizaciji Free Software Foundation in Open Source Initiative.

V drugem poglavju so podrobno opisani procesi ustvarjanja odprte kode, nosilci tega ustvarjanja, organizacija ustvarjalnega dela ter komunikacija med udeleženimi v procesu. Opisane so osnovne razlike med razvojem klasičnih (lastniških) ter odprtih programov, glavne smernice, ki naj bi jih razvijalci upoštevali pri ustvarjanju ter vplivi tehnologije in licenc na ustvarjalni proces.

V zadnjem poglavju je opisana izraba odprtih standardov na področju Slovenije in Evropske Unije ter odnos slovenske politike do uvedbe odprtih programov v javni sektor. Opisana so vidnejša gibanja za razvoj odprte kode na področju Slovenije ter prizadevanja vlade za uvedbo le-te v javni in zasebni sektor. Izpostavljeni pa so tudi nekateri uspešni primeri rabe odprte kode na področju Evropske Unije.

Pri pisanju prvih treh poglavij sem se večinoma zgledoval po knjigi Stevena Webra, »The Succes of Open Source« oziroma »Uspeh odprte kode« (Weber, 2009). Weber je predavatelj politologije na kalifornijski univerzi Berkeley v Združenih državah Amerike (ZDA), ki tudi sicer velja za enega izmed centrov nastanka odprte kode. Weber v knjigi podaja svoja mnenja in teorije vendar se pri tem mnogokrat opira na dela ostalih poznavalcev področja, predvsem računalničarjev in raziskovalcev, ki so aktivno sodelovali v razvoju računalništva in odprte kode. Izpostavitev tega dejstva se mi zdi zlasti pomembna zato, ker je ta diplomska naloga z vidika uporabljenih virov in literature nekoliko specifična, saj se večinoma torej zanaša na omenjeno Webrovo knjigo ter spletne vire.

2 ZGODOVINA ODPRTE KODE

Povsem na začetku razvoja računalništva nismo mogli razlikovati med programsko opremo (angl. *software*) oziroma strojno opremo (angl. *hardware*). Ni bilo niti razlik med programerji, uporabniki in razvijalci strojne opreme; obstajali so le posamezniki, ki so z računalniki delali. Na samem začetku je obstajalo zgolj stikalo, ki je poznalo vklopljeno ter izklopljeno stanje, torej stanje nič ali ena. Več takšnih stikal skupaj je tvorilo prvi dvojiški računalnik. Poleg tega, da so bili prvi računalniki dragi, veliki in okorni, tudi niso bili široko uporabni. Podjetje International Business Machines Corporation (IBM) je leta 1952 na trg poslalo prvi komercialni računalnik, imenovan 701. Govorimo o veliko ožjem trgu kot tistemu, ki ga poznamo danes. Samo strošek mesečnega najema te naprave je znašal 15 tisoč ameriških dolarjev, medtem ko je leta 1953 povprečna cena znašala neverjetnih 1,6 milijona dolarjev. Zaradi visoke cene si je nakup oziroma najem lahko privoščilo le ameriško obrambno ministrstvo, računalnik 701 so zato poimenovali kar Obrambni računalnik. Največja težava tega računalnika, poleg visoke cene, je bila pomanjkanje ustrezne programske opreme ter navodil, kaj naj računalnik dejansko počne. Ljudje, ki so se ukvarjali s pisanjem kode, niso imeli prav nobenih pripomočkov, ki bi jim delo olajšali. Pričeti so morali iz nič, programska koda pa je bila že takrat zelo obsežna. Samo za program, ki je vračal dinamično radarsko sliko je bilo potrebno spisati 80 tisoč vrstic navodil oziroma programske kode. (Weber, 2004, str. 20-22)

Zaradi teh težav in kompleksnosti procesa razvoja programske kode, so se inženirji odločili, da zberejo skupaj vse ljudi, ki so kdajkoli in kjerkoli uporabljali ta računalnik. Skupaj naj bi sestaviti orodje, ki bo omogočalo bolj enostavno uporabo računalnika. Tako je nastal projekt PACT (angl. *Project for the Advancement of Coding Techniques*) v katerem so bile združene vse večje zahodnoameriške družbe, ki so bile pogodbeno vezane z ameriškim obrambnim ministrstvom. Projekt se je ukvarjal z razvojem bolj naprednih tehnik za generiranje in razvoj programske kode.

Sam razvoj orodij za uspešnejšo uporabo računalnikov ne bi bil dovolj za informacijsko revolucijo, ki je sledila in smo ji priča še danes. Drugi predpogoj je bil občutno znižanje cene računalniške opreme. Na začetku tega trenda nižanja cen je ključno vlogo odigralo podjetje Digital Equipment Corporation (DEC) skupaj z ustanoviteljem Kenom Olsenom. Ta je zavračal IBM-ov elitistični odnos do računalnikov kot dragih naprav ki naj bodo na voljo le nekaj izbranim posameznikom. DEC je leta 1960 podkrepil svojo paradigmo z izdelavo računalnika imenovanega PDP-1, ki je bil na voljo za približno 120 tisoč dolarjev. Leta 1965 so pri DEC izdali še različico PDP-8, ki je bila na voljo že za razmeroma ugodnih 18 tisoč dolarjev. Resnično inovacijo pa je predstavljala izvedenka iz leta 1970, to je prvi miniračunalnik, poimenovan PDP-11 (PDP-11, 2009). Ta je z začetno ceno 11 tisoč dolarjev in zadovoljivo zmogljivostjo bil zanimiv tudi za univerze in razne raziskovalne oddelke nekaterih korporacij in podjetij.

2.1 POJAV OPERACIJSKEGA SISTEMA UNIX

Vse večja razpoložljivost in priljubljenost računalnikov je vedno bolj klicala po iznajdbi učinkovitih in nezapletenih orodij za delo z njimi. Poleg projekta PACT je bilo še nekaj poizkusov združevanja računalniških uporabnikov, da bi ustvarili uspešen operacijski sistem. Vsi poizkusi so se izkazali za neuspešne, končni izdelek je bil običajno preveč kompliciran, da bi bil tudi dejansko uporaben (Weber, 2004, str. 23-28).

Preboj na tem področju se je zgodil leta 1969. V tem letu je računalniški znanstvenik Ken Thompson (Ken Thompson, 2009) v zgolj štirih tednih na svojem domu spisal program, ki ga je poimenoval Uniplexed Information and Computing Services oziroma UNICS, kasneje preimenovan v UNIX. Podvig mu je uspel klub dejstvu, da je uporabljal v tem času že zastarel DEC-ov računalnik PDP-7. Thompsonu je tako v mesecu dni uspelo to, kar prej ni organizirani skupini ljudi z razmeroma obsežnim znanjem in izkušnjami ter dostopom do precej modernejših računalnikov. To je še danes eno izmed osrednjih vodil razvoja odprte kode. Namesto velikih zapletenih projektov se je potrebno osredotočiti na osnove, na preproste stvari, ki kasneje prerastejo v kompleten končni izdelek. UNIX s svojo filozofijo razvoja lahko označimo kot osrednji del tako ekonomskega kot tudi kulturnega razvoja odprte kode ki jo poznamo danes.

Ken Thompson je nadaljeval z razvojem UNIX-a skupaj s sodelavcem Dennisom Ritchie-jem (Dennis Ritchie, 2009). Oba sta bila zaposlena v podjetju Bell Telephone Laboratories (BTL) kjer sta že predhodno sodelovala na nekaterih ponesrečenih projektih razvoja operacijskih sistemov. Leta 1970 sta na voljo dobila nov računalnik PDP-11/20 skupaj s priloženim DEC-ov operacijskim sistemom. Le tega nista nikoli uporabila, na računalnik sta naložila Thompsonov UNIX in sistem vzpostavila presenetljivo hitro. Za potrebe ostalih raziskovalcev podjetja BTL sta morala iznajti tudi preprost urejevalnik besed. To jima ni povzročalo pretiranih preglavic, rezultat je bil pravzaprav tako dober, da sta Thompson in Ritchie dobila na voljo še močnejši računalnik za potrebe njunih raziskav.

Naslednja ovira na poti popularizacije operacijskega sistema UNIX je bilo pomanjkanje priročnika oziroma navodil, kako z njim upravljati. Do tedaj so uporabniki vse, kar so želeli vedeti o UNIX-u izvedeli tako, da so program preizkušal ali kar neposredno vprašali njegovega avtorja. Zahteve po pisni dokumentaciji so se stopnjevale, še zlasti zato, ker se je izkazalo da utegne biti UNIX mnogo več kot le igrača. Prva izdaja priročnika za uporabo je ustalila še eno pomembno tradicijo UNIX-a in odprte kode: navedbo vsakega podprograma z njegovim lastnikom oziroma avtorjem, ki je načeloma zadolžen za njegov nastanek in vzdrževanje.

UNIX se je torej počasi širil, do začetka leta 1973 je doživel 16 namestitev v sklopu družbe AT&T (angl. *American Telephone & Telegraph*), največjega ponudnika telefonskih storitev v Združenih Državah Amerike, katerega del je bilo tudi podjetje BTL.

Naslednja pomembna inovacija UNIX-a je bil koncept »cevi« (angl. *pipes*), ki je bil namenjen povezavi produkta (izhoda) enega programa z izhodiščem (vhodom)

nekega drugega programa (angl. *output - input* povezava). Gre za ključen element t.i. modularnosti, ki omogoča da neko kompleksno in zapleteno računalniško operacijo razdelimo na več manjših in bolj enostavnih, ki jih je lažje razrešiti. Ideja modularnosti je privedla do razmišljanja, da programska oprema ni zgolj orodje za dosego določenega cilja temveč kar nekakšna škatla, ki vsebuje mnogo različnih orodij. Uporabniki UNIX-a nanj niso več gledali kot na integriran operacijski sistem, temveč raje kot množico različnih modulov, s katerimi lahko nato sestavijo neko kompleksnejšo funkcijo. Zasnova cevi je privedla tudi to treh glavnih zapovedi UNIX-a:

- Vsak napisan programi naj opravlja eno stvar vendar naj to stvar opravi dobro;
- Napisani programi naj bodo med seboj združljivi;
- Napisani programi naj bodo sposobni obdelati tekstovne nize saj so le-ti univerzalni vmesnik.

Izhod vsakega programa je torej tekstovni niz, ki ga lahko drugi programi uporabijo kot svoj vhod.

2.2 ŠIRJENJE OPERACIJSKEGA SISTEMA UNIX

Na začetku je tako UNIX imel zelo ozko skupino uporabnikov znotraj AT&T, ni pa imel nekega širšega občinstva. To se je spremilo leta 1973, ko sta Thompson in Ritchie predstavila UNIX zbranim računalniškim strokovnjakom zbranim na ACM (angl. *Association for Computer Machinery*, ACM, 2009). Od tega trenutka dalje je bilo zanimanje za njun operacijski sistem izjemno, na naslov BTL-a pa se je usulo ogromno prošenj za dobavo posameznih kopij (Weber, 2004, str. 28-29).

Zdelo se je, da je bila to izjemna poslovna priložnost za AT&T toda temu ni bilo tako. Že leta 1956 je namreč pravosodno ministrstvo v ZDA izdalo odlok, katerega vsebina je preprečevala podjetjema Western Electric ter AT&T udejstvovanje na trgih, ki niso povezani z njunima primarnima dejavnostma, telefonom in telegrafom. Ker prodaja računalniške programske opreme ne sodi v to kategorijo, se je moralo podjetje AT&T trženju UNIX-a odreči. Dejansko odlok ministrstva ni bil tako zavezujoč, saj je dopuščal več manevrskega prostora okrog ostalih dejavnosti omenjenih podjetij, kot so to dojeli večinoma konservativni pravniki zaposleni pri AT&T.

V praksi je to torej pomenilo, da AT&T programske opreme ne sme prodajati, oziroma z njeno prodajo ustvarjati dobička. Odlok pa z ničemer ni omejeval distribucije programske opreme in zaračunavanja stroškov nastalih z njo. UNIX so tako patentirali in licencirali, licenca pa je vsebovala minimalna določila. V njej je bilo med drugim zapisano, da AT&T operacijski sistem dostavi takšnega kot je, brez kakršnekoli podpore uporabnikom ali popravkov napak oziroma hroščev. AT&T Se je tako izognil nesoglasjem z ameriško zakonodajo, njihov način licenciranja pa je imel še vsaj eno pomembno posledico. Uporabniki UNIX-a so se namreč zavedli, da kakšne uradne podpore s strani proizvajalca ne bodo deležni, zato so pričeli

sodelovati med seboj in si izmenjavali izkušnje ter popravke za napake, ki so jih odkrili in ustvarili med svojim delom.

Širitev in razvoj UNIX-a sta bila predvsem zelo dobrodošla za univerzitetne računalniške oddelke. Za samo nekaj 100 ameriških dolarjev so univerze dobile operacijski sistem skupaj z njegovo izvorno kodo. Program je bilo mogoče naložiti na večino tedanjih računalnikov in ga prosto uporabljati za učenje, spreminjanje ter eksperimentiranje. Tako ni presenetljivo, da se je ena izmed pomembnejših smernic razvoja UNIX-a in odprte kode nadaljevala na ameriških fakultetah, tu predvsem izstopa Berkeley.

Do leta 1976 se je UNIX razširil v več kot štirideset inštitucij samo v ZDA. Poleg tega je imel aktivno bazo uporabnikov v Angliji, Avstraliji ter na Japonskem. Uporabniki so se predvsem na nacionalni ravni redno srečevali na različnih srečanjih in si izmenjevali izkušnje ter znanja in dosežke povezane z operacijskim sistemom.

Takšna srečanja so bila za obstoj in nadaljnjo širitev UNIX-a ključna. Ta operacijski sistem se je namreč lahko širil le na najbolj tradicionalne možne načine, torej na podatkovnih kasetah, diskih in ostalih fizičnih nosilcih podatkov. Leta 1969 je sicer bil vzpostavljen projekt ARPANET (angl. *Advanced Research Projects Agency Network*, Hauben, 2009), servis namenjen izmenjavi podatkov in prednik današnjega interneta. Od leta 1970 dalje je bila elektronska pošta že razmeroma razširjeno sredstvo komunikacije. Težava je bila v tem, da na voljo ni bilo vmesnika, ki bi omogočal izmenjavo podatkov preko elektronske pošte na računalnikih z naloženim UNIX operacijskim sistemom. Prvi tak vmesnik se je pojavil leta 1976 in to težavo tudi odpravil.

2.3 UNIX NA UNIVERZI BERKELEY

Kot sem že omenil zgoraj, so profesorji ter študenti na kalifornijski univerzi Berkeley prispevali pomemben delež k razvoju UNIX-a. Začetnik in pobudnik uvedbe UNIX-a na Berkeleyu je bil profesor računalniške znanosti, Robert Fabry. Leta 1973 so na univerzi zato tudi nabavili svoj prvi računalnik, DEC-ov PDP-11/45 ter nanj prvič v zgodovini brez pomoči avtorja naložili operacijski sistem UNIX. Ta namestitev je položila temelje za uspešno in dolgotrajno sodelovanje med AT&T ter univerzo Berkeley.

Tudi v Kaliforniji sprva ni šlo brez težav. Največji problem je bil v tem, da so si edini računalnik morali deliti trije različni oddelki, za računalništvo, matematiko in statistiko. Medtem ko so želeli računalničarji delati z UNIX-om, pa so matematiki in statistiki hoteli uporabljati DEC-ov operacijski sistem. Zaradi tega je bilo potrebno uvesti osem urne izmene in pogostokrat se je zgodilo, da je računalnik poganjal UNIX operacijski sistem samo sredi noči. Toda študenti in profesorji se na to niso posebno ozirali kar samo dodatno potrjuje, kakšen fenomen je bil Thompsonov operacijski sistem v tistem času.

UNIX se je nato počasi pričel razširjati tudi po Berkeleyu. Najprej sta ga pričela uporabljati še dva profesorja, ki sta se ukvarjala z izdelavo svoje baze podatkov. Dobila sta tudi pooblastila za nabavo več računalnikov, med njimi tudi zelo zmogljiv DEC-ov PDP-11/70.

Leto 1975 pa je bilo za uveljavitev UNIX-a na univerzi Berkely ključno. Spomladi je namreč skupaj z novim računalnikom PDP-11/70 v Kalifornijo v vlogi gostujočega profesorja prišel tudi Ken Thompson. S seboj je prinesel tedaj že šesto različico UNIX-a, ki so jo nemudoma pričeli uporabljati. Prav tako sta v tistem času bila na Berkeleyu prisotna dva študenta, Chuck Haley in Bill Joy (Bill Joy, 2009). Pričela sta z delom na Pascalovem prevajalniku (angl. *compiler*), ki ga je prav tako s seboj prinesel Thompson. Omenjeni prevajalnik je služil kot vmesnik za pretvorbo izvorne kode v strojno kodo. Rezultate njunega dela so kmalu pričeli uporabljati na vsej univerzi.

Ko je Thompson poleti 1976 zapusti Berkeley pa sta Joy in Haley pričela z delom na samem jedru UNIX-a in izboljšave ter predloge glede izboljšanja same zmogljivosti operacijskega sistema posredovala Bellovim laboratorijem.

Novice o dosežkih predvsem Billa Joya pa so se razširile tudi v ostale dele UNIX-ove skupnosti po Ameriki in svetu. Ker so tudi na Berkeley pričele prihajati zahteve po njegovih izdelkih, je Joy sestavil paket programskih orodij in pripomočkov (med njimi npr. tudi zgoraj omenjeni prevajalnik Pascal), ki ga je poimenoval Berkeley Software Distribution (BSD, 2009). Brezplačno je razdelil približno 30 takšnih paketov orodij.

Peter Salus, računalniški znanstvenik navaja, da gre v tem primeru za ključne elemente kulture sodelovanja, kot tudi za preprost mehanizem distribucije programske opreme in njenega nastajanja kar pojasni tudi s primerom:

»V Bellovih tehnoloških laboratorijih so ustvarili računalniški program. Program so distribuirali v izvorni obliki. Potem je ta program vzel nekdo iz Velike Britanije in ga nekoliko spremenil, nastal je nekoliko drugačen program. Nato je nekdo tretji prevzel prvotni in spremenjeni program ter iz njiju ustvaril nekaj novega. To novost (program) je nato ponudil skupnosti v zameno za plačilo, hkrati pa je bila vključena tudi v naslednjo izdajo podjetja Bell.« (Salus v: Weber, 2004, str. 31)

Tudi Berkeleyev programski paket, torej BSD, je šel skozi podoben cikel. Medtem ko je skupnost določene stvari spreminjala in izboljševala pa Bill Joy ni miroval in je s pomočjo svojih pomočnikov tudi sam iskal izboljšave ter inovacije. Tako je s številnimi izboljšavami leta 1978 izšla druga izdaja BSD, bolj znana pod imenom 2BSD, ki je bila obsežnejša kot prva[1].

Naslednji pomemben korak v razvoju BSD-ja se je zgodil leta 1979 in je bil posledica bolj ali manj nenamernega sodelovanja med AT&T ter Berkeleyem. Ko so pri Bell

[1] Na tem mestu naj poudarim da sta bili izdaji BSD in 2BSD skupek programskih orodij in pripomočkov, ne pa samostojna operacijska sistema.

Labs izdali že svojo sedmo različico UNIX-a je ta sicer mnoge navdušila z vsemi novostmi, je pa zato bolj trpela njegova zmogljivost oziroma hitrost delovanja v primerjavi z verzijo 6.

Tako so se mnogi raziskovalci, vključno Bill Joy s sodelavci, takoj lotili predelave izvorne kode sedme različice UNIX-a. Že nekaj časa je bilo tudi jasno, da k razvoju ne prispevajo samo raziskovalci in profesorji, pač pa poleg študentov celo nekateri srednješolci iz različnih strokovno usmerjenih šol povezanih z računalništvom. Verjetno pa je najpomembnejši aspekt dela na sedmi verziji UNIX-a predstavljala njega predelava na večino računalniških sistemov, ki so bili na voljo v tem obdobju. V to kategorijo spada tako predelava, ki je omogočala delovanje v navezi z Intelovimi čipi 8086 (Intel 8086, 2009), ki so tedaj domovali v osrčju večine osebnih računalnikov. Zraven spada tudi predelava za Motoroline sisteme, ki so jih med drugim takrat uporabljali pri podjetju Apple. Med najpomembnejše preobrazbe UNIX-a pa zagotovo spada tudi predelava, poimenovana UNIX 32V (UNIX/32V, 2009). To izvedenko UNIX-a je bilo mogoče naložiti na najnovejše računalnike podjetja DEC imenovane VAX-11/780, enega izmed njih si je lastila tudi univerza Berkeley.

Smiselno je omeniti da vodilni možje podjetja DEC, na čelu s Kenom Olsenom, niso bili najbolj naklonjeni ideji predelave UNIX-a, predvsem ne na njihovo najnovejšo napravo, VAX-11/780. Podjetje DEC je namreč imelo več svojih lastniških operacijskih sistemov in je tako smatralo UNIX kot konkurenco. Kljub tem pomislekom so DEC-ovi inženirji le dobili dovoljenje, da predelavo tudi dejansko uresničijo. Pri tem so naleteli na nekaj težav, tako da so se morali po pomoč zateči k Bellovim inženirjem. S skupnimi močmi jim je uspelo izdelati medtem že težko pričakovano predelavo 32V, ki jo je nujno potrebovalo že precej fakultet.

Vse te aktivnosti, povezane z UNIX-om, kot tudi njegova čedalje večja prepoznavnost so v vrste AT&T-ja pričele vnašati nemir. Vedno bolj so se namreč zavedali, da je Bellova programska oprema, na čelu z UNIX-om izvrstna poslovna priložnost. Sicer so se še vedno čutili tesno zvezane z odlokom ameriškega pravosodnega ministrstva iz leta 1956, vendar je imelo njihovo zaznavanje tržnih priložnosti za posledico to, da so predelavo UNIX 32V smeli uporabljati le na Berkeley-ju zgolj v raziskovalne namene. Tudi to je bil vzrok, da različica 32V nikoli ni upravičila visokih pričakovanj. Pestile pa so jo tudi druge težave, ena ključnih je bila predvsem prevelika poraba navideznega spomina.

Vse zapleti so privedli do tega, da je pričela Berkeleyeva skupina raziskovalcev predelovati izvorno kodo UNIX-a 32V in znova spisala nekatere dele jedra. S tem so optimizirali predvsem porabo navideznega spomina, rezultat njihovega dela pa je bila tretja izdaja Berkeleyevega programskega paketa oziroma 3BSD.

3BSD pa je za razliko od predhodnikov že predstavljal celovito distribucijo operacijskega sistema, namenjeno DEC-ovim napravam VAX. Ta izdaja je bila pomembna tudi iz vidika nadaljnjih razvojnih smernic UNIX-a. Medtem ko se je AT&T počasi pričel osredotočati na bolj stabilne in komercialno usmerjene različice, pa je

Berkeley prevzel vlogo nadaljnjega raziskovanja in razvoja tega operacijskega sistema.

2.4 UNIX IN INTERNET

Omenil sem že, da zametki interneta segajo v leto 1968 z ustanovitvijo projekta ARPANET. Med leti 1970 in 1980 je bilo v to omrežje povezanih večina raziskovalnih centrov Agencije za raziskovanje naprednih vojaških projektov (angl. *Defense Advanced Research Projects Agency*[2], DARPA, 2009). To je privedlo do potrebe po nekem standardnem komunikacijskem protokolu in odgovor na to je bil protokol TCP/IP (angl. *Transmision Control Protocol/Internet Protocol,* TCP/IP, 2009), ki je še dandanes izjemno pomemben v internetnem prostoru. DARPA pa je poleg pomanjkanja takšnega protokola imela še nekaj težav, njihova strojna oprema je bila zastarela, poleg tega pa jih je pestila tudi slaba združljivost strojne in programske opreme, zato so potrebovali neko skupno osnovo (Weber, 2004, str. 33-35).

Da bi vsi raziskovalni centri uporabljali enako strojno opremo ni bila realna opcija saj so potrebovali različne naprave tudi za sam proces eksperimentiranja s tehnologijami. Tako so morali enotnost iskati v programski opremi, predvsem operacijskem sitemu ki predstavlja temelj za delovanje vseh ostalih programov. Tu se je kot najprimernejši kandidat pojavil UNIX. UNIX je imel ključno prednost pred ostalimi, predvsem DEC-ovimi, lastniškimi operacijskimi sistemi; na voljo je bila njegova izvorna koda. To ga je naredilo zanimivega za raziskovanje in eksperimentiranje, hkrati pa je bil tudi bolj prilagodljiv za nameščanje na različne tipe strojne opreme.

DARPA je tako z Berkeleyem leta 1980 tudi podpisala pogodbo. Profesor Bob Fabry se je zavezal, da bo prilagodil verzijo UNIX-a 3BSD njihovim potrebam. S tem namenom je ustanovil Skupino za raziskovanje računalniških sistemov (angl. *Computer Systems Research Group,* CSRG, 2009), v kateri so sodelovali tudi študenti kakršen je bil Bill Joy. Tako so konec leta 1980 izdali različico 4BSD, ki pa je imela precej težav in je zaostajala tudi za DEC-ovim operacijskim sistemom. Leta 1981 je zato izšla popravljena verzija 4.1BSD.

AT&T se je medtem pripravljal na izdajo svoje komercialne različice UNIX-a, imenovane UNIX System V, zato so morali na Berkeleyu spremeniti način poimenovanja svojih izdaj, da bi se izognili morebitni zmedi.

V tem času so pri DARPI pričeli z uporabo različice 4.1BSD, in v Bostonskem podjetju BBN (pogodbeno vezanim z DARPO) so lahko izdali zgodnjo verzijo težko pričakovanega TCP/IP protokola. Le-tega je Bill Joy nato vključil v Berkeleyev programski paket in poleg tega predelal spletni vmesnik, da je bil primeren tudi za uporabo izven DARPINIH standardov. Septembra 1983 je Berkeley tako izdal

[2] Kratici ARPA in DARPA predstavljata isto organizacijo. Ustanovljena je bila kot ARPA vendar so ji do leta 2009 večkrat dodali in odvzeli naziv Obrambna oziroma Defense. Nazadnje so jo preimenovali leta 1996, od tedaj dalje ostaja DARPA.

operacijski sistem 4.2BSD, ki je poleg vseh sprememb kot prvi vseboval delujočo podporo TCP/IP protokolu. Berkeley je dobavil preko 1000 licenc svoje različice UNIX-a 4.2BSD, kar je bilo več kot kdajkoli prej. Uporabniki so se namreč raje odločali za Berkeleyevo različico kot pa komercialno izdajo AT&T-ja UNIX System V.

4.2BSD je v praksi dejansko predstavljal začetek interneta, kakršnega poznamo danes in ki je hkrati tudi zaslužen za popularnost in razmah odprte kode.

Osemdeseta leta preteklega stoletja sicer niso znana po tem, da bi UNIX uspeval na številnih trgih in bil v uporabi na večini računalniških sistemov – ravno nasprotno. Se je pa zato neprestano tehnološko razvijal in obstal v pomembnih nišah računalništva. Vse bolj je na površje prihajal Microsoft s svojo izrazito lastniško usmerjeno tržno politiko, poleg tega pa je UNIX izgubil precej pomembnih ljudi. Le-ti so se zaposlovali v različnih podjetjih in svoj čas raje posvečali razvoju lastniške programske opreme kot pa izdelku, ki ni prinašal nekih izrazitih dobičkov.

Pomembno pa je tudi dogajanje povezano s podjetjem AT&T ter njihovimi omejitvami glede trženja programske opreme. Ameriško sodstvo je leta 1982 sprejelo nov predlagan proti kartelni odlok, namesto tistega iz leta 1956, ki je AT&T-ju onemogočal vstop na trge nepovezane s telefonom in telegrafom. Pravno je pričel veljati šele dve leti kasneje, med drugim pa je tudi določal, da se podjetje AT&T razbije na več enot. Leta 1984 je tako podjetje Bell Labs postalo avtonomno, pod svojim okriljem je združevalo nekaj hčerinskih podjetij, Westeren Electric pa so razpustili.

Sprejetje tega odloka je torej pomenilo, da AT&T ni več vezan na omejitev vstopa na trg računalniške opreme, kar so nemudoma izkoristili. Ustanovili so poseben oddelek imenovan Unix System Laboratory (USL, 2009) in povsem spremenili licenčne pogoje izdaje UNIX-a. Leta 1988 so cene njihove verzije operacijskega sistema narasle na 100 tisoč ameriških dolarjev za posamezno kopijo, nekaj let kasneje pa celo na 250 tisoč. Te cene so pomenile, da si lahko AT&T-jev UNIX privoščijo le še dobro stoječa podjetja kot sta bila DEC in IBM, vse manj dostopen pa je postal univerzam. Hkrati je to predstavljalo tudi veliko pobudo za nadaljnji razvoj BSD-ja, ki se postajala vedno večja alternativa in konkurenca AT&T-ju.

2.5 RAZVOJ BSD RAZLIČICE UNIX-a

Izdaja 4.2BSD leta 1983 je potrebovala nekaj izboljšav, tako kot verzija 4BSD je namreč imela določene težave z delovanjem. Skupina CSRG je potrebovala dve leti, preden je lahko objavila izid izboljšane verzije 4.3BSD, zaradi težav z vključitvijo TCP/IP protokola pa je dejansko izšla šele leta 1986 (Weber, 2004, str. 39-43).

Naslednji pomemben korak je bila ločitev samega jedra BSD-jeve različice UNIX-a na strojno-odvisni ter strojno-neodvisni del. Ta ločitev je omogočala veliko lažje predelave na vse ostale računalniške sisteme. CSRG je to verzijo poimenoval 4.3BSD-Tahoe.

Tahoe verzija je bila tista, ki je izšla ravno v času velikih sprememb AT&T-jevega licenciranja UNIX-a. Ta podatek je še zlasti pomemben zato, ker je večina kode BSD-jevih izdaj še vedno pripadala AT&T-ju oziroma so jo spisali njihovi uslužbenci. Glede na to, da je bil vsak izvod BSD-ja dejansko povsem odprte narave, kar pomeni da je bil opremljen z izvorno kodo, to ni več sovpadalo z novo AT&T-jevo politiko. Na Berkeleyu so zato morali poiskati primerno rešitev.

Tako so se v skupini CSRG odločili, da izdajo lastno verzijo TCP/IP protokola v samostojnem programskem paketu imenovanem Networking Release 1. Ta protokol je postajal vedno bolj priljubljen med uporabniki računalnikov in predvsem interneta, Berkeleyeva verzija pa ni vsebovala prav nobene povezave z AT&T-jem. Šlo je sicer za korak nazaj, od izdaje samostojnega operacijskega sistema so na Berkeleyu znova prešli zgolj na izdajo orodja oziroma pripomočka, ki je lahko deloval le v okviru drugega operacijskega sistema. Kljub temu pa je Networking Release 1 naletel na izjemen odziv med uporabniki, za kar je bila zaslužna tudi zelo velikodušna licenca, ki je pogojevala uporabo te izdaje.

Licenčne omejitve Networking Releasea 1 so bile minimalne. Lastnik je imel pravico početi s tem orodjem karkoli si je zaželel. Lahko jo je izdal oziroma distribuiral spremenjeno ali nespremenjeno, z izvorno kodo ali brez nje, v vsakem primeru brez kakršnihkoli obveznosti do Berkeleya. Iz originalnega izvoda je lastnik lahko naredil lastniško programsko opremo in jo prodajal po kakršnikoli ceni je želel. Eden izmed redkih pogojev, ki ga je licenca postavljala je bil, da bodo zasluge v dokumentaciji pripisane Berkeleyu ter da bodo avtorske pravice (angl. *copyright*) v izvorni datoteki ostali nedotaknjeni. Berkeley je sicer zaračunal tisoč dolarjev za originalni izvod te izdaje, vendar je licenca lastniku omogočala, da ga je nemudoma skopiral in razdal komurkoli je hotel. To je bil prvi primer licence, ki so jo kasneje imenovali licenca tipa BSD (Laurent, 2004, str. 14).

Dober odziv na omenjeni programski paket je pozitivno vplival tudi na skupino CSRG. Eden izmed ključnih članov te skupine, Keith Bostic (Keith Bostic, 2009) je predlagal, da poskušajo v Networking Release 1 vključiti čim več ostale BSD kode, ki ni imela nobene povezave z AT&T-jem. Na Berkeleyu so to operacijo poimenovali »čista soba« (angl. *clean room*). Postopek je potekal tako, da so posamezniki brez pomoči izvorne kode in samo z vpogledom v dokumentacijo programa, ki je opisovala kaj naj bi program izvajal, šli po vzvratni poti. Vzeli so torej končni izdelek in v vzvratnem procesu izločali vse dele kode, ki so vsebovale AT&T-jeve elemente.

Šlo je za izjemno obsežen projekt, za katerega niti ni bilo povsem jasno, če je v celoti izvedljiv, v kar so dvomili tudi nekateri člani skupine CSRG. Keith Bostic je zato organiziral večjo skupino prostovoljcev, to je bil poizkus uvedbe pisanja programske opreme temelječega na internetni komunikaciji. Spodbujal jih je preko govorov na združenjih uporabnikov UNIX-a, z javnimi govori ter tudi preko spleta. Edina stvar, ki jim jo je obljubil v zameno za sodelovanje, je bila navedba njihovih imen ter zaslug v spremni dokumentaciji programa.

Bosticevi poizkusi dela so počasi pričeli dobivati vse več privržencev. V osemnajstih mesecih se je projektu pridružilo preko štiristo razvijalcev. Kmalu so bila preurejeni vsi glavni elementi BSD-ja razen samega jedra. Tega se je lotil Bostic s sodelavci, bilo pa je to še eno izjemno zahtevno ter kompleksno opravilo: V veliki meri so ga opravili do pomladi leta 1991. Vsem skupaj je uspelo na novo spisati praktično kompleten operacijski sistem, razen šestih datotek, ki so še vedno vsebovale AT&T-jevo kodo. Datoteke so bile preveč kompleksne tudi za Berkeleyevo skupino in njihove prostovoljne sodelavce. Tako so se na fakulteti odločili, da bodo izdali nepopoln operacijski sitem ter upali, da bodo manjkajoče datoteke spisali uporabniki.

Junija 1991 je izšel Networking Release 2. Ime ni odražalo dejanske vsebine paketa, šlo je za precej več kot samo spletni vmesnik. Takšno ime je dobil zato, da pravnikom kalifornijske univerze ni bilo potrebno ponovno vzpostaviti drugega licenčnega sistema. Networking Release 2 so tako izdali pod enakimi pogoji kot prvo različico: tisoč dolarjev za kopijo, izvorna koda vključena, prosto razširljiv dokler zasluge ostanejo Berkeleyeve.

To izdajo je nato prevzel William Jolitz (William Jolitz, 2009). Pravilno je predvidel, da prihodnost računalniških procesorjev leži v Intelovi arhitekturi čipa x86 (Intel x86, 2009). Ti čipi so bili takrat srce IBM-ovih osebnih računalnikov ter predhodniki kasnejših in bolj znanih Pentiumov (Intel Pentium, 2009) in so se razvijal veliko uspešneje kot konkurenčni procesorji, denimo DEC-ovi. Jolitz je nato le šest mesecev po izdaji Networking Releasea 2 dokončal šest manjkajočih datotek ter izdal prvi operacijski sistem za Intelove osebne računalnike 386 (Intel 386, 2009), ki se je širil s pomočjo interneta (mnogo ljudi je zmotno prepričanih, da je bil prvi tak operacijski sistem delo Linusa Torvaldsa (Linus Torvalds, 2009), avtorja Linux-a). 386/BSD, kakor se je operacijski sistem imenoval, je bil licenciran za prosto redistribucijo in spreminjanje, dokler so zasluge ostale nespremenjene. Kmalu si je 386/BSD preko spleta naložilo več tisoč uporabnikov, ki so Jolitzu pošiljali popravke napak, kode ter ideje za nadaljnji razvoj. Ravno v tem času je na Finskem študent Linus Torvalds iskal operacijski sistem temelječ na UNIX-u, s katerim bi lahko eksperimentiral na svojem osebnem računalniku. Kljub dejstvu, da so razvijalci UNIX-a med seboj že relativno dolgo sodelovali preko interneta, svetovni splet, kakršnega poznamo danes takrat še ni obstajal. Splet je bil razdeljen na večje skupine, nekakšne otoke, ki med seboj še niso bili neposredno povezani. Zaradi tega razloga Linus Torvalds tudi ni pravočasno izvedel za 386/BSD operacijski sistem in je tako moral pričeti z razvojem svojega lasnega.

Kljub temu pa je 386/BSD predstavljal velik napredek v razvoju in je služil kot osnova mnogim nadaljnjim izvedbam Berkeleyevega programskega paketa. Pomebnejše so predvsem tri kasnejše izdaje: NetBSD, FreeBSD in OpenBSD. V osnovi so bile enake, vsaka pa je bila specializirana na svojem področju. Glavna ideja NetBSD izdaje je tako bila, da je čim bolj združljiva s številnimi tipi strojne opreme, FreeBSD je bila osredotočena za olajšano uporabo na Intelovih x86 čipih, medtem ko je bil OpenBSD osredotočen predvsem na varnost.

Networking Release 2 pa je imel še enega potomca. Na Berkeleyu so izkoristili možnost, ki jo je ponujala BSD licenca in ustanovili novo podjetje Berkeley Software Design Incorporated (BSDI). V BSDI-ju so vzeli Jolitzovo izvorno kodo šestih manjkajočih datotek, jih napisali ter leta 1992 izdal lasten lastniški operacijski sistem za nekaj manj kot tisoč dolarjev na kopijo. Ta operacijski sistem so tudi precej agresivno oglaševali ter ga neposredno primerjali z AT&T-jevo različico, kar je kasneje pripeljalo tudi do pravnega spora med AT&T-jem ter Berkeleyem.

2.6 PRAVNI SPOR MED UNIVERZO BERKELEY IN AT&T

BSDI-jeva izdaja Networkinig Release 2 je torej bila kompleten operacijski sistem. Težava z AT&T-jevega stališča je bila v tem, da je s ceno tisoč dolarjev na kopijo bil veliko cenejši od AT&T-jeve različice. V AT&T-jev oddelku za UNIX (Unix System Laboratories – USL) so se zato odločili za tožbo proti BSDI-ju s katero so želeli dokazati, da so v Kaliforniji zlorabili AT&T-jeve pravice do UNIX-a kot blagovne znamke. Med drugim so v USL trdili, da so pri BSDI-ju ukradli njihovo lastniško izvorno kodo. To so pri BSDI-ju zanikali in trdili da so uporabili zgolj kodo, ki jo je prosto distribuirala kalifornijska univerza ter jo tržili v skladu z njihovimi licenčnimi omejitvami skupaj s šestimi datotekami, ki so jih spisali sami. Sodišče je zato odločilo, da mora AT&T razrešiti spor s kalifornijsko univerzo, z BSDI-jem pa samo del, ki se je dotikal šestih spornih datotek (Weber, 2004, str. 49-52).

AT&T se je znašel v neprijetnem položaju, morali bi namreč vložiti tožbo proti njihovemu dolgoletnemu partnerju, kalifornijski univerzi. Poleg tega pa je negotovost okrog nerešenih pravnih vprašanj imela precejšnje predvsem negativne posledice na razvoj tako UNIX-a kot tudi odprte kode. Težava je bila v tem, da so razvijalci izgubili motivacijo za delo, saj niso vedeli kakšna usoda čaka njihove končne izdelke. V najslabšem primeru bi se lahko zgodilo, da bi določen program zaradi sodne prepovedi preprosto zaklenili oziroma prepovedali njegovo uporabo, ali pa bi se takšen program spremenil celo v lastniško programsko opremo. V času, ko se je računalništvo razvijalo izjemno hitro je tako prišlo velike upočasnitve, preden so razrešili spor sta minili kar dve leti (za računalniške razmere skoraj večnost).

AT&T je ob koncu leta 1992 namreč ponovno vložil tožbo tako proti kalifornijski univerzi kot tudi BSDI-ju ter zahteval sodno prepoved distribucije Networking Releasea 2 kot tudi BSDI-jeve programske opreme. Leta 1993 je sodišče zahtevo zavrnilo. Že naslednji dan po odločitvi sodišča pa je kalifornijska univerza vložila tožbo zoper USL, da naj bi bili oni tisti ki so kršili zakon saj naj bi izrabili ohlapne pogoje BSD licence in odvzeli avtorske pravice oziroma zasluge Berkeleyu ter s tem kršili praktično edino zavezujočo klavzulo BSD licence. Primer ni nikoli šel na sodišče, kmalu po vložitvi te tožbe so pri AT&T-ju oddelek za razvoj UNIX-a (torej USL) prodali podjetju Novell. Pri Novellu so priložnost izkoristili za otvoritev pogajanj z Berkeleyem glede pravic in zaslug glede UNIX-a ter leta 1994 tudi sklenili dogovor.

Vsebina dogovora je še vedno zaupna, zagotovo pa je prišlo do nekaterih kompromisov. Junija leta 1994 je Berkeleyeva skupina CSRG izdal nov programski

paket imenovan 4.4BSD-Lite, pri USL pa so se strinjali da je vsa vključena koda nesporna in v skladu z dogovorom. Vsi, ki so uporabljali pretekle različice CSRG-jevih izdaj so morali svoje izdelke uskladiti s 4.4BSD-Lite (npr. BSDI, FreeBSD, OpenBSD itd.). To je povzročilo novo upočasnitev v razvoju UNIX-a. Junija 1995 pa so pri CSRG izdali še zadnji programski paket, 4.4BSD-Lite Release 2, nato pa so skupino razpustili.

2.7 FREE SOFTWARE FOUNDATION

Poleg AT&T-ja ter Berkeleya so seveda obstajala tudi druga razvojna središča programske opreme. Eden pomembnejših z vidika odprte kode je bil tudi Center za umetno inteligenco na inštitutu MIT (angl. *Massachusetts Institute of Technology*, MIT, 2009). V šestdesetih in sedemdesetih letih 20. stoletja je bil to eden izmed ključnih središč razvoja programske opreme in predvsem računalniške komunikacije. Poleg tega je znano, da so se na MIT-ju stvari odvijale s poudarjeno moralno ter etično komponento, prav tako pa so prvi uporabili izraz prosto programje in vzpostavili intelektualno kulturo sodelovanja ter odprtosti (Weber, 2004, str. 46-49).

Velik vpliv na skupnost MIT je imel tudi vedno večji porast lastniške programske opreme v osemdesetih letih 20. Stoletja. Velika podjetja so neprestano najemala MIT-ove najboljše programerje kar je pripeljalo tako daleč, da so od njih zahtevali, da podpišejo sporazume o ne-razkrivanju podatkov. Poleg tega pa je bila velika večina strojne opreme dobavljena z operacijskimi sistemi, ki niso imeli priložene izvorne kode in tako niso bili pretirano zanimivi za proučevanje.

Kaplja čez rob, ki je posredno tudi pripeljala do ustanovitve Free Software Foundationa je bil laserski tiskalnik, ki so ga leta 1979 kupili MIT-ovi laboratoriji. Tiskalnik je imel veliko težav z zatikanjem papirja. Ko so to pomanjkljivost želeli odpraviti, jim podjetje, ki je dobavilo tiskalnik ni hotelo predati izvorne kode programa, ki je s tiskalnikom upravljalo. To je povzročilo precej frustracij in negodovanja pri zaposlenih na MIT-ju, predvsem pa pri kasnejšemu ustanovitelju FSF-ja Richardu Stallmanu.

Richard Stallman se je vse bolj nagibal k mišljenju, da je lastniška programska oprema moralno sporna, da iz običajnih ljudi, ki si želijo med seboj zgolj pomagati, dela pirate, pravna ureditev tega področja pa zgolj omejuje sodelovanje skupnosti. Stallman v programski opremi ni videl zgolj orodij za dosego neki ciljev temveč predvsem odraz človeške kreativnosti ter tudi sredstvo, s katerim obstoječa skupnost ustvarja splošno družbeno korist.

Leta 1984 je zato Stallman pustil službo pri MIT in se posvetil razvoju prostega programja. V podporo temu projektu je torej ustanovil neprofitno organizacijo Free Software Foundation. Poglavitni cilj FSF-ja je bil ustvariti povsem prost operacijski sistem, ki bi ga lahko vsakdo pridobil iz interneta, ga uporabljal ter prosto spreminjal ter širil. Za zgled je vzel UNIX, da pa se je povsem odcepil od AT&T-jeve vse bolj lastniške logike je svoj projekt poimenoval GNU, kar je rekurzivni akronim za »GNU is

not UNIX« ali »GNU ni UNIX«. Leta 1984 je izšel manifest GNU, ki je opisoval temeljne ideje projekta in predvsem razjasnil pomen besede prosto programje.

Beseda zveza prosto programje, po Stallmanovi razlagi, nikoli ni predstavljala nične cene temveč zgolj svobodo do uporabe, spreminjanja ter širjenja takšnih programov. Stallman je celo zagovarjal distribucijo po določeni ceni, saj tudi projekti razvoja prostega programja potrebujejo finančna sredstva za delovanje. Zmeda okrog besedne zveze prosto programje, ki je v veliko primerih prisotna še danes, sicer izhaja iz angleščine, saj se prevaja kot »free software« kjer v angleščini beseda »free« lahko pomeni tudi brezplačno.

V kontekstu svobodnega oziroma prostega programja je Stallman opredelil štiri poglavitne točke, ki jih mora upoštevati tak program (Definicija prostega programja, 2009):

- Svobodo do uporabe takega progama v kakršnekoli namene (svoboda 0);
- Svobodo do preučevanja delovanja programa in njega spreminjanja na način, da nam bolje služi (svoboda 1);
- Svobodo do razširjanja kopij programa brezplačno ali v zameno za plačilo (svoboda 2);
- Svobodo do spreminjanja programa ter razširjanja spremenjene verzije, da imajo tudi ostali korist od tega (svoboda 3).

Ključ do realizacije teh točk je seveda dostop do izvorne kode prostega programa.

Stallman se je kmalu zavedel, da samo te točke ne bodo dovolj za vzdrževanje ravnotežja v odprtokodni skupnosti. Družbeni sistem intelektualne lastnine, ki se je vzpostavil z uresničitvijo omenjenih svobod je potreboval še nekatere pravne omejitve ki bi preprečile morebitne zlorabe. Stallman namreč nikakor ni želel, da nekdo vzame izvorno kodo nekega programa, z njeno pomočjo ustvari nek nov program in ga nadalje trži kot lastniškega brez dodane izvorne kode. Takšno možnost dopušča licenca tipa BSD, Stallman je zato moral iznajti nekaj drugega.

Posledica njegovih prizadevanj je bila General Public Licence (Laurent, 2004, str. 35), katere bistvo je, da omejuje oziroma celo izniči možnost ohranjanja izvorne kode skrite, tako v osnovnem programu kot tudi v vseh naslednjih izvedbah. Avtorsko zaščiteno delo, v angleščini imenovano »copyright« pod GPL licenco postane avtorsko odprto, Stallman je to poimenoval »copyleft«. Program, izdan pod pogoji GPL licence v nobenem primeru ne more postati lastniški, vedno mora biti izdan skupaj z izvorno kodo. Vsi potomci takšnega programa morajo biti prav tako izdani pod omenjeno licenco kar znova pomeni, da ne morejo postati lastniški. Licenca tudi zahteva, da se niti najmanjši delci prostega programa ne smejo pojaviti v lastniškem programu, razen če bo tudi ta izdan pod enakimi pogoji. Takšne zahteve GPL licence so znane tudi kot virusna klavzula. Gre za pozitiven pomen besede virus v smislu, da prosto programje »okuži« drugo prosto programje in tako se odprtokodna skupnost širi.

Free Software Foundation je zaslužna za mnogo popularnih programov, povsem združljivih tudi z UNIX-om, ki ga je tako še dodatno pomagala utrditi na mestu operacijskega sistema po okusu odprtokodne skupnosti. Množica teh programov pa je povzročila, da so jim morali razvijalci GNU-ja posvečati veliko pozornosti saj so morali neprestano dodajati izboljšave ter popravljati hrošče. Zaradi tega je trpel osnovni cilj FSF-ja; izdelati kompleten prosto dostopni operacijski sistem. Razvijalcem GNU-ja je do konca osemdesetih let 20. stoletja vendarle uspelo izdelati ogrodje omenjenega operacijskega sistema, niso pa imeli ene ključne sestavine – jedra. To pa je bila izvrstna priložnost za vstop na sceno naslednje izjemno pomembne komponente odprte kode – Linuxa.

2.8 LINUX

Morda iz dosedanjega besedila ni povsem jasno razvidno, toda številne komercialne in odprte različice UNIX-a so na tržišču programske opreme povzročile pravo zmešnjavo. Glavna težava je bila, da jih večina med seboj ni bila združljiva in ogromno število različnih izdaj je povzročalo zmedo med uporabniki in razvijalci.

Leta 1991 pa je nase opozoril finski študent računalništva Linus Torvalds. Iskal je alternativo Microsoftovemu operacijskemu sistemu DOS (angl. *Disc Operating System*, DOS 2009), ki je bil takrat prednaložen na večino osebnih računalnikov. Želel je uporabljati operacijski sitem podoben UINX-u, toda zardi omejene dosegljivosti in strojne zahtevnosti namestitev UNIX-a na osebni računalnik ni bila najbolj optimalna izbira. Torvalds je nato izvedel za Minix, poenostavljeno različico UNIX-a, ki jo je razvil nizozemski profesor Andrew Tanenbaum (Andrew Tanenbaum, 2009) kot učno orodje. Torvalds je kupil različico Minixa, ga namestil na svoj osebni računalnik in nemudoma pričel s pisanjem jedra lastnega operacijskega sistema, Minix pa uporabljal kot ogrodje. Že jeseni leta 1991 Minixa ni več potreboval in je na splet objavil izvorno kodo jedra svojega operacijskega sistema, ki ga je poimenoval Linux (Zgodovina Linuxa, 2009).

Odziv ljudi je bil izjemen. Do konca leta 1991 se je odzvalo več kot sto ljudi, ki so Linusu Torvaldsu preko elektronske pošte pošiljali predloge ter popravke kode in napak. V letih 1992 ter 1993 je število privržencev Linuxa stalno naraščalo, čeprav se je zdelo vedno bolj očitno, da se pod taktirko predvsem Microsofta počasi bliža konec operacijskih sistemov tipa UNIX. Leta 1994 je Torvalds izdal prvo uradno verzijo Linuxa označeno z 1.0. Od tega trenutka dalje se je projekt pričel razvijati z veliko hitrostjo.

Glavna prednost Linuxa pred že dokončanim Minixom je bila, da se je razvijal in omogočal eksperimentiranje. Torvalds je z Linuxom uspel prepričati ljudi, da se projekt lahko razvije v nekaj velikega in pomembnega. Konec leta 1991 se je Linux preselil na dva večja strežnika v Nemčiji in ZDA, vedno večje zanimanje pa je Torvaldsa prisililo, da spremeni svoje začetno mnenje o licenciranju. Pred izdajo različice 0.12 januarja 1992 je Torvalds zagovarjal uporabo svoje lastne licence, ki je prepovedovala zaračunavanje kakršnihkoli stroškov, tudi v povezavi z samo

distribucijo. Internet v tem obdobju še ni bil tako masovno razširjen, zato je ta pogoj zelo omejeval širjenje. To je hitro spoznal tudi Torvalds in z izdajo verzije 0.12 prevzel licenco GPL, kar je verjetno ena izmed najpomembnejših odločitev, ki jih je sprejel v času razvoja Linuxa. Prehod na GPL je tako omogočil zaračunavanje minimalnih stroškov, nastalih zaradi distribucije, poleg tega pa dodal tudi virusno klavzulo licence, ki onemogoča nastanek kakršnekoli lastniške programske opreme nastale na podlagi izvorne kode.

Naslednji pomemben korak v razvoju Linuxa je bila vpeljava uporabniškega vmesnika v operacijski sistem. Uporabniški vmesnik pretvarja uporabnikove želje oziroma ukaze v strojni jezik. Gre torej za pomemben del (vsakega) operacijskega sistema, brez njega praktično ne moremo uporabljati računalnika. Torvalds je za ogrodje vmesnika uporabil prosto dostopen GNU-jev program imenovan Bash (Bash, 2009), ki je uporabniku predstavil preprosto orodno vrstico, v katero je vpisoval ukaze. Skupnost pa si je želela bolj prefinjen, grafični vmesnik, nekakšen sistem oken, kakršnega uporablja istoimenski Microsoftov lastniški operacijski sistem Windows (Okna). Skupnost pod vodstvom Oresta Zborowskega (Hughes, 1995) je za model vzela vmesnik imenovan X, ki je bil dobro poznan UNIX-ovim uporabnikom, in ga predelalo za Linux. Znova je bila izjemno pomembna Torvaldsova reakcija; ne samo, da je povsem sodeloval z razvijalci vmesnika X, ki je medtem postal samostojni razvojni projekt izdan pod posebno MIT licenco, vmesnik X je uporabil kot orodje za testiranje temeljev Linuxovega jedra. S to potezo je zgolj utrdil sodelovanje med obema stranema.

Maja leta 1992 je izšla verzija 0.96 z podporo vmesniku X. Naslednji izjemno pomemben korak v razvoju Linuxa pa je bil vgradnja podpore za internet. Skupnosti uporabnikov-razvijalcev je bilo povsem jasno, da Linux potrebuje vrhunsko podporo protokolu TCP/IP. Bill Joy je za BSD Unix že spisal uporaben TCP/IP protokol, ki bi ga z nekaterimi izboljšavami bilo možno uporabiti tudi v Linuxu, toda skupnost na čelu s Torvaldsom ni hotela sprejeti takšne odločitve. Ravno v tem obdobju je namreč potekala pravna vojna za pravice do kode med BSD-jem in AT&T-jem. Zaradi tega so pričeli povsem na novo programirati internetni vmesnik.

Pojavila sta se dva kandidata za razvoj, Fred van Kempen (Hughes, 1994) ter Alan Cox (Alan Cox, 2009), imela pa sta različni razvojni filozofiji. Medtem ko je van Kempen zagovarjal pristop, kjer naj bi razvil kompleten in optimalno delujoč vmesnik je bil Cox zagovornik pristopa, da naj stvar zgolj deluje, popravke ter izboljšave pa bodo vgradili kasneje. Torvalds je moral sprejeti odločitev, kateri model je primernejši za Linux in odločil se je za pristop Alana Coxa. To je povzročilo nekaj jeze in zamer s strani van Kemperja, ki je nekaj časa nadaljeval tudi s samostojnim delom in razmišljal celo o odcepitvi in samostojnemu nadaljnjemu razvoju Linuxa, vendar se v praksi to ni obneslo.

Po izidu različice 1.0 marca leta 1994 je bil naslednji pomemben korak v razvoju Linuxa predelava le-tega na različne računalniške sisteme. Do tega trenutka je Linux deloval zgolj na Intelovih x86 procesorjih, od tu dalje pa so se za Linux pričeli zanimati predvsem pri DEC-u, kjer so izdelali nov, 64-bitni računalnik imenovan

Alpha. To je bil za Linux velik napredek, podjetje DEC je bilo eno izmed vodilnih na področju računalniške tehnologije in se je povsem navezalo na prosti Linux, čeprav so imeli tudi svoj komercialni operacijski sistem. Tudi iz tehnološkega vidika je bil to velik korak naprej, arhitektura 64-bitne Alphe in Intelovega čipa x86se namreč precej razlikuje. Jedro Linuxa je bilo torej potrebno prilagoditi tako, da koda ne bi bila več odvisna od strojne konfiguracije. To je tudi pomenilo, da je bil Linux z manjšimi modifikacijami primeren tudi za ostale računalniške sisteme, ne samo DEC-ove ali Intelove.

Še en pomemben korak k hitremu širjenju Linuxa je bila pojava prvih distribucijskih paketov. Linux je bil do tedaj namreč namenjen predvsem ljudem, ki so se spoznali na programiranje. Njegova namestitev je bila precej zahtevno opravilo, sistem je bilo potrebno vzpostaviti v več delih in poiskati ustrezne gonilnike za naprave. Pogostokrat je vzpostavitev sistema povzročala težave tudi izkušenim programerjem, zato se je pojavila potreba po nekom, ki bi zbral vse potrebne dele skupaj in napravil namestitveni proces bolj prijazen uporabniku. Tako se je pojavil Ian Murdock (Ian Murdock, 2009), sicer študent ekonomije, ki s podporo FSF-ja ustanovil projekt Debian (Debian, 2009), s katerim je postavil zgodnje standarde distribucije Linuxa. Sledili so mu mnogi, ki so vsak na svoj način (večinoma v zameno za plačilo – vendar zgolj stroške distribucije) promovirali svojo izdajo Linuxa. Poleg Debiana je bila proti plačilu na voljo tudi distribucija Slackware (Slackware, 2009); distribucija Yggdrasil (Yggdrasil, 2009) je bila prva, ki je Linux ponudila na CD-Romu, mediju ki je tedaj pričel izpodrivati diskete.

Naslednja novost, ki je pomagala k širjenju Linuxa pa je bila preureditev razvojnega procesa. Torvalds se je veliko posvečal predvsem predelavam Linuxa na različne strojne konfiguracije, kar je povzročalo zastoje v napredku. Prav tako se je pojavila vrzel med naprednimi uporabniki, ki so si želeli predvsem raziskovati in izboljševati operacijski sistem in uporabniki, ki so posegali predvsem po distribucijskih paketih ter so predvsem želeli stabilno verzijo Linuxa. Torvalds se je tako domislil dvotirnega sistema izdajanja različic, tiste z sodo oznako so bile stabilne (1.0, 1.2, 1.4 itd.), tiste z liho oznako pa preizkusne in namenjene raziskovanju ter testiranju. V končni fazi se je tako testna (liha) oznaka spremenila v stabilno (sodo), odprla pa se je nova testna.

Leta 1996 je torej izšel Linux 2.0 ki je imel vgrajeno podporo različnim strojnim konfiguracijam in mnoge druge izboljšave. V tem letu se je Linus Torvalds tudi preselil v ZDA in sprejel službo pri podjetju Transmeta. To podjetje sicer ni bilo neposredno povezano z Linuxom vendar je Torvaldsu obljubilo dovolj časa, da bo lahko nemoteno nadaljeval z razvoje operacijskega sistema. Mnogi pripadniki Linuxove skupnosti so se zbali, da utegne ta korak povzročiti počasnejši razvoj in imeli so prav.

Dve leti kasneje, leta 1998, je bil Torvalds že medijsko precej prepoznaven, podjetja, ki so se ukvarjala z izgradnjo lastniških operacijskih sistemov pa so Linux dojemala kot veliko grožnjo. Torvalds je bil tudi razpet med svojo službo pri Transmeti in nadaljnjim razvojem Linuxa, kar je terjalo veliko naporov. Linuxova skupnost je to

dojemala preprosto; njihov vodja ni bil dovolj odziven, zato je trpel projekt. Pojavila se je različica pod vodstvom Davida Millerja, ki je vsebovala popravke, ki jih Torvlads sicer ni odobril in to je pomenilo prvo večjo krizo in možnost odcepitve od prvotnega na povsem samostojen projekt.

Napetost se je umirila predvsem po zaslugi Erica Raymonda, ki predvsem razelektril ozračje. Trdil je, da spori niso osebni in so zato nesmiselni. Glavni akterji so se nato zbrali na poslovnem kosilu, tudi Torvalds in Miller, in začrtali nekoliko bolj uradno organizacijsko strukturo in predvsem razbremenili Torvaldsa, vsaj kar se tiče medijske izpostavljenosti.

Odločitve na tem sestanku so začrtale pomembne in stabilne smernice nadaljnjega razvoja Linuxa. To se odraža tudi na sami uporabi. Linux je postal prefinjen operacijski sistem, ki je že leta 2000 poganjal tretjino strežnikov, ki so tedaj sestavljali svetovni splet. Postal je tudi tržni fenomen, sicer povsem brezplačen toda izjemno zanimiv z vidika distribucije.

2.9 OPEN SOURCE INSTITUTE

Kot nakazuje že naslov gre za definicijo odprte kode. Iz dosedanje vsebine diplomske naloge je razvidno, da se je razvoj odprte kode pričel sočasno z razvojem (modernega) računalništva. Organizacija OSI je izraz odprta koda zgolj uveljavila v praksi in dodala svojo perspektivo, ki se razlikuje od npr. Stallmanove in Free Software Foundationa. (Zgodovina OSI, 2009)

Ključno ime, ki se pojavlja ob imenu OSI je Eric Raymond (Eric Raymond, 2009). Leta 1997 je s predstavitvijo publikacije The Cathedral and the Bazzar močno vplival na podjetje Netscape, ki je objavilo izvorno kodo za svoj popularni spletni brskalnik in s tem presenetilo tako odprto kodno skupnost kot tudi poslovni svet. Izraz odprta koda je bil skovan leta 1998 na konferenci v Kaliforniji, katere udeleženci so bili predvsem veljaki iz različnih odprtih projektov. Netscape je pomembno vplival na smernice razvoja OSI, udeleženci konference so se namreč odločili, da je potrebno opustiti moralne in etične komponente, ki so se držale Stallmanovega izraza »prosto programje« (torej free software) in zasnovo OSI postaviti na povsem pragmatičnih, ekonomskih temeljih, kakor je to storil Netscape. Na sami konferenci je Chris Paterson tudi predlagal ime odprta koda ali open source in ostali so se s predlogom strinjali.

V naslednjih dneh po konferenci so pričeli njeni udeleženci širiti vest o dogovoru in spodbujati skupnost k uporabi novega pristopa ter izraza. Odzvali so se mnogi, tudi Linus Torvalds, o pobudi je razmišljal celo Richard Stallman vendar je na koncu zavrgel možnost spremembe svojih načel. K širjenju nove ideje in izraza se je zavezala večina ključnih odprtih projektov, med njimi tudi Linux in Apache.

Uradno je organizacija OSI nastala le nekaj dni po zgoraj omenjeni konferenci, februarja 1998. Njen prvi predsednik je bil Eric Raymond in je na tem položaju ostal do leta 2005. Organizacija je bila zamišljena kot izobraževalna ustanova, ki hkrati

ozavešča ljudi o odprti kodi ter sam izraz in idejo tudi zagovarja. V ta namen so tudi sestavili seznam pogojev, ki jih mora izpolnjevati programska koda, da lahko upravičeno nosi naziv odprta koda. Seznam je bolj znan pod imenom Open Source Definition (OSD) oziroma Definicija odprte kode in je na koncu tega poglavja.

Od leta 2003 dalje je OSI registrirana kot neprofitna organizacija. Eden izmed najvidnejših uspehov njene prve uprave na čelu z Raymondom je ta, da so uspeli uveljaviti organizacijo skupaj z odprtokodno definicijo kot priznano telo, tako iz strani skupnosti kot tudi poslovnega sveta in vlade. OSD danes velja kot vodilni standard licenciranja odprtih programov.

Kljub temu da je OSI postala ena vodilnih organizacij na področju odprte kode pa se ironično izogiba javnemu, glasnemu delovanju. Svoje cilje raje dosegajo na prefinjen način za kulisami, brez javnih izpostavljanj. Razlaga za takšno delovanje je, da so se na ta način uspešneje soočili z določenimi grožnjami uperjenimi v skupnosti in se izognili velikim krizam, še preden se je javnost zavedla da kriza dejansko obstaja.

Leta 2005 pa je OSI postala tudi mednarodna organizacija v pravem pomenu besede s podružnicami tudi v Evropi, Južni Ameriki, Japonski in Indiji.

OSD oziroma definicija odprte kode je izpeljana iz dokumenta z imenom Debian Social Contract (DSC, 1997). Od nastanka ni doživela praktično nobenih sprememb, naknadno je bila dodana zgolj zadnja deseta točka. Definicija odprte kode (Uradna definicija odprte kode, 2009):

- Odprtokodno programsko opremo je mogoče svobodno redistribuirati. Lahko jo redistribuira kdorkoli brezplačno ali proti plačilu;
- Izvorna programska koda je dostopna uporabniku. Licenca mora dovoljevati distribucijo v prevedeni kakor tudi v izvorni obliki;
- Licenca mora dovoljevati spremembe osnovne kode in izvedene oblike nove kode;
- Kljub temu, da mora biti izvorna koda dostopna, lahko izvorni avtorji zahtevajo, da se morebitne spremembe jasno ločijo od originalne kode in tako ohranijo ločnico med prvotno in modificirano kodo (npr. v obliki popravkov ali različnih verzij);
- Licenca ne sme omejevati katerekoli osebe ali skupine;
- Licenca ne sme biti omejevalna glede na področje dela, v okviru katerega se programska koda uporablja;
- Distribucija licenc mora biti enakovredna za vse uporabnike, brez dodatnih omejitev;
- Licenca za isto programsko kodo se ne sme razlikovati, če se jo uporablja v kombinaciji z drugo programsko opremo;
- Licenca ne sme omejevati uporabe druge programske opreme;
- Licenca mora biti tehnološko nevtralna.

3 KAJ JE ODPRTA KODA IN KAKO DELUJE

Bistvo razumevanja odprte kode se skriva v procesu, s katerim le-ta nastaja. Ob tem je potrebno odgovoriti tudi na štiri ključna vprašanja (Weber, 2004, str. 54-57):

- Kdo so ljudje ki ustvarjajo odprto kodo
- Kaj točno ti ljudje počnejo
- Kako med seboj sodelujejo ter
- Kako med seboj razrešujejo nesoglasja

Izgradnja nekega programa je zelo kompleksno in eksaktno opravilo. Programer se ob soočenju z določenim kompleksnim problemom pogosto ne more zanašati na neke osnovne zakonitosti, kot se lahko denimo fizik, ki si vedno lahko pomaga z osnovnimi fizičnimi zakoni, okrog katerih lahko nato zgradi neko novo spoznanje. Poleg tega uporabniki programsko opremo uporabljajo na vse mogoče načine ter na različnih, ne nujno medsebojno združljivih strojnih konfiguracijah. To v praksi pomeni, da je nemogoče predvideti velik del takih situacij, poleg tega pa večina tehnologije, na katerih bodo izdelani programi tekli, niti ni bila še odkrita. Za primerjavo: avtomobilski oblikovalec ima vseeno manj dela s predvidevanjem, v kakšnih okoliščinah bodo stranke vozile njegov avto, poleg tega pa tudi ceste ostajajo nespremenjene za daljše časovno obdobje.

Pisanje programa lahko zato primerjamo z ustvarjanjem neke pesmi. Pesem je po navadi delo enega uma, kreativnosti, ki pa mu lahko ostali na njegovi poti pomagajo. Bistvo nekega programa pa je prav tako v ustvarjalnosti njegovega avtorja in ustvarjalnosti ne more zamenjati niti tehnologija niti izboljšana delitev nalog.

3.1 RAZVOJ KLASIČNE TER PROSTE PROGRAMSKE OPREME

Glede na dejstvo, da gre v obeh primerih za razvoj programske opreme sta pristopa izgradnje precej različna (Weber, 2004, str. 57-65).

3.1.1 Klasična programska oprema (pristop katedrale)

Moderni operacijski sistem, kot primer kompleksnega programa, vsebuje milijone vrstic kode, ki jih morajo programerji napisati. Program lahko znova primerjamo z avtomobilom, ki vsebuje veliko različnih elementov kot so motor, zavore ali elektronika. Vsi ti elementi, tako kot pri izgradnji operacijskega sistema, potrebujejo različna specializirana znanja za izgradnjo, na koncu pa morajo še vedno tvoriti delujočo celoto prilagojeno uporabniku. Kompleksnost izgradnje programa torej nujno pogojuje nekakšno delitev dela. Že v samih začetkih razvoja programske opreme pa se je pojavilo vprašanje: kako takšno delitev izvesti?

Frederick Brooks (Frederic Brooks, 2009), računalniški znanstvenik, je zbral nekatera razmišljanja svojih predhodnikov in izgradnjo kompleksnega programa primerjal z

izgradnjo srednjeveške katedrale. Po Brooksovem mnenju (Brooks v: Weber, 2004, str. 59) je ključ za izgradnjo programske opreme njena enotna zasnova, pri izgradnji katedrale bi to bil arhitektov načrt, po katerem se nato ravnajo vsi ostali. Po njegovem mnenju lahko samo posameznik ali pa tudi majhna skupina zelo podobno mislečih ljudi ustvari nek osnovni načrt, po katerem bodo programerji tak program tudi izdelali.

Delitev dela pri programiranju je torej zelo jasna. V prvi fazi je potrebno potegniti jasno mejo med zasnovo ter izgradnjo. Za zasnovo je odgovoren arhitekt, ki naredi glavni načrt in ima tudi jasno predstavo, kakšen naj bi bil končni izdelek. Arhitekt je zadolžen tudi za delitev projekta na manjše podsisteme, ki jih je mogoče razvijati in implementirati v končni izdelek kar se da neodvisno. V drugi fazi pa je potrebno programerje organizirati kot kirurške ekipe, ki bodo projekt realizirale. Vsaka skupina ima svojega pod-arhitekta (v kirurški ekipi bi bil to glavni kirurg), ki je odgovoren za dodeljeno nalogo. Projekt se lahko v končni fazi spremni v mnogo plastno delitev dela, odvisno seveda od njegove kompleksnosti. Takšna delitev dela je dejansko malenkostno spremenjen pristop, ki ga je uvedel že Ford s prvo masovno proizvodnjo avtomobilov na svetu.

Kljub preizkušenemu modelu delitve dela pa še zdaleč ne gre za popolno rešitev izgradnje programske opreme. Težko je namreč nadzorovati in ocenjevati kvaliteto dela programerjev oziroma napisane kode, učinkovitejše metode za takšno ugotavljanje so zelo drage. Kot marsikje drugje tudi za programerje velja rek, da je kvaliteta pomembnejša kot kvantiteta. Prav tako je težko izločiti ljudi, ki zasluge pri delu pridobivajo na račun tujega dela. Veliko teoretikov ter programerjev se osredotoča na pisanje literature za izboljšanje metod izgradnje programske opreme, toda nekako še vedno prevladuje hudomušno mnenje, izpeljano iz Churchillovega mnenja o demokraciji: »Opisana metoda izgradnje programske opreme je najslabša možna, razen vseh ostalih, ki so še na voljo«.

Toda Brooks meni, da bistvo problema razvoja kompleksnih programov leži že v sami zasnovi takega programa, in bo ostal prisoten ne glede na izboljšave, ki so narejene v sami realizaciji projekta. Iz tega izhaja tudi Brookov zakon (Brookov zakon, 2009), ki je sledeč: Če procesu nastajanja programske opreme, ki zaostaja za urnikom, dodamo dodatno delovno silo, to ne bo pospešilo procesa nastajanja, temveč ga bo celo upočasnilo. Če denimo število programerjev, ki delajo na projektu, narašča proti neskončno (n), delo ki ga ti programerji opravljajo v najboljšem primeru tudi narašča proti neskončno (n), vendar dovzetnost za napake in hrošče narašča s kvadratom neskončnosti (n2). To v praksi pomeni, da povečano število programerjev hitreje ustvarja težave kot pa rešitve. Toda ta zakon kot tudi nekatere druge ugotovitve iz zgornjega sestavka ne veljajo celoti v primeru razvoja odprte kode.

3.1.2 Prosta programska oprema (pristop tržnice)

Razvoj proste programske opreme poteka na precej drugačen način kot razvoj klasične, ki je v večini primerih sinonim za lastniško programje. Zelo priljubljena

podoba razvoja nelastniške odprtokodne programske opreme je orientalska tržnica. Na njej mrgoli trgovcev (razvijalcev) ter kupcev (uporabnikov) in med njimi včasih ni prave ločnice. vsakdo dela tisto, kar mu najbolj ustreza, hkrati pa vsi skupaj še vedno tvorijo celoto, tržnico oziroma bazaar.

Ključni lastnosti razvoja proste programske opreme sta prostovoljna udeležba v procesu razvoja ter povsem svobodno izbiranje zadolžitev, s katerimi se razvijalec želi ukvarjati.

V proces razvoja se lahko vključi ali iz njega odide kdorkoli in to kadarkoli želi. V takšnem razvojnem procesu ni zavestno organizirane delitve dela, saj lahko vsakdo sam izbira s čim se bo ukvarjal. Določena delitev dela je vsekakor prisotna, saj drugače ne bi dočakali zahtevnejših prostih programov, vsekakor pa industrijska definicija delitve nalog odprtokodnemu procesu ne ustreza.

Bistvo vsakega odprtokodnega razvojnega projekta ali procesa je v prosto dostopni izvorni kodi, ki jo največkrat pridobimo preko interneta. Od te točke dalje so smernice razvoja odvisne predvsem od licence, pod katero je takšna izvorna koda izdana. Licence tipa BSD uporabnika izvorne kode omejujejo minimalno in omogočajo, da izvorno kodo nekdo spremeni ter jo v nadaljevanju izda kot lastniško programsko opremo. Na drugi strani licence tipa GPL poznajo mnogo več omejitev in ne tolerirajo praktično nobenega udejstvovanja v smeri lastniških programov.

Ključ do uspeha prostih programov pa ni samo v individualnem udejstvovanju programerjev, pomembno je, na kakšen način ti posamezniki sodelujejo s skupnostjo. Tu se znova pojavijo razlike med licencami, ki pogojujejo uporabo in razširjanje odprte kode in njenih derivatov. V projektu pod licenco tipa BSD smo največkrat priča majhni skupini razvijalcev, ki sama spiše večino kode za celoten projekt. Zunanji uporabniki lahko vedno prispevajo svoje izboljšave in predloge, vendar se razvijalci projekta največkrat na te prispevke pretirano ne zanašajo, poleg tega pa navadno tudi nimajo nekega standardiziranega vmesnika za sprejem takšnih izboljšav in predlogov.

V projektih, ki so zaščiteni z GPL licenco (med njih spada tudi Linux) pa je največkrat ključno, da uporabniki neprestano ponujajo svoje rešitve vodjam projekta. Tu niso mišljena zgolj opozorila na napake in predlogi izboljšav, skupnost k projektu prispeva dejanske izboljšave same kode. Celoten proces oziroma organizacija je zasnovana tako, da v največji možni meri spodbuja vsakogar, da prispeva k razvoju projekta. Ker gre za ogromno število razvijalcev in uporabnikov marsikdaj ni mogoče postaviti meje med njimi. Takšen pristop se v največji meri zgleduje po Stallmanovem FSF-u.

Na začetku tega poglavja sem kot idealno prispodobo razvoja prostega programja omenil orientalsko tržnico. Takšna tržnica morda deluje kaotično, toda pri razvoju največjih odprtokodnih projektov gre za organizirano obliko kaosa. Eden najuspešnejših odprtokodnih projektov je zagotovo razvoj Linuxa. V primeru Linuxa razvijalci med seboj v največji meri komunicirajo preko elektronske pošte, izražajo svoja različna mnenja, pogostokrat pride tudi do nesporazumov ali prepirov. Toda ko uporabnik – programer prijavi neko izboljšavo Linuxove kode, obstaja točno določen

postopek, preko katerega to izboljšavo zavrnejo ali morda vključijo v naslednjo izdajo programa. Vsaka izboljšava se mora prebiti skozi več hierarhičnih nivojev, od t.i. vratarjev do vzdrževalcev, ki imajo pregled nad določenim delom kode, do poročnikov, ki so zadolženi za večje segmente kode. Končno odločitev o vključitvi popravka v naslednjo izdajo pa ima edino Linus Torvalds. Manjši odprtokodni projekti imajo seveda precej enostavnejše postopke in organizacijsko strukturo, obstajajo pa tudi taki, ki poznajo dejansko formalno ureditev. Eden izmed takšnih je odprtokodni projekt Apache (Apache, 2009), najbolj popularen spletni strežnik na svetu. Apache ima denimo svoj lasten odbor, ki z večino glasov potrdi vsako spremembo v kodi. Najpoglavitnejše vprašanje pri vseh omenjenih projektih pa je predvsem, zakaj je njihova organizacija dejansko stabilna? Zakaj ljudje upoštevajo postavljena pravila, če jih nič ne zavezuje k njih upoštevanju?

V praksi se zgodi, da danih pravil dejansko ne upoštevajo vsi. Ko nekomu npr. zavrnejo popravek, v katerega je vložil mnogo truda, ima taka oseba zelo jasno alternativo; vedno se lahko odcepi od originalnega projekta ter ustanovi svojega lastnega in k sodelovanju povabi somišljenike. Gre za tako imenovani »forking«. Po mnenju nekaterih razvijalcev je to celo ključna lastnost odprtokodnih projektov. Kljub temu pa so primeri takšnih cepitev dokaj redki saj zahtevajo ogromno časa in znanja.

V nadaljevanju se bom nekoliko podrobneje posvetil osebam, ki ustvarjajo odprto kodo, njihovi organiziranosti in motivom za sodelovanje.

3.2 SODELUJOČI V PROCESU NASTANKA ODPRTE KODE

Na začetku bi bilo smiselno povedati, koliko ljudi se dejansko ukvarja z razvojem odprte kode vendar je takšno število nemogoče določiti. Težko pa je tudi opredeliti, kdo dejansko je razvijalec oziroma avtor, kdo pa zgolj uporabnik. Med avtorje namreč lahko prištejemo vsakega študenta, ki za svoje lastne potrebe preuredi določen program ali pa predlaga neko izboljšavo ostalim programerjem. Seveda obstajajo poizkusi merjenja števila ustvarjalcev. Eden takšnih je tudi SourceForge (Sourceforge, 2009), projekt ki pomaga razvijalcem programov z naborom orodij ter hrani evidenco vseh registriranih izdelkov in avtorjev. Februarja leta 2009 je imela njihova spletna stran registriranih peko 2 milijona uporabnikov in več kot 230 tisoč različnih projektov, večinoma sicer zelo majhnih.

The Linux Counter (The Linux Counter, 2009) je spletna stran, ki skuša z različnimi metodami izmeriti število aktivnih uporabnikov Linuxa. Ena izmed teh je štetje uporabnikov ki se registrirajo na spletni strani. Ocenjujejo, da je registriranih med 2,5 in 5 odstotki vseh uporabnikov kar v skrajnem primeru pomeni preko 60 milijonov aktivnih uporabnikov. Veliko ostalih poizkusov merjenja se prav tako osredotoča na Linux, še vedno gre namreč za najbolj znan in razširjen odprtokodni projekt.

Veliko raziskav se osredotoča na količino oziroma pomembnost prispevka posameznika. Vsi namreč nikoli ne prispevajo enakega deleža k razvoju projekta in

pogostokrat je v veljavi pravilo 80-20; 80 odstotkov dela je postorjenega s strani 20 odstotkov razvijalcev. Projekt imenovan The Orbiten Free Software Survey (OFSS, 2009), katerega namen je bil v osemnajstih mesecih v letih 1999 in 2000 izmeriti količino prispevka posameznika k odprtokodnemu projektu. Program, ki ga je v ta namen napisal Rishab Ghosh so namestili v ozadje nekaterih odprtokodnih projektov (šlo je za relativno majhen vzorec) in prišli do ugotovitev da 10 odstotkov najbolj prizadevnih razvijalcev naredi 71 odstotkov vsega dela. Te ugotovitve so potrdile tudi kasnejše raziskave na nekaterih ostalih projektih. Veliko večino dela postori ozek krog programerjev. Raziskave so tudi potrdile, da večina sodelujočih, kar 90 odstotkov, sodeluje le v enem projektu, približno 2 odstotka pa tudi v šestih in več.

Izsledki raziskav potrjujejo sliko orientalske tržnice kot model razvoja odprtokodne programske opreme. Tudi struktura sodelujočih je zelo raznolika in mednarodno razgibana. Ena izmed raziskav je tudi primerjala končnice domen elektronskih naslovov sodelujočih v projektu Linux Software Map (Boutell.com, 2009). Projekt je omogočal, da so avtorji javno objavili svoje prispevke k Linuxu ter opisali njihove funkcije. Predpostavili so, da nekdo s končnico elektronskega naslova .de prihaja iz Nemčije, nekdo s končnico .it iz Italije itd. Ugotovili so, da je 37 odstotkov vseh sodelujočih iz evropskih držav, odstotek pa je v praksi še večji saj za nekatere končnice (denimo .com ali .org) ni mogoče ugotoviti geografskega izvora. Triindvajset odstotkov vseh programerjev, ki so objavili svoje delo na LSM je imelo končnico elektronskega naslova .com. Iz tega je možno posredno razbrati tudi to, da določen odstotek sodelujočih v odprtokodnih projektih to počne tudi v službeno korist. Izjave določenih razvijalcev so namreč pokazale, da uslužbenci v podjetjih radi uporabljajo prosto programje saj ga lahko prilagodijo svojim potrebam, sproti pa odpraviti tudi napake.

3.3 KAJ DEJANSKO DELAJO RAZVIJALCI?

Kot sem že omenil uporabniki proste programske opreme niso nujno klasični uporabniki. V večini primerov so namreč vključeni v razvojni proces prostega programja. Seveda pa to ne velja za vse. Vsakdo se lahko odloči da bo odprtokodni program uporabljal kot običajen lastniški program, vendar smisel odprte kode ostaja, da spodbuja ravno sodelovanje, kopiranje in nadaljnje širjenje prostih programov v skladu z licenčnimi določili (Weber, 2004, str. 72-82).

Razvijalci prostega programja se učijo iz napak svojih predhodnikov. Ta proces se ni razvil ali nastal na podlagi teoretičnih osnov. Se pa programerji navadno držijo sedmih ključnih smernic, opisanih v nadaljevanju.

3.3.1 Naredi projekt zanimiv in izvedljiv

Sodelujoči programerji lahko prostovoljno izbirajo svoje zadolžitve in navadno izberejo zanimive, privlačne, predvsem pa pomembne naloge ki bodo na ostalo skupnost naredile pozitiven vtis. Programiranje je večinoma sicer težavno in tudi težaško opravilo. Pobudnik nekega odprtokodnega projekta lahko zato le upa, da se

bodo nekomu izmed množice sodelujočih zdela zanimiva tudi tista težavnejša in manj privlačna opravila, ki jih večina sicer spregleda. Iz tega vidika je zelo pomembna velika baza udeležencev, ki imajo različna znanja, motive in zanimanja. Nekateri vodje oziroma pobudniki procesa zato pogostokrat oglašujejo določene manj privlačne zadolžitve da pritegnejo skupnost tudi k tem nalogam. Najbolj karizmatični voditelji projektov, med katere zagotovo spada tudi Linus Torvalds, pa lahko napravijo še korak dlje. Javno lahko izjavijo, da določen del projekta sicer ni zanimiv, vendar je nujno da nekdo prevzame odgovornost zanj. Takšen segment projekta hitro dobi privržence, saj mnogokrat obstaja neka nagrada znotraj skupnosti za osebe, ki uspešno opravijo nalogo. Poleg tega pa je veliko vredno že spoznanje, da jih je k delu pozval sam (karizmatični) vodja.

Vodja projekta se tako sooči z izzivom: najti mora ravnovesje med obstoječimi izzivi in zagotovili, da bodo ti izzivi v končni fazi tudi izpolnjeni. Uporabniki-ustvarjalci namreč iščejo vedno nove izzive in gledajo predvsem na to, da projekt ne bo obtičal v slepi ulici temveč se bo razvijal naprej in bo iz njega nastalo še veliko uporabnih produktov.

3.3.2 Reakcija na dražljaj

Veliko programerjev (ne samo tistih, ki se ukvarjajo z odprto kodo) uživa pri reševanju otipljivih problemov, ki se pojavijo v nekem projektu. Gre za neke vrste dražljaj, ki v neki meri pojasnjuje tudi prostovoljno udeležbo pri ustvarjanju odprte kode. Bistvo je, da je ta dražljaj povezan z zadovoljstvom ob rešitvi problema. Lahko ga primerjamo s članstvom v določenih organizacijah, kjer v zameno za članarino plačnik dobi neko dobrino, npr. majico, ki je še nek dodaten motiv ali dražljaj poleg samega članstva.

Celotna industrija programske opreme se lahko primerja z ledeno goro, povprečen opazovalec ali celo uporabnik vidi samo približno petindvajset odstotkov celotne strukture. V praksi to predstavlja programsko opremo, ki jo neko podjetje napiše za prodajo in uporabo zunaj svojih meja, torej je dostopna na trgu. Kar petinsedemdeset odstotkov vse programske opreme pa je napisane zgolj za interno rabo in narejene za zelo specifična opravila in nikoli ni na voljo v prosti prodaji. Večina programerjev, ki so za svoje delo plačani torej skrbi zgolj za vzdrževanje in odpravljanje napak pri takšnih programih.

Večina zgoraj omenjenih programskih problemov oziroma dražljajev se torej pojavi in reši zgolj za interne potrebe znotraj podjetij ali celo v domačem okolju. Podoben princip deluje tudi v odprtokodnih skupnostih. Glavna razlika in hkrati prednost teh skupnosti je, da so razvile nekakšen sistem ali zbiralnik, v katerega kapljajo vsi ti individualni prispevki in tako vsaj določen del le teh koristi skupnostnim projektom. Takšen pristop zamegli mejo med javnimi ter privatnimi dobrinami in se neposredno navezuje na naslednjo točko.

3.3.3 Ne izumljaj tople vode

Glede na to, da je programiranje zelo zahtevno in predvsem zamudno opravilo mora dober programer vedno iskati že odkrite rešitve, ki so mu na voljo. To je še zlasti izrazito v odprtokodnem procesu, kjer udeleženci največkrat niso neposredno nagrajeni za svoj trud in programirajo v svojem prostem času. V njihovem primeru je torej nesmiselno pričakovati, da bodo neko stvar pričeli graditi od začetka, če je nekdo pred njimi že zgradil povsem solidno osnovo. Prav tako imajo programerji odprte kode vsaj dve veliki prednosti pred ustvarjalci lastniške programske opreme:

- Koda se prosto giblje naokrog in spotoma raste saj uporabniki stalno prispevajo nove dodatke;
- Uporabniki in ustvarjalci vedo, da bo koda, izdana pod licencami podobnimi kot je GPL vedno dostopna in jim ni potrebno skrbeti, da bo nekoč postala nedostopna ali celo lastniška.

Ko gre za lastniško programsko opremo, katere koda ni dostopna prosto se v organizacijskih okvirih pogosto pojavi sindrom »ni narejeno pri nas«. S tem sindromom se torej soočajo programerji zaposleni v podjetjih kjer ustvarjajo lastniško programje. Ti programerji se zavedajo da za problem, s katerim se soočajo že obstaja povsem ustrezna rešitev, vendar je zaradi pravnih ali poslovnih razlogov ne smejo uporabiti. To v praksi pomeni, da morajo programsko opremo spisati od samega začetka. Vendar s stališča korporacije takšne zahteve niti niso iracionalne. Uporaba neke druge kode (zaprte ali odprte) največkrat s seboj prinese dolgoročne nezaželjene odvisnosti od konkurence.

3.3.4 Vzporedno reševanje težav

Računalniški znanstvenik Daniel Hillis (Daniel Hillis, 2009) je dejal: »Poznamo samo dva načina kako ustvariti zelo zapletene stvari: s pomočjo tehnike ali pa z evolucijo.« (Hillis v: Weber, 2004, str. 76.) Računalniški program je največkrat ena izmed takšnih kompleksnih stvari.

Frederick Brooks zagovarja tehničen način izgradnje tradicionalne programske opreme, kjer je arhitekt odgovoren za glavni načrt in razdelitev nalog svojim podrejenim. Vsaka izmed podrejenih skupin se nato ukvarja s svojimi zadolžitvami, ključna vloga arhitekta pa ostaja, da jih pravilno usmerja skozi številne različne možnosti, ki se ponujajo sproti.

Odprtokodni proces pa za razliko od tradicionalnega uporablja evolucijski pristop. Vodja ali začetnik takega projekta še vedno lahko poda neke smernice, kaj je pomembnejše od nečesa drugega, vendar to ni bistveno. Če je naloga pomembna bo najverjetneje pritegnila veliko različnih ljudi ali timov, ki jo bodo poizkušali rešiti in tako dejansko delovali vzporedno, vendar na mnogih različnih krajih in celo celinah. Takšen pristop omogoča veliko različnih rešitev in poti skozi določen proces, v neki kasnejši fazi pa nato izločitveni proces loči tiste slabše od boljših.

Oba pristopa imata tudi nekatere pomanjkljivosti. Tehnični se dobro obnese le pod pogojem da ima glavni inženir ali arhitekt dober glavni načrt in med samim izvajanjem procesa sprejema same dobre odločitve. Evolucijski pristop pa je lahko tudi krut, podobno kot narava, saj brezobzirno izloči najšibkejše člene. Težko je določiti, kateri način razvoja je boljši, bistvo odprtokodnega oziroma evolucijskega načina je, da pritegne kolikor je mogoče različnih ljudi, ki potem sami odločajo in izbirajo sredstva in poti do cilja.

3.3.5 Vpliv velikih števil

Ta točka se nanaša na preizkušanje produkta. Naloga preizkuševalcev je, da predvidijo čim več različnih situacij, v katerih se bo znašel preizkušani proizvod. Takšne situacije je mnogo lažje predvideti v primeru pralnih strojev ali avtomobilov, saj obstaja določen vzorec uporabe. Povsem drugače je pri programski opremi, ki mora delovati na številnih različnih strojnih konfiguracijah in se soočati z nešteto različnimi željami in pričakovanji uporabnikov. Glede na to, da obstaja milijone različnih poti skozi kodo nekega srednje velikega programa, je testiranje omejeno zgolj na majhen del teh poti. Velja pravilo: več napak ali hroščev kot se jih odkrije med testiranjem, boljše so možnosti da bo večina izmed njih odpravljenih in bo na trg prišel spodobno delujoč izdelek. V industriji programske opreme se tako večno pojavlja vprašanje: Koliko pomanjkljivosti ima še lahko nek končni izdelek, da je primeren za trg? Iz stališča razvijalcev je zaželjena čim bolj zgodnja izdaja (imenovana tudi beta verzija), z ekonomskega oziroma tržnega vidika pa je nedokončan program poln hroščev lahko prava katastrofa. Stranke namreč pričakujejo vsaj spodobno delujoč izdelek, slab izdelek lahko zato uniči morebitno dobro ime celotnega podjetja.

Ko primerjamo testiranje tradicionalne oziroma lastniške programske opreme s testiranjem odprtokodnih produktov, je prednost več kot očitno na strani slednjih. Podjetja, ki razvijajo lastniško programsko opremo svoje produkte predčasno izdajo ožjemu krogu ljudi, tako imenovanim beta preizkuševalcem, ki poizkušajo odkriti čim več napak in predvideti kar največ različnih načinov uporabe. Istočasno se podjetja soočajo z roki za dokončanje in izdajo svojega izdelka, ki mora biti na tržišču pred konkurenco. Ravno iz tega razloga smo mnogokrat priča polovično dokončanim programom, za katere avtorji naknadno (torej po izdaji), največkrat brezplačno ponujajo popravke dosegljive na njihovih spletnih straneh.

Odprtokodni proces je v tem pogledu precej različen od tradicionalnega in mnogo bolje izkorišča svojo veliko bazo uporabnikov-razvijalcev. Odprt program dejansko ni nikoli končan v pravem pomenu te besede. Vedno se nahaja v beta fazi. Znova je ključen element dostopnost izvorne kode, ki omogoča aktivno sodelovanje komurkoli, razvijalci pa nase gledajo kot na udeležence v stalno razvijajočem se procesu. Odpravljanje težav in hroščev nastopa kot izziv, ki ga je potrebno premagati in ne kot težava, zaradi katere lahko izgubi celotno podjetje.

Tako kot v ostalih evolucijskih procesih tudi v tem primeru veliko število udeležencev in njihova raznolikost zagotovi hitrejšo prilagoditev razmeram, konkretno gre za odkrivanje in popravljanje hroščev. Večje število ljudi kot se ukvarja s programom, več težav oziroma izzivov bodo odkrili. In več kot je udeležencev, hitreje bodo ti izzivi rešeni.

3.3.6 Delovna dokumentacija

Izvorna koda programa je poznavalcu sicer povsem berljiva, to pa še zdaleč ne pomeni, da je njeno branje lahko. Celo v zmerno velikem programu včasih tudi iz izvrstno napisane kode ni povsem razvidno, kaj je avtor skušal doseči. Kvalitetna dokumentacija navadno avtorju vzame veliko časa, nje pisanje pa je precej dolgočasno opravilo. Namenjena je predvsem ostalim ustvarjalcem, da lahko jasno razberejo, kaj je skušal izvorni avtor doseči.

V organizaciji, ki ustvarja lastniško programsko opremo ima formalna dokumentacija dokaj nizko prioriteto, več komunikacije poteka neposredno in zato imajo ostali segmenti prednost. Odprtokodni razvijalci pa se morajo mnogo bolj zanašati na dobre spremne dokumente. Različni uporabniki-razvijalci, ki so največkrat v medsebojnem stiku samo preko interneta morajo poglobljeno razumeti naravo nekega izdelka, če želijo uspešno nadaljevati z njegovim razvojem.

Realnost pa je mnogokrat drugačna od želja bodočih avtorjev odprtokodnega projekta. Dejstvo je, da avtorji odprte kode mnogokrat zanemarjajo dokumentiranje lastnega dela in navadno niti niso preveč spretni pri pisanju priročnikov. V veliko primerih to povzroča nesporazume in zaostanke v razvoju projektov ter onemogoča ponovljivost določenih procesov, kar je ključno z vidika znanosti.

3.3.7 Pogovor

Avtorji odprte kode največkrat radi govorijo o svojem delu, vse od specifičnih tehničnih ugank do splošnih smernic razvoja industrije programske opreme. Zmotno je prepričanje, da so te debate, ki večinoma sicer potekajo na daljavo preko elektronske pošte, vedno umirjene in vljudne. Debate sicer večinoma potekajo v prepričanju da za vsak tehnični problem obstaja tehnična rešitev, kar pa niti slučajno ne pomeni da se vsi strinjajo kako do nje priti. Veliko različnih mnenj največkrat ponudi tudi veliko število rešitev, zato je sodelovanje ustvarjalcev odprte kode v pogovorih dokaj pomemben vidik sodelovanja v procesu.

3.4 SODELOVANJE RAZVIJALCEV ODPRTE KODE

Vsi veliki produkcijski procesi, izjema ni niti odprtokodni, se soočajo z vprašanjem uspešnega sodelovanja. Posamezniki se trudijo, vendar je njihov trud zaman če niso povezani med seboj oziroma niso povezani njihovi izdelki. Proces ustvarjanja odprte kode je zato moral iznajti svoj lasten način sodelovanja. Za razumevanje tega vidika

so predvsem pomembni naslednji točki: Vpliv tehnologije ter razvoj odprtokodnih licenc (Weber, 2004, str. 82-88).

3.4.1 Vpliv tehnologije

Že relativno dolgo časa je internet glavno sredstvo komuniciranja, širjenja in deljenja tako mnenj kot tudi datotek, kar je izjemno pomembno za razvoj odprte kode. Na začetku tega besedila sem že omenil, da je bilo pred iznajdbo interneta sodelovanje na področju razvoja informacijskih tehnologij (pa tudi ostalih panog) oteženo. V ZDA so obstajali trije večji centri razvoja programske opreme, Bellovi laboratoriji, MIT-ov laboratorij za raziskovanje umetne inteligence in Kalifornijska univerza Berkeley. Bolj kot so bili navdušenci oddaljeni od teh centrov, manj jih je bilo in težje so prišli v stik z novostmi. Prenašanje podatkov na diskih s pomočjo avtobusov in letal pa je bilo tako drago kot naporno.

Internet je bil torej ključ do razmaha odprte kode. Zbližal je ljudi in izbrisal geografske razlike. Razširjanje datotek in izkušenj je tako postal samoumeven proces. Z razvojem interneta pa so se stalno povečevale možnosti za razvoj programske opreme. Vzpostavljene so bile različne baze podatkov, na voljo je bilo vedno več orodij za delo in ne nazadnje je bilo sodelovanje omogočeno vedno širšemu krogu ljudi.

3.4.2 Razvoj odprtokodnih licenc

V nasprotju s pričakovanji so odprtokodni avtorji pogostokrat izjemno goreči zagovorniki intelektualne lastnine ter avtorskih pravic. Redko se primeri, da ti avtorji svoje izdelke dajo na trg in se enostavno odpovedo vsem zaslugam. Odprtokodni proces temelji na seriji licenc, ki temeljijo na povečevanju uporabe, razvoja, rasti ter distribucije odprtokodnih izdelkov.

Odprtokodni proces temelji na domnevi, da želijo biti ljudje ustvarjalni in ne potrebujejo veliko dodatnih pobud za delovanje. Praktično edina ovira, ki se lahko pojavi v odprtokodnem procesu je tako nedostopnost osnovne dobrine za delo, torej izvorne kode. Poglavitna naloga licenc, ki pokrivajo področje odprte kode je ustvariti družbeno strukturo, ki se bo še naprej širila in jo najbolje zaznamujejo naslednje točke:

- Pooblaščajo uporabnike k dostopu do izvorne kode;
- Večino pravic do uporabe kode raje prenesejo na uporabnika kot da jo zadržijo za avtorja;
- Preprečujejo uporabnikom, da bi ti omejili dostopnost ostalim uporabnikom in tako ščitijo osnovno načelo odprte kode.

Vsaka licenca ta načela udejanja nekoliko drugače, kot sem že omenil zgoraj, je denimo licenca tipa BSD precej manj zavezujoča kot licenca tipa GPL.

3.5 REŠEVANJE NESPORAZUMOV MED RAZVIJALCI

Nesporazumi med zelo inteligentnimi, visoko motiviranimi in največkrat pretirano samozavestnimi ustvarjalci so pravzaprav stalnica vsakega razvojnega procesa in izjema ni niti odprtokodni. Vzroki za konflikte so največkrat sledeči trije (Weber, 2004, str. 88-92):

- Kdo je pristojen za sprejemanje končnih odločitev, ko gre za več dobronamernih vendar medsebojno nezdružljivih rešitev v korist nekega projekta?
- Komu bodo dodeljene zasluge za določeno (uspešno) opravljeno zadolžitev?
- Možnost odcepitve od projekta, predvsem v kontekstu kdo je upravičen in sposoben prevzeti nase takšno odgovornost in pod kakšnimi pogoji (pravica do same odcepitve vedno na voljo).

Podobni vzroki konfliktov se pojavljajo tudi v tradicionalnem procesu razvoja programske opreme vendar obstajajo mehanizmi, s katerimi jih je mogoče odpraviti. Težava odprtokodnega procesa je, da ti mehanizmi v njem ne delujejo. Največkrat namreč ni izrazitega vodje, ki bi lahko udejanjil neko odločitev, kako nadaljevati s projektom; ni poslovodje z močjo odpuščanja in zaposlovanja delavcev; navsezadnje pa tudi ni formalnega postopka za pritožbe na določeno odločitev.

V odprtokodnem procesu sta za reševanje konfliktov ključna predvsem narava vodstva in struktura odločanja.

Glede vodstva se javnost največkrat osredotoča na Linusa Torvaldsa, avtorja Linuxa. Njegov slog vodenja je unikaten. Čeprav gre po naravi za dokaj sramežljivo osebo, ga razvijalci spoštujejo in priznavajo kot avtoriteto, predvsem zaradi njegove predanosti projektu in intelektualnih sposobnosti. Skozi vsa leta razvoja je Torvalds ohranil vizijo razvoja Linuxa in v praksi se je ta vizija največkrat izkazala za pravilno ter še bolj utrdila zaupanje vanj. Eden izmed najbolj prepoznavnih elementov Torvaldsovega vodenja je tudi široka obrazložitev večine bolj zapletenih odločitev, ki jih je sprejel. Pozitivno na skupnost vpliva tudi njegova lastnost, da brez težav priznava napake ki jih je napravil med odločanjem. Vse te njegove lastnosti pomenijo, da lahko svoje privržence razočara edino še s tem, da postane neodziven, torej preneha z odgovarjanjem na pobude, mnenja, vprašanja. Javnost tako Torvaldsa glede na njegov način vodenja pogosto označuje za dobrohotnega diktatorja, diktatorja, ki ga večina nikoli ne bo srečala v živo, vseeno pa se prostovoljno podreja njegovim odločitvam.

Seveda pa Torvaldsov stil vodenja še zdaleč ni edini uspešen. Richard Stallman, ustanovitelj FSF-ja, je svoje vodstvene sposobnosti razvil predvsem po zaslugi izjemnega znanja o programiranju. V nasprotju s Torvaldsovo filozofijo Stallman svojo vlogo v FSF-ju uporablja za goreče zagovarjanje predvsem etičnih in moralnih načel glede odprte kode.

Prav tako kot vodstvene sposobnosti posameznikov je v odprtokodnem procesu pomembna struktura odločanja. Večina majhnih projektov funkcionira tako, da ena oseba sprejema vse odločitve o spremembah in dodatkih k izvorni kodi ali jedru

programa in v zgodnji fazi razvoja ni bil izjema niti Linux. Linux pa ima danes povsem drugačno strukturo odločanja. Torvalds se veliko zanaša na ožji krog programerjev imenovanih tudi poročniki, vsak izmed njih je zadolžen za določen del programa in koordinira delo ostalih razvijalcev. Poročniki največkrat delegirajo zadolžitve več vzdrževalcem, struktura se tako razteza do povsem navadnih uporabnikov-programerjev. Pomembno je, da gre za neuradno strukturo organiziranosti. Komunikacija in sprejemanje odločitev sicer poteka proti vrhu piramide, vendar ta struktura ni nikjer jasno zapisana, jo pa udeleženci projekta največkrat priznavajo.

Če gre v primeru Linux-a za projekt, ki temelji na licenci GPL, pa se v projektih, ki nastanejo na podlagi licence tipa BSD navadno razvije drugačen sistem odločanja, ki temelji na koncentričnih krogih. Majhna skupina programerjev ima popoln dostop do jedra programa in ta skupina daje ali odvzema vse dostopne pravice naslednjemu krogu ali skupini programerjev. Število programerjev v posameznem krogu pa se razlikuje od projekta do projekta.

Vsi načini vodenja in odločanja imajo svoje prednosti ter pomanjkljivosti. Vsem je skupno to, da ne obstaja nobena sila ali avtoriteta, ki bi lahko prisilila kogarkoli v opravljanje določene naloge. Prav tako ni nobene omejitve ki bi preprečevala, da posameznik ali skupina iz projekta kadarkoli izstopi.

3.6 PREDNOSTI ODPRTE KODE

Zagovorniki odprte kode radi poudarjajo njene prednosti in pri tem izpostavljajo predvsem področje varnosti in stabilnosti odprtih programov. Medtem poudarjajo, da avtorji lastniških programov preveč energije in truda vlagajo v vizualne atribute in zanemarjajo delovanje. Za avtorje odprte kode to nikakor ni značilno, saj svoje zanje usmerjajo predvsem v lastnosti programa, ki bodo pri kolegih cenjene in sprejete z zanimanjem. Takšna usmerjenost izhaja tudi iz dejstva, da večina odprtokodnih avtorjev v programiranju išče izziv in želi poglobiti svoje znanje. Pri razvoju programa se zato raje osredotočajo na lastnosti kot so jasen načrt dela, zanesljivost, enostavno vzdrževanje programa ter spoštovanje načel odprtih standardov ter skupnosti. (Primožič, 2005, str. 29-32)

- Zanesljivost

Zanesljivost prostega ali lastniškega programa v praksi pomeni odsotnost hroščev oziroma napak, ki bi povzročile nepravilno delovanje programa in izgubo podatkov. Pri odpravljanju takšnih napak imajo odprtokodni programi ogromno prednost saj se odprt program praktično v vsakem trenutku nahaja v preizkusni dobi, morebitne pomanjkljivosti pa odkriva in popravlja celotna skupnost. Večja zanesljivost odprtih programov v primerjavi z lastniškimi torej izhaja predvsem iz »vpliva velikih števil«. Sama količina ljudi, ki preizkuša program je neprimerno večja kot pri testiranju lastniške programske opreme, prav tako pa je pomembno tudi dejstvo, da se odprto programsko opremo izdaja na trg veliko pogosteje kot lastniško. Zaradi te lastnosti se popravki nekaterih napak pojavijo samo nekaj ur za izdajo izvorne kode.

- Stabilnost

Spremembe v odprtih programih niso tako pogoste niti nujne kot je to v navadi pri lastniških programih. Pri slednjih namreč podjetja pogosto spodbujajo uporabnike k nadgradnji zastarelih različic ali dodajanju podpore novim formatom četudi to ni nujno potrebno, saj ta podjetja žene ekonomski motiv oziroma zaslužek. Pojavljajo se celo pritiski v obliki ukinitve tehnične podpore starejšim različicam programov ali nezdružljivostjo starejših verzij z novejšimi. Odprti standardi pa takšnih motivov nimajo zato je svoboda uporabnika v smislu nadgrajevanja in spreminjanja programske opreme večja. Odprti standardi se največkrat tudi ne spreminjajo v tolikšni meri, da starejše različice ne bi zadostovale za uspešno opravljanje dela in tudi težav z združljivostjo navadno ni.

- Preglednost

Gre za s strani uporabnikov dokaj zanemarjen vidik, ki pa v zadnjem obdobju pridobiva na pomembnosti. S strani avtorjev in ponudnikov lastniške programske opreme smo mnogokrat priča obljubam o varnosti njihovih izdelkov, neobstoju stranskih vrat (angl. *backdoors*), prilagodljivosti različnim standardom in fleksibilnosti za bodoče spremembe. V resničnost teh obljub se je nemogoče prepričati brez dostopne izvorne kode, kar pogosto povzroča preglavice tudi pristojnim inštitucijam. V primeru dostopne izvorne kode pa je možno vse takšne trditve preveriti, avtorji programa z objavo vlijejo v uporabnika zaupanje.

- Fleksibilnost in svobodnost

V poslovnem kontekstu fleksibilnost programske opreme pomeni možnost izbire rešitve, ki bo zadovoljila vse potrebe uporabnikov. V primeru vpeljave informacijskega sistema v organizacijo se pogosto pojavi prevelika odvisnost od enega proizvajalca programske opreme. Le-ta ima nato velik vpliv na zmožnost opravljanja delovnega procesa. Uporaba prostega programja zmanjšuje takšno tveganje in uporabnikom ponuja več svobode pri izbiri ali nakupu ostalih programov, ki jih potrebujejo za delo.

Odvisnost od enega proizvajalca programske opreme je problematična predvsem za to, ker se ta proizvajalec lahko odloči prenehati z razvojem programa. Ker je program zaščiten z licenco zaprtega tipa brez dostopne izvorne kode, ima podjetje, ki program uporablja zvezane roke v smislu nadaljnjega razvoja aplikacije (razen v primeru da jim razvijalec proda pravice in omogoči dostop do izvorne kode, kar pa v večini primerov pomeni velike stroške). Tako je edina možnost da takšno podjetje zamenja programsko opremo kar poleg nepričakovanih stroškov povzroči še težave pri prilagajanju zaposlenih. Ob uporabi odprte programske opreme takšnih težav ni, tudi če odprtokodna skupnost izgubi zanimanje za določen program, lahko podjetje vedno najame skupino programerjev, ki bo zadolžena za nadaljnji razvoj ter vzdrževanje.

Poleg tega pa odprtokodni projekti omogočajo veliko večjo prilagodljivost in kombinacijo posebnosti določenega programa. Veliko ljudi si namreč prost program

povsem prilagodi svojim lastnim potrebam zato je mnogokrat mogoče najti različico ki nam ustreza ali pa za nje razvoj najeti strokovnjake

- Stroški

Nižji stroški pri uporabi odprtokodne programske opreme se navadno izkažejo šele na daljši rok, kratkoročno so največkrat celo višji v primerjavi z uporabo lastniške programske opreme. Dejavnikov, ki govorijo v prid uporabe proste programske opreme predvsem v večjih podjetjih je mnogo. Eden izmed glavnih se nanaša na največkrat nične stroške nakupa programa, brezplačno je tudi nadaljnje obnavljanje licenc. Odprto programje je možno brez licenčnih stroškov naložiti na neomejeno število računalnikov, prav tako je brezplačno kakršnokoli posodabljanje programov. Prosto programje je tudi manj ranljivo za viruse ter napade hekerjev in ima navadno manj varnostnih lukenj. V tem primeru se tako zmanjšajo predvsem stroški pri pregledovanju in čiščenju sistema za virusi, zmanjšajo se tudi stroški sistemske administracije. Pomemben dejavnik pri zniževanju stroškov pa je tudi možnost uporabe starejše strojne opreme saj odprto programje običajno ni tako strojno zahtevno kot lastniško.

- Podpora

Poleg namenskih podjetij, ki se ukvarjajo s plačljivim svetovanjem za določene proste programe, obstaja velika verjetnost, da je odgovor na določeno vprašanje mogoče dobiti brezplačno preko različnih spletnih klepetalnic. Poleg tega pa razvijalci prostih programov v razvoj izdelka nemalokrat vložijo ogromno truda zato jim je v interesu, da bodo ljudje njihov produkt uporabljali in priznali. Iz tega razloga pogostokrat žrtvujejo še nekaj dodatnega časa, ki ga posvetijo reševanju težav uporabnikom.

3.7 SLABOSTI ODPRTE KODE

Najbolj izpostavljene slabosti prostega programja so slabša združljivost z ostalimi programi in zapleteni uporabniški vmesniki. Pogostokrat pa se kot resna težava pojavi tudi zahtevna namestitev takšne programske opreme. Glavne slabosti so opisane v nadaljevanju (Primožič, 2005, str. 32-34):

- Združljivost

Prosto programje ni vedno najbolje združljivo z ostalimi aplikacijami. Uporabniki to najbolj občutimo v primeru Microsoftovega operacijskega sistema Windows. Odprto programje namreč navadno Windows okolja ne izkoriščajo dovolj, zato se pojavijo napake in nevšečnosti. Glede na to, da ima Microsoft z Windowsi še vedno prevladujoč položaj med povprečnimi uporabniki nezdružljivost predstavlja precejšno težavo odprtega programja.

- Uporabniški vmesnik

Uporabniški vmesniki prostega programja so največkrat bolj kompliciran kot vmesniki lastniških programov. Napisani so »od programerjev za programerje« in ne za

nezahtevne uporabnike, ki posedujejo malo tehničnega znanja. To izhaja predvsem iz tega, da nezahtevni uporabniki ne sodelujejo v procesu nastanka prostega programa (ker nimajo dovolj tehničnega znanja), zato tudi ne podajo svojih želja in pričakovanj razvijalcem, ki vidik vmesnikov zato največkrat »zanemarijo«.

- Podpora

Podpora je največkrat izpostavljena kot močna stran odprtega programja vendar ima tudi svoje pomanjkljivosti. Pogosto je namreč namenjena razvijalcem, zato vsebuje mnogo tehničnih izrazov in za njeno razumevanje uporabnik potrebuje veliko tehničnega znanja. Dokumentacija je pogosto neažurna zaradi pogostih izdaj programa, tako da je edino pravo vodilo kar izvorna koda sama, le-ta pa je neposredno podlaga za programsko dokumentacijo.

- Namestitev aplikacije

Večina lastniške programske opreme je tipa »vstavi in zaženi« (angl. *plug and play*) kar pomeni enostavno namestitev in takojšnjo uporabo. Veliko odprtokodnih programov močno zaostaja za tem principom, namestitev mnogih aplikacij je prezahtevna za neveščega uporabnika in pogosto niso podrti vsi gonilniki za naprave. Stanje na področju odprtega programja se sicer popravlja. Nekatere bolj razširjene odprte aplikacije vodijo uporabnika skozi namestitev kot je to običaj pri komercialnih produktih, vendar še vedno med potekom namestitve uporabnika obremenjujejo z nepotrebnimi zahtevki.

- Intelektualna lastnina in pomanjkanje lastništva

Pri odprti programski opremi zaradi velikega števila razvijalcev mnogokrat obstaja možnost, da se bo v takšnem programu pojavil del komercialne oziroma zaprte kode. Takšna koda je največkrat avtorsko zaščitena na način, ki izključuje uporabo v kateri koli drugi aplikaciji. To se največkrat pripeti v primeru, ko programer zaposlen v določenem podjetju napiše del kode za komercialni program, v svojem prostem času pa to isto kodo vključi še v odprti program pri razvoju katerega sodeluje. Podjetja ki razvijajo lastniške programe so veliko bolj pozorna na to težavo in svojim uslužbencem prepovedujejo uporabo kode, ki je bila napisana izven podjetja saj kršitev avtorskih pravic predstavlja trdno podlago za tožbo. Poleg tega se prednost komercialnih produktov skriva v dejstvu, da lahko podjetje nastopa kot zaupanja vreden skrbnik in zagotavlja npr. združljivost določenega programa s predhodnimi različicami. Če se podjetje obljube ne drži je možno pravico izterjati preko sodišča, medtem ko v primeru odprtokodne skupnosti tak pristop ni možen in se uporabniki lahko zanesejo zgolj na dobro voljo razvijalcev.

- Težak začetek in konec

Ena izmed največjih težav odprtokodnega projekta je ustvariti dovolj široko skupnost razvijalcev oziroma ljudi, ki bodo kakorkoli pripomogli k razvoju in širitvi projekta. Razvijalci morajo imeti jasno definiran skupen cilj kar pa ni vedno najlažje izvedljivo. Veliko takšnih projektov pa ima tudi težave s prehodom v stabilno stanje saj

mnogokrat zanimanje za razvoj upade že pred tem. Tudi to je vzrok da večina odprtokodnih projektov nikoli ne zapusti beta faze razvoja.

4 ODPRTA KODA V SLOVENIJI IN EVROPI

Poglavje govori o možnostih uporabe odprte kode v javnem sektorju. Opisane so nekatere organizacije za promocijo odprte kode v Sloveniji in Evropi. Predstavljena je pobuda poslancev za prenovo informacijskih sistemov v slovenski javni upravi in odgovor Vlade ter nekatere splošne ugotovitve Vlade Republike Slovenije glede uvajanja odprte kode. Podani pa so tudi primeri uporabe odprtokodnih aplikacij v več evropskih državah.

4.1 ORGANIAZCIJE ZA PROMOCIJO ODPRTE KODE

4.1.1 Center odprte kode Slovenije

Center odprte kode Slovenije (COKS, 2009) je nacionalni spodbujevalec razvoja, uporabe in znanja o odprtokodnih tehnologijah in rešitvah. Razvojno podporni center odprte kode Slovenije nudi uporabnikom centraliziran sistem storitve pomoči in podpore ter zagotavlja rešitve za potrebe javnega in zasebnega sektorja. Poslanstvo COKS je pospeševanje razvoja odprtokodnih rešitev, spodbujanje uporabe in oblikovanja dobrih praks ter širjenje znanja in vedenja o prednostih odprtokodnih informacijskih rešitev tako v gospodarstvu, javnem sektorju, kakor tudi med končnimi uporabniki (Poročilo o ustanovitvi in pričetku dela, 2007).

Center odprte kode Slovenije je nastal na podlagi pogodbe o sodelovanju, podpisane sredi leta 2007, kot konzorcij osmih organizacij komercialnega, raziskovalnega in izobraževalnega oz. akademskega značaja. Pobudo zanj je dalo podjetje Agenda d.o.o., ki je tudi koordinator konzorcije.

Delovanje COKS temelji na filozofiji odprte kode, ki je razumljena kot inovacija v pogledu načina produkcije programske opreme in njene distribucije. Delovati je pričel konec leta 2007, ko je pred tem uspel s prijavo na razpis Ministrstva za visoko šolstvo, znanost in tehnologijo. Ministrstvo je preko razpisa iskalo izvajalca za razvojno podporni center za odprto kodo v Sloveniji. COKS uporabnikom nudi storitve preko spletnega portala (www.coks.si), na voljo pa je tudi brezplačno svetovanje preko elektronske pošte in klicni center za pomoč uporabnikom.

COKS precej pozornosti posveča tudi trenutno monopolnemu položaju Microsoftove programske opreme v slovenski javni upravi ter izobraževalnih inštitucijah. Njihova prizadevanja so usmerjena tudi v uvajanje prostega programja v omenjene ustanove. Pri tem se sklicujejo na usmeritve Evropske Unije, ki prednost predpisujejo odprti kodi in katerih podpisnica je tudi Vlada Republike Slovenije.

COKS v skladu z odprtokodnimi načeli ostaja odprt konzorcij, ki k sodelovanju vabi vse posameznike ali podjetja in organizacije, ki bi na kakršenkoli način pripomogla k širitvi odprte kode v Sloveniji in svetu. Najbolj iskana so podjetja, ki bi na območju Slovenije nudila celovite informacijske rešitve s pomočjo odprte kode. Udejstvovanje posameznikov pa je predvsem možno v obliki pomoči pri urejanju spletnih virov in foruma na portalu COKS, pomoč pri prevajanju odprtokodnih programov, navodil oziroma priročnikov ter nudenje pomoči in podpore drugim uporabnikom.

4.1.2 Kiberpipa

Kiberpipa je center, ki je bil ustanovljen leta 2000 na pobudo navdušencev iz številnih področij kot so kultura, informacijske tehnologije ter tudi z udeležbo političnih ter medijskih aktivistov. Glavna cilj je bil ustanovitev odprte informacijske družbe v Sloveniji ter spodbujanje njenega razvoja. Center je bil eden izmed prvih v Sloveniji, ki je ponujal možnost brezplačnega dostopa do interneta in ostalih modernih informacijskih tehnologij (O Kiberpipi, 2009).

Projekt Kiberpipe je sicer zastavljen zelo široko in pokriva mnogo različnih tehnoloških področij ter prepričanj. S stališča odprte kode je pomemben zato, ker se zavzema za uporabo odprtih standardov ter prostega programja ter zato tudi spodbuja njihovo uporabo in razvoj. Deluje na prepričanju, da je neoviran pretok znanja oziroma informacij nujno potreben za napredek družbe in posameznika v njej. V ta namen organizirajo številne projekte, aktivnosti, predavanja in delavnice preko katerih želijo ozaveščati tudi širšo javnost o prednostih odprtih standardov. Tako kot odprtokodno gibanje oziroma proces razvoja odprtokodnega programa, tudi delovanje Kiberpipe sloni predvsem na prostovoljnem udejstvovanju posameznikov. Skozi vsa leta delovanja pa Kiberpipa udejanja tudi eno izmed glavnih vodil organizacije, to je prost dostop do vseh aktivnosti za vsakogar.

V devetih letih obstoja je Kiberpipa navezala veliko stikov z različnimi podjetji, raziskovalnimi ustanovami ter znanstveniki. Najbolj aktivna področja trenutno so Linux in odprto programje, internet, globalna sredstva komunikacij, razvoj ter brezžično omreženje. Pozornost usmerjajo tudi na produkcijo, distribucijo in arhiviranje digitalnega videa ter prenose v živo, svoboden pretok informacij, patentiranje programske opreme in intelektualno lastnino v digitalni družbi. Kiberpipa ima med drugim tudi podporo Študentske organizacije Univerze v Ljubljani, Ministrstva za visoko šolstvo, znanost in tehnologijo, Ministrstva za Kulturo ter Mestne občine Ljubljana.

Število aktivnosti, ki se organizirajo v okviru Kiberpipe presega število 200. Center je odprt med oktobrom in junijem pet dni tedensko, letno pa beležijo preko šestdeset tisoč obiskovalcev. Nekatere aktivnosti organizirane v okviru Kiberpipe:

- Pipini odprti termini

Gre za redna tedenska srečanja namenjena predvsem uporabnikom Linuxa ter tudi ostalih odprtih tehnologij. Udeleženci skozi predavanja izvejo o različnih novih

programih in njihovih funkcijah. Ti termini so primerni tudi za manj vešče uporabnike odprtih tehnologij.

- Računalniški muzej

Namen muzeja je ohranjati računalniško in tehnološko dediščino, ki je bila osnova za razvoj današnjih modernih informacijskih tehnologij. V muzeju je moč računalniško zgodovino spoznati preko stalne razstave, predavanj in različnih tematskih dogodkov. Stalna razstava je zasnovana interaktivno kar pomeni, da je večino razstavljenih predmetov možno preizkusiti v praksi.

- Spletne urice

Dejavnost je namenjena oblikovalcem spletnih strani. Vsak teden spletni profesionalci predstavijo določeno področje oziroma novosti spletnega oblikovanja. Cilj spletnih uric je večsmerni prenos znanj med spletnimi razvijalci in grajenje skupnosti le-teh.

4.1.3 Društvo uporabnikov Linuxa Slovenije

Društvo je sicer bolj znano pod kratico LUGOS (angl. *Linux User Group of Slovenia*). Namen društva LUGOS je združevanje vseh slovenskih uporabnikov prostega operacijskega sistema Linux ter širjenje in promoviranje Linuxa med ostalimi uporabniki računalnikov. Društvo je bilo ustanovljeno 23. julija 1997. Naloge društva obsegajo združevanje računalniških uporabnikov, izobraževanje preko tečajev konferenc in delavnic ter širjenje Linuxa v akademske in izobraževalne organizacije. Poleg tega so zadolženi tudi za izdajanje revij, časopisov in brošur z informacijami oziroma navodili za uporabo UNIX-a, Linuxa in aplikacij zanje. Prevajajo tudi določene dele Linuxa, če je to smiselno in se pokaže potreba po slovenskem prevodu. Sicer pa se v društvu borijo tudi proti monopolu operacijskih sistemov na slovenskem trgu s pomočjo propagande, informativnih brošur in predstavitvami pozitivnih lastnosti Linuxa na različnih sejmih. (LUGOS, 2009)

Eden izmed pomembnejših projektov društva LUGOS je Pingo Linux. Gre za slovensko distribucijo Linuxa, ki temelji na popularnih odprtih distribucijah podjetij Fedora in Red Hat. Prva izdaja Pingo Linux 1.0 je nastala kot spremljevalna distribucija Linuxa h knjigi »Linux z namizjem KDE«. Izšla je leta 2000, temeljila pa je na takrat najbolj popularni distribuciji Red Hat 6.2.

Druga izdaja Pingo Linux 2.0 je izšla oktobra 2003 in je temeljila na Red Hat 9.0. V sklopu razpisov Ministrstva za šolstvo, znanost in šport je bil Pingo Linux 2.0 spomladi leta 2004 kot sekundarna zagonska možnost nameščen na šest tisoč novih računalnikov, ki jih je ministrstvo sofinanciralo vzgojno izobraževalnim zavodom. Tretja in četrta izdaja Pingo Linuxa sta izšli septembra leta 2004 in 2005. Za razliko od predhodnih dveh sta temeljili na distribuciji Fedora 2 in Fedora 4.

Ostali vidnejši projekti, pri katerih sodeluje društvo LUGOS pa so že omenjeno slovenjenje različnih odprtokodnih aplikacij, sodelujejo na odprtih terminih društva Kiberpipa in pomagajo pri vzpostavitvi odprtega brezžičnega omrežja v Ljubljani.

4.1.4 Open Source Observatory and Repository

Open Source Observatory and Repository (OSOR) je organizacija ki zbira, opisuje, preučuje in ponuja odprtokodne rešitve in aplikacije uporabljene v javnih sektorjih držav evropske unije. Vsem potencialnim uporabnikom odprte kode preko spletnega portala nudijo najnovejše informacije s tega področja. Poleg tega nudijo možnost varne hrambe izvorne kode in dokumentacije programja namenjenega uporabi v javnih sektorjih. OSOR pa nudi tudi prostor oziroma neke vrste spletno delavnico, ki omogoča lažje sodelovanje razvijalcev odprtih aplikacij na mednarodni ali lokalni ravni. V ta namen ponuja različna orodja za razvijanje programske opreme, v to kategorijo spadajo tudi spletni forumi in dopisni seznami. Namen organizacije je namreč tudi spodbujanje sodelovanja razvijalcev ter izmenjava znanja o odprti kodi. OSOR ima v evropskih državah številne partnerje, med njimi tudi COKS (OSOR, 2009).

Preko njihove spletne strani je bilo v času pisanja diplomske naloge mogoče najti preko 1750 različnih odprtokodnih projektov namenjenih različnim uporabam v javnem sektorju. Na voljo je bilo tudi preko šestdeset študij primerov rabe odprte kode.

4.2 ODPRTA KODA IN JAVNA UPRAVA V SLOVENIJI

4.2.1 Politika Vlade Republike Slovenije pri uvajanju in uporabi programske opreme temelječe na odprti kodi

Vlada Republike Slovenije je oktobra 2003 sprejela dokument z naslovom Politika Vlade Republike Slovenije pri uvajanju in uporabi programske opreme temelječe na odprti kodi (Politika Vlade RS pri uvajanju in uporabi programske opreme temelječe na odprti kodi, 2003). V dokumentu je opredeljena temeljna politika vlade glede uvedbe odprtokodne programske opreme v širši javni sektor s smiselno navezavo na gospodarstvo in zasebno uporabo. Iz dokumenta je razvidno, da je tedanja vlada prepoznala potencial odprte kode in proste programske opreme Izpostavljeno je dejstvo, da odprta koda lahko predstavlja alternativo lastniški programski opremi, odprta koda pa bi lahko znatno pripomogla tudi k uveljavitvi slovenskega jezika v računalniške aplikacije. Poleg izjave o spoštovanju licenčnih določitev tako lastniške kot proste programske opreme je v dokumentu še zlasti izpostavljena povezljivost z ostalimi informacijskimi standardi. Državni organi si ne morejo privoščiti nezdružljivih standardov, ki bi ovirali komuniciranje s civilno družbo, gospodarstvom ali drugimi državnimi upravami. Ta ugotovitev se nanaša tudi na lastniško programsko opremo,

ta nikakor ne sme biti omejena s patenti ali pa morajo lastniki teh patentov omogočiti enostavno povezovanje z ostalimi sistemi.

Prednosti, ki so izpostavljene v dokumentu se nanašajo predvsem na omejitev stroškov pri nakupu licenc in racionalizaciji stroškov podpore pri poslovnih procesih. Z uporabo odprtokodne programske opreme se poveča tudi prilagodljivost potrebam in željam naročnikov ter zaupanje v varnost in zasebnost. Povečana je medsebojna povezljivost in prilagodljivost, nadgrajevanje in redistribuiranje enkrat izvedenih rešitev pa je preprostejše. Zapisana je tudi ideja o prilagoditvi izobraževalnega sistema, kjer bi bilo potrebno vzpostaviti nevtralnost med prosto ter lastniško programsko opremo ter učeče generacije seznaniti tako s prednostmi kot tudi slabostmi ene in druge.

Navedeni so tudi pomisleki, kot so nedorečeni odprtokodni poslovni modeli in manj dodelani mehanizmi vplivanja na hitrost in kakovost teh rešitev. Ena izmed večjih bojazni se skriva tudi v sami dostopnosti kode saj to pomeni, da se lahko ta spreminja dokaj nenadzorovano predvsem z namenom reševanja kratkoročnih problemov, kar pa bi lahko posledično vodilo do težav in nestabilnosti sistemov v prihodnosti. Predvsem zaradi tega bi bilo potrebno bistveno več pozornosti posvetiti tudi dokumentaciji in testiranju sprememb pred vključitvijo v glavno vejo kode. Pojavljajo se tudi dvomi glede varnosti in razpoložljivosti podpore ter tveganjem, ki jih prinesejo številni različni licenčni sistemi.

V dokumentu vlade sicer nikjer ni omenjena možnosti, da bi delovanje javnega sektorja v celoti lahko prešlo na prosto programje, odprta koda je vedno omenjena kot alternativa lastniškemu programju, v katerega je bilo vloženih že precej sredstev.

Glavne točke, ki jih je vlada sprejela z omenjenim dokumentom so torej (Politika Vlade RS pri uvajanju in uporabi programske opreme temelječe na odprti kodi, 2003, str. 6 in 7):

- Vlada Republike Slovenije bo s svojim stalnim delovanjem ter aktivnostmi na področju informacijsko-komunikacijskih tehnologij in ne nazadnje tudi lastnim zgledom pripomogla k širjenju informacij in znanja o značilnostih ter priložnostih uporabe programske opreme in rešitev temelječih na odprti kodi.

- Vlada Republike Slovenije bo enakopravno obravnavala odprtokodne in lastniške programske rešitve. Pri razvoju in nakupu informacijskih rešitev bo izbor rešitve temeljil na finančni in funkcionalni učinkovitosti posamezne rešitve, ne glede na njen poslovni ali licenčni model.

- Vlada Republike Slovenije bo v prihodnje, v največji možni meri načrtovala, gradila in kupovala take informacijske rešitve, ki bodo temeljile na odprtih standardih in protokolih.

- Vlada Republike Slovenije se bo izogibala omejevanju na uporabo informacijskih rešitev, ki ne omogočajo povezljivosti in izmenjave podatkov z drugimi informacijskimi rešitvami in sistemi. Kjer pa že obstajajo uveljavljene in delujoče

zaprte informacijske rešitve, bo ob dopolnitvah in nadgradnjah podpirala odpiranje le-teh.

• Vlada Republike Slovenije se bo, v vseh primerih, kjer je to ekonomsko upravičeno, trudila pridobiti polne pravice do lastništva, sprememb in redistribucije izvorne kode naročenih informacijskih rešitev. Prevzete in s proračunskimi sredstvi financirane informacijske rešitve bo Vlada Republike Slovenije izdala v javno uporabo pod licenco kot jih je sama pridobila, razen če obstajajo tehtni varnostni ali drugi pomisleki, ki terjajo drugačno ravnanje.

• Vlada Republike Slovenije bo podpirala usposabljanje, izobraževanje zaposlenih in njihovo uvajanje v delo z odprtokodno programsko opremo, kakor tudi aktivno spodbujala prenos znanja ter dobre prakse s tega področja med uporabniki.

• Vlada Republike Slovenije se bo zavzemala za čim širšo uporabo programskih rešitev, temelječih na odprti kodi, tako za lastne potrebe, kakor tudi za rešitve, ki so sofinancirane iz javnih sredstev (na primer za potrebe šolstva, raziskovalnih dejavnosti, prejemnikov državnih sofinanciranj), kadar je to tehnološko smiselno in ekonomsko upravičeno. Hkrati se bo zavzemala tudi za doseganje čim bolj ugodnih komercialnih in drugih pogojev za uporabo lastniške programske opreme.

• Vlada Republike Slovenije bo spodbujala razvoj programske opreme in rešitev temelječih na odprti kodi in tudi sama razvijala informacijske rešitve na osnovi tovrstnih rešitev. Spodbude bodo namenjene predvsem novim kakovostnim in uporabnim izdelkom ter lokaliziranim obstoječim rešitvam. Vlada RS bo še naprej spodbujala tudi razvoj domače lastniške in komercialne programske opreme, ki bazira na odprtih standardih, saj tudi ta prispeva k razvoju informacijske panoge v Sloveniji.

• Vlada Republike Slovenije bo znanja in vsebine s področja programske opreme in rešitev temelječih na odprti kodi vpeljala v vzgojno-izobraževalne programe. V okviru šolskih programov moramo mladi generaciji jasno predstaviti prednosti in slabosti lastniške in odprtokodne programske opreme in jo naučiti, da se samostojno, strokovno in odgovorno odloča za rabo posameznega pristopa.

• Vlada Republike Slovenije bo aktivno spodbujala uporabo programskih rešitev temelječih na odprti kodi tudi zunaj javnega sektorja (v gospodarstvu in civilni družbi). Vlada bo s politiko javnih razpisov in z neposrednimi (so)financiranji projektov aktivno spodbujala razvoj izvirne programske opreme v slovenščini ter lokalizacijo (prevajanje v slovenščino) pri vpeljevanju tuje programske opreme in rešitev temelječih na odprti kodi. Obseg namenjenih sredstev bo določala tudi na osnovi ocene prihrankov in povečane konkurenčnosti v gospodarstvu ter posledično višjimi razpoložljivimi sredstvi v proračunu Republike Slovenije.

4.2.2 Poslanska pobuda za prenovo informacijskih sistemov in odzivi nanjo

12. januarja 2009 so trije poslanci Socialnih demokratov, Luka Juri, Matevž Frangež in Dejan Levanič, na Vlado Republike Slovenije poslali poslansko pobudo v zvezi s

prenovo informacijskih sistemov (Jurij et al., 2009). Predlagali so, da naj Ministrstvo za javno upravo v sodelovanju z Ministrstvom za visoko šolstvo, znanost in tehnologijo pripravi program postopnega prehoda na odprtokodno programje v javnem sektorju in sčasoma tako opusti komercialno licenčno programje. Poslanci kot glavni razlog za ta prehod navajajo prihranek pri nakupu licenc, ki so pri odprtokodnem programju brezplačne. V pobudi omenijo, da sam prehod pomeni določene stroške vendar naj bi bili ti stroški občutno nižji že takoj, ko se zaključi prvi val izobraževanja uslužbencev. Kot veliko prednost navajajo tudi to, da bi v primeru prehoda na odprto kodo večina porabljenih sredstev ostala v Sloveniji, država bi tako privarčevala kot tudi dobila nova delovna mesta. Odprtokodna oprema pa je po navedbah poslancev tudi varnejša pred zlorabami in manj dovzetna za okužbe z virusi. Konkretno poslanci predlagajo prehod na odprti internetni brskalnik Mozilla Firefox (namesto Internet Explorerja), pisarniška orodja Open Office (namesto Microsoft Office), nadaljnji korak pa bi bil prehod iz programskega okolja Windows na Linux.

26. marca 2009 je vlada na redni seji sprejela tudi odgovor na zgoraj povzeto pobudo (19. redna seja Vlade RS, 2009). V odgovoru so zapisali, da je pobuda preveč splošna in zato težko izvedljiva. Ministrstvo za javno upravo (kot skrbnik informacijskih sistemov) že sedaj uporablja nekatere odprtokodne rešitve, predvsem za poganjanje strežnikov. Ob tem priznavajo, da so odprtokodne rešitve na področju strežniške infrastrukture celovito gledano cenejše kot bi bile komercialne alternative. Prehod na odprto kodo pa ne pomeni celotnega prihranka stroškov v primerjavi z lastniško programsko opremo, saj je za vzdrževanje potrebno najeti ustrezne uslužbence. Tu se pojavi pomislek glede kakovosti vzdrževalcev odprtokodnega programja v primerjavi s kadrom, ki je usposobljen za vzdrževanje lastniškega programja, še zlasti na področju Slovenije. V odgovoru se je vlada opredelila tudi do konkretnih predlogov poslancev.

V primeru uporabe odprtih spletnih brskalnikov ne vidijo težav, prav tako pa ne možnosti za prihranek, saj je Internet Explorer tako ali tako vključen v operacijski sistem Windows. Tudi glede uporabe aplikacije OpenOffice.org kot alternative Microsoftovemu Wordu ne vidijo težav, nekateri to aplikacijo v javni upravi uporabljajo že sedaj. Ministrstvo za javno upravo v povezavi s tem tudi že pripravlja pobudo za določitev odprtega standarda kot obveznega standarda za izmenjavo datotek znotraj javne uprave (ISO/IEC 26300:2006, 2009).

Težavo pa vlada vidi pri prehodu iz operacijskega sistema Windows na Linux. Skrbi jih predvsem veliko število aplikacij, ki jih uporablja javna uprava, ki delujejo zgolj v okolju Windows. Težava je tudi, da so Microsoftovi Windowsi 95 odstotno prisotni na delovnih postajah v svetovnem merilu. Poleg tega pa je Linux po mnenju vlade še vedno neprimeren za končno širšo uporabo zaradi preveč odprte kode, prevelike ranljivosti zaradi množične uporabe in nezadostnega znanja za uporabo. Prav tako bi prilagajanje posameznim organom, ki imajo specifične zahteve, zahtevalo obsežne prilagoditve in testiranja, prehod je torej nemogoč brez poizkusnih projektov. Ob vsem tem pa je potrebno zagotoviti tudi nemoteno delovanje vseh institucij.

Odgovor vlade se dotakne tudi strežniške infrastrukture. Na strežnikih se v veliki meri že uporablja Linux, stroški vzdrževanja pa so nižji kot bi bili v primeru uporabe Microsoftovega strežniškega programja. Odgovor se dotakne tudi varnostnega vprašanja, zapisano je, da je lastniško programje bolj privlačno za različne napade vendar tudi Linux nanje ni imun. Kot razlog za to je navedena ravno dostopnost izvorne kode. Sicer pa Ministrstvo za javno upravo pozdravlja vse pobude, ki bi pripeljale k izboljšanju in boljši gospodarnosti informacijskih rešitev v državni upravi.

Odzivi na pobudo so mešani. Zagovorniki odprte kode pobudo pozdravljajo in v komercialnem programju zaznavajo splošno zlo, nasprotniki jo zavračajo s standardnimi argumenti o slabostih odprte kode. Tudi odzivi na odgovor vlade so mešani. Predvsem zagovorniki odprte kode pozdravljajo, da je vlada pripravljena razmišljati o alternativnih oziroma odprtih rešitvah, po drugi strani pa ji očitajo pomanjkanje znanja na tem področju. V odgovoru vlade je namreč veliko posploševanj in nasprotij, vendar tudi sama pobuda ni dovolj obširna in natančna. Strokovna javnost pa ima na zadevo svoj pogled.

Junija se je v reviji Moj mikro (Banovič, 2009) pojavil članek, ki govori ravno o opisani pobudi in reakciji vlade. Pisec meni, da je v bližnji prihodnosti popoln prehod na odprto kodo nemogoč. Smiselno pa bi bilo uvesti poizkusne projekte na nekatere manjše organe, denimo upravno enoto ali občino. Postopka bi se bilo potrebno predvsem lotiti postopoma. Pisec članka pa tudi izpostavlja specifično situacijo v Sloveniji. Glede na to, da je največji odjemalec Microsofta v Sloveniji ravno javni sektor imata obe stranki v postopku svoje prednosti; Microsoft Slovenija bi brez javne uprave težko preživel in je zato prisiljen sklepati kompromise, po drugi strani pa nima konkurence, tako da je javna uprava prisiljena sklepati posle z njim. Avtor članka tako odprto kodo vidi predvsem v luči ponujanja alternative javni upravi, ki bi imela tako v roki boljše izhodišče za pogajanja. Seveda pa Microsoft ni edini komercialni ponudnik programske opreme, javna uprava pogodbe sklepa tudi z ostalimi ponudniki, Microsoft je brez komercialne konkurence zgolj na področju pisarniških orodij.

4.3 PRIMERI UPORABE ODPRTE KODE V JAVNEM SEKTORJU

V mnogih državah evropske unije uvajajo odprto kodo v različne segmente javnega sektorja. Kot razloge za uvedbo odprtokodne programske opreme se največkrat navaja nižje nabavne stroške in večjo fleksibilnost v primerjavi z zaprto kodno programsko opremo. Navedeni so nekateri novejši in odmevnejši primeri uporabe.

4.3.1 FriKomPort

Na norveškem so leta 2006 v regiji Kongsberg vzpostavili poseben portal z namenom organiziranja in usklajevanja tečajev za usposabljanje zaposlenih na občinah. Portal so razvili s pomočjo odprtokodnih orodij v sodelovanju s sedmimi občinami , ki se nahajajo v omenjeni regiji. Ko so projekt realizirali so zanimanje zanj kmalu pokazale tudi ostale regije, občine in organizacije. Regija Kongsberg je zato program izdala

pod licenco GPL kot prost program. Danes portal uporablja preko 50 različnih organizacij in več tisoč norveških javnih uslužbencev dnevno. S prihajajočo izdajo angleške različice pa avtorji pričakujejo še več uporabnikov tudi drugod v Evropi in svetu (FriKomPort, 2009).

4.3.2 Open Portal Guard

Open Portal Guard (OPG) je odprtokodna aplikacija razvita v manjši občini Grosseto v Italiji. Gre za program namenjen branju in obdelavi podatkov, ki se nahajajo na čipih elektronskih osebnih izkaznic. Leta 2003 se je Italijanska vlada odločila uvesti poizkusni projekt uvedbe elektronskih osebnih izkaznic in za sodelovanje zaprosila občine. Vlada je preko Ministrstva za notranje zadeve izbrala tudi najprimernejšega ponudnika informacijskih rešitev. Mnoge občine z učinkom izbranega izvajalca niso bile zadovoljne zaradi nenehnega pojavljanja tehničnih težav in pretirano omejujočih licenčnih določil. Poleg tega je vzpostavitev komercialnega informacijskega sistema močno bremenila proračune predvsem manjših občin. V občini Grosseto so se zato lotili razvoja lastne rešitve temelječe na odprti kodi, za začetnika razvoja velja Ben Brugger. Brugger je navezal stike tudi z ostalimi občinami, ki so prav tako želele razviti samostojno rešitev, nekatere podobno misleče pa je našel tudi na mednarodni ravni. Konkreten rezultat prizadevanj Bruggerja in njegovih sodelavcev je trenutno močna razširjenost odprtokodnega sistema OPG v Grossetu. Pomembneje pa je, da je občina skupaj z razvijalci sistema pridobila pomembno izhodišče za razvoj sistemov za branje elektronskih kartic tudi na mednarodni ravni (Open Portal Guard, 2009).

4.3.3 Švicarsko Zvezno ter Administrativno sodišče

Odprta koda je bila na Zvezno sodišče v Švici uvedena že leta 2001 ob prenovi zastarele informacijske tehnologije. Pristojni za posodobitev so že tedaj prepoznali prednosti odprte kode, predvsem v večji prilagodljivosti in varnosti tovrstnih produktov. Uporabljali so sicer lastniški operacijski sistem podjetja Sun Microsystems imenovan Solaris v navezi s Sunovim lastniškim urejevalnikom besedila Staroffice, vendar so v sistem vključili številne odprtokodne aplikacije kot je internetni brskalnik Firefox in odjemalnik elektronske pošte Novell Evolution. Poleg tega omenjena lastniška proizvoda povsem podpirata ostale odprtokodne standarde (Švicarsko Zvezno ter Administrativno sodišče, 2009).

Leta 2007 je bilo ustanovljeno Zvezno administrativno sodišče, ki je združilo 37 različnih resorjev, ki so do združitve delovali samostojno. Za učinkovito delovanje so tako potrebovali kvaliteten informacijski sistem. Zahtevno nalogo je dobil oddelek za informacijske tehnologije na Zveznem sodišču, kar pa že na samem začetku ni ugajalo novoustanovljenemu administrativnemu sodišču. Največja težava je bila, da pri implementaciji sistema niso imeli nobene vidne vloge. Klub temu, da so se odprtokodni standardi na Zveznem sodišču trdno uveljavili, pa se na administrativnem sodišču z novo tehnologijo ne morejo sprijazniti. Eden izmed pomembnejših vzrokov je tudi to, da je večina uslužbencev v posameznih resorjih v

preteklosti delala z Microsoftovimi programi in urejevalniki besedil, zato jim prehod na odprtokodne rešitve predstavlja preglavice. Pojavljajo se pomisleki glede funkcionalnosti kot tudi združljivosti tovrstnih tehnologij. Predvideno je, da se bo tako sodelovanje na področju uvedbe informacijskega sistema med sodiščema končalo leta 2010.

Opisan primer dokazuje, kako je lahko posamezen projekt uspešen v neki organizaciji, medtem ko v drugi izpade neučinkovito. Medtem ko na administrativnem sodišču razmišljajo o zamenjavi odprtokodne tehnologije pa na zveznem sodišču načrtujejo popoln prehod na operacijski sistem OpenSolaris, odprto različico lastniškega Solarisa.

4.3.4 Francoska policija in Ubuntu Linux

Francoska policija velja za eno izmed največjih javnih teles v evropskem prostoru. Telo je sestavljeno iz redne in vojaške policije in zaposluje preko 100 tisoč ljudi. Organizacija je centralizirana, večino odločitev sprejmejo v Parizu. Ravno zaradi velikosti in razvejanosti policije je zanesljiv informacijski sistem še bolj pomemben dejavnik kot navadno in ni priporočljivo da je odvisen še od nekoga tretjega. S tem namenom je skupina strokovnjakov leta 2001 sklenila, da iz lastniške zaprto-kodne programske opreme postopoma preidejo na odprtokodno (Francoska policija in Ubuntu Linux, 2009).

Spremembe so uvajali postopoma. Prva velika sprememba se je zgodila šele leta 2005, ko so pričeli z menjavo Microsoftovih pisarniških orodij Microsoft Office z njihovo odprtokodnim nadomestkom Open Office. Nato je sledila menjava spletnih brskalnikov, iz Microsoftovega Internet Explorerja na odprtokodni brskalnik Firefox. Ko pa je leta 2006 Microsoft najavil izdajo novih operacijskih sistemov Windows Vista pa so ljudje, odgovorni za presnovo informacijskih sistemov policije sprejeli odločitev, da bodo postopoma vse Microsoftove operacijske sisteme Windows zamenjali z Ubuntu izdajo Linuxa. Pomembno je tudi dejstvo, da pri končnih uporabnikih ni bilo opaženih večjih težav pri privajanju na novo tehnologij .

Sicer pa Francoska policija ni osamljen primer uporabe odprte kode v javnem sektorju, Ubuntu operacijske sisteme uporablja tudi Narodna skupščina Francije, Linux uporabljajo tudi na Ministrstvu za kmetijstvo in ribištvo, nekatere odprtokodne aplikacije pa so v uporabi tudi v Pariškem mestnem svetu.

5 ZAKLJUČEK

Skozi petdeset let razvoja modernega računalništva smo bili priče izjemno hitremu napredku tehnologije in številnim odkritjem, ki so pomagala človeku pri delu in razvoju. Velik del zaslug za ta hitri napredek gre pripisati tudi odprti kodi, ki je v takšni ali drugačni obliki prispevala k razvoju od samega začetka. Od njenih skromnih začetkov, ko še nihče ni slutil kolikšen vpliv bo imela, se je razvila v nepogrešljiv del modernega računalništva. Danes predstavlja resno alternativo in konkurenco programju, ki ga je večina bolj vajena uporabljati – komercialnemu programju. Toda počasi se ta trend spreminja kar še zlasti ni pogodu podjetjem in korporacijam, ki se ukvarjajo z razvojem in prodajo strogo komercialne programske opreme. Toda zaradi samega procesa nastanka odprtokodnega programja, ki temelji na prostovoljnem udejstvovanju posameznika in skupnosti, omenjena podjetja v rokah nimajo toliko vzvodov za izničevanje odprte kode, kot bi si jih sami želeli. Edino pravo »orožje« je zgolj to, da tudi sami postanejo boljši.

V primeru odprte kode gre za fenomen, zlasti z vidika ekonomije. Ne samo, da je velika večina odprtokodnega programja brezplačna in jo je možno povsem legalno pridobiti preko interneta. Bistvo fenomena se skriva v njenem nastanku. Tisoče programerjev v svojem prostem času piše vrstice kode, iz katere nastane program, in ta program nato delijo z ostalimi uporabniki. Obstaja mnogo razlogov zakaj to počno oziroma zakaj se »žrtvujejo« za druge. V diplomskem delu se motivom razvijalcev sicer nisem podrobneje posvečal. V veliko primerih programe razvijajo predvsem zaradi svojih potreb in nato omogočijo uporabo tudi ostalim, nekateri želijo zgolj pokazati svoje znanje, spet drugi pripadati skupnosti. Dejstvo je, da ti prispevki večinoma prostovoljcev danes predstavljajo javno dobrino in korist.

Fenomen se zagotovo skriva tudi v možnosti zaslužka s pomočjo odprtega programja. Obstaja več načinov za ustvarjanje dobička preko odprte kode, odvisni so predvsem od načina licenciranja posameznega produkta. Produkti izdani pod določili licence GPL se osredotočajo predvsem na nudenje tehnične podpore in zaračunavanju stroškov povezanih z distribucijo. Licenca BSD pa omogoča pretvorbo odprtega programja v zaprto-kodno programsko opremo in nadaljnjo prodajo. Obstaja še veliko ostalih licenc, ki v diplomskem delu sicer niso podrobneje predstavljene. Kljub temu je mogoče razbrati, da je napreden sistem licenciranja odprtega programja eden izmed stebrov dosedanjega in nadaljnjega razvoja odprte kode. Ravno sistem licenciranja namreč daje razvijalcem vpogled v možnosti, ki lahko doletijo njihov izdelek. Največji strah, ki se pojavlja pri odprtokodnemu razvijalcu je namreč ta, da bo njegov izdelek bodisi nekdo izkoristil za svoj osebni dobiček ali prepoznavnost, ali pa bo program postal zaprto-kodni. S pomočjo licenc je možno podobne zlorabe preprečiti oziroma posamezniku ali skupnosti vsaj pripisati primerne zasluge.

Glede na to, da je odprta koda največkrat cenejša od komercialne programske opreme se pojavlja vprašanje, zakaj ni razširjena še veliko bolj kot dejansko je. Dejstvo je, da ima tudi odprtokodno programje svoje pomanjkljivosti, ki največkrat najhuje prizadenejo ravno povprečnega uporabnika. Največja pomanjkljivost je težavnost namestitve takšnih programov in zapletenega uporabniškega vmesnika. To razumljivo odbija ne napredne uporabnike računalnikov. Težava je lahko tudi v nezdružljivosti posameznih formatov z različnimi sistemi. To izhaja predvsem iz dejstva, da večina današnjih nezahtevnih uporabnikov uporablja razširjeno lastniško programsko opremo (npr. Microsoftovo), le-ta pa večinoma ne deluje v okolju odprtokodnih programov. Tudi to je eden izmed poglavitnih razlogov, da se odprta koda le počasi širi v različne javne ustanove. Iz vidika nabavnih stroškov licenc in stroškov vzpostavitve informacijskih sistemov, temelječih na odprti kodi bi bil tak prehod logičen. Težave se pojavijo pri uvajanju zaposlenih in združevanju standardov, kar kratkoročno v večini primerov pomeni večje začetne stroške, to pa omejuje možnost sprememb. Močni pa so tudi različni komercialni lobiji, ki ponujajo vedno večje ugodnosti ob uporabi dosedanje, komercialne programske opreme.

Kljub vsem težavam pri preboju odprte kode na mesto komercialnega programja se odprtokodna skupnost iz dneva v dan povečuje. Z odprto kodo se je vsaj posredno verjetno srečal že vsak uporabnik interneta, čeprav nevede. Eden izmed največjih odprtokodnih projektov je namreč spletni strežnik Apache. Na Apachejevem programu danes »stoji« skoraj polovica svetovnega spleta, torej spletnih strani ki jih vsakodnevno obiskujemo. Tudi to diplomsko delo se v veliki meri zanaša na enega izmed odprtokodnih projektov. Osrednje vodilo pri pisanju je bila sicer Webrova knjiga o uspehu odprte kode, kot sem omenil že v uvodu. Toda pogled na vire razkrije, da je ogromno podatkov mogoče pridobiti tudi z uporabo proste spletne enciklopedije, Wikipedie. Pisci člankov so anonimni prostovoljci, ki jih določeno področje zanima, pri pisanju pa se opirajo na druge vire. Razširjena skupnost in bralci pa vseskozi opozarjajo na morebitne napake ki se tako v določenem obdobju izločijo iz besedila.

Odprta koda je vse bolj prepoznavna in uporabljana tudi v slovenskem prostoru. Javni sektor se zaveda potenciala, ki ga odprto programje predstavlja. V tem trenutku je državni aparat še preveč navezan na tradicionalne oziroma komercialne rešitve, vendar se stvari obračajo tudi v prid odprte kode. Dokaz za to je Center odprte kode Slovenije, ki je nastal na pobudo vlade. Že dlje časa je v Sloveniji prisotno tudi društvo slovenskih uporabnikov Linuxa. Tudi v zasebnem sektorju je situacija podobna. V Sloveniji uspešno deluje organizacija Kiberpipa, ki obiskovalcem preko različnih seminarjev in aktivnosti razširja zavest in znanje o odprti kodi in ostalih informacijskih tehnologijah.

V okviru Evropske unije deluje organizacija OSOR, ki je povezana tudi s Centrom odprte kode Slovenije. Na evropski ravni zbira primere uporabe odprte kode v javnih sektorjih držav ter hkrati nudi podporo sedanjim in bodočim razvijalcem odprtokodnih aplikacij. Iz OSOR-ja so povzeti tudi trije uspešni primeri uporabe odprtokodnega programja v primeru francoske policije, povezovanja občin na norveškem in uvedba odprtokodne aplikacije za branje elektronskih osebnih izkaznic v Italiji. Predstavljen

je tudi primer poizkusa uvedbe odrte kode na Zvezno administrativno sodišče v Švici, ki pa ni bil tako uspešen.

Kakšnih konkretnejših primerov uvedbe odprte kode v slovensko javno upravo trenutno še ni moč zaslediti. Vlada Republike Slovenije je sicer že leta 2003 sprejela dokument, s katerim spodbuja k uvedbi odprtega programja v organe javne uprave, vendar se z izjemo nekaterih osamljenih poizkusov do danes na tem področju ni veliko spremenilo. Predstavljena je tudi pobuda vladi treh poslancev iz stranke Socialnih demokratov iz leta 2009 in odgovor vlade. Gre sicer za jasen signal, da se stvari premikajo v prid odprte kode, vendar trenutno iniciativa še vedno ostaja na ravni pobud in strateških dokumentov. Veliko primerov dejanske uporabe rešitev, ki bi slonele na odprti kodi trenutno torej še ni.

Odprta koda je med nami neizogibno prisotna, njena prisotnost pa se bo zgolj povečevala. Potrebno jo je sprejeti in kar najbolje uporabiti in izkoristit. Izkoristiti v skladu z licenčnimi določili in ne zgolj izrabiti. Ravno to je ključ, da se je odprta koda tako razširila in uveljavila, zaradi poštenega udejstvovanju posameznikov, ki v določenih primerih korist drugih ljudi postavljajo celo pred svoje lastne.

LITERATURA

BANOVIČ, Zoran. So »odprti« poslanci prehitri? Moj mikro. URL=«http://www.mojmikro.si/mreza/odprta_koda/so_odprti_poslanci_prehitri«. Marec, 2009.

HAUBEN, Michael. History of ARPANET. URL=«http://www.dei.isep.ipp.pt/docs/arpa.html«. 12.8.2009.

HUGHES, Phil. Interview: Fred van Kempen. URL=«http://www.linuxjournal.com/article/2772«. 1.6.1994

HUGHES, Phil. Interview: Orest Zborowski. URL=«http://www.linuxjournal.com/article/70«. 1.7.1995.

JURIJ, Luka, FRANGEŽ, Matevž in LEVANIČ, Dejan. Pisna poslanska pobuda. URL=«http://arhiv-cns.gov.si/index-arhiv-cns.php?vie=cnt&gr1=grObr&gr2=a08&url=18a6b9887c33a0bdc12570e50034eb54/ba4118e50cadae7ec125758300550739?OpenDocument« 12.1.2009.

LAURENT, M. St. Andrew. Understanding Open Source and Free Software Licensing. O'Reilly Media Inc. URL=«http://oreilly.com/catalog/osfreesoft/book/«. 2004

PRIMOŽIČ, Peter. Uporaba odprte kode kot osnova za razvoj programa. Diplomsko delo, Fakulteta za računalništvo in informatiko, Ljubljana, 2005.

WEBER, Steven. The Success of Open Source. Harvard University Press, Cambridge, Massachusetts in London, Anglija, 2004.

VIRI

19. redna seja Vlade RS. Odgovor na pisno pobudo poslancev dr. Luke Jurija, Frangež Matevža in Dejana Levaniča. URL=«http://www.mju.gov.si/si/splosno/cns/novica/article/14/9969/e4dc975755/«. 26.3.2009.

ACM – Association for Computer Machinery. URL=«http://www.acm.org/«. 12.8.2009.

Alan Cox. Iz Wikipedie, proste enciklopedije. URL=«http://sl.wikipedia.org/wiki/Alan_Cox«. 15.8.2009.

Andrew Tanenbaum. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Andrew_S._Tanenbaum«. 15.8.2009.

Apache. HTTP Server Project. URL=«http://httpd.apache.org/«. 16.8.2009.

Bash. Iz Wikipedie, proste enciklopedije. URL=« http://en.wikipedia.org/wiki/Bash«.
15.8.2009

Bill Joy, Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Bill_Joy«. 12.8.2009.

Bitkeeper. URL=« http://www.bitkeeper.com«. 17.8.2009.

Boutell.com. Linux Software Map. URL=«http://www.boutell.com/lsm/«. 16.8.2009.

Brookov zakon. URL=«http://www.jargon.net/jargonfile/b/BrookssLaw.html«.
16.8.2009.

BSD. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Berkeley_Software_Distribution«. 31.8.2009.

BSDI. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Berkeley_Software_Design«. 14.8.2009.

COKS – Center odprte kode Slovenije.
URL=«http://www.coks.si/index.php5/Glavna_stran«. 18.8.2009.

CSRG – Computer Systems Research Group. Iz Wikipedie, proste enciklopedije.
URL=« http://en.wikipedia.org/wiki/Computer_Systems_Research_Group«.
14.8.2009

DARPA. Zgodovina DARPA-e. URL=«http://www.darpa.mil/history.html«. 14.8.2009.

Daniel Hillis. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/W._Daniel_Hillis«. 16.8.2009.

David Miller. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/David_S._Miller«. 15.8.2009.

Debian. URL=« http://www.debian.org/«. 15.8.2009.

Definicija prostega programja. Free Software Foundation.
URL=«http://www.fsf.org/licensing/essays/free-sw.html«. 18.8.2009.

Denis Ritchie. Iz Wikipedie, proste enciklopedije. URL=«
http://en.wikipedia.org/wiki/Dennis_Ritchie«. 12.8.2009.

DOS – Disk Operating System. Iz Wikipedie, proste enciklopedije.
URL=«http://sl.wikipedia.org/wiki/DOS«. 15.8.2009.

DSC – Debian Social Contract.
URL=«http://www.debian.org/social_contract.html#guidelines«. 5.7.1997.

Eric Raymond. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Eric_S._Raymond«. 14.8.2009.

Francoska policija in Ubuntu Linux. OSOR.eu - Open Source Observatory and Repository. URL=«http://www.osor.eu/case_studies/towards-the-freedom-of-the-operating-system-the-french-gendarmerie-goes-for-ubuntu«. 18.8.2009.

Frederic Brooks. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Fred_Brooks«. 16.8.2009.

Freeware definition. URL=«http://www.linfo.org/freeware.html«. 14.8.2009.

FriKomPort. OSOR.eu - Open Source Observatory and Repository.
URL=«http://www.osor.eu/case_studies/frikomport-sharing-code-costs-and-benefits«. 18.8.2009.

FSF – Free Software Foundation. URL=«http://www.fsf.org«. 4.8.2009.

Ian Murdock. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Ian_Murdock«. 15.8.2009.

Intel 386. Iz Wikipedie, proste enciklopedije. URL=«
http://en.wikipedia.org/wiki/Intel_80386«. 14.8.2009.

Intel 8086. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Intel_8086«. 14.8.2009.

Intel Pentium. Iz Wikipedie, proste enciklopedije. URL=«
http://en.wikipedia.org/wiki/Pentium«. 14.8.2009.

Intel x86. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/X86«. 14.8.2009.

ISO/IEC 26300:2006. Open Document Format for Office Applications (Open Document) v 1.0. URL=«http://standards.iso.org/ittf/licence.html«. 3.9.2009.

Ken Thompson. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Ken_Thompson«. 12.8.2009.

Keith Bostic. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Keith_Bostic«. 14.8.2009

Linus Torvalds. Linus Torvalds Bio. URL=«http://www.linux.org/info/linus.html«. 14.8.2009.

LUGOS – Društvo uporabnikov Linuxa Slovenije.
URL=«http://www.lugos.si/drustvo/drustvo«. 28.8.2009.

MIT - Massachusetts Institute of Technology, URL=«http://web.mit.edu/«. 14.8.2009.

Moj mikro. So »odprti« poslanci prehitri?

O Kiberpipi. Kiberpipa. URL=«http://www.kiberpipa.org/sl/about/«. 18.8.2009.

OFSS – The Orbiten Free Software Survey. URL=«http://131.193.153.231/www/issues/issue5_7/ghosh/index.html«. 16.8.2009.

Open Portal Guard. OSOR.eu - Open Source Observatory and Repository. URL=«http://www.osor.eu/case_studies/breaking-the-mould-grosseto-develops-the-openportalguard-eid-system«. 18.8.2009.

OSI – Open Source Initiative. URL=«http://www.opensource.org«. 4.8.2009.

OSOR – Source Observatory and Repository. URL=« http://www.osor.eu/«. 27.8.2009.

PDP-11. Iz Wikipedie, proste enciklopedije. URL=«http://en.wikipedia.org/wiki/PDP-11«. 10.8.2009.

Politika Vlade RS pri uvajanju in uporabi programske opreme temelječe na odprti kodi. Vlada Republike Slovenije, Ministrstvo za informacijsko družbo. URL=«http://mid.gov.si/mid/mid.nsf/V/K0A29A3F8E51AD220C1256DC00030FF40/$file/Politika_OSS_Koncna.pdf«. 16.10.2003.

Poročilo o ustanovitvi in pričetku dela. COKS. URL=«http://www.coks.si/images/f/f7/C_O_K_S_porocilo_2007.pdf«. 10.6.2007.

Red Hat. URL=« http://www.redhat.com«. 17.8.2009.

SourceForge, Inc. URL=«http://sourceforge.net/«. 16.8.2009.

SourceForge, Inc. Iz Wikipedije, proste enciklopedije. URL=«http://en.wikipedia.org/wiki/SourceForge,_Inc«. 17.8.2009.

Švicarsko Zvezno ter Administrativno sodišče. OSOR.eu - Open Source Observatory and Repository. URL=«http://www.osor.eu/case_studies/open-source-on-the-desktops-of-the-swiss-federal-court-and-federal-administrative-court-organisational-challenges«. 18.8.2009.

TCP/IP. Iz Wikipedie, proste enciklopedije. URL=«http://sl.wikipedia.org/wiki/TCP/IP«. 14.8.2009.

The Linux Counter. URL=«http://counter.li.org/«. 16.8.2009.

The Slackware Linux Project. URL=«http://www.slackware.com/.« 15.8.2009.

UNIX/32V. Iz Wikipedie, proste enciklopedije. URL=«http://en.wikipedia.org/wiki/UNIX/32V«. 14.8.2009

Uradna definicija odprte kode.
URL=«http://www.coks.si/index.php5/Vse_o_Odprti_kodi«. 15.8.2009.

USL – Unix System Laboratories. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Unix_System_Laboratories« 14.8.2009.

William Jolitz. URL=«http://william.telemuse.net/william-jolitz-vital-in-silicon-valley«. 14.8.2009.

Yggdrasil. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/Yggdrasil_Linux/GNU/X«. 15.8.2009.

Zgodovina Linuxa. Iz Wikipedie, proste enciklopedije.
URL=«http://en.wikipedia.org/wiki/History_of_Linux«. 18.8.2009.

Zgodovina OSI. URL=«http://www.opensource.org/history«. 15.8.2009

Zgodovina UNIX-a. The Open Group.
URL=«http://www.unix.org/what_is_unix/history_timeline.html«. 31.8.2009.

SEZNAM UPORABLJENIH KRATIC

ARPANET - Advanced Research Projects Agency Network

ACM - Association for Computer Machinery

AT&T - American Telephone & Telegraph

BTL - Bell Telephone Laboratories

BSD - Berkeley Software Distribution

BSDI - Berkeley Software Design Incorporated

COKS – Center odprte kode Slovenije

CSRG - Computer Systems Research Group

DARPA - Defense Advanced Research Projects Agency

DEC - Digital Equipment Corporation

DOS - Disc Operating System

DSC - Debian Social Contract

FSF - Free SOftware Foundation

GNU - GNU is not UNIX

GPL - General Public Licence

IBM - International Business Machines Corporation

LUGOS - Linux User Group of Slovenia

MIT - Massachusetts Institute of Technology

PACT - Project for the Advancement of Coding Techniques

OFSS - The Orbiten Free Software Survey

OPG – Open Portal Guard

OSD - Open Source Definition

OSI - Open Source Initative

OSOR - Open Source Observatory and Repository

TCP/IP - Transmision Control Protocol/Internet Protocol

UNIX - Uniplexed Information and Computing System

USL - Unix System Laboratory

ZDA - Združene države Amerike

SEZNAM PREVODOV

source code – izvorna koda

machine code – strojna koda

compiler – prevajalnik

freeware – brezplačno programje

software – programska oprema

hardware – strojna oprema

pipes – cevi

outout - izhod

input – vhod

copyright – avtorske pravice

clean room – čista soba

IZJAVA O AVTORSTVU IN NAVEDBA LEKTORJA

Spodaj podpisani Matija Koncilja izjavljam, da sem s pomočjo mentorja doc. dr. Ljupča Todorovskega in navedene literature avtor te diplomske naloge. Strinjam se z objavo te diplomske naloge na internetu. Diplomsko nalogo je lektorirala Milena Južnik.