Open Source Software

Naomi Hoffman

Table of Contents

Table of Contents

Introduction

There are various ways to distribute software. For example, with proprietary software a fee is charged for the executable program and the source code is typically not available. This is a common method of distributing software. Proprietary software is typically distributed only in binary form (meaning the program is only readable as ones and zeros). Other methods of distributing software include:

ShareWare is usually free for the initial program but a fee is charged for continued use, newer versions and/or upgrades (i.e., WinZip, also see the TuCows website www.tucows.com).

FreeWare is without cost to the user but the source code is not shared. Microsoft at one time gave away its Windows operating system and continues to give Internet Explorer away. However, Microsoft does not give its source code. This was a clever way to expand its market share. This is a commonly used strategy for gaining market share is sometimes referred to as a 'loss leader.'

Open Source Software (OSS) (the source code is available without charge -, i.e., Linux) is free software with the source code.

There is really nothing new about Open Source. In fact, the Internet was the first open source entity even though it wasn't called that at the time. The strength of Open Source software development is its ability to leverage the skills, knowledge, and programming intellect of a large developer community.

I discovered that many, if not most, of the open source companies use OS tools that primarily programmers would be familiar with --- not the end user **(Appendix F).** Compilers, debuggers, and editors are the three primary tools programmers use on a day-to-day basis.[1] In fact, few open source applications can be understood by the end user.

Understanding the Open Source Concept

Open-source software development works like this: a programmer or a company develops a rough version of a software product and releases the source code for the public (typically a community of developers) to use and modify. Source code is the underlying code as it is written in a programming language, before it is compiled into the 1's and 0's that computers need to run it. In other words, source code is the human (programmer) readable language instructing a computer to execute a program. By contrast, in a conventional software distribution, usually called closed or proprietary source, the end-users obtain code that has been compiled into binary form and is not easily decipherable or reusable by software developers.

After the source code is released, developers obtain an open-source license (See 'Licensing' below) from the overseer, which grants the user four rights: to possess a copy of the

code and to compile, modify, and redistribute it. Unlike a traditional software license, in which a customer pays a fee to use (not own) the code, a strict open-source license, as defined by the OpenSource Initiative (discussed later) may not require a royalty or other fee. Usually, changes to the original source code are then sent back to the author for possible incorporation into the official version of the product. Some types of open-source code licenses do not require the return of the enhanced code. The process continues indefinitely, until interest declines or a company decides to close the source and sell the product commercially -- which, to date, has been rare.

History of the Free Software Movement

GNU (GNU's Not Unix) and the Free Software Foundation (FSF) were founded by Richard Stallman, who started the liberal free software movement due to strong personal opposition to intellectual property. The FSF is seen by many as too restrictive and there was no clear way for a business to be successful using this concept.

Richard Stallman firmly believes in the golden rule --- do until others as you would have them do unto you. In the GNU Manifesto Stallman states "I consider the golden rule requires that if I like a program I must share it with other people who like it. Software sellers want to divide the users and conquer them, make each user agree not to share with others. I refuse to break solidarity with other users in this way."[2] Stallman also believes that because helping others is the basis of society, computer users should be free to modify programs to fit their needs and, likewise, to share software.[3]

According to Stallman the rule made by the owners of proprietary software is, "If you share with your neighbor, you are a pirate. If you want any changes, beg us to make them."[4] Stallman provides the following assumptions made by most proprietary software companies: [5]

1. Software companies have an unquestionable natural right to own software and thus have power over all its users.
2. The only important thing about software is what jobs it allows you to do.
3. Developers would have no usable software if we [proprietary software company] did not offer a company power over the users of the program.

Further, Stallman believes the term "free software" is sometimes misunderstood. Free software has nothing to do with price — it is about freedom. Stallman defines a program as 'free' if it meets the following criteria: [6]

1. You have the freedom to run the program, for any purpose.
2. You have the freedom to modify the program to suit your needs. To make this freedom effective in practice, you must have access to the source code, since making changes in a program without having the source code is exceedingly difficult.
3. You have the freedom to redistribute copies, either gratuitous or for a fee.
4. You have the freedom to distribute modified versions of the program, so that the community can benefit from your improvements.

However, companies like Red Hat Software and Cygnus Solutions are examples of successful companies using the Open Source model but sticking with Stallman's philosophy. Even so, Stallman's philosophy is too restrictive for companies supporting libraries and toolkits.

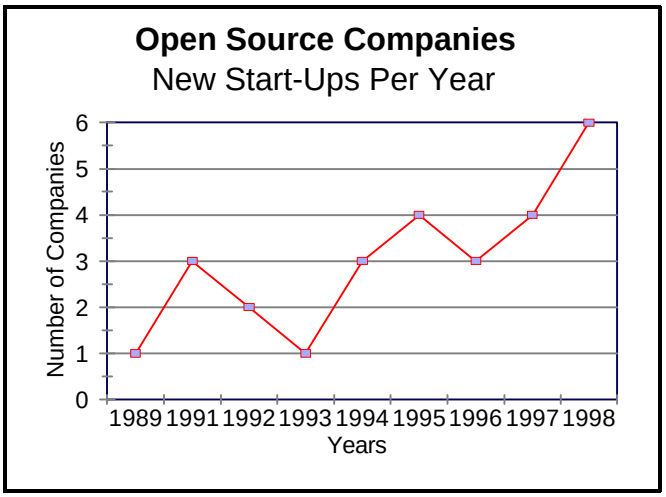
Richard Stallman does not agree on all fronts with the Open Source initiative because he is against proprietary tools being added onto OSS. Although, the free software philosophy rejects a specific widespread business practice, it is not against business. As long as businesses respect the users' freedom, everyone wins.[7]

What Fueled the OS Movement?

Shock waves went rippling through the commercial-software marketplace with the emergence of the open-source Apache Web server program as the market leader in April 1996; Netscape's decision in early 1998 to release the source code to the Navigator Web browser; and the steadily rising use of Linux.[8] Around the same time that Netscape decided to release the source code for its browser a new organization called the OpenSource Initiative (www.opensource.org) was started in an attempt to educate the business community. The term 'open source' was chosen because of the negative perceptions attached to the terms 'freeware' and 'free software.'

Cygnus Solutions was the first to start supporting an open source product --- the Gnu C Compiler (GCC). In fact, Cygnus was around before the term 'open source' was coined. GCC was critical in the development of Linux. Red Hat distributes and provides support for the Linux operating system. Approximately 10 million users in 1998 used the Linux operating system. AT this point Linux's market share is insignificant compared to the Microsoft market share of 230 million DOS/Windows users.[9] However, Linux's new popularity is largely to do with Red Hat's distribution efforts and is expected to continue to grow rapidly.

Today, many companies are entering the open source market as the following graph depicts.



Source: Appendix E-1
Note: Based on a sample of 27 companies.

Licensing

How do you keep others from taking an open source program, selling it, and thereby making it proprietary? An easy answer is to restrict the use of the program with a license. Licensing agreements require users to share modifications to existing code with the rest of the world. Richard Stallman coined the phrase "copyleft" which used the copyright law but reverse its premise. Instead of means of privatizing software, it becomes a means of keeping software free.[10]

The GNU Public License (GPL), used for most GNU software, states that anything added to or combined with a copylefted program must be such that the larger combined version is also free and copylefted (**Appendix A**).

The GNU C library uses a special kind of copyleft called the GNU Library General Public License (LPGL), which gives permission to link proprietary software with the library. This was a strategic move to encourage the use of the GNU system and encourage development of free applications.

All OSS licenses do not follow the GPL and the LPGL format. For example, unlike the GNU license, one cannot redistribute Kitware's VTK with permission from one of Kitware's founders (**Appendix B**). The Open Source Initiative has established a mailing list to review licenses submitted to certification@opensource.org. (For examples of other license agreements see **Appendices C** and **D**.

Benefits of The OSS Model

The open source concept can unify the efforts of programmers around the world and companies that provide commercial services (customization, enhancements, bug fixes, and support). Further, companies can capitalize on the economies of scale and broad appeal of this new kind of software.[11]

A company using OSS could enjoy the following benefits typical with the open source development model:

- *Internet.*

  The Internet has created a new infrastructure off of which open source can leverage. The Internet has lowered the barriers to entry and the costs of distribution.[12] For example, any file transfer protocol (ftp) or web server can now serve as a distribution point that is cheap and effectively instantaneous. The Internet's infrastructure makes the best possible environment for OSS. The infrastructure includes the Domain Name Server (DNS), sendmail, the various open-source TCP/IP stacks and utility suites, and the open-source scripting languages such as Perl.[13] These are the running gears of the

Internet. Without these tools, the Internet would not be as effective as it is today.

According to Vivek Mehra, Cobalt Networks cofounder "The [open source] model has demonstrated that it can evolve more quickly and with higher quality than the proprietary model in the Internet space, but not necessarily in other application spaces."[14]

- *Pool of talented programmers.*

  Today, good programmers are hard to come by. Hiring experienced programmers may be cost prohibitive for many companies. In addition, larger companies have difficulty attracting and keeping the good programmers because of the aversion for corporate bureaucracy. The open source concept provides an unlimited universe of programmers --- the good, the bad, and the evil. In an open source company, at a minimum, you have a healthy mix of talent and perspectives that can, and usually does, lead to better software.

- *Reliable testing*.

  One programmer cannot be expected to account for all possible test conditions. Nor does one programmer necessarily have the test environment to fully test every aspect of a hypothesis. In the Open Source development model, programs are used and tested in a wider variety of contexts than one programmer could generate, and bugs get uncovered that would otherwise not be found. This same principle is expressed as "Given enough eyes, all bugs are shallow." [15]

- *Faster development time.*

  The OSS company can develop and debug new software with the speed and creativity of science.[16] The increased speed with which an outside co-developer can have a needed bug fixed will often translate into a substantial opportunity gain for that customer.

- *Lower overhead.*

  Lower overhead costs are primarily realized because R&D costs are minimized. Most of research and development (R&D) expenses are shouldered by the unlimited universe of programmers. The open-source model allows software shops to (in effect) out-source some of their work, paying for it in values less tangible than money.[17] Of the companies surveyed, I found that few budget for R&D costs.

- *Results reviewed by peers.*

  Computer science has only one means of enabling peers to replicate results: sharing the source code.[18] In industry, peer reviews allows objective and, hopefully, timely feedback. Such feedback allows for efficient focusing on problematic areas.

- *Customer satisfaction*

  Increased customer satisfaction will result by being able to respond quickly and effectively to the customers needs. In addition, smaller shops will be able to handle bigger projects. Thus, providing the customer with more choices.

Challenges, Mis-perceptions And Threats

Challenges

Many of the challenges faced by companies providing open source service are not unlike those faced by companies offering proprietary software. Michael Tiemann, founder of Cygnus Solutions, identified six challenges faced by companies: Sanity, Scalability, Sustainability, Profitability, Manageability, and Investibility. [19]

1. *Sanity*

   Why would a customer pay for a competitor's advantage? Software should not be viewed as a competitive advantage but rather as a commodity. A company's competitive advantage is realized through value added to the software (distribution, support, etc.) as well as benefits (mentioned earlier) derived from the open source model.

2. *Scalability*

   How can a serviced-based business scale? Every company, whether it is selling proprietary or open source software, must consider growth and address how it plans to meet demand. By having a focused business plan, with a sound marketing strategy, a company can better anticipate and meet future demand.

   Eighty percent of the companies reviewed (thirty in total) provide some form of software support **(Appendix E-1)**. This clearly indicates that companies are moving to meet a need (demand).

   For example, the Linux-based Beowolf clusters have become popular supercomputers at several national laboratories, government agencies and universities. But they have been rare in the private sector because Chief Information Officers (CIOs) have only begun to consider Linux as a reliable, supportable operating system.[20] However, the more CIOs consider them as a viable option the more companies will spring up to provide service.

3. *Sustainability*

   Will an OSS company (i.e., Cygnus or Red Hat) be around when customers need support? Commercial support is critical for the success of any product or service. In an age where the technology stocks are shooting up in the market one day and the next day the same company is closing shop, it's important to know a business will be around when needed. With an open source product, if an OSS company were to go out of business then the source code would not disappear along with the company. With closed source software, if another company did not buy the right then the product would become obsolete.

4. *Profitability*

   How can open-source software be profitable? As stated before, companies such as Red Hat Software and Cygnus Solutions have been very successful (and profitable) distributing and supporting open source software, respectively. With proper planning, an OSS company has a promising chance of success.

5. *Manageability*

How can open-source software be managed to deliver quality consistently? Manageability is more of a challenge for small start-up companies. Tiemann went from having undeniably the best compiler in almost every regard to a more complex set of tradeoffs the customer would have to evaluate.[21] In such cases, having a software manual and mailing lists helps to manage training and technical questions.

What will happen if the software fragments? OSS opponents will argue that a benefit of proprietary software is only one "true" version exists, whereas open source software can fragment into many out-of-sync releases, not one of them a legitimate "standard."[22] Linux has proven that this can be overcome. With the Linux kernel and most other OSS products, changes to code are not automatically integrated. Typically, code changes are reviewed and integrated with future releases if deemed essential.

6. *Investibility*

How can a company with no software investment potential ever attract investors? Today, the measure of success is determined by stock value and most of the OSS companies have not made an initial public stock offering (IPO). Typically, the measure of a successful company is not necessarily profits but whether or not it has made an IPO. Since my study began, both Red Hat and Cygnus announced plans to make an IPO.

Unfortunately for the open-source software community, most venture capitalists (VCs) are not convinced that the movement will provide many promising business opportunities. Such quotes as the following emphasizes those educating investors in terms they understand may prove challenging.

> *"Startups - The open-source business model has VCs baffled." said Brian E. Taptich* [23]

Other challenges

- *The premise that a company must differentiate its product from its competitors.*

  Naturally, if the source code is open then everyone knows the family secrets. However, it is the service, not the software, that differentiates a company from its competitors.
- *Staying focused.*

  A company should carefully weigh all the options before selecting a strategy (i.e., marketing and distribution). For example, Cygnus Solutions did a great job engineering its product to the customer's specific needs whereas Cygnus' competitors had finished products that were sold very effectively in their respective market niches. When Cygnus decided to build and sell a shrink-wrapped product, they were changing their attack plan from an elaborate flanking maneuver to a frontal assault against companies that had 10 to 100 times the revenue.[24] This was a challenge Cygnus did not foresee but could of avoided if it remained focused.

Misperceptions

- *Fear that quality will be lost and support not provided.*

  Understandably, a small company with limited resources will not want to purchase a product without some guarantee that it will be around tomorrow. However, more and more open source companies are springing up to address that demand**.**
- *The 'hacker' stigma.*

  Many programmers in the open source arena take great pride in being referred to as a 'hacker'. However, to many non-programmers, the term 'hacker' leaves them with images of someone trying to break into a system. 'Hack' also might cause one to envision patch work with no audit trail or documentation. Why would companies want to expose themselves to unnecessary risk? Therefore, it might be prudent to avoid the term if it causes concern for the customer.
- *Contributors' fixes will be self serving.*

  Most OSS is written by programmers striving to solve a problem unique to their own circumstances. It is far from clear that 'itch-scratching' will lead to user-friendly interfaces that can match the sophistication of software such as Microsoft Windows. However, this should not be viewed as a problem because creativity leads to innovation which most likely will benefit users down the road.
- *The customer being seen as a freeloader.*

  A company may not want their image to be that of benefitting from others hard work. In fact, one company I surveyed stated that a customer insisted upon paying for code that was built using open source software. The same customer went so far as to request the software be scaled back so that they could make their version proprietary.

Threats

- *Patents on algorithms.*

  At this point, it is uncertain how a small company with limited resources will defend itself against patent infringement suits. According to Richard Stallman the worse threat we face comes from software patents, which can put algorithms and features off-limits to free software for up to twenty years. For example, the LZW compression algorithm patents were applied for in 1983, and free software still cannot be released to produce proper compressed graphical interchange formats (GIFs). In 1998, a free program to produce MP3 compressed audio was removed from distribution under threat of a patent suit.[25] Stallman's advice is to not give up hope and use such an opportunity to explore an alternate and, hopefully, better way.

  Proprietary companies selling OSS and/or causing fragmentation.

  Preventing companies like Microsoft from buying an open-source software product, altering it and selling it as proprietary does not seem likely.
- *Proprietary companies stealing the key programmers leading OSS efforts.*

  This does not appear to be a real risk - or at least not for some time. Being part of the open source wave provides motivation to many programmers. In some cases, large companies, like IBM and GE, encourage their programmers to contribute to an open source product. This not only directly benefits in being the first to have access to a new

technology but also encourages creativity.

- *Product liability.*

  It is uncertain how a small company with limited resources will defend itself against liability suits. However, an unlimited pool of developers can provide an impressive defense even if each contribution is small in comparison.

- *Key programmers will become less motivated and/or withdraw from contributing to OS initiative.*

  What if Linus Torvalds wearies of overseeing the Linux development process? Who will make the ultimate decision on what software patch is incorporated into the all-important Linux kernel? And how exactly will the open-source "community" enforce the "give back what you create" open-source licensing terms? Change the leaders, and usually you lose the community, leaving nothing but unprotected source code.

  However, efforts like the sourceXchange will minimize such a risk. The goal of sourceXchange is to provide a forum:

  - where developers and sponsors can connect,
  - where developers can be rewarded for their work, and
  - sponsors can meet specific needs with OSS.

  Through sourceXchange, creating code will create value for sponsors, for developers, and for all members of an expanding Open Source community.[26]

  A company distributing, supporting and/or servicing an OS product can also attractive programmers looking to continue their contribution regardless of where they work. Further, it would provide some comfort to a company that the programmer will continue to contribute to the software project after he/she leaves the company. The following items motivate a programmer to contribute to the OS effort: [27]

  ✓ *Intellectual exercise.* A very complex project like Apache or the Linux kernel brings the satisfaction of the ultimate in intellectual exercise.

  ✓ *Intellectual self interest.* Sharing of ideas is done through sharing code. Coding is what programmers love to do and it is how they define their intellect. Without coding, a programmer feels less of a person, much like an athlete deprived of an opportunity to compete. Further, many programmers do not like maintaining a piece of code after having mastered it. However, great pride is achieved when one is on the cutting edge of new technology or the first to come up with the next standard.

  ✓ *Maintaining the Open Source philosophy.* The belief that profits should not come at the expense of the pursuit of knowledge.

  ✓ *Empowerment* is what Erich Raymond calls "scratching an itch." Most open source projects began with frustration: looking for a tool to do a job and finding none, or finding one that was broken or poorly maintained. Now a programmer has a means for finding the best or most effective tools.

  *** Note that monetary rewards was not mentioned in the list but is nonetheless a form of motivation. It was not listed above because history has shown that scientific success outlives financial success.[28]

The Business Model

Even though Nikki Goth Itoi, Senior Editor for the Red Herring, does not believe OS will be the force to drive the next wave of startups to success he does believe that " . . . Open-source development has found an important place in the commercial software arena. But it won't radically alter the competitive landscape . . . " [29]

Many companies are profiting from OSS. For example, global oil giant Amerada Hess saved millions of dollars by replacing a costly IBM supercomputer with a high-end parallel cluster (Beowolf system) running Linux. At $130,000, the Linux-based Beowolf system performs the task in about the same time (two weeks) as the 32-node IBM SP2 system running AIX that the company paid $2 million to lease for three years, according to Vic Forsyth, Amerada Hess Houston-based manager of geophysical systems.[30]

There are at least four known business models for making money with open source software: [31]

1. *Support Sellers*

   In this model, you give away the software product, but sell distribution, branding, and after-sale service. This is what Red Hat and Cygnus are currently doing.

2. *Loss Leader*

   In this model, you give away open-source as a loss-leader and, temporarily, market position for potential future sales for proprietary software. This is what Netscape is currently doing.

3. *Widget Frosting*

   In this model, a hardware company (for which software is a necessary addition but strictly a cost rather than profit center) goes open-source in order to get better drivers and interface tools cheaper. Silicon Graphics, for example, supports and ships Samba because it makes hardware more attractive.

4. *Accessorizing*

   Selling accessories -- books, compatible hardware, complete systems with open-source software pre-installed. It is easy to trivialize this (open-source T-shirts, coffee mugs, Linux penguin dolls) but at least the books and hardware provide some clear successes: O'Reilly Associates, SSC, and VA Research are among them.

The following pie chart shows the marketing strategies used for twelve OSS companies surveyed.

Source: Appendix E-2

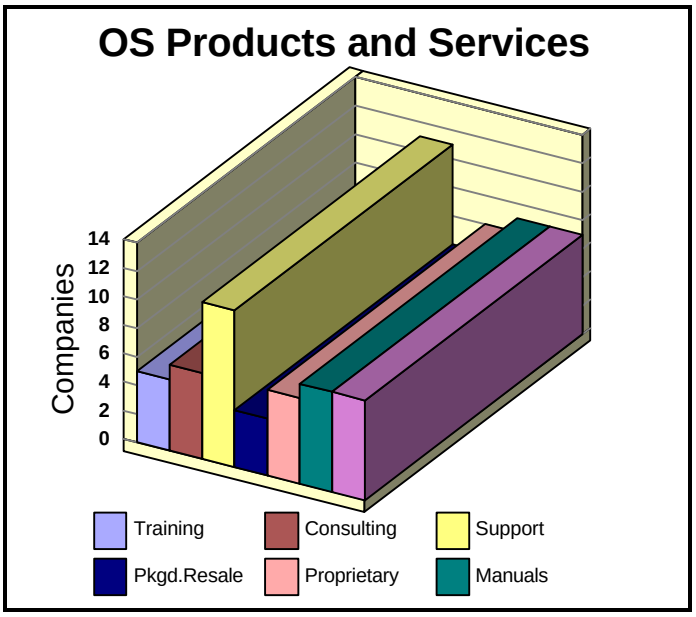
OSS Tools and Applications

It appears OSS is used primarily to implement frameworks and platforms (TCP/IP and the kernel) not the end-user applications (commercial offerings) (**Appendix F)**.

Products and Services

When deciding upon what products and services to offer a company should first consider a demand analysis (forecasting market and sales potential, and estimating the competitive reaction), cost analysis (estimating R&D and marketing costs), and profitability analysis (considers both demand and cost analysis).[32]

As the following chart depicts, support is the most common service provide by an OS company.



Source: Appendix E-1.
Note: Based on a sample of 14 companies.

A breakdown by products and/or services typically offered by an OSS company is as follows **(Appendix E-1)**:

Training

Five (42%) out of 14 companies surveyed via questionnaire offer training.

Consulting Services

Eighteen (60%) out of thirty companies surveyed provided consulting services.

Support Services

Twenty-four (80%) out of thirty companies surveyed provided software support. This is realistic because many companies make a larger percentage of their income from support services, not from selling the software. Mailing lists are an effective forum for users to ask questions but not always the most effective means of providing support. For example, Sun Microsystems makes the majority of its money from supporting and servicing their operating system and hardware. It is estimated that Sun enjoys fully 50% of their profits from support, training and consulting services.[33]

Packaged resale of OSS products

Four (28.6%) out of 14 companies surveyed via a questionnaire offer packaged resale of an OSS product. For example, Red Hat charges to distribute Linux.

Proprietary packages built on OSS

Six (43%) out of 14 companies surveyed via a questionnaire offer proprietary software built on top of an OSS product. Demand exists for these types of tools because many companies would rather pay a little more for the value added from a usable interface

than to pay a salary for someone to learn how to do the required task.

For example, Sendmail uses the two-tiered approach. The Sendmail Pro is proprietary. Sendmail Pro and one year ahead in the development cycle of the free (open source) sendmail version.[34]

Literature (i.e., Manuals)

Seven (50%) out of 14 companies surveyed via questionnaire offer some form of literature (typically a manual) for an OS product **(Appendix ____).** Literature such as a manual allows potential customers who are not technically familiar with the software a means to learn and a reference for trouble shooting. For example, Red Hat and Kitware provide manuals for Linux and VTK, respectively.

Marketing

> *"Historically, open-source communities have been led by individuals with great technical acumen but low commercialization skills, or no ambition to commercialize at all," says David Cowan of Bessemer Venture Partners.*

As with proprietary software companies, marketing is critical for a company's success in the competitive arena. Is Microsoft's window's a superior product? Absolutely not. So then why does Microsoft have a large share of the operating system market? Because Microsoft has an aggressive and very effective advertising and marketing campaign. In fact, the Unix proprietary players knew so little about marketing that Microsoft snatched away their market.

The following marketing points should be considered by a start-up company:

1. *Timing - when to enter the market.*

The company first to market with a large installed base and oversees the OS process will be better positioned to be a market leader. The primary developer has the advantage of being first to market. The only companies that stand a reasonable chance of success are those that are first to market. They begin with a large installed base, and oversee the open-source process on which their business depends. It also helps to have the software's creator on board.

> *"Open Source is like fighting a Moebius strip, and everything flows to the side of the primary maintainer. So, while our competitors may get some tactical advantage in the "me-too" GNU space, Cygnus benefits in the long run. Founded in 1989, our first-mover advantage is ten years ahead of the competition."* [35]

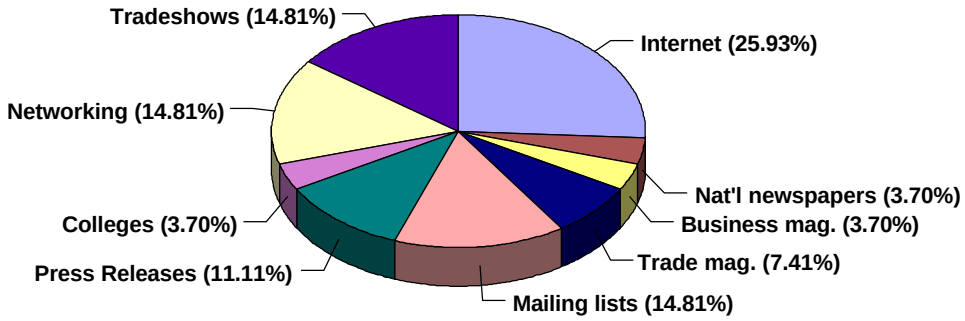
2. *Advertising.*

Advertising is necessary to let the customer know that a company has a product/service exists that will satisfy the customer's want or need. Typically, advertising is any paid form of nonpersonal communication through the mass media about a good or service. However, there are forms of advertising that requires little or no capital investment. The following forms of advertising should be considered by an OS company:

- ✓ Internet (i.e., search engines, links from other companies webpages).
- ✓ Company website
- ✓ Trade shows

- ✓ Radio
- ✓ Television
- ✓ Local Newspaper(s)
- ✓ National Newspapers (i.e., Wall Street Journal)
- ✓ Business magazines (i.e., Business Week)
- ✓ Technical Magazines (i.e., Wired, Dr. Dobbs)
- ✓ Mailing lists (including electronic mail)
- ✓ Press Releases
- ✓ Colleges/Universities
- ✓ Networking (Word of mouth and seminars)

Advertising Strategies



Source: Appendix E-2

I found the Internet, trade shows, technical magazines are the most commonly used means of advertising by open source companies (**Appendix E-2)**. Those methods are also the least costly but not necessarily the most effective in reaching the target market. I would recommend targeting business (i.e., Wall Street Journal, Forbes) and technical (Dr. Dobbs and Wired) magazines.

3. *Brand management*.

A brand is a name, term design, symbol, or any other feature that identifies one seller's good or service as distinct from those of other sellers. Red Hat's strength comes from brand management: consistent marketing and community outreach that makes the community recommend them with their friends ask them which distribution to use.[36]

4. *Sales.*

Selling should not focus only on the technical merits of the software. Instead of focusing on the software's technical merits, Cygnus' sales person explained the intrinsic

complexity of the job they were doing and the business value of the software. This helped in showing the customers why they should buy from them instead of trying to do the work in-house.[37]

Electronic commerce (e-commerce) is an effective way of processing transactions on-line. In fact, 7 (50%) out of the 14 companies surveyed use e-commerce **(Appendix E-1)**. These same companies tend to be the larger companies in terms of employees and age of the company.

5. *Distribution.*

Distribution is the process of getting the product to the customer. One good way to deliver free software is to package it as a full featured program with a nice manual.[38] In fact, Red Hat Linux has been very successful distributing Linux. A critical factor in the success of the X Window System was that the X developers were willing to give the sources away for free in accordance with the hacker ethic, and were able to distribute them over the Internet.[39]

5. *Exit strategy.*

The exit strategy is a companies plan to leaving a market or changing strategy. The exit strategy should include whether the company plans to sell or make an IPO after it reaches its goals (i.e., profits).

What Are the 'Other' Companies Doing?

Even if most corporate managers do not recognize the importance of OSS, Bill Gates certainly does. At the 1998 annual Microsoft meeting with financial analysts Bill Gates stated that, in the Web server market, "Apache is our biggest competitor. It's gaining market share faster than Netscape." Since January of 1997, the UNIX-only Apache server has consistently held more than 40% of the web server market.[40] Though Microsoft's Internet Information Server product currently dominates the growing Windows-NT web server market, Apache will be released for the NT platform by the end of this year.[41] Linux also poses a threat to Microsoft's operating system market **(Appendix I).**

Firms such as Cobalt Networks and Cygnus Solutions, as well as more seasoned companies like IBM, Silicon Graphics, HP, and Compaq are embracing open source for specific needs. Some companies like Oracle and Informix are simply jumping on the bandwagon by porting versions of their existing products to run on Linux but keeping their code close to their vests. IBM is porting its popular DB2 database to the Linux operating system and porting Apache to the AS400 platform.

Others, like IBM and SGI, devote full-time engineers to enhancing Apache and Samba, respectively, and make any resulting new source code freely available. However, the degree of commitment varies. For example, while companies like Dell merely provides more customer choice by offering Linux pre-installed on a new servers or PCs, Netscape has actually taken its own star application and made its source code available for free.[42] In addition, Netscape Communications Corporation has opened the source to its Netscape Communicator and employs

the developers. Sun Microsystems provided OSS support with donations of hardware and resources to help the SPARC port to Linux or through supporting Tcl (Jon Ousterhout's scripting language).[43]

Summary

It is best to think of source code as a commodity NOT a competitive advantage. When closely examined there is nothing strange or magical about open-source development from a business point of view. It should neither be shunned as impractical nor embraced as a panacea. There is no one single model one must follow, and it is not an "all or nothing" proposition. Open source is simply a new way of developing, distributing, and licensing software. For companies which understand the economic, cultural, and political factors that go into implementing an effective open-source strategy, the open-source model offers the promise of helping a business better survive and thrive in an increasingly demanding environment.

I recommend the following process for any company considering the open source model:

1. Identify your competitive advantage. Typically, the competitive advantage is the expertise of its people but could also be an effective means of distributing a product (i.e., Red Hat).
2. Assess your resources (the available people and tools).
3. Identify your products and services. In addition, identify a pricing strategy that is in line with the competition and develop a licensing agreement (refer the Open Source guidelines) if you are maintaining software or distributing add-on tools.
4. Identify your target/niche market and the market share needed to survive.
5. Identify your competitive environment. This includes identifying competitors and barriers (i.e., cost and market share) to entry and exit.
6. Develop a plan to educate your market on the benefits of using an open source product or service.
7. Develop a marketing campaign.
   a. Advertising (Internet, business and technical magazines)
   b. Sales (e-commerce)
   c. Distribution (if feasible offer through your website)
8. After completing steps 1 - 7 you can now develop a ***business plan*** that should include the following:
   a. Vision and mission statement
   b. Five/ten year plan (this includes the Exit strategy).
   c. Target market
   d. Competitive environment (competitors, their market share and strategies).
   d. Products and services (the value your company will add).
   e. Marketing plan.
      (1) Advertising campaign.
      (2) Sales campaign.
      (3) Distribution (website or middle-man)

In the future I envision immediate growth with toolkits, libraries and developer tools. Beyond that, I believe end-user application growth will be driven by the hardware vendors.

Andrew Leonard best summed it up when he said "As the high tech competitive landscape becomes ever more brutal - as product cycles speed up, as consumers become more discerning, and international pressure intensified - the tactic of going open source may best be envisioned as a necessity, not a business plan." (27)

Bibliography

1. DiBona, C., Ockman, S., and Stone, M. , 'OpenSources - Voices from the Open Source Revolution' O'Reilly ©1999 p. 73

2. Ibid p. 72 & http://www.fsf.org/gnu/manifesto.html.

3. Ibid p. 54

4. Ibid p. 54

5. Ibid p. 54

6. Ibid p. 58

7. Ibid p. 61

8. Leonard, Andrew - "Open Season - Why an industry of cutthroat competition is suddenly deciding good karma is great business", Wired May 1999, pp. 183

9. DiBona, C., Ockman, S., and Stone, M. p.115

10. Ibid p. 59

11. Ibid p. 72

12. Ibid p. 15

13. http://www.opensource.org/for-suits.html

14. Somogyi, Stephan - "Free Enterprise – An open-source success story.–", Wired Magazine, May 1999 pp.183

15. DiBona, C., Ockman, S., and Stone, M. p. 7

16. Ibid p. 17

17. http://www.opensource.org/for-suits.html

18. DiBona, C., Ockman, S., and Stone, M. p. 7

19. Ibid p. 84

Bibliography (continued)

20. http://www.computerworld.com/home/print.nsf/all/990503linux

21. DiBona, C., Ockman, S., and Stone, M. p. 79

22. Ibid p. 79

23. The Red Herring magazine, February 1999 issue.

24. DiBona, C., Ockman, S., and Stone, M. p. 80

25. Ibid p. 68

26. http://www.sourcexchange.com/info.html

27. DiBona, C., Ockman, S., and Stone, M. p. 13

28. Ibid p. 17

29. The Red Herring magazine From the February 1999 issue.

30. http://www.computerworld.com/home/print.nsf/all/990503linux

31. http://www.opensource.org/for-suits.html

32. Schoell, William F, and Guiltinan, Joseph P. 'Marketing - Contemporary Concepts and Practices 5th edition," Allyn and Bacon © 1992 pp.319.

33. DiBona, C., Ockman, S., and Stone, M. p. 9

34. Ibid p. 6

35. Ibid p. 84

36. Ibid p. 5

37. Ibid p. 83

38. Ibid p. 5

Bibliography (continued)

39. Ibid p. 25

40. http://www.netcraft.com/survey

41. http://www.netaction.org/articles/freesoft.html

42. Somogyi, Stephan " Free Enterprise – An open-source success story", Wired Magazine May 1999 pp146

43. DiBona, C., Ockman, S., and Stone, M. p. 9

METHODOLOGY

I started out by contacting all companies using open source software as a means to make money. I'm in the process of contacting companies to determine how successful they have been using the various strategies. I sent fourteen companies a questionnaire (Appendix H). Of the fourteen, six (43%) companies responded with varying levels of detail. I also met with business associates from OS companies and read various technical and business magazine articles.

Appendix A

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

675 Mass Ave, Cambridge, MA 02139, USA

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect

making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

Appendix A

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it.

For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program 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.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in

other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public 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.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS
AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY

OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE
TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A
FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix B

VTK License

Appendix C

TargetJr License

Appendix H

Survey Questionnaire

Company: _____

Name:_____

Title: _____

Length of Time employed by the company: _____

Date: _____

Administrative:

Does the company have a vision and mission statement? If so, please provide.
When did the company start using or supporting OSS?
How many people work for the company (# of technical and clerical staff)? If possible, provide an organization chart.

Products and Services:

Please identify the following products and services provided by the company.

_____ Training
_____ Consulting
_____ Support Services
_____ Packaged resale of OSS products
_____ Proprietary packages built on OSS
_____ Literature (i.e., manuals)
_____ Other - please explain
_____

What is the percent of net income and sales attributed to the following products and service:

_____ Training
_____ Consulting
_____ Support Services
_____ Packaged resale of OSS products
_____ Proprietary packages built on OSS
_____ Literature (i.e., manuals)
_____ Venture capital
_____ Government Contracts
_____ Other - please explain
_____

How much (percentage of NI or sales) is being spent on R&D?

Does the company use e-commerce? Yes ___ No ___

What is the company's product and service pricing strategy? Please provide the product/service fee schedule.

Advertising/Marketing:

Who is the company's target market? (Domestic and International)

Who are the company's competitors? (Domestic and International)

Which of the following marketing strategies does the company use (check all that apply):

_____ Support Selling (maintenance of software or add-on applications)
_____ Widget frosting (a hardware company uses OSS to display product features)
_____ Accessorizing (coffee mugs)
_____ Loss Leader (give away OSS as a loss-leader to gain market position)
_____ Other - please explain _____

Please identify the advertising strategies used by the company (check all that apply:

_____ Internet (i.e., search engines)
_____ Trade shows
_____ Radio
_____ Local Newspaper(s)
_____ National Newspapers (i.e., Wall Street Journal)
_____ Business magazines (i.e., Business Week)
_____ Technical Magazines (i.e., Wired)
_____ Mailing lists (including electronic mail)
_____ Press Releases
_____ Colleges/Universities
_____ Networking (Word of mouth and seminars)
_____ Other - please explain
_____

Other:
Please comment here with information you believe to be pertinent to this topic:
_____
_____
_____

The survey can be e-mailed (nhoffman@nycap.rr.com), faxed (518-371-3971) or mailed (3 Springwood Court Clifton Park, NY 12065). If you have any questions, please email or call (518-383-3565 in the evenings or 518-485-7756 during the day). Thank you.

Appendix I

The Microsoft Halloween Documents
Selected Quotes

“OSS poses a direct, short-term revenue and platform threat to Microsoft, particularly in server space. Additionally, the intrinsic parallelism and free idea exchange in OSS has benefits that are not replicable with our current licensing model and therefore present a long term developer mind share threat.”(Http://www.opensource.org/halloween.html)

“Recent case studies (the Internet) provide very dramatic evidence ... that commercial quality can be achieved / exceeded by OSS projects.”
(Http://www.opensource.org/halloween.html)

“Linux and other OSS advocates are making a progressively more credible argument that OSS software is at least as robust -- if not more -- than commercial alternatives. The Internet provides an ideal, high-visibility showcase for the OSS world.”
(Http://www.opensource.org/halloween.html)

“The ability of the OSS process to collect and harness the collective IQ of thousands of individuals across the Internet is simply amazing. More importantly, OSS evangelization scales with the size of the Internet much faster than our own evangelization efforts appear to scale.” (Http://www.opensource.org/halloween.html)

“Linux represents a best-of-breed UNIX, that is trusted in mission critical applications, and - due to it's open source code - has a long term credibility which exceeds many other competitive OS's.” (Http://www.opensource.org/halloween2.html)

“Linux's (real and perceived) virtues over Windows NT include: Customization ... Availability/Reliability ... Scaleability/Performance ... Interoperability ...
(Http://www.opensource.org/halloween2.html)

“Using today's server requirements, Linux is a credible alternative to commercial developed servers in many, high volume applications.”
(Http://www.opensource.org/halloween2.html)

“The effect of patents and copyright in combatting Linux remains to be investigated.”
(Http://www.opensource.org/halloween2.html)