Софтуерът и софтуерната индустрия

Радостин Раднев <radnev@yahoo.com>

Този документ има за цел да запознае читателите с особеностите на софтуера, с начините на производство му. Да обясни как е възможно съществуването на свободния софтуер, мотивацията и печалбата на хората, които разработват свободен софтуер. В заключение има опит да се предскаже бъдещето на софтуерната индустрия, съвети за мигриране към свободен софтуер и отговори на най-често задаваните въпроси. Документът е насочен към широката аудитория и не съдържа много технически термини. Документът се разпространява под условията на GNU Free Documentation License [http://www.fsf.org/licenses/fdl.html].

Съдържание

1. Програма, изходен код и машинен код

Програмите се пишат на езици за програмиране. Съществуват различни видове езици за програмиране, които решават различни типове задачи. Една програма представлява обикновен текст, написан на текстов редактор. Обикновено командите са на английски език и са интуитивни. Това се нарича изходен код на програмата. Т.е. изходният код се разчита лесно от човек. За съжаление, компютърът не разбира изходния код на програмата. Той трябва да бъде транслиран (преведен) до команди, които се разбират от компютъра. В резултат на превеждането (транслирането) се получава машинен код, който представлява поредици от нули и единици, които се изпълняват от компютъра. Машинният код практически е нечитаем от човек и може да се изпълнява само на платформата, за която е преведен (компилиран1).

Нека да разгледаме един пример, с който да илюстрираме разликата между изходния код и машинния код. Примерите са реализирани с помощта на езика Паскал2. Компилаторът, който е използван, е със свободен изходен код и може да бъде изтеглен безплатно от следния адрес http://www.freepascal.org/. Всъщност, цялата статия е реализирана изцяло с използването на свободен софтуер.

```
Program Calculator;

var
    operand1   : real;
    operand2   : real;
    result     : real;
    operation  : char;
    error      : string;


BEGIN
    write('Въведете число: ');
```

1 Транслаторите се делят на два вида: интерпретатори и компилатори. Интерпретаторите четат изходния код и изпълняват (превеждат) командите на машинен код една по една по време на изпълнението на програмата. Компилаторите генерират (превеждат) цялата програма на машинен код предварително. След това програмата може да се изпълнява самостоятелно без наличието на компилатор. Това деление не е много точно, понеже напоследък навлязоха нови концепции, като псевдо-компилатор и виртуална машина. Но целта на статията не е да разглежда различните видове транслатори и да прави сравнение между тях.

2 Езикът за програмиране Паскал бе избран пред другите езици, поради факта, че се изучава(ше) в почти всички училища и университети, лесен е за разчитане и е идеален за обучение, въпреки че няма толкова голямо практическо приложение като Джава или C/C++.

```
    readln(operand1);

    write('Въведете операция: ');
    readln(operation);

    write('Въведете число: ');
    readln(operand2);

    error := '';

    case operation of
        '+': result := operand1 + operand2;
        '-': result := operand1 - operand2;
        '*': result := operand1*operand2;
        '/': result := operand1/operand2;
        otherwise error := 'невалидна операция';
    end;

    if error = ''
        then writeln('Резултатът е: ', result:6:5)
        else writeln('Грешка: ', error);

END.
```

Въпросният пример представлява калкулатор. Програмата изисква въвеждането на две числа и операция. След това операцията се извършва върху числата и резултатът се извежда на екрана. За момента се поддържат операциите събиране, изваждане, умножение и деление, което е напълно достатъчно за демонстрационни нужди.

Както виждате, изходният код на програмата е разбираем (разбира се, читателят трябва да има известни познания по английски език). За да преведем изходния код на машинен код, използваме командата **fpc calculator.pp**, която се явява компилатор на езика Паскал. Ето и самата сесия:

```
[radnev@localhost linux]$ fpc calculator.pp
Free Pascal Compiler version 1.0.10 [2003/06/26] for i386
```

```
Copyright (c) 1993-2003 by Florian Klaempfl
Target OS: Linux for i386
Compiling calculator.pp
Assembling calculator
Linking calculator
35 Lines compiled, 0.1 sec
[radnev@localhost linux]$
```

В резултат се получава изпълнимият файл **calculator**. Терминологията варира леко и може да срещнете други наименования, като двоичен файл[3], машинен файл или най-използваното наименование програма. Ето част от съдържанието на изпълнимия файл или т.н. машинен код:

```
01111111 01000101 01001100 01000110 00000001 00000001 00000001 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000010 00000000 00000011 00000000 00000001 00000000 00000000 00000000
10000000 10000000 00000100 00001000 00110100 00000000 00000000 00000000
11111100 10110100 00000000 00000000 00000000 00000000 00000000 00000000
00110100 00000000 00100000 00000000 00000010 00000000 00101000 00000000
00000101 00000000 00000100 00000000 00000001 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 10000000 00000100 00001000
```

Нека да изпълним програмата, за да видим дали работи. Ето няколко изпълнения на програмата:

```
Въведете число: 23
Въведете операция: *
Въведете число: 45
Резултатът е: 1035.00000

Въведете число: 12345
Въведете операция: /
Въведете число: 23
```

3 Текстов файл и двоичен файл е двойка термини, подобна на изходен код и машинен код. Смисълът е пак същия. Текстовият файл може да се чете от човек директно, докато двоичният файл не може.

```
Резултатът е: 536.73913

Въведете число: 2
Въведете операция: ^
Въведете число: 32
Грешка: невалидна операция

Въведете число: 345
Въведете операция: /
Въведете число: 0
Runtime error 200 at 0x08052F63
  0x08052F63
  0x080480B0
```

Програмата работи за тестваните случаи на умножение и деление, и показва резултата с точност до петия знак след десетичната запетая. Липсва операцията по степенуване и при деление на нула програмата се чупи, вместо да покаже съобщение за грешка.

Очевидно, ако разполагаме само с машинния код на програмата, няма как да я оправим. Но ако разполагаме с изходния код на програмата, може сравнително лесно да добавим операцията по степенуване и да направим съответната проверка за грешка. Ето модифицираната част на новата програма:

```
case operation of
    '+': result := operand1 + operand2;
    '-': result := operand1 - operand2;
    '*': result := operand1*operand2;
    '/': if operand2 = 0                        -- Проверка за
             then error := 'деление на нула'    -- деление на
             else result := operand1/operand2; -- нула
    '^': begin
             result := 1;                       -- Новата
             for i := 1 to trunc(operand2)      -- операция за
                 do result := result*operand1; -- степенуване
         end
    otherwise error := 'невалидна операция';
```

```
end;
```

След като компилираме програмата, изпълняваме я отново, за да видим дали работи. Ето няколко нови примера от изпълнението на програмата:

```
Въведете число: 34
Въведете операция: /
Въведете число: 0
Грешка: деление на нула

Въведете число: 2
Въведете операция: ^
Въведете число: 5
Резултатът е: 32.00000
```

Както се вижда от изложените примери, ако имаме изходния код на една програма и разбираме от програмиране, може сравнително лесно да оправим грешки и/или да добавим нова функционалност. При липсата на изходния код на програмата, очевидно, това е невъзможно.

2. Видове софтуер

Както вече се досещате, софтуерът може да се раздели на три групи в зависимост от правата за достъп до изходния код на програмата: затворен, отворен и свободен.

2.1. Софтуер със затворен изходен код

Софтуер със затворен изходен код е когато получавате само машинния код (изпълнимия файл, крайната програма). Изходният код остава тайна за потребителите. Повечето от комерсиалните програми се разпространяват по този начин. Друго име на този вид е собственически софтуер.

2.2. Софтуер с отворен изходен код

Софтуер с отворен изходен код е когато получавате достъп до изходния код на

програмата при определени условия. Има различни видове лицензи и условия, при които потребителят получава достъп до изходния код. Дори Майкрософт обяви, че отваря кода на някои от продуктите си за някои техни клиенти. Нека да разгледаме случая с Българското правителство и Майкрософт. Копие от текста на споразумението може да бъде намерен тук [http://d.linux-bg.org/index.php?folder=docs%2Fproekto-zakoni/ms].

Преди да продължим с разглеждането на споразумението, трябва да направим уговорката, че изходният код е отворен само за някои потребители, а не за всички. По този начин, изходния код на Уиндоус е затворен за мен и за Вас, но в някаква степен може да бъде отворен за Българското правителство (ако се постигне споразумение). Ето и частта от споразумението описваща правата на потребителя:

2.1 *Обвързване. „Обвързаното лицензиране“ позволява на Правителството на Република България да използва Източниковия Код като средство за информация; във форма която позволява само четене; за целите посочени в Оправомощаването. Компетентният орган може да използва Източниковия Код в цифров или разпечатан вид, както е доставен от Майкрософт. Компетентният орган може да го разглежда и с помощта на търсачка на програмни грешки.*

2.2 *Валидиране и изпробване на Източниковия Код. „Валидирането“ позволява на персонала да работи с персонала на Майкрософт върху Източниковия Код в помещенията на развойните отдели на Майкрософт в САЩ или други държави в рамките на взаимно договорен проект, както е описано и съгласувано в Оправомощаването.*

Малко по-надолу е описано, че споразумението се отнася за три години, разбира се. Както се вижда дори и при такива ограничения, все пак отвореният код е малко по-добър от затворения код. От текста се вижда, че може само да четете кода, но не и да го транслирате и валидирате[4]. Както е описано, валидирането може да се извършва само в помещения на Майкрософт при сътрудничество на служители на Майкрософт. От текста не става ясно, дали може да си вземете изходния код и Уиндоус с Вас от къщи или там ще Ви дадат други, за които предварително е сигурно, че се валидират без проблеми. Но за сметка от това от текста става ясно,

4 Валидиране означава да транслирате (компилирате) изходния код на Уиндоус и да го проверите след това, дали съответства на това, което сте си закупили. В световен мащаб Майкрософт отнесоха много критики, че оставят задни вратички в Уиндоус и в последствие могат да изтеглят чрез тях информацията, която съхранявате. Отговорът на Майкрософт към тези критики, е точно тази програма за отваряне на кода на големи клиенти, чието споразумение разглеждаме.

че преди това трябва да имате „*взаимно договорен проект, както е описано и съгласувано в Оправомощаването*“. С две думи, не е ясно може ли изобщо да проверите изходния код или не може.

Разбира се, има различни видове лицензи за разпространение на софтуер с отворен код. В никакъв случай понятието „*отворен изходен код*“ не бива да се ограничава до Майкрософт и тяхната лицензионна политика.

2.3. Софтуер със свободен изходен код

Висшата форма от всички тези е софтуерът със свободен изходен код или наричан за кратко „*Свободен софтуер*“. В този случай потребителят получава изходния код на програмата и свободата да прави каквото си поиска с изходния код и програмата.

Дефиницията на свободен софтуер е описана най-добре на сайта на Фондацията за Свободен софуер [http://www.fsf.org/]. Самият текст се намира на следния адрес http://www.fsf.org/philosophy/free-sw.html. Понеже сайтът е на английски, тук е даден преводът на дефиницията за свободен софтуер[5].

И така, един софтуер е свободен тогава, когато потребителят получава следните права:

1. Свободата да изпълнява програмата за каквато и да е цел.
2. Свободата да изучава как работи програмата и да я променя за своите нужди.
3. Свободата да разпространява копия на програмата.
4. Свободата да подобрява програмата и да публикува промените, така че цялото общество да има полза.

Точките 2 и 4 предполагат наличието на достъп до изходния код на програмата.

3. Видове лицензи за свободен софтуер

5 В оригинала правата (свободите) се броят от нула. Това е така, защото програмистите имат навика да броят от нула, докато нормалният човек, обикновено, започва да брои от едно.

Важно е да се знае какви видове лицензи съществуват за разпространение на свободен софтуер. Има един доста голям списък с лицензи и сравнение между тях, който може да бъде намерен тук [http://www.fsf.org/philosophy/license-list.html].

Основната разлика между лицензите за разпространение на свободен софтуер и комерсиалните лицензи е, че лицензите за свободен софтуер защитават правата на потребителите, за разлика от комерсиалните, които защитават правата на производителите на софтуер. Разбира се, лицензите за свободен софтуер защитават и правата на авторите. Тук ще разгледаме само два лиценза, като основно наблегнем на лиценза ДжиПиЕл (GPL).

3.1. Лицензът БиЕсДи (BSD)

Под лицензът БиЕсДи се разпространяват няколко операционни системи, като ФриБиЕсДи (FreeBSD [http://www.freebsd.org/]), НетБиЕсДи (NetBSD [http://www.netbsd.org/]) и ОпънБиЕсДи (OpenBSD [http://www.openbsd.org/]). Текстът на лиценза на английски език може да бъде намерен тук [http://www.freebsd.org/copyright/license.html]. За съжаление, няма превод на български език.

Накратко, лицензът дава пълни права на потребителите да правят каквото си искат с изходния код. Изходният код може да бъде подобрен, дори може да бъде създадена нова програма на базата на съществуващата и тя да се разпространява със затворен код. Т.е. може да вземете изходния код, да го промените, да го затворите и да продавате новата програма. Има само едно, единствено условие. В новата програма да споменете имената на авторите на оригиналния код.

Причините за този тип лиценз са дългите съдебни дела относно правата върху софтуера. Накрая авторите решават да се откажат от всякакви права и разрешат на всеки да прави каквото си иска с кода, включително и да го затвори.

Преди няколко години компанията за компютри Ейпъл (Apple [http://www.apple.com/]) използва направеното от разработчиците на ФриБиЕсДи, подобри го, затвори го и сега го продава като операционната система МакОС Хикс (MacOS X [http://www.apple.com/macosx/]).

3.2. Лицензът ДжиПиЕл (GPL)

Лицензът ДжиПиЕл[6] е най-използваният, най-превъзнасяният и най-атакуваният

лиценз. Целият свободен софтуер, принципите на разработката му, дори цялата тази статия се крепят на идеите заложени в лиценза ДжиПиЕл.

Авторът на лиценза - Ричард Столман - си поставя следните цели:

- Гарантирана на правата на потребителите, които разгледахме в Глава 2.3, „Софтуер със свободен изходен код“.
- Запазване на кодът, който се разпространява под него, винаги свободен. Това всъщност е основната разлика между лиценза БиЕсДи и лиценза ДжиПиЕл.
- Гарантиране авторските права на автора и запазване на репутацията му.

Оригиналният текст на английски език може да бъде намерен тук [http://www.fsf.org/licenses/gpl.html]. Лицензът има два превода на български език, които могат да бъдат намерени тук [http://linux.gyuvet.ch/html/docs/gnu_bg.html] и тук [http://bulgaria.sourceforge.net/prava/gplbg.html]. Няма да разглеждаме текста на целия лиценз, а само въведението и основните му цели. Ето част от въведението:

Когато говорим за свободен софтуер, имаме предвид свободата, а не цената[7]*. Нашият GPL е направен така, че да Ви осигури свободата да разпространявате копия на свободен софтуер (и да взимате такса за тази услуга, ако желаете), също да Ви предостави изходните текстове на програмите или възможността да ги получите, ако искате, също да Ви позволи да променяте софтуера или да използвате части от него за създаването на нови свободни програми; също и да сте наясно как да правите тези неща.*

За да защитим Вашите права, се налага да направим ограничения, които забраняват на който и да е да Ви откаже тези права или да Ви помоли да се откажете от тях. Тези ограничения водят и до определени отговорности за Вас, ако разпространявате копия на свободен софтуер, или ако го променяте.

Например, ако разпространявате копия на някаква програма, безплатно или срещу заплащане, трябва да предоставите на получателя всички права, които имате и Вие. Трябва да сте сигурен, че той, както и Вие, ще получи или може

6 Съкращението идва от „General Public License“ или преведено на български означава „Общ публичен лиценз“.

7 На английски „free“ има две значения: безплатен и свободен. Уточнението е необходимо само за английски език. В българския език разликите между безплатен и свободен са очевидни. За да определят значението по-добре, авторите използват следното сравнение *„Free software“ is a matter of liberty, not price. To understand the concept, you should think of „free“ as in „free speech“, not as in „free beer“.* Или преведено на български, става въпрос за свободата на словото, а не за безплатни кебапчета и бира по време на предизборна кампания.

да получи изходните текстове на програмата. Освен това трябва да му покажете тези условия, за да може той да си знае правата.

Ние защитаваме Вашите права по два начина: (1) чрез авторски права за софтуера и (2) предлагаме Ви това Разрешение, което дава законно основание за копиране, разпространение и/или промяна на софтуера.

Също, за защита на всички автори и всеки от нас, ние искаме да сме сигурни, че всеки разбира, че няма никаква гаранция за работата на свободния софтуер. Ако програмата е променена от някой и предоставена на друг, ние изискваме получателят да знае, че това което има не е оригинала, тъй че всички проблеми, предизвикани от промените не трябва да се отразят на репутацията на оригиналния автор.

Накрая, всяка свободна програма се третира еднакво от софтуерните патенти. Искаме да предотвратим опасността разпространителите на свободни програми да придобият патентни разрешения, приватизирайки по този начин програмите. За да се предпазим от това, ние искаме всеки издаден патент да разрешава свободно ползване от всички, или да не се издава въобще.

Както сте разбрали, може да правите всичко с програмата, ако сте потребители. Ако направите подобрения, може да си ги ползвате само Вие. В момента, в който решите да продавате или разпространявате копия на програмата, трябва да дадете същите права, каквито имате Вие, на Вашите потребители. Т.е. ако вземете една програма и я подобрите, трябва да направите публично достояние промените. Или ако използвате код от такава програма, за да създадете нова програма, новата програма също трябва да се разпространява под лиценза ДжиПиЕл. С други думи, имате права докато сте потребител. Ако станете производител, трябва да дадете същите права на клиентите си. Разбира се, тази идея изисква достъп до изходния код на програмата.

4. Мотивация на разработчиците на свободен софтуер

И въпреки всичко, кое и какво мотивира програмистите да разработват свободен софтуер? Лицензът ДжиПиЕл защитава потребителите, а не производителите на софтуер. Очевидно компаниите не го харесват, защото няма как да правят бизнес и съответно пари чрез него. Все пак, една фирма съществува, за да развива бизнес и да прави пари. В пълен противовес на това, програмистите го харесват, разработват и разпространяват софтуер под лиценза ДжиПиЕл.

4.1. Разлики между идеалното и материалното

Софтуерът е интелектуален продукт. Софтуерът не може да се пипне. Софтуерът представлява поредица от нули и единици, които се изпълняват от компютъра. Софтуерът не е материален продукт. При него няма триене, стареене на материала и изхабяване. Една програма както работи днес, така ще работи и утре, така ще работи и след 100 години. Не бъркайте компактдиска със софтуера. Компактдискът е обикновен материален носител, който струва около 0.50 лв. Софтуерът може да се разпространява и в Интернет без видим материален носител.

Други интелектуални продукти това са: книги, филми, музика, клипове, телевизионни и радио предавания. Всичко, което може да се запише в цифров вид, е интелектуален продукт. Разбира се, една книга има разходи по печат, оформление, подвързия и пр. Но съдържанието на книгата остава интелектуален продукт. Хартията е просто носител, както в случая със софтуера и компактдиска. По същия начин може да се разглеждат и филмите, и песните, и клиповете. Всички те могат да се разпространяват в Интернет или на различни материални носители.

Разликата между интелектуалните и материалните продукти е повече от ясна. Интелектуалните продукти имат цена на първоначално производство и после разходите за разпространение и дублиране са нищожни. Един празен компактдиск струва 0.50 лв. На него могат да се запишат огромно количество програми или доста часове музика, или един филм. Материалните продукти освен разходите за проектиране, включват разходите за материали и труд, които са необходими да се произведе (сглоби) едно копие на съответния материален продукт. Примерно, един автомобил има разходи по проектиране, които са единични. Т.е. инженерният проект (дизайнът) на автомобила се извършва веднъж. Самото проектиране отново се води интелектуален продукт, но когато започнем да произвеждаме автомобили, всяка отделна бройка има разходи за материали и разходи за труд, които са доста високи и няма как да бъдат избегнати.

Разликите между софтуера и другите интелектуални продукти са, че повечето от тях ги ползваме само по един път. Да прочетем една книга веднъж, дваж или три пъти. Най-много да гледаме един филм 3-4 пъти. Да слушаме една песен 100 пъти. Радио и телевизионните предавания обикновено ги слушаме или гледаме само по веднъж. За разлика от това, ние ползваме някои програми почти ежедневно, по осем часа на ден. Фактически ние работим с тях. Повечето сървъри в Интернет трябва да работят 24 часа на ден, седем дена в седмицата, без прекъсване.

Друга основна разлика между софтуера и останалите продукти е, че има една специална програма, която се нарича операционна система. Операционната система (ОС) е специален вид софтуер, който управлява ресурсите на компютъра и се явява интерфейс (преводач) между останалите програми и хардуера. Програмите, които работят на тази ОС, обикновено не ползват директно хардуерните устройства - клавиатура, мишка, диск, флопи, монитор, принтер и пр. Те ползват функциите на ОС за работа с тези устройства. Което означава, че ако ОС си промени функциите за работа с хардуерните устройства, останалите програми няма да могат да работят. Или ще трябва и те да се преработят, за да ползват новите функции.

4.2. Средства за производство на софтуер

В Глава 1, „Програма, изходен код и машинен код“ разгледахме програмата калкулатор. Какво ни трябваше за да напишем тази програма? Нека да изброим необходимите неща:

- Текстов редактор, за да въведем изходния код. Това е вид програма.
- Транслатор на езика Паскал, за да преведем кода на разбираем за компютъра език. Това също е програма.
- Горните две програми, както и програмата калкулатор, за да работят, изискват операционна система. Операционната система е вид програма (софтуер), на която работят всички останали програми.
- Знания да се напише програмата - квалифицирани кадри за изпълнение на задачата.

Както виждате, за производството на софтуер трябва софтуер. Парадоксално или не, това е истината. Изключваме компютърът като необходимо средство за производство на софтуер. В този случай компютрите и Интернет се разглеждат като необходима инфраструктура за съществуването на софтуера. Нещо като автомобилостроенето и пътищата (инфраструктурата). За развитието на автомобилостроенето са необходими пътища. Въпреки че това сравнение не е много точно, защото връзката между хардуера (компютрите) и софтуера (програмите) е много по тясна, отколкото зависимостта между пътищата и автомобилостроенето.

Да обобщим, за производството на софтуер е необходим софтуер. От което

следва, че производителите на софтуер са и потребители. От което стигаме до въпроса *„Кои права искате да Ви бъдат защитени, тези на производител или тези на потребител?“* Отговорът, мисля, че е ясен. Ако сте голяма компания, производител на софтуер, най-вероятно ще искате правата на производителя да са защитени. Обаче, ако сте малка компания или независим разработчик, или работите в университет, или сте студент, ще искате правата на потребителя да бъдат защитени.

4.3. Косвена и пряка печалба

Пряка печалба е когато получавате пари за продажбата на нещо или извършването на услуга. Софтуерните фирми произвеждат софтуер и го продават. Т.е. те имат пряка печалба от тази дейност.

Косвена печалба е обратно на пряка печалба. Не печелите пари директно, а косвено спестявате някои и друг лев. Именно косвената печалба стимулира програмистите да разработват свободен софтуер.

Нека да разгледаме следния случай. Въпреки, че е чисто теоретичен, на практика нещата стоят точно по този начин.

Представете си, че има 1000 човека. Всеки дава по 1 лв. Общо имаме 1000 лв. или някакъв материален продукт за 1000 лв. Понеже продуктът е материален, само един от участниците в процеса може да го ползва в даден момент.

Сега да си представим, че тези 1000 човека решават да напишат една програма. И всеки написва по един модул, който се състои от по 5-10 реда код. Т.е. всеки дава труд за 1 лв. Накрая имаме една програма, за която е изразходван труд за 1000 лв. Нека да приемем, че съществува подобна програма със затворен код и нейната цена е 20 лв. За съжаление, ценообразуването на софтуер е малко сложно, то зависи от различни фактори и е трудно да се каже колко е пазарната цена на въпросната програма. Но ако приемем, че е 20 лв., това означава, че програмистите, участници в процеса, са спестили 19 лв. Защото софтуерът е интелектуален продукт и въпросната програма, която са разработили, може да бъде изтеглена по Интернет от всеки един от тях. За разлика от материалният продукт, който може да бъде ползван само от един човек, програмата може да бъде ползвана от всички участници в процеса на разработка. И още повече, тази програма може да бъде ползвана от всички хора в Света, независимо дали са участвали или не в процеса на разработка. Просто тя няма да се изхаби, няма да свърши, ще работи вечно и няма нужда след 3 години да плащате нови лицензни такси.

Нека да разгледаме един чисто практически пример, който описва горната теоретична ситуация.

Навярно повечето от читателите знаят, че аз съм автор на база с думи за проверката на правописа под офис пакета ОпънОфис орг (OpenOffice.org [http://openoffice.org/] - съкратено ООо). Който не знае, може да посети проекта БГ Офис [http://bgoffice.sourceforge.net/]. Някъде в началото на 2002 година излезе свободният офис пакет ОпънОфис орг, който привлече вниманието ми, защото наложително ми трябваше текстообработваща програма с поддръжка на проверка на правописа на български език. По това време на пазара се предлагаше проверка на правописа от фирмата Датекс. Ако не ме лъже паметта, цената на пакета бе около 100 лв. Пакетът на Датекс работеше на Майкрософт Уърд (Microsoft Word), който струваше около 1000 лв. (цената е на целия Майкрософт Офис, защото Уърд не можеше да се купи отделно). И както знаете, Майкрософт Офис работи само на Уиндоус, който струваше около 200 лв. По това време аз бях щастлив потребител на Линукс и нямах нужда от закупуването на Майкрософт Уиндоус. Още повече, че току що беше излязъл и офис пакетът ОпънОфис орг, който също можеше да се изтегли безплатно. От една страна, мога да ползвам платено решение, което струва около 1300 лв. От друга стана, имам безплатна операционна система и безплатен офис пакет, но пък нямам проверка на правописа на български език за това решение. Реших, че мога да спестя 1300 лв. и да създам свободна база от думички за проверка на правописа. Занимавам се професионално с програмиране и имам достатъчно знания да създам такъв продукт. Речено, сторено.

Докато създам първата версия ми отне около 2 месеца по 1-2 часа труд на ден в свободното ми време. Версията нямаше много думи, около 30 000. Но ми отне време докато дефинирам и създам структурата и формата на данните. След това развитието на проекта вървеше доста по-лесно и бързо. На няколко пъти добавях по няколко хиляди думи. Последната версия на проекта съдържа около 65 000 думи в основна форма, което прави малко над един милион словоформи. Поради натурата на свободния софтуер, аз трябваше да създам само база от думи. Алгоритмите за проверка и генериране на предложенията за замяна съществуват в съответните програми. Днес имаме проверка на правописа на български езика на почти всички свободни програми, които поддържат тази функционалност. Става въпрос за офис пакета ОпънОфис орг, за браузъра Мозила (Mozilla [http://mozilla.org/]), за двата свободни модула за проверка на правописа Испел (Ispell [http://fmg-www.cs.ucla.edu/fmg-members/geoff/ispell.html]) и Аспел (Aspell [http://aspell.net/]). С развитието на тези продукти се развива и проверката на правописа на български език. До момента над 10 човека са помогнали с идеи и/

или труд на проекта. Като допълнение, всеки може да си нагажда проверката на правописа да отговаря на неговите нужди и да я ползва в комбинация със софтуера, с който иска, защото тя е свободен софтуер. От другата страна имаме платени продукти със затворен код, чиято първоначална цена е 1300 лв. и евентуални лицензионни такси на всеки три години.

Мисля, че с този пример се илюстрира много точно движещата сила на свободния софтуер и мотивацията на авторите му. Датекс разработва техния пакет за проверка на правописа с цел да го продава и от тези продажби печели пари (пряка печалба). Моята основна дейност е свързана с програмиране, което също така включва неприятните задачи по писане на задания, спецификации, документации, ръководства за потребителя и пр. Аз печеля пари от нещо друго. Но решавам, че мога да спестя някои и друг лев, ако си разработя собствена проверка на правописа, която да я ползвам в основната си дейност. В този случай аз се явявам производител и потребител на софтуера, който съм разработил. Аз създавам проверката на правописа не да я продавам и да печеля от продажбата, а за да я ползвам. За да си подобря производителността на труда и качеството на документите, които създавам.

Нека да разгледаме още един пример. Става въпрос за сървъра Апаче (Apache [http://apache.org/]). Положението при него е същото. Авторите на Апаче са хора свързани директно с Интернет - системни администратори, администратори на уеб сайтове, доставчици на Интернет, компании за хостване на уеб сайтове и пр. Те не правят този сървър да го продават и да печелят пари от него. Те го създават, за да го ползват. Основната им дейност е доставката на услуги, свързани с Интернет. Според изследванията на НетКрафт (Netcraft [http://news.netcraft.com/]) сървърът Апаче се ползва на 70% от сървърите в Интернет. Майкрософт имат 20% и другите сървъри са по 1%. Ако предположим, че цената на един такъв сървър е 500 долара, ще получим, че са спестени около 16 милиарда долара (32280582*500 = 16140291000).

Нека да разгледаме класическия пример с Линус Торвалдс. Преди 2-3 години излезе филмът „Революционната операционна система“ (Revolution OS). В него той споделя, как и защо е започнал работа по Линукс. Основната причина е парите. Линус по това време е бил студент в Университета Хелзинки и там са ползвали компютри Сън, заредени с операционната система Юникс (Сън Соларис). Един такъв компютър по онова време струваше около 8 000 долара. По същото време се появиха и персоналните компютри с процесор Интел (Intel), които струваха около 2 000 долара. Проблемът е, че липсваше добра ОС за тези компютри. По това време те се разпространяваха с ДОС (Дискова операционна система), която няма голяма част от функционалността на Юникс. Най-много се

усещала липсата на поддръжка на мрежа. Линус не можел да се включи към Интернет с ДОС. Поради тази причина, той започва работа по Линукс и след 8 месеца получава работеща ОС. В интервюто Линус заявява *„След осем месеца работа по проекта имах два пъти по-бърз компютър на една четвърт от цената на Сън.“*

Нека да разгледаме още един пример. В този случай Правителството на Германия поръчва създаването на свободен софтуер, лицензиран под ДжиПиЕл. Става въпрос за проекта Kroupware [http://www.kroupware.org/]. Това е софтуер за синхронизиране на работата в екип. Идеята е да се създаде свободен, сигурен и по-евтин заместител на наличните продукти - най-вече на продукта Майкрософт Ексчейндж (Microsoft Exchange). Основните причини са парите и сигурността. Немският бизнес плаща годишно няколко милиарда Евро за софтуер. Защо тези пари да не останат в Германия. Повече подробности за проекта и начина на финансиране могат да се намерят на следния адрес http://www.kroupware.org/faq/faq.html.

5. Предпоставки за развитие на свободния софтуер

Предпоставките за развитие на комерсиалния софтуер са ясни. Има търсене - ражда се предлагане. Има свободна пазарна ниша за дадена програма - намират се фирми, които да запълнят пазарната ниша и да предложат програмата. Поради тази причина ще се фокусираме само на предпоставките за развитие на свободния софтуер.

5.1. Потребител - разработчик

В предната глава разгледахме кое мотивира разработчиците на свободен софтуер. Очевидно, за да се създаде една свободна програма, сред потребителите и трябва да има достатъчно квалифицирани разработчици.

Проектът Апаче започва именно поради тази причина. Много квалифицирани кадри имат нужда от сървъра Апаче - и сървърът Апаче се ражда. Същата история е и със Линукс. Линус Торвалдс има нужда от операционна система, с която да се включи в Интернет. По това време за персоналните компютри има само ДОС (DOS), който няма възможности за включване в Интернет. Абсолютно същите мотиви ме стимулират и мен да създам проверката за правописа на български език.

Да обобщим, първо някой трябва да има нужда от съответната програма. И второто условие е, този някой трябва да е квалифициран разработчик. При наличието на тези две условия съответната програма се ражда. Разбира се, колкото повече потребители и програмисти има съответната програма, толкова по-бързо се развива.

5.2. Самосъзнание на разработчиците

Много хора имат нужда от дадено нещо, но малко се захващат да го направят. Друго качество, което трябва да притежава разработчика, е самосъзнание и желание за споделяне на идеи и код. Т.е. ако Линус Торвалдс беше разработвал сам Линукс, очевидно, днес ние нямаше да си говорим за Линукс. Това би коствало неимоверни усилия на Линус. Системата щеше да се развива много бавно и сигурно щеше да е във версия 0.2 (за сведение, в момента на писане на статията версията на Линукс е 2.6).

Друг много нагледен пример е проверката на правописа, на която аз съм автор. Лицензът не ме задължава да я направя публично достояние. След като съм я разработил, аз мога да си я ползвам сам. Но ако останалите разработчици правеха така, то очевидно, че нямаше да има Линукс, нямаше да има ОпънОфис орг и, съответно, аз нямаше дори да мога да си мечтая да направя проверка на правописа.

С други думи, участниците в процеса трябва да разберат простата истина, която е написана на сградата на нашето Народно събрание, че „*Съединението прави силата*“. Трябва да разберат, че споделянето на идеи, код и съвместната работа водят до неимоверно ускоряване на процеса на разработка на дадената програма и от това печелят всички.

Естествено, важна роля играе и лицензът ДжиПиЕл. Той се явява като гарант, че работата на разработчиците ще остане свободна и всички бъдещи подобрения, от други автори, ще спомогнат за по-бързото развитие на софтуера.

5.3. Натрупване и повторно използване

В Глава 4.2, „Средства за производство на софтуер“ доказахме, че за разработка на софтуер е необходим софтуер. А именно операционна система и средства за разработка като: компилатори, редактори, интегрирани среди и пр. Очевидно, че аз не мога да създам проверката за правописа без наличието на свободна ОС, без наличието на свободни средства за производство[8] и без наличието на

свободен офис пакет. От своя страна разработчиците на свободния офис пакет ОпънОфис орг, ползват Линукс и компилатора на езика C/C++.

Наличието на свободна операционна система и наличието на свободни средства за разработка се явяват най-важния фактор за развитието на свободния софтуер.

Калоян Доганов в творбата си „Теоретични импликации на софтуерната революция“ е разгледал много подробно и с много примери същността на натрупването на софтуера. Любознателните читатели могат да намерят четивото тук [http://revolution.sourceforge.net/].

Накратко, общността от разработчици на свободен софтуер преди 10 години беше заета да създава свободна ОС и свободни средства за разработка. Естествено, това по никакъв начин на влияе на обикновените потребители. Обикновените потребители искат потребителски програми като: ОпънОфис орг, графични среди, браузъри, счетоводни програми и пр. Т.е. преди да се премине към създаването на потребителските програми, трябваше да се създаде базата за развитието на свободния софтуер. В момента базата е създадена и фокусът е преместен към създаването на потребителските програми. Което от своя страна привлича повече обикновени потребители, а не само разработчици. Разбира се, това води до отражение на свободния софтуер в медиите, в живота на обикновените потребители, в политиката на правителствата и фирмите.

Или казано по друг начин, трябваше да се акумулира (натрупа) едно достатъчно голямо количество от свободен софтуер. Трябваше да се създаде базата на свободния софтуер, за да може той да се явява фактор в днешно време.

Натрупването е възможно благодарение на един основен принцип, който е широко разпространен в свободния софтуер. Ерик Реймънд (Eric S. Raymond) в своето есе „Как да стана хакер“ дефинира много просто въпросния принцип „*Никой и никога не трябва да решава един проблем два пъти*“. Български превод на есето може да бъде намерен тук [http://linux.gyuvet.ch/html/paper/hackers.html]. Въпросният принцип се реализира много лесно със средствата на свободния софтуер. Той е заложен в принципите на свободния софтуер. Принципът помага неимоверно много за развитието на свободния софтуер, понеже увеличава производителността на труда, спестява създаването на едни и същи модули по няколко пъти. Също така, спомага за избягването на другия принцип дефиниран в същото есе „*Скуката и рутината са злини*“. Същият автор доразвива въпросните принципи в едно друго есе - „Катедралата и базарът“. Български превод на

8 За сведение при разработката на проверката на правописа се използва езика Пърл (Perl), локализацията (българизацията) на Линукс от Антон Зиновиев (пакетът БГ Линукс - bglinux). Естествено, ползва се и Линукс и изключително високопроизводителните средства за обработка на текстови файлове като командите: sort, uniq и др.

„Катедралата и базарът“ може да бъде намерен тук [http://catb-bg.sourceforge.net/].

5.4. Развитие на технологиите

В миналото софтуерът се е разпространявал на дискети. Дискетите съдържаха максимум 1.4 МБ информация и струваха около 5 лв. С развитието на технологиите се появиха компактдисковете, които могат да съдържат над 500 пъти повече информация и струват 10 пъти по-евтино от дискетите. В днешно време софтуерът се разпространява или на компактдискове, или чрез Интернет.

Преди 10 години свободният софтуер се е разпространявал по доста допотопен начин. Изпращате 10 лв. в плик на даден адрес и от там Ви пращат колетче с 2-3 дискети. Само за сравнение, в днешно време е необходимо едно щракване с мишката, за да изтеглите някоя програма. Още едно щракване и програмата се инсталира. Ако програмата е свободен софтуер, няма нужда от регистрация, няма нужда да се притеснявате за лицензни такси, няма нужда да се притеснявате, че имате нелегален софтуер на компютъра си.

Един прост пример, зареждате следния адрес http://www.openoffice.org/ и изтегляте офис пакета ОпънОфис орг. В зависимост от връзката може да се наложи да изчакате повечко. Но ако имате бърза връзка, ще Ви отнеме 5-10 мин. След това инсталирате пакета и проверявате дали Ви върши работа, дали отговаря на изискванията Ви. Няма регистрация, няма такси, няма притеснения за нелегален софтуер. Ако Ви върши работа - го ползвате. Ако не - го изтривате. Дори и да го оставите, той не пречи. Никой не може да Ви обвини, че притежавате нелегален софтуер.

Друг важен фактор това е, намаляването на цените на компютрите и на Интернет. В днешно време компютрите са много повече достъпни, отколкото преди 10 год. Интернет също е по-достъпен. Това доведе до увеличаване на потребителите. Някои от тях са разработчици или имат желание да помогнат с нещо, което от своя страна доведе до включване на нови хора в процеса на разработка на свободен софтуер.

Интернет е нова технология, която промени Света. Интернет предоставя много нови начини за общуване, правене на бизнес, обмяна на мисли, идеи, съвместна работа и др. Очевидно това ускорява неимоверно много процеса на разработка на свободен софтуер. Развитието на Интернет допринася изключително много за развитието на свободния софтуер.

6. Кратка история на софтуера

6.1. История на комерсиалните лицензи

В началото когато компютрите са били достъпни само за изследователски центрове или много големи и богати компании, софтуерът се е разпространявал свободно. В началото на 80-те почти всичкия софтуер става платен и лицензиран. По това време лицензът е разрешавал ползването на софтуера само на един компютър, което е логично. С развитието на софтуера лицензите за сървърен софтуер започват да се калкулират на базата на броя процесори, които има компютъра, на който ще го инсталирате или колко потребителя ще се включват към този сървър.

Отначало сървърите са били с доста скромни възможности и ако потребителите искат да изградят голяма система, трябва да закупят, примерно, десет сървъра, което означава и десет лицензa за софтуера, който ще работи на тях. С развитието на хардуера, компютрите са ставали по-мощни. И вместо да купуват десет компютъра, потребителите започнали да купуват, примерно, два компютъра, но с четири процесора всеки. По този начин се е постигала същата изчислителна мощ, но се е спестявало пари от лицензи. Защото във втория случай трябва да закупите само два лицензa. Естествено, производители на софтуер се усетили, че губят и обявяват лицензиране на базата на броя на процесорите (или изчислителната мощ на компютъра). В момента изчислителната мощ на компютрите е нараснала неимоверно много спрямо последните 5 год. Затова сега сървърният софтуер се продава на базата на броя потребители, които могат да се включват към него.

Всички тези неща остават скрити за обикновените потребители, понеже те ползват само персонални компютри и софтуер за тях. Но и те са засегнати от промени в лицензионната политика на Майкрософт. Всички знаят, че Майкрософт обявиха, че вече няма да продават софтуер, а ще го отдават под наем за срок от три години. Това означа, че на всеки три години, потребителите трябва да закупуват нов лиценз, независимо дали искат или не искат да се обновяват.

Много интересен е въпросът *„Малко ли са или са много три години в сферата на ИТ?“* От една страна, три години са много време, защото на всеки шест месеца се появява нещо ново. От друга страна, в момента има работещи системи, които са правени през 80-те и работят успешно над 20 год.

За да придобием по-точна представа много или малко са три години в софтуерната индустрия, нека да разгледаме случая с програмата, която разпространява Данъчната администрация в България. Става въпрос за добре известната програма за ДДС, която се разпространява безплатно. Тя е написана

на Clipper. Clipper е компилатор на езика и системата за управление на бази от данни dBase (СУБД dBase), която от своя страна е продукт на компанията Аштън-Тейт (Ashton-Tate). Аштън-Тейт е една от най-известните и големи компании за производство на софтуер в началото на 80-те. Днес Аштън-Тейт не съществува. Компанията е на път да фалира в началото на 90-те, когато Борланд (Borland) я купува през 1991 год. на безценица. Clipper се разработва от Компютър Асоциейшън (Computer Association - съкратено CA). Clipper има две версии широко използвани в България - Clipper'87 и следващата, подобрена версия Clipper'92. Clipper повече не се поддържа от Компютър Асоциейшън. Поддръжката му е прехвърлена на друга компания. На уеб сайта на Компютър Асоциейшън много трудно може да се намери информация за Clipper.

Както виждате в ДА ползват технология от преди 20 год., като последната версия на продукта е излязла преди 12 год. В този случай, ако трябваше да се плащат лицензни такси, то трябваше да се платят поне четири пъти (4 * 3 = 12).

6.2. История на Майкрософт

В 1981 год. компанията АйБиЕм (IBM) създава персоналния компютър (ПК). По това време фирмата Seattle Computer Products разработва QDOS 0.10 (Quick and Dirty Operating System). Майкрософт откупува правата, за да продава продукта на АйБиЕм. Така се ражда ДОС (Дискова операционна система). ДОС има няколко реализации: на Майкрософт - MS DOS; на АйБиЕм - PC DOS; на Диджитал Ресърч (Digital Research) - DR DOS и още няколко др. Може да намерите кратка история на ДОС тук [http://en.wikipedia.org/wiki/Disk_operating_system].

Майкрософт започва разработката на графична среда, която в последствие прераства в операционната система Уиндоус. Уиндоус версии 3.0, 3.10, 3.11 имат добър успех и се приемат добре от потребителите. Големият бум идва с разработката на Уиндоус 95. От тук нататък успехът на Майкрософт е гарантиран и на практика те успяха да монополизират пазара на ОС за персонални компютри. Разбира се, това доведе и до установяване на монопол върху по-апетитните парчета от пазара на приложни програми за Уиндоус. Може да намерите кратка история на Уиндоус тук [http://en.wikipedia.org/wiki/History_of_Microsoft_Windows].

Как така от почти никому неизвестна компания, Майкрософт се превърнаха в световен лидер в производството на софтуер за персонални компютри? Наистина ли са толкова добри, че разбиха конкуренцията или са родени под щастлива звезда? Или може би използват твърде много нелоялна конкуренция?

Отговорът на горните въпроси е *„По малко от всичко.“* Трябва да се отдаде

дължимото на Майкрософт, че започват да разработват и налагат графична операционна система, нещо необикновено за онова време. Всъщност, графичният потребителски интерфейс (ГПИ) не е разработка на Майкрософт. Компанията, която първа разработва идеята за графичния интерфейс и посочващото устройство тип мишка, е Ксерокс (Xerox). Но по същото време Ксерокс са заети да се борят с конкуренцията от страна на японските производители на електроника и изпускат шанса. От него се възползват Ейпъл и Майкрософт. Може да намерите кратка история на ГПИ тук [http://en.wikipedia.org/wiki/History_of_the_GUI].

В началото има различни версии на ДОС, от различни производители. Майкрософт разработват графичната си среда, която работи на базата ДОС. Има доказателства, че Майкрософт са използвали нелоялни практики, за борба с конкуренцията. Уиндоус 3.10 е отказвал да тръгне, ако не е стартиран на ДОС, произведен от Майкрософт. Както и да е, това едва ли е оказало голямо значение на развитието на Майкрософт и Уиндоус, поради факта, че е имало свободна пазарна ниша за графична операционна система. След успехът на Уиндоус 3.10, идва тоталният успех на Уиндоус 95. От тук нататък, мисля, че всички знаят историята на Майкрософт и няма смисъл да бъде преразказвана.

Фактически Майкрософт използват факторите натрупване и повторно използване, които разгледахме в предната глава. Майкрософт успяха да натрупат голямо количество софтуер, което ги изстреля на върха. Забележете, че те не само акумулираха софтуер, но и го контролираха. Те контролираха операционната система, те контролираха средствата за разработка на софтуер за тази ОС. При тези условия конкуренцията се отказа сама.

За да стане по-ясно и да разбулим напълно тайната на Майкрософт, нека да разгледаме следния пример. Представете си, че Вие сте софтуерна компания, която иска да разработва офис пакет за Уиндоус. Нека да сравним какво Ви трябва на Вас и какво трябва на Майкрософт, за да може да разработите успешно програмата.

1. Вие трябва да закупите Уиндоус - Майкрософт не трябва, понеже те са авторите на Уиндоус и го имат. Значи Майкрософт Ви водят едни гърди напред в надпреварата.

2. Вие трябва да закупите средства за разработка на софтуер - Майкрософт не трябва, понеже те имат разработени средства за производство на софтуер, които са ги използвали за разработка на Уиндоус. Отново Майкрософт са едни гърди пред Вас.

3. Вие трябва да разучите функциите на Уиндоус, за да може да програмирате за него. Майкрософт не трябва, защото те са автори на Уиндоус и имат знания и информация, която Вие нямате, а и няма начин как да я придобиете.

Както виждате битката е с предизвестен край. При така поставените първоначални параметри, за Майкрософт е детска игра да установят монопол върху по-апетитните пазари. Някои от по-любознателните читатели може би ще направят забележката, че Майкрософт не се опитват да монополизират, примерно, пазара на издателски системи. Да, Майкрософт не се се стремят да заемат всички пазарни ниши, защото може да ги обвинят в монополизъм. От друга страна, пазарът на издателски системи е много малък в сравнение с пазара на офис пакети. Издателските системи се правят по-трудно, имат по-малко на брой клиенти и от тук границата на печалба на този пазар е много малка в сравнение с границата на печалба при офис пакетите.

Между другото, имаше заведено дело за монополизъм към Майкрософт в САЩ и се чуха гласове за разделяне на компанията на две. Едната от тях да прави само операционната система, а другата да се занимава с производство на потребителски софтуер. По този начин втората компания ще бъде поставена на равни начала с останалите участници на софтуерния пазар. Но като се вземе факта, че Майкрософт е златната кокошка на американската икономика и много голяма част от приходите на компанията идват от продажби извън САЩ, то не е логично да се поставят пречки пред тези приходи на средства в американската икономика. По тази причина делото беше решено в полза на Майкрософт.

За Уиндоус 3.10 съществуваха различни офис пакети, от различни производители. През 1995 год. Майкрософт започнаха да продават Уиндоус 95. Уиндоус 95 имаше редица предимства пред предшественика си Уиндоус 3.10. Също така, графичните елементи бяха коренно променени. Уиндоус 95 има различен, по-добър външен вид. В частта, която не се вижда директно от потребителя, също има добавени нови функции. Два месеца след излизането на Уиндоус 95, Майкрософт пуска на пазара и офис пакета Майкрософт Офис 95. За тези кратки срокове конкуренцията още не може да разучи новите функции в Уиндоус 95. Очевидно е, че Майкрософт заедно с разработката на Уиндоус 95, са разработвали и Офис 95. Нещо, което конкуренцията няма как да направи.

Както споменахме по-рано ОС е специален вид софтуер, който управлява ресурсите на компютъра и се явява интерфейс между останалите програми и хардуера. Който контролира дадена ОС, може да контролира и пазара на останалите програми за тази ОС. Това е все едно да определяте правилата на

играта. Ако Вие определяте правилата на играта, едва ли някой може да Ви бие на тази игра. Трябва да сте абсолютен некадърник, за да допуснете загуба.

След успехът на Уиндоус 95 и установяване на монопол за персоналните компютри, Майкрософт почти спря да се развива. Ако разгледате Уиндоус 95 и Офис 95, от една страна, и Уиндоус ЕксПи и Офис ЕксПи (XP), от друга страна, ще видите, че няма кой знае какви разлики между тях. Има добавени нови драйвери, поддръжка на нови устройства, нови файлови формати и пр. Но това са неща, които са необходими за развитието на продуктите. Последните почти 10 години няма качествен скок във функционалността или възможностите на Уиндоус и Майкрософт Офис.

Майкрософт се оправя много добре с продуктите, за които контролират средата, в която се изпълняват. Но нека да разгледаме пазара за сървъри, където Майкрософт нямат пълен контрол над ОС, на която да работят сървърите. Става въпрос за сървъра Апаче (Apache), който го разгледахме в предната глава. Майкрософт пуснаха подобен продукт - Интернет Информейшън Сървър (IIS), който с излизането си на пазара зае 30% дял. После успя да стигне до 35%. Това бе и най-високата стойност, която успя да достигне. От тогава насам, Интернет Информейшън Сървър губи по 0.5-1% на месец, за да достигне до 20% пазарен дял в момента. Статистика може да се вземете от сайта НетКрафт (Netcraft [http://news.netcraft.com/archives/web_server_survey.html]). Както се вижда в този случай, Майкрософт не могат да контролират правилата на играта изцяло и затова имат само 20% пазарен дял.

В заключение, трябва да отдадем заслуженото на Майкрософт за техния труд и успехи до 95 год. Но след това - последните 10 год. - след като станаха монополисти те се държат и развиват като монополисти. Също така, трябва да кажем, че която и компания да беше на мястото на Майкрософт, щеше да действа по същия начин. И накрая, голяма част от успехите на Майкрософт се базират на особеностите на софтуера, които разгледахме в Глава 5.3, „Натрупване и повторно използване“, а не изключителни умения и знания на компанията.

6.3. История на ФСС и Линукс

Ричард Столман започва работа в лабораторията за изкуствен интелект в Масачузетския технологичен институт през 1971 год. По това време софтуерът се с обменял свободно. В едно интервю Ричард Столман споделя *„Разменянето на софтуера не беше ограничено само до нашата общност. Тази практика е стара, колкото и компютрите. Както и разменянето на рецепти е старо, колкото и готвенето. Но при нас, в лабораторията, тази практика бе много*

силно изразена.“

Ситуацията се променя коренно в началото на 80-те. Почти всичкия софтуер по това време става комерсиален. Общността от хакери в лабораторията се разпада. Повечето от тях са наети от компании за производство на софтуер. Тези, които са останали, трябва да подписват договори за неразпространение на софтуера, който ползват.

След като общността се разпада, Ричард Столман се изправя пред морален проблем. Той може да стане програмист в някоя компания и да печели добри пари. Но това означава да създава комерсиален софтуер, който той не харесва. Един дребен факт оказва влияние на решението на Столман да създаде Фондацията за Свободен софтуер. Драйверът за принтера, който ползват в лабораторията, няма много възможности и прави използването му кошмар. Ричард Столман има знанията да поправи този драйвер, но компанията производител отказва да предостави изходния код, а и новата версия се бави.

През януари 1984 год. Ричард Столман напуска МТИ и започва да пише свободен софтуер. Първата програма, която написва, е редактора Emacs. Интересът към Emacs нараства. Също така, други хора започват да се включват в проекта. По този начин се появява необходимостта от някакъв формален и легален начин да се управляват нещата. Така през 1985 год. се създава Фондацията за Свободен софтуер. Това е благотворителна организация, освободена от данъци, която се занимава с разработването на свободен софтуер.

До 1990 год. ФСС създава почти всичко необходимо за една операционна система като: редактор, компилатор на езика C/C++, команден интерпретатор, програми за е-поща и др. Но липсва най-важната част, а именно сърцето на една операционна система - ядрото.

В началото на 90-те един финландски студент на име Линус Торвалдс търси добра операционна система, която да замени ДОС, който той ползвал на персоналния си компютър. По това време изходния код на Юникс вече бил тайна и бил затворен. Холандският професор Ендрю Таненбаум (Andrew S. Tanenbaum) разработил малка операционна система - Миникс (MINIX), на която да може да обучава студентите си. Миникс била за персонални компютри с дизайн, заимстван от Юникс. Изходният код на Миникс бил достъпен и се разпространявал с учебника по ОС на професора. Линус Торвалдс използвал Миникс за база на ядрото, което смятал да създаде.

Мисля, че не е нужно да разказваме историята от 1991 год. до сега с подробности. Но на кратко, Линус пуска своето произведение в Интернет и моли за коментари,

идеи и помощ. Ядрото е лицензирано под лицензa ДжиПиЕл. Линус посочва, че основната причина да избере този лиценз е факта, че компилатора на езика C, който ползва, е разработен от Фондацията за Свободен софтуер и той от своя страна се разпространява под ДжиПиЕл.

Повече подробности за историята на Линукс и Фондацията за Свободен софтуер, може да намерите тук [http://www.linux-bg.org/cgi-bin/y/index.pl?page=article&id=history&key=352608315] и тук [http://linux-bg.org/cgi-bin/y/index.pl?page=article&id=history&key=334234041].

7. Бизнес модели за свободния софтуер

На теория свободният софтуер не е безплатен. Но на практика свободния софтуер се явява безплатен. Става въпрос за масово използваните програми. Те на практика са безплатни за крайните потребители. Никой не Ви спира да изтеглите ОпънОфис орг или някоя друга програма без да плащате. Тогава възниква въпроса, как да се прави бизнес на базата на свободния софтуер.

7.1. Софтуерът - продукт или услуга

Интересен въпрос *„Какво е софтуерът, продукт или услуга?“* Май повечето хора си мислят, че е продукт. В ранните години на софтуерната индустрия, софтуерът се е разглеждал като продукт. Но в последствие, започна да се разглежда като услуга. Дори Майкрософт обяви, че разглежда софтуера като услуга.

Сървърите за бази от данни (а и не само те) имат различна цена в зависимост от това, дали го ползват 10 или 100 човека. Ако разгледаме един типичен продукт като кафе машина, ще видим, че подобно ограничение не съществува за кафе машината. Когато си купувате кафе машина, никой не Ви пита дали ще я ползват 10 или 100 човека. Значи има разлика между кафе машината и сървърите.

Нека да разгледаме и нашумялия натиск над Интернет залите, че не могат да ползват Уиндоус 98, защото нямат право да го отдават под наем. И отново примера с кафе машината. Когато си купувате кафе машина, никой не Ви пита дали ще я ползвате у Вас или ще правите кафене с нея.

И последния пример, разбира се, заявката на Майкрософт, че те не няма да продават повече Уиндоус, а ще го отдават под наем за срок от 3 години.

Големите фирми, държавните агенции и др. сключват договори за цялостно софтуерно обслужване. Доставка на софтуер, инсталиране, разработка на нови

системи, интегриране на системи, обучение на персонала и пр. Ако разгледаме една такава сделка, то дялът на софтуера в цялата сделка е много малък. В крайна сметка, стигаме до заключението, че софтуерът е услуга.

7.2. Разпространение и поддръжка на Линукс

Преди да преминем към тази тема трябва да разгледаме някои определения и разлики между Линукс и Уиндоус.

Потребителите, които идват от света на Уиндоус, знаят, че след като инсталират Уиндоус, те трябва да инсталират още няколко приложения, за да стане използваем. Това е така, защото Уиндоус има много малко програми, включени в инсталационния компактдиск. За разлика от него, Линукс идва на няколко компактдиска (от 3 до 7, според дистрибуцията). Обикновено, по време на инсталиране на Линукс се инсталират всички необходими програми за Вашата работа. Поради тази причина, когато се спомене Линукс или дистрибуция на Линукс, се разбира пълен комплект от програми, необходими за всекидневната работа. Става въпрос за програми и библиотеки от порядъка над 10 000.

Линукс е структуриран по различен начин от която и да е друга операционна система. Поради свободния характер на системата и факта, че не може всичко да се изпълнява от един проект (човек), имаме множество малки проекти, които не са свързани помежду си. Примерно, Линус Торвалдс ръководи програмирането на ядрото, което като дял е около 2% от цялата система. Други проекти създават различни програми, с чиято помощ се окомплектова цялата система.

Окомплектоването на Линукс не е лесна задача. То включва събирането на различни програми от Интернет заедно и създаването от тези програми на една цялостна, завършена система. С тази нелека задача са се заели различни фирми. Те събират програмите от Интернет, компилират ги, записват ги на компактдиск, добавят инсталираща програма и ги разпространяват. Това се нарича дистрибуция на Линукс. По-известните дистрибуции на Линукс са Ред Хат (RedHat [http://redhat.com/]), СуЗе (SuSE [http://suse.com/]), Мандрейк (Mandrake [http://mandrake.com/]). Пълен списък на дистрибуциите може да се намери на сайта http://distrowatch.com/.

Изброените по-горе дистрибуции са комерсиални. Т.е. правят се от фирми. Разбира се, има и некомерсиални дистрибуции. Т.е. дистрибуции, които се правят от доброволци. Дистрибуции, които се развиват по същия начин, както се развиват проектите с отворен код. Най-известната, некомерсиална дистрибуция е Дебиан (Debian [http://debian.org/]).

Основната разлика между комерсиалните и некомерсиалните дистрибуции е същата, както между комерсиалния и свободния софтуер. Компаниите правят дистрибуциите, за да печелят пари. Некомерсиалните дистрибуции съществуват, за да бъдат ползвани от създателите им, а не за да бъдат продавани (пряка и косвена печалба).

В днешно време почти всички компютри са включени в Интернет. Това ги прави изключително много уязвими за атаки от най-различни злонамерени индивиди. Това от своя страна, прави поддръжката и обновяването на една операционна система изключително важен фактор. Всяка една модерна операционна система има начин за обновяване на програмите. Често се откриват грешки или пропуски в сигурността на софтуера. След като някоя грешка бъде открита и оправена се пускат сервизни пакети или кръпки[9] за съответната програма. Въпросните сервизни пакети или кръпки най-често се разпространяват по Интернет. Обикновено, операционната система автоматично проверява за наличието им и Ви уведомява.

7.3. Продажба на услуги

Правенето на бизнес със свободен софтуер трябва да се различава от продажбата на несвободен софтуер. Лицензът ДжиПиЕл задължава авторите на свободния софтуер да предоставят същите права на потребителите. Т.е една фирма не може да се издържа от продажбата на свободен софтуер, защото всеки клиент има право да използва програмата на неограничен брой компютри, да я разпространява и дори да я продава.

Вместо продажбата на софтуер, компаниите, занимаващи се със свободен софтуер, се насочват към услугите. Събиране на няколко хиляди програми на компактдиск и продажбата им. Създаване на автоматична система за обновяване и поддръжка. Изграждане на нови компютърни системи на базата на свободен софтуер, мигриране на съществуващи системи към свободен софтуер с цел намаляване на разходите по експлоатация и др.

7.4. Двойно лицензиране

Друг начин за правене на бизнес това е, двойното лицензиране на даден продукт. Някои от по-известните продукти, които се разпространяват под два лиценза, са библиотеката Qt, системата за управление на бази от данни MySQL, офис пакетът StarOffice. Ако искате, може ползвате този софтуер за лични нужди или за

9 В Уиндоус и възприет терминът „Service Pack“, докато в Линукс се използва терминът „Patch“.

създаване на свободни програми. Но ако искате да правите несвободен софтуер с въпросните програми, трябва да си закупите, копие което Ви дава правото да правите несвободни програми и да ги продавате. Нека да разгледаме отделните случаи.

Библиотеката Qt започва развитието си като много-платформена библиотеката за създаване на графични приложения. В последствие се добавят още един куп функции, които я превръщат в конкурент на Java и .NET. С помощта на библиотеката Qt, може да създавате програми за Уиндоус, Линукс, Юникс и Макинтош. Под Линукс библиотеката има версия, която се разпространява под лицензa ДжиПиЕл. Което означава, че може да я ползвате, за да създавате само свободни програми, лицензирани под ДжиПиЕл. Под останалите платформи библиотеката Qt се разпространява с комерсиален лиценз. Също така, и под Линукс има версия, която се разпространява с комерсиален лиценз. Това дава свободата на потребителите на избират. Ако искате да пишете свободна програма или програма за свои собствени нужди, може да използвате свободната версия под Линукс. Ако искате да създавате комерсиален софтуер, поръчвате си комерсиалната версия и може да създавате комерсиален софтуер за Линукс, Юникс, Уиндоус и Макинтош. Също така, комерсиалната версия има повече възможности от свободната.

Положението със системата за управление на бази от данни (СУБД) MySQL е подобно. Разбира се, свободната версия няма поддръжка от страна на компанията. Ако искате поддръжка, трябва да си платите.

И третият случай, който ще разгледаме, е офис пакетът СтарОфис (StarOffice). Това е версия на ОпънОфис орг. СтарОфис е собственост на фирмата Сън Майкросистемс (Sun Microsystems). Преди няколко години фирмата взема решение да отвори изходния код на СтарОфис. По този начин се ражда ОпънОфис орг. Изходният код на офис пакета ОпънОфис орг се разпространява под два лицензa - под ДжиПиЕл и под специален лиценз на Сън. Това дава право на Сън да използва изходния код на ОпънОфис орг, да го модифицира, да го затвори и да продава комерсиалния офис пакет СтарОфис. В сравнение с ОпънОфис орг, СтарОфис има добавени няколко допълнителни модула и функции. Разбира се, СтарОфис има поддръжка от страна на Сън.

8. Има ли безплатен обяд

8.1. Край на безплатното

Преди няколко години, когато Интернет не беше толкова развит, печеливш бизнес модел беше да се продават дистрибуциите на Линукс на компактдискове. Със закупуването на дадена дистрибуция на Линукс, потребителят получаваше повече, отколкото със закупуването на Уиндоус, а и цената определено беше по-ниска.

С развитието на Интернет и поевтиняването на записващите устройства, стана сравнително лесно да се изтегли дадена дистрибуция от Интернет и да се запише на компактдискове в домашни условия. Дори стана по-евтино и по-бързо да се използва този начин, отколкото да се отиде до магазина и да се закупи дистрибуцията за 50 долара. От друга страна, за да привлекат повече потребители, компаниите, занимаващи се с дистрибуция на Линукс, дълго време разрешаваха безплатно изтегляне на техните дистрибуции.

Днес за Линукс се говори все повече и повече. Все повече и повече организации го избират, вместо Уиндоус. Някои изследвания, за мотивите за миграция или избор на Линукс пред Уиндоус, нареждат причините за избора на Линукс в следния ред:

1. Стабилност - Линукс е по-стабилен от Уиндоус, не забива и няма нужда да се рестартира.
2. Сигурност - Линукс е по-сигурен и устойчив на вируси и атаки, отколкото Уиндоус.
3. Възможността да нагласяш почти всички параметри на Линукс, нещо което го няма в другите ОС.
4. По-ниска цена на закупуване и поддръжка.

Както виждате цената е поставена на четвърто място. Разбира се, става въпрос за изследвания в САЩ, където 500 долара е средната седмична заплата и заделянето на пари за покупката на Уиндоус не е голям проблем, както в България. В България, може би, цената ще бъде поставена на първо място заедно със стабилността, ако се проведе подобно изследване. Наблягам на тези факти, за да стане ясно, че с развитието на Линукс, компаниите, занимаващи се с дистрибуция му, ще спират безплатните оферти и ще започнат да слагат цени на услугите, които извършат.

Много хора преди се чудеха как така можеш да си изтеглиш Линукс безплатно от

Интернет. И съвсем логично задаваха въпроса *„Има ли безплатен обяд?“* Голяма част от тях го ползваха и без да си задават този въпрос. В края на 2003 год. компанията Ред Хат (RedHat), лидер сред дистрибуциите на Линукс, обяви, че прекратява безплатното разпространение на Линукс, спира безплатната поддръжка и обновяване по Интернет и съответно обяви цени за въпросните услуги. Доста хора, свикнали на безплатното, се ядосаха. Други, които бяха само наблюдатели, си казаха *„Що да мигрираме на Линукс, след като и той стана платен?“* Трети мъдро заключиха *„Няма такова нещо като безплатен обяд.“* Даже министър Калчев в становището на Министерството на държавната администрация, отправено по повод искане от Комисията по транспорта и съобщенията към Народното събрание, заяви *„Ред Хат спря безплатната поддръжка на продуктите си.“*

Голяма част от хората не разбират икономическите процеси, а ако ги разбират, тогава пък идея си нямат от ИТ или разглеждат софтуера, като нормален продукт. Въпросните стъпки от страна на компаниите, разпространяващи Линукс, са напълно нормални и очаквани.

Една компания съществува, за да печели пари. Тя трябва да има приходи, защото фирмите имат разходи по заплати на персонала, наеми на офиси, данъци, осигуровки и др. Ако бизнес моделът на фирмата е да печели пари от дистрибуция на Линукс, от поддръжка по Интернет и автоматично обновяване, в такъв случай след първоначалните промоции за привличане на потребители, е абсолютно логично фирмата да спре с безплатните неща и да започне да иска пари за услугите, които предлага. Не забравяйте простата истина, на която се крепи икономиката. Основната цел на една фирма е да печели пари. Фирмите се занимават с дистрибуция на Линукс, за да печелят пари.

8.2. Пречки в комерсиалните дистрибуции

В бъдеще фирмите, занимаващи се с дистрибуция на Линукс, ще използват различни похвати, за да задължават потребителите да плащат. Забележете, че не става въпрос за трикове за заобикаляне на лицензa ДжиПиЕл или други нелоялни похвати. Това са абсолютно нормални похвати.

Ограничения при обновяването. Обновяването на операционната система е критичен фактор в днешно време. Услугата за автоматично обновяване е успешен бизнес модел. Ако изтеглите дадена дистрибуция на Линукс безплатно от сайта на компанията, нямате право на обновяване. Ако си закупите копие на дистрибуцията, имате право на обновяване в рамките на даден период. Лицензът не Ви спира да инсталирате дистрибуцията на 10 компютъра, въпреки че сте

закупили само едно копие. Проблемът е, че ще имате автоматично обновяване само за единия от компютрите. По същия начин, лицензът не Ви спира да копирате кръпките на останалите компютри. Проблемът е, че операцията отнема време и ресурси и излиза по-евтино да си поръчате обновяване за 10 компютъра, а не да се занимавате да разгадавате как работи системата за обновяване, за да може да я заобиколите.

Някои дистрибуции включват в себе си и несвободен софтуер. Примерно, някои версии на дистрибуцията СуЗе съдържат офис пакета СтарОфис, който не е свободен софтуер. По този начин може да инсталирате дистрибуцията само на един компютър. Ако я инсталирате на втори компютър, трябва да сте сигурни, че на втория компютър инсталирате само свободен софтуер.

8.3. Некомерсиални дистрибуции на Линукс

Както съществуват комерсиални дистрибуции и програми, така съществуват и некомерсиални дистрибуции. Некомерсиалните дистрибуции се правят от доброволци и движещата сила отново е косвената печалба. Разработчиците събират различните програми и окомплектоват дистрибуция на Линукс. Компаниите правят дистрибуции на Линукс, за да печелят пари. Некомерсиалните дистрибуции съществуват, за да бъдат ползвани от създателите им, а не за да бъдат продавани. Разработчиците направиха толкова свободен софтуер и сега не върви да го купуват от компаниите, занимаващи се с дистрибуция на Линукс (или да купуват услуги по обновяване). Това е все едно да продаваш краставици на краставичар. Най-известната некомерсиална дистрибуция е Дебиан (Debian [http://debian.org/]). Няколко факта за Дебиан:

- По същия начин както общността от разработчици успя да създаде свободен софтуер, по-добър от комерсиалния, така същата общност успя да създаде и дистрибуция, която е по-добра от комерсиалните дистрибуции на Линукс. Без никакви скрупули, Дебиан може да бъде обявена за най-добрата дистрибуция.
- Дебиан има най-мощната система за обновяване на софтуера.
- Понеже е некомерсиална дистрибуция, Дебиан няма ограничения за изтегляне и обновяване, които ги имат комерсиалните дистрибуции..
- Дистрибуцията на Дебиан включва над 10 000 програми.
- Дебиан има над 1 000 разработчика, които са отговорни за пакетиране и поддръжка на програмите, които разпространяват.

- По-важните решения в Дебиан се вземат чрез гласуване. Право на глас имат регистрираните разработчици.

- Лидерът на Дебиан се избира чрез гласуване всяка година. Повече информация за ролята и начина на избиране на лидерът на Дебиан може да видите тук [http://www.debian.org/devel/leader].

- Дебиан има Обществен договор, в който се споменават какви са задълженията от страна на Дебиан, към неговите потребители. Общественият договор може да бъде намерен тук [http://www.debian.org/social_contract].

- Дебиан има годишен бюджет от около 10 000 до 30 000 долара. С този нищожен, направо смешен, бюджет разработчиците на Дебиан успяват да създадат най-добрата дистрибуция.

- Половината от разработчиците на свободен софтуер използват Дебиан. Както споменахме по-горе, на компаниите е трудно да продават краставици на краставичар. Може да видите статистиката тук [http://www.infonomics.nl/FLOSS/floss1/stats_13.html].

- Последните шест месеца Дебиан има най-бърз темп на разпространение спрямо останалите дистрибуции. Отчасти това се дължи на факта, че комерсиалните дистрибуции започнаха да слагат цени за услугите си. Може да видите статистиката тук [http://news.netcraft.com/archives/2004/01/28/debian_fastest_growing_linux_distribution.html].

Пълно изследване и информация за Дебиан има тук [http://telemetrybox.org/tokyo/]. Статията е задължителна за мениджерите в сферата на ИТ, за да видят как нараства броя на поддържаните пакетите, броя на отговорниците по поддръжка и производителността на труда. В документа се споменава, че един отговорник е отговорен средно за 350 софтуерни пакета и производителността на труда е нараснала 3-5 пъти с разработването на системата за автоматично управление на пакетите. Няма комерсиална дистрибуция, която може да се похвали с подобни успехи. Частичен превод на документа на български език може да бъде намерен тук [http://linux-bg.org/cgi-bin/y/index.pl?page=article&id=history&key=345321592]. Може да прочетете много аргументирани твърдения *„Защо Дебиан“* тук [http://people.debian.org/~srivasta/talks/why_debian/talk.html]. Статията е на английски.

8.4. Отговор на въпроса за обяда

Отговорът на въпроса *„Има ли безплатен обяд?“* е *„Да, има безплатен обяд, ако може да готвите.“* За мен обядът е безплатен, защото мога да готвя. Като професионалист, занимаващ се с ИТ и софтуер, мога да инсталирам, да настроя и да ползвам ОС Линукс. От тази гледна точка на мен ми излиза безплатно или по-евтино. Все пак аз заделям от свободното си време да създавам свободен софтуер. Но има доста голям брой потребители, които само ползват, без да създават свободен софтуер. Те са инвестирали само време и усилия да се запознаят с Линукс, но не и средства за закупуване, обновяване и лицензи. От тази гледна точка, в сравнение с Уиндоус на тях им излиза безплатно, защото не са похарчили пари.

Ако разгледаме въпроса от гледна точка на обикновените потребители, то той трябва да се перифразира по следния начин *„Може ли да се постигне по-качествен и по-евтин обяд?“* Отговорът на така поставения въпрос е *„Твърдо да.“*

Обикновените потребители имат полза от конкуренцията. Изборът и конкуренцията са майка на прогреса. Изборът и конкуренцията означават по-ниски цени, по-добро качество, постоянно развитие и усъвършенстване на продуктите. Монополът е точно обратното на изброените неща. Потребителите имат избор между Уиндоус, Юникс, комерсиална или некомерсиална дистрибуция на Линукс. Некомерсиалните дистрибуции също имат поддръжка. Потърсете за специалист или фирма, която се занимава с Линукс. Сигурен съм, че ще намерите платена поддръжка, обучение, инсталиране, настройки, обновяване и пр. на разумни цени.

9. Проблеми в софтуерната индустрия

9.1. Провали в софтуерната индустрия

Последните години софтуерната индустрия изпитва изключително сериозни проблеми. Нека да разгледаме следното проучване IT project failure is rampant - KPMG [http://www.theregister.co.uk/content/7/28299.html], което показва състоянието на софтуерната индустрия. Ето превод на няколко цитата от проучването:

Проучването на КаЕмПиДжи (KMPG), което покрива 134 компании в Англия, САЩ, Африка, Австралия и Европа отчита, че 56% от фирмите са имали поне един провален проект в сферата на ИТ през последната година. Средната загуба от такива проекти е около 12.5 милиона Евро, като най-голямата единична загуба е била 210 милиона Евро.

Проучването също така твърди, че само 9% от фирмите смятат, че завършването на проекта в рамките на бюджета е най-важната мярка за успех на проекта. Други 21% смятат, че най-важната мярка за успех е завършването на проекта навреме.

Най-често като причини за провал се споменават лошото управление, лошата комуникация между техническия и бизнес персонал. Около 67% от компаниите са споделили, че управлението на програмирането (създаването на софтуера) има нужда от подобрение.

В изследване на Световната Банка за софтуера с отворен код, което ще разгледаме по-късно, между другото се споменава за проучване, подобно на проучването на КаЕмПиДжи, което показва, че само 26% (един от четири) от софтуерните проекти са имали успешен край. В проучването също така се твърди, че успехът на даден проект не зависи от използваните технологии (свободни или несвободни). Отново като главни причини за провала на проектите се изтъкват неясни цели, честа смяна на заданието, лошо управление и пр. субективни фактори. Изследването се намира тук [http://www.infodev.org/symp2003/publications/OpenSourceSoftware.pdf].

И накрая, софтуерната индустрия реализира спад от 12% през 2001 и 10% през 2002. За съжаление, все още няма данни за 2003 год.

9.2. Липса на подготвени кадри

Основният проблем, който се премълчава или се поставя на последно място от всички, е липсата на подготвени кадри.

За създаването на софтуер са необходими подготвени хора, докато за някои други отрасли освен кадри са необходими и природни дадености, рудни залежи и пр. Примерно, за развитието на туризма трябва природа. Повечето от провалите в софтуерната индустрия се отчитат като обективни: лоша комуникация, лошо планиране, неясни задания от страна на клиента и др. Но всички те водят към липсата на подготвени кадри. Не може да се оправдава провалът на даден софтуерен проект със сушата, кишата, комунизма и „таварищ Путин“ (в стария вариант на вица бе империализма и силите на злото).

В това няма нищо изненадващо, компютрите навлязоха със шеметна скорост във всяка една сфера на нашия живот - стопанската, финансовата, политическата и пр. Търсенето на кадри изпревари подготовката им. За обучението на един добре подготвен специалист в сферата на ИТ са необходими 5 год. в университета и

после около 5 год. практика и чиракуване. Точно толкова, колкото и за подготвянето на един добър инженер или лекар. Като говорим по темата, мисля, че трябва да отдадем дължимото на Тодор Живков и политиката на компютаризация в средните училища с добре известния персонален компютър Правец 82, чиято цел бе да запознае колкото се може по-рано бъдещите кадри с компютрите. По този начин се целеше изместване на необходимия срок за обучение в по-ранна възраст.

Голямото търсене на специалисти доведе до покачване на заплатите в бранша. Което от своя страна доведе до наплив от хора от други професии и/или без завършено висше образование. Според някои данни над 70% от хората, заети в софтуерната индустрия са от други професии. Това от своя страна доведе до изброените по-горе провали в сферата на ИТ.

За радост, България е от страните с достатъчно подготвени кадри и гладът за професионалисти не се усеща толкова много, колкото в някои други страни. От горното правило има изключения. ЦАПК Прогрес открито си признават, че нямат подготвени кадри. Един цитат, който може да бъде намерен тук [http://egateway.government.bg/FAQs.htm]: *„Поради необходимостта от интегриране на допълнителна функционалност в модулите на Електронния портал на Българското правителство, с която да се осигури възможността Порталът да се ползва не само от потребители на продуктите на Microsoft Co., ще сме изключително благодарни на всеки, който ни помогне по някакъв начин и ни насочи към решаването на проблема с цифровото подписване на документи под Linux ОС. Фирма ЦАПК Прогрес ООД не разполага със специалисти, които биха могли да реализират едно такова решение, включващо модул за цифрово подписване в браузърите, различни от Internet Explorer v5.5, които се ползват от потребителите на Linux ОС.“*

9.3. Неспазване на стандартите

Един от най-големите проблеми на софтуерната индустрия е липсата на стандарти и/или неспазването на наличните стандарти. Последните 5-10 год. се наблюдават усилия за създаване на стандарти. Трябва да се отбележи, че днес има стандарти в почти всяка една подсфера на софтуерната индустрия. Все още липсват някои крайно необходими стандарти като стандарт за обмяна на електронни документи. В момента като такъв се е наложил форматът на Майкрософт Уърд. Проблемът със съществуващите стандарти е, че почти никой не е запознат с тях и съответно никой не ги спазва. Трудно е да се обясни на някой взел 1-2 курса и станал програмист какво е стандарт, от къде може да се

намери и защо трябва да се спазва. От друга страна, компаниите толерират неспазването на стандартите и използването на техните собствени спецификации. По този начин се цели дългосрочно обвързване на клиента с продуктите на компанията.

За съжаление, много малко се говори за стандартите. Обикновено, темата се измества в друга посока. За сега проблемът не е много наболял, но в бъдеще това ще се превърне в един от най-големите проблеми на софтуерната индустрия.

Най-болезнените случаи това са уеб сайтовете, които се отварят само с браузъра на Майкрософт - Интернет Експлорър (Internet Explorer). В българското Интернет пространство имаше няколко дискутирани случая със сайтове, изискващи само продукти на Майкрософт. Става въпрос за сайта на Националната здравна каса, за сайта на БДЖ, където човек може да провери разписанието на влаковете, за сайта на е-правителството.

Няма видими причини тези страници да не спазват стандартите. Няма проблеми със спазването на стандартите да се постигне същата функционалност. Проблемите са в недостатъчната подготовка на хората, създали съответните страници.

За да изясним проблемът със стандартите, ще направим съпоставка с мобилните телефони и уеб страниците.

Ако си купите мобилен телефон, поддържащ стандарта ДжиЕсЕм (GSM), може да се включите с него към всеки доставчик, поддържащ същия стандарт. Може да си купите телефон Нокия, Сименс, Панасоник или някоя друга марка. С този телефон може да се включите към Мобилтел или Глобул. Също така, в чужбина, имате връзка с други оператори, които предоставят роуминг. От една страна, имаме стандарт ДжиЕсЕм. От друга страна, имаме телефонни централи, поддържащи този стандарт. От трета страна, имаме телефонни апарати, поддържащи стандарта. Всичко това води до факта, че централите на един производител могат да комуникират с телефоните на друг производител. Това означава избор, конкуренция, постоянно развитие на продуктите, по-ниски цени и пр.

Връзката между уеб сървър и браузър е точно същата, както връзката между телефонна централа и телефонен апарат. В случая с телефоните стандартът се спазва. В случая с уеб страниците също има стандарти, но те не се спазват. Вместо това се използват собствените спецификации на Майкрософт, които водят до нарушение на стандарта и визуализиране на страницата само на продукти на

Майкрософт. Това е пълна противоположност на конкуренцията и избора. Това означава дългосрочно обвързване с една компания, липса на избор, липса на конкуренция, спиране на развитието на продуктите и плащане на лицензни такси на всеки три години.

9.4. Преодоляване на проблемите

Проучването на Световната банка показва, че изборът на софтуер (свободен или затворен) не указва влияние на успеха на един проект. Това е абсолютно вярно. Ако някой не може да го направи със затворен софтуер, той няма да може да го направи и със свободен софтуер. И обратно. Ако някой може да го направи със затворен софтуер, той ще го направи и със свободен. От тази гледна точка, свободният софтуер не е решение на всички проблеми, но въпреки всичко той може да помогне в някои аспекти. Преди да стигнем до разглеждането на свободния софтуер и как може той да помогне на един проект, нека да разгледаме задължителните неща, които трябва да се направят.

Първото важно нещо е подборът на кадрите. Съответно с дългосрочен план създаване на такива кадри или поне създаване на предпоставки за създаване на висококвалифицирани специалисти.

Второто нещо, разбира се, са стандартите. Приемането на международните стандарти като български и използването им. Или просто използването им в създаването на бъдещи софтуерни проекти. Всъщност повечето клиенти, които сключват договор за разработка на специфичен софтуер, трябва да ги включват в договорите и да задължават фирмите изпълнителки да ги спазват. Това е задължително условие. Подценяването на този фактор може да доведе до изключително сериозни проблеми в бъдеще. Ако два модула комуникират помежду си използвайки стандарти, то замяната на единия модул няма да окаже влияние на система. Ако двата модула комуникират по нестандартен начин, то ако се наложи замяната на единия модул, това автоматично ще повлече и замяната на втория модул.

Затова абсолютно задължително е използването на установените стандарти, там където ги има. Забележете, че тук не става въпрос за свободен софтуер, софтуер с отворен или затворен код. Тук става въпрос за много важен принцип използван в създаването на който и да е продукт. Просто стандартите са навсякъде в нашия живот и тяхната задача е да го улеснят. За съжаление, това не е точно така в софтуерната индустрия. Подценяването на проблема ще доведе до ескалацията му.

Въпреки че свободният софтуер не е панацея, той може да помогне в някои аспекти. Най-очевидното е липсата на лицензи такси, които трябва да се плащат на всеки три години. Свободният софтуер може да помогне с повторното използване на модули. Всъщност, повторното използване на модули е основното нещо, което може да помогне за преодоляване на проблемите. Също така, свободния софтуер е по-надежден, по-сигурен, има по-ниски разходи на поддръжка и, най-важното, не обвързва клиента с даден производител.

10. Мигриране към свободен софтуер

10.1. Състояние на свободния софтуер

В момента Линукс е готов заместител за Уиндоус. Има голям избор от графични среди, подходящи за начинаещи потребители. Някои от комерсиалните дистрибуции като Мандрейк и СуЗе са по-лесни за инсталиране и настройка, отколкото Уиндоус. Разпознаването на хардуера и настройката на системата става автоматично. Има около десетина стъпки, където трябва да потвърдите или въведете информация от сорта на потребителски име и парола за влизане в системата.

За Линукс има няколко браузъра. По известните са Mozilla и Konqueror. Те са свободен софтуер. Браузърът Opera, който е комерсиален, също го има за Линукс.

Има много богат избор на клиенти за е-поща. С Mozilla идва един. В КДЕ има вграден, който се казва KMail. Има две системи за синхронизиране на работата в екип. Става въпрос за проекта Kroupware, който разгледахме по-рано и за който казахме, че е създаден по поръчка на Германското правителство. Той се интегрира напълно с графичната среда КДЕ. Evolution е друг продукт за работа в екип.

Има няколко офис пакета с отворен изходен код. Най-известният е OpenOffice.org. Друг по-известен е KOffice, който се интегрира с КДЕ. Трябва да спомена и AbiWord и Gnumeric за графичната среда Гном.

В Линукс има изключително голямо разнообразие от езици за програмиране, редактори и интегрирани среди за разработка. Това е система направена от разработчици и е логични изборът на средства за разработка да е голям.

До момента разгледахме накратко използването на Линукс като работна станция. Не знам дали е нужно да разглеждаме възможностите на Линукс като сървър. Фактът, че сървърът Апаче работи на 70% от сървърите в Интернет мисля, че е

повече от достатъчен. 90% от сайтовете в България използват свободен софтуер и по специално: Апаче, базите от данни MySQL или PostgreSQL, езиците за програмиране на уеб приложения (уеб сайтове и портали) PHP, Python, Perl. Всъщност въпросната комбинация е позната като LAMP (Linux, Apache, MySQL, PHP или Python, или Perl).

Почти всички производители на сървърен софтуер са прехвърлили продуктите си за Линукс, въпреки че те от своя страна не са свободен софтуер. Става въпрос за АйБиЕм (IBM) и техните продукти като базата от данни DB2 и сървъра WebSphere. Всички продукти на Оракъл (Oracle) ги има за Линукс. Даже от компанията заявиха, че минават изцяло на Линукс. Всички компютри, които ползват във фирмата, ще работят на Линукс. Продуктите на Сън Майкросистемс (Sun Microsystems) също ги има за Линукс. Става въпрос за Java и технологиите, свързани с Java.

Има няколко интересни изследвания, които са много задълбочени и описват много добре състоянието на свободния софтуер. Изследванията могат да бъдат намерени в секцията „Библиография“ и са задължително четиво за мениджърите в сферата на ИТ.

10.2. Състояние на свободния софтуер в България

За да сме наясно дали може да се използва свободен софтуер за работна станция, трябва да разгледаме състоянието на строго специфичните програми, насочени към българския пазар и потребител.

Линукс е локализиран доста отдавна от Антон Зиновиев. Той е автор на пакета bglinux. Датите, часът, превключване между кирилица и латиница, и др. неща, специфични за българския потребител са направени. Пакетът bglinux отдавна е част от по-големите дистрибуции на Линукс и автоматично се инсталира, ако изберете в началото български език.

Проектът БГ Офис [http://bgoffice.sourceforge.net/], на който аз съм автор, има за цел да направи свободна проверката на правописа за продуктите със свободен код. Освен проверка на правописа, има модул за сричкопренасяне, синонимен речник, двупосочен българско-английски речник. Всички те се интегрират с почти всички свободни програми, които имат нужда от такива модули. Става въпрос за клиенти за е-поща, офис пакети, текстови редактори и др. Проектът има и модул за граматическа проверка на български език, който е в начална фаза на разработка. Плановете включват и разработка на система за автоматичен превод от и на английски език. Продуктите на проекта си пробиват път към дистрибуциите

на Линукс. Дебиан и някои от комерсиалните дистрибуции предлагат части от проекта БГ Офис, които се инсталират автоматично.

Голяма част от свободните програми са преведени на български, но все още качеството на преводите не е задоволително. Има още какво да се желае в тази насока. За съжаление, още много работа трябва да се свърши, за да стане системата използваема за хората, които не разбират английски език.

За съжаление с това се изчерпват специфичните български програми. А определено има нужда от счетоводни и финансови програми, понеже те са най-често използваните от бизнеса.

В България съществуват няколко различни неформални групи свързани с Линукс. Има официално регистрирано сдружение за свободен софтуер, чиито цели са да рекламира Линукс. Сдружението се занимава с организиране на инсталационни фестове и семинари. За съжаление, сдружението не развива софтуерни проекти. Адресът на сдружението е http://www.fsa-bg.org/.

Един от най-известните сайтове за информация, новини, статии и взаимопомощ в България е Линукс за българи [http://www.linux-bg.org/].

10.3. Бъдещето на свободния софтуер

В предните глави разгледахме движещите принципи на свободния софтуер. Надявам се, че успях да убедя читателите в ползата на свободния софтуер. При така зададените параметри на системата е ясно, че свободния софтуер ще продължи да съществува и да се развива с по-бързи темпове, от несвободния софтуер. След като има свободни средства за разработка на софтуер, след като има много голямо натрупване на свободен софтуера, мисля, че е ясно, че фокусът ще се премества към потребителските програми. Свободният софтуер достигна критична маса, която ще му позволява да се разраства лавинообразно.

Ако дадена програма има много потребители, които са и разработчици, то тя ще се развива бързо. По тази логика прогнозата за бъдещето е лесна. Програмите за всекидневна употреба, които имат достатъчно критична маса от потребители - разработчици, ще се развиват бързо. Програмите, които имат ограничен брой клиенти, ще продължат да струват скъпо. Ако Вие сте единствения потребител на дадена програма, не може да изисквате някой друг да я направи без пари. Ако я поръчате на фирма, то програмата ще струва скъпо. Независимо от това дали се разработва със свободен или комерсиален софтуер. Предимството в този случай може да дойде от липсата на лицензни такси и разходи за закупуване на комерсиален софтуер, необходими за разработката и експлоатацията на

системата. Разбира се, и свободата да избирате и управлявате по-гъвкаво ИТ във Вашата фирма.

10.4. Практически съвети

Изграждането на дадена информационна инфраструктура или система, зависи от много неща. Изискванията за 1 компютър се различават от изискванията за 10 компютъра. Те пък от своя страна се различават от изискванията на организация със 100 компютъра. А има организации и системи, които трябва да поддържат над 1 000 компютъра. В този случай изискванията са съвсем различни. Поради тази причина, няма универсална рецепта за мигриране или изграждане на дадена информационна инфраструктура на базата на свободен софтуер. Разбира се, съществуват няколко общовалидни правила, които трябва да се спазват.

Ключовата дума при мигриране или вземане на решение за това какъв вид софтуер да се използва е *„планиране“*. То всъщност навсякъде е така и едва ли казвам нещо ново. В докладът на Световната банка, който разгледахме в предната глава, именно думичката планиране е най-често споменаваната. Докладът е задължително четиво за тези, които искат да мигрират или да ползват свободен софтуер. В него се разглеждат няколко успешни проекта за използване и мигриране към свободен софтуер. Също така, се разглеждат и по-важните проекти със свободен софтуер, които придават облика на Линукс днес. Докладът се намира тук [http://www.infodev.org/symp2003/publications/OpenSourceSoftware.pdf].

Друг важен момент при изграждането на системи е използването на стандарти. Темата бе дискутирана в предната глава. Тук отново ще повторим, че използването на стандарти е просто задължително, независимо от това дали се използва свободен или комерсиален софтуер.

Разбира се, преди да вземете каквото и да е решение, разучете дали програмите, които мислите да използвате ги има за Линукс. Очевидно, че ако Ви трябва счетоводна програма, мигрирането няма да е възможно или ще е по-трудно. Но ако Ви трябват програми за достъп до Интернет и офис пакет, мигрирането е напълно възможно. Т.е. вижте какви случаи на използване имате във Вашия бизнес. Някои хора правят обратното. Първо купуват нещо и после се чудят какво да го правят. Т.е. елиминирайте бизнес случая *„Купих си мотор, после обаче се оказа, че съм в Сибир и ми трябва не мотор, ами моторна шейна.“*

Възможно е да мигрирате и частично. Примерно, имате легално закупен Уиндоус, който Ви върши работа, но Ви трябва офис пакет. Пробвайте ОпънОфис орг. Той

върви и под Уиндоус. Никой няма да Ви иска пари, за да го изтеглите и да го пробвате. Ако нямате бърза връзка с Интернет, проверете дали го няма в някое компютърно списание. Често компютърните списания, които се разпространяват в България, включват компактдиск със софтуер, който се разпространява свободно. Също така може да го поръчате за 5 лв. от някой сайт, който се занимава с продажба на свободен софтуер.

В заключение, мигрирането или изграждането на информационна инфраструктура е процес, а не еднократно действие. Ако имате някоя система, която работи на комерсиален софтуер, не означава, че трябва да я мигрирате веднага към свободен софтуер. Използвайте я и когато трябва да я подмените или разширите, обмислете възможността за използване на свободен софтуер.

10.5. Избор на дистрибуция

Ако решите да мигрирате или да използвате Линукс, идва следващия въпрос *„Коя дистрибуция да използвате?“* Уиндоус има само един производител и това улеснява много нещата. Но Линукс има различни дистрибуции. Някои от тях комерсиални, други некомерсиални.

Голяма част от потребителите използват Линукс не заради цената, а заради факта, че е по-стабилен, по-надежден, по-сигурен и може да се настройва за всякакви нужди. В България средната заплата не може да се сравнява със средната заплата в САЩ или Западна Европа и затова цената е много важен фактор. Забележете, че има версии на Линукс, по-скъпи и от Уиндоус.

Ако искате да плащате, аз бих Ви препоръчал да използвате СуЗе (SuSE [http://www.suse.com]). Все пак немците са с доказани традиции в техниката. Немското качество и перфекционзъм са се превърнали в нарицателно. Немците имат същите проблеми с тяхната азбука, каквито имаме ние с кирилицата10. Затова СуЗе обръща голямо внимание на локализацията и в това отношение са перфектни. При използването на комерсиална дистрибуция, трябва да обърнете внимание и на цената. Защото цените на отделните дистрибуции и цените на услугите, който предлагат компаниите, варират. Някои от тях са по-скъпи от Уиндоус дори.

Ако искате да минимизирате Вашите разходи за ИТ, Дебиан е отличен избор. Ако

10 Кирилицата има точно толкова проблеми, колкото всяка една азбука, различна от английската. Немци, французи и др. имат същите, че даже и по-големи, проблеми от нашите. Факт е, че българският е точно толкова лесен за компютърна обработка, колкото английския и много след това се нареждат останалите езици. Както и да е, това не е проблем, но някои фирми го правят на голям проблем с цел да правят бизнес. За повече информация вижте сайта на инжИнера. Добре, че в Линукс по-голямата част от софтуера е свободна и няма пишман специалисти да го кирилизират и него.

използвате Дебиан, не сте зависими от компания. Няма да плащате такси. Имате гарантирано обновяване. Ако имате проблеми, може да наемете някой да Ви ги разреши и да Ви проведе един курс за работа с Дебиан. Със сигурност този вариант ще Ви струва по-малко, отколкото ако използвате комерсиална дистрибуция. Силата на Дебиан се показва, ако имате нужда да инсталирате и управлявате голям брой компютри. Тогава може да възложите инсталацията на една фирма, поддръжката на друга, обучението на трета и пр. Защото ресурсите на Дебиан са достъпни в еднаква степен на всички хора, независимо дали става въпрос за голяма или малка компания, или независим разработчик. Всеки един от тях има равноправен достъп до сайта на Дебиан и до документацията. Това означава повече участници на пазара, по-добра конкуренция и съответно по-ниски цени.

10.6. Примерен план за образованието

След толкова много приказки е добре да дадем един примерен план за въвеждането на свободен софтуер в образованието. Има няколко причини да се избере образованието, пред другите сфери:

- В образованието няма какво да се мигрира. Там компютрите се използват за обучение. Няма ситуации, в които дадена система зависи от друга или да трябва да се конвертират 10 000 изключително важни документа.
- Всички налични програми, които трябват за образованието, ги има като свободен софтуер.
- Линукс има изключително много езици и средства за програмиране, които са идеални средства за обучение по програмиране.
- Линукс има офис пакет, който става за обучение по начална компютърна грамотност на бизнес паралелките.
- Линукс има средства и програми за работа с Интернет - браузъри, клиенти за е-поща и др. Изборът е голям.
- Принципите на текстообработката или електронните таблици са едни и същи. Те съществуват още преди появата на Майкрософт. Всички офис пакети, имат идентичен изглед и работят по идентичен начин. Няма значение дали студентите или учениците изучават Майкрософт офис или някой друг офис пакет. След това могат да приложат наученото с друг офис пакет.

- В много малък процент от случаите се налага използването на други програми.
- Голяма част от компютрите в образованието са много слаби и на тях не може да върви последната версия на Уиндоус. Линукс се справя много добре на по-слаби компютри. Т.е. слабичките компютри вместо да се заменят, могат да влязат отново в употреба.

Преимуществата на Линукс пред Уиндоус са очевидни. Уиндоус е със затворен изходен код и в занятията по програмиране няма как да се покаже на студентите как работи Уиндоус. Линукс е със свободен изходен код. Който има желание може да го разглежда, да си играе с него, да експериментира. Ако трябва да направим сравнение, Уиндоус е като количка с дистанционно управление. Можеш да си играеш с нея, но бързо омръзва, защото не можеш да видиш как работи. За сравнение, Линукс е като конструктор. С него можеш да си направиш кола или камион, може да преместиш двигателя отпред, може да разместиш бутоните на дистанционното управление и да видиш дали ще работи. Може да видиш как работи и да го подобриш. Да се преподава операционни системи с помощта на Уиндоус е все едно да се преподава специалността Двигатели с вътрешно горене и да няма разрез на един двигател. Вместо това да се показва една и кола и да се обяснява на студентите *„Ето там вътре е двигателя, само че не можем да отворим капака и да го разглобим, за да видим как работи.“*

Накрая трябва да споменем, че използването на Дебиан в този случай е задължително. Една хомогенна среда се управлява и поддържа по-лесно, отколкото ако имаме различни дистрибуции. Ако се толерира някоя дистрибуция, става монопол и образованието започва да зависи от дадена компания. Разбира се, мигрирането от една дистрибуция на Линукс към друга не е кой знае колко трудна задача, но все пак е разход. Използването на Дебиан не е монопол, защото Дебиан не е компания. Това е некомерсиална дистрибуция. Инсталирането, поддръжката и обучението ако са наложителни, могат да се поръчат от различни фирми.

Единственото, което трябва да се направи, е списък с необходимите програми, които трябва да се инсталират. Споменахме, че Дебиан има над 10 000 програми. Инсталирането на всички от тях е безумие. Затова трябва да се направи списък с минималните изисквания към системата. Ето нещо примерно:

1. Базови програми, необходими за функциониране на системата. Без тях не може, ама да ги споменем за всеки случай.

2. Българска локализация.

3. Средства за програмиране, които се изучават в българската образователна система: Паскал, C/C++, Джава и пр.

4. Офис пакет: ОпънОфис орг.

5. Средства за работа в Интернет: Mozilla; KMail и др.

6. Още нещо, ако трябва.

При така подготвеното задание всяка образователна институция може да реши дали сама да се справи с проблема или да потърси помощ от фирми. Заданието трябва да е едно, за да бъде хомогенна система и да може да се управлява и поддържа по-лесно.

Библиография

Български статии

Калоян Доганов. 2001. *Теоретични импликации на софтуерната революция [http://revolution.sourceforge.net/]*.

Линукс БГ. 2004. *Какво е Линукс [http://linux-bg.org/cgi-bin/y/index.pl?page=article&id=advices&key=360040421&p=2]*.

Преводи

Калоян Доганов, Никола Колев. 2000. *Превод на „Катедралата и базара“ [http://catb-bg.sourceforge.net/]*.

Владимир Герджиков. 2002. *Частичен превод на „Free/Libre and Open Source Software - Survey and Study (FLOSS)“ [http://floss-bg.sourceforge.net/]*.

Йовко Ламбрев. 2000. *Превод на „Как да стана хакер“ [http://linux.gyuvet.ch/html/paper/hackers.html]*.

Йовко Ламбрев. 2000. *Превод на „История на ядрото на Линукс“*

[http://linux.gyuvet.ch/html/paper/kernel.html].

Андрю Иванов. 2001. *Превод на „GNU операционната система и движението за отворен код“ [http://linux-bg.org/cgi-bin/y/index.pl?page=article&id=history&key=334234041]*.

Андрю Иванов. 2003. *Превод на „Историята на Линукс“ [http://linux-bg.org/cgi-bin/y/index.pl?page=article&id=history&key=352608315]*.

Никола Антонов. 2002. *Превод на „Дебиан: минало, настояще, бъдеще - I част“ [http://linux-bg.org/cgi-bin/y/index.pl?page=article&id=history&key=345321592]*.

Цвятко Йовчев. 2000. *Превод на ДжиПиЕл (GPL) [http://linux.gyuvet.ch/html/docs/gnu_bg.html]*.

Атанас Атанасов. 2001. *Превод на ДжиПиЕл (GPL) [http://bulgaria.sourceforge.net/prava/gplbg.html]*.

Международни статии

Manoj Srivastava. 2004. *Why Linux? Why Debian? [http://people.debian.org/~srivasta/talks/why_debian/talk.html]*.

Christoph Lameter. 2002. *Debian GNU/Linux: The Past, the Present and the Future [http://telemetrybox.org/tokyo/]*.

Правни документи на Фондацията за Свободен софтуер

ДжиПиЕл (GPL) в оргинал [http://www.fsf.org/licenses/gpl.html].

Списък и сравнение на различни видове лицензи [http://www.fsf.org/philosophy/license-list.html].

Философия на свободния софтуер [http://www.fsf.org/philosophy/free-sw.html].

История на движението ГНУ (GNU) [http://www.fsf.org/gnu/gnu-history.html].

Изследвания на международни институции

World Bank. 2004. *Open Source Software - Perspectives for Development [http://www.infodev.org/symp2003/publications/OpenSourceSoftware.pdf]*.

Rishab Aiyer Ghosh, Ruediger Glott. 2002. *Free/Libre and Open Source Software - Survey and Study [http://www.infonomics.nl/FLOSS/report/index.htm]*.

Международни периодични издания

The Register. 2002. *IT project failure is rampant - KPMG [http://www.theregister.co.uk/content/7/28299.html]*.

GNU Free Documentation License

GNU Free Documentation License
Version 1.2, November 2002

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the

software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant

Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose

title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify

you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document,
   unaltered in their text and in their titles. Section numbers
   or the equivalent are not considered part of the section titles.
M. Delete any section Entitled "Endorsements". Such a section
   may not be included in the Modified Version.
N. Do not retitle any existing section to be Entitled "Endorsements"
   or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation;

```
    with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Tex
    A copy of the license is included in the section entitled "GNU
    Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts,
replace the "with...Texts." line with this:

    with the Invariant Sections being LIST THEIR TITLES, with the
    Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other
combination of the three, merge those two alternatives to suit the
situation.

If your document contains nontrivial examples of program code, we
recommend releasing these examples in parallel under your choice of
free software license, such as the GNU General Public License,
to permit their use in free software.
```