

Open Source Software

Topics covered

What is open source software?

Licensing terms

The Cathedral and the Bazaar

Bezroukov rebuttal

Halloween Document

Recent developments

Topics covered

- **Licensing**
- **Theory of open source development**
- **Linux**
- **PHP**
- **Apache**
- **MySQL**
- **Perl**

What is open source software?

- **Software in which you can read the source code**
- **Advantages:**
 - * **Understanding**
 - * **Debugging**
 - * **Modification**
 - * **Porting**
 - * **Often free (or at least cheaper)**
- **Advantages are limited by:**
 - * **quality of code**
 - * **quantity of code**

opensource.org Definition

- **Free redistribution**
- **Source code**
- **Derived works**
- **Integrity of author's source code**
- **No discrimination against persons or groups**
- **No discrimination against fields of endeavor**
- **Distribution of license**
- **License must no be specific to a product**
- **The license must not restrict other software**
- **The license must be technology-neutral**

Major open source players

- **Richard Stallman**
 - * **Founder of *Free Software Foundation (GNU project)***
 - * **Zealous advocate of using free software**
- **Linus Torvalds**
 - * **Head of Linux project**
 - * **Started with MINIX project**
 - * **Superseded the GNU Unix effort (Hurd)**
- **Eric Raymond**
 - * **Open source evangelist**
- **Larry Wall**
 - * **Creator of Perl**

Licensing terms

- **Considerations which determine the licensing status:**
 - * **Is it *gratis*?**
 - * **Is the source available?**
 - * **Can you redistribute?**
 - * **Can you distribute modifications?**
 - * **Can you restrict your distributions?**
- ***Public domain* means you can do anything you want. It may or may not be open source.**
- ***Open source* means the source is available -- it may or may not be free or restricted.**

- ***Fearer*** just means you don't have to pay for it. It may or may not be open source or redistributable.
- ***Copyleft*** means you can use, modify and redistribute but you can't restrict your modifications or redistributions.
 - * **GNU General Public License** is the most prevalent example.
 - Linux** uses this license.
 - MySQL** and **Perl** may be used with this license.
 - * **Can't build a commercial product on top of it.**
 - * **A simpler variant called the *Artistic License*** may also be used by Perl users.

- ***MIT style licenses*** retain a copyright while allow you to use, modify and redistribute. You *may* restrict your modifications.
 - * **Apache, PHP, X11, BSD Unix**
 - * **Also called permissive, non-copylefted licenses**
- ***Sun Community Source Licensing*** is a restrictive license that allows you to view source code, but not to redistribute.
 - * **Used by Java**
 - * **Apache Software Foundation negotiated relaxations for open source work** http://www.marketwire.com/mw/release_html_b1?release_id=39838
- ***Semi-free licenses*** allow individuals and/or non-profits to use for free.
 - * **Lightweight version of PGP (Pretty Good Privacy)**

Comparison table of licenses

License / Description	Gratis?	Source available?	Redistributable?	Can restrict redistributions ?
Sun Community Source Licensing	Yes	Yes	No	
Open		Yes		
Closed		No		
Apache / PHP / X11 / "MIT License"	Yes	Yes	Yes	Yes
Gnu General Public Library	Yes	Yes	Yes	No
Public Domain	Yes		Yes	Yes
Freeware	Yes			

- **List of open source style licenses can be found at <http://open-source.org/licenses/>.**

Cathedral and the Bazaar (CatB)

- **Seminal paper (manifesto) of open source software development** http://www.firstmonday.dk/issues/issue3_3/raymond/
- **Written by Eric Raymond in 1997**
- **Intentionally emulated Linux process for fetchmail internet mail utility**
- **Key analogy of paper**
 - * **Traditional commercial software development is controlled by a few select people in a closed, controlled secretive fashion -- the "Cathedral"**
 - * **Open source is "a great babbling bazaar of differing agendas and approaches" -- a free market of ideas.**

Key lessons of Linux and fetchmail projects

- “Every good work of software starts by scratching a developer’s personal itch”
- “Good programmers know what to write. Great ones know what to rewrite (and reuse).”
- “Plan to throw one way; you will, anyhow.”
 - * Fred Brooks, *The Mythical Man-Month*
- “Release early, release often. And listen to your customers”

- “Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone.”
 - * “Given enough eyeballs, all bugs are shallow.”
 - * Raymond calls this ‘Linus’s Law’
 - * Apparently contradicts Brook’s Law: *Adding manpower to a late software project makes it later.*
- “If you treat your beta-testers as if they’re your most valuable resources, they will respond by becoming your most valuable resources.”
- “The next best thing to having good ideas is recognizing good ideas from your users. Sometimes the latter is better.”
 - * *Original thought is highly overrated. -gcw*

- **“Provided the development coordinator has a medium at least as good as the Internet, and knows how to lead without coercion, many heads are inevitably better than one.”**
 - * **Linux started around 1991**
 - * **Internet usage started to become widespread in the early 90’s**

Preconditions for successful open source

- **Initiator must provide a starting point**
 - * **“plausible promise”**
 - * **basic design of Linux was already in place (UNIX)**
 - * **charismatic, effective communicator**

Hacker reward system

- **Why contribute to open source?**
- **Recognition is often a more powerful incentive than money**
 - * **ego satisfaction**
 - * **reputation among community of developers**

Other design concepts in CatB

- * Not specific to open source
- “Often, the most striking and innovative solutions come from realizing that your concept of the problem was wrong.”
- “Perfection (in design) is achieved not when there is nothing more to add, but rather when there is nothing more to take away.”
 - * *Programmers should be paid not by how much they write but by how much they delete. -gcw*
- “Smart data structures and dumb code works a lot better than the other way around.”
 - * Open source culture is “C” oriented
 - * Old timer bias (haven’t groked objects yet)

Criticism / Limitations of CatB

- **Nikolai Bezroukov** http://www.firstmonday.dk/issues/issue4_12/bezroukov/index.html
- **Brook’s Law remains valid**
 - * **Parallelism of Linux effort only applies because design complete**
reference implementation exists
 - * **net number of bugs isn’t going down**
 - * **architectural problems can’t be solved in distributed fashion**
 - * **waste of effort to have multiple developers attacking same bugs**

- **Successful open source projects are not a free-market bazaar**
 - * **Torvalds maintains control of key Linux components delegated to a select few in some cases**
 - * **other projects (e.g. Apache) managed by a self-selected meritocracy**
 - * **truly democratic projects lead to code forking (splitting)**
- **Linux model not all that different from Microsoft's**
 - * **Release early, release often**
 - * **Importance of having users**
 - * **Microsoft's success was built upon 'plausible promise'**
also low entry cost -gcw
 - * **Importance of reuse and rewriting**
 - * **Early entry into market**

- **Open source culture is not new**
 - * **Continuation of academic culture**
 - status seeking**
 - collaboration**
 - peer review**
 - * **GNU / Free Software Foundation arose from MIT culture**
 - * **Linux connection to University of Helsinki critical during incubation period**

- **Source code not sufficient**
 - * **RTFS doesn't reveal architecture**
 - * **Key design structure remain proprietary**
 - Knowledge is power (status)**
 - Reticence as survival tactic -gcw*

- **Open source reward structure deficiencies**
 - Politics**
 - * **hierarchical structure**
 - * **favoritism**
 - * **poisoning of peer review process**
 - * **overload and burnout**
 - * **role of the press**
 - * **immaturity among developers**
 - 16-25 year olds**
 - 'all night hacking' hero ethics**

Limitations

- **Success limited to those things hackers care about**
 - * **Operating systems, languages, networking**
 - * **Office suites and the like not highly successful**
- **Need clear target (i.e. HTTP protocol, UNIX, awk/sed)**
 - * **May not lead to innovation**
- **Support for wrong reasons**
 - * **Not Microsoft**
 - * **Not entirely altruistic**
 - Red Hat, IBM supporting Linux**
 - O'Reilly supporting Perl et. al.**

“Halloween” Document (HD)

- **Internal Microsoft memo leaked to open source community**
 - * **Written by Vinod Valloppillil**
- **Review of open source principles (CatB, et. al.)**

<http://www.scripting.com/misc/halloweenMemo.html>

HD - Open Source Strengths

- * **Grow with the Internet**
- * **Winner take all**
- * **Developers seek to contribute to largest OSS platform**
- * **Larger projects solve more “problems at hand”**
- **Long term credibility**
 - * **OSS is long term credible**
 - * **Lack of code-forking enhances long term credibility**
 - GPL licensing discourages forking**
 - Avoids evolutionary dead end**
 - Prevents FUD (fear-uncertainty-doubt) attack**
- **Parallel Debugging and development**

- **OSS = ‘perfect’ API evangelization / documentation**
 - * **open source provides underlying code -- attractive to “enthusiast developers”**
 - * **closed source is based on trust and support of vendor -- attractive to novice/intermediate developers**
- **Release rate**

HD - Open Source Weaknesses

- **Management costs**
 - * **Starting OSS project difficult**
 - Large future “noosphere”**
 - Scratch big itch**
- **Right amount of problems first**
 - * **Post parity development**
 - * **Unsexy work**
 - * **Integration / Architecture work**

- **Process issues**
 - * **Iterative cost**
 - * **Non-expert feedback**
 - Ease of use, UI intuitiveness must be built in from ground up**
- **Organizational credibility**
 - * **Support model**
 - Tends to limit open source to more tech savvy users**
 - * **Strategic futures**

HD - Open Source Business Models

- **Secondary services**
 - * e.g. IBM Apache / Linux
- **Loss leader - market entry**
 - * Linux as “not Microsoft”
 - * Perl -- book sales
- **Commoditizing downstream suppliers**
 - * Hardware vendor support for linux
- **First mover -- Build now, \$\$ Later**

HD - Product specific analysis

- **Linux**
 - * **Credible OS and development process threat**
deployed in mission critical applications
Best of breed UNIX
Gaining market share
Iterates very fast
 - * **short/medium-term threat in servers**
 - * **unlikely to be threat in desktops**
 - * **Beating linux**
Beat UNIX
Extended functionality in protocols / services

- **Netscape**
- **Apache**
 - * **Organization ~19 core administrators**
 - * **Strengths**
 - Market share**
 - limits Microsoft's ability to extend HTTP protocol**
 - Encourages spread of UNIX / Linux**
 - 3rd party support**
 - * **Weaknesses**
 - Performance**
 - HTTP Complexity & Application services**
- **Perl, BIND, sendmail, majordomo, et. al.**

HD - Microsoft response

- **Vulnerability - server**
 - * **Clients task switch**
 - * **Servers more task specific**
 - * **Commodity servers less commitment**
 - * **Servers professionally managed**
- **Capturing OSS - Developer Mindshare**
 - * **Parallel debugging via code licensing**
 - * **Low cost entry tools**
 - * **Distribute source code**
 - * **More extensibility and develop community/noosphere**
 - * **Monitor OSS newsgroups**

- **Capturing OSS - Microsoft Internal processes - limitations**
 - * **Different development modes, tools, and source code managers**
 - * **No central repository / code access**
 - * **Closed developer communication**
 - * **More component robustness**
 - * **Integration**
 - * **Iterative costs and dependencies**
- **Improve service infrastructure (MSDN et. al.)**

- **Blunting attacks**
 - * **De-commoditize protocols & applications**
 - Integrate DNS with Directory**
 - HTTP-DAV (Distributed Authoring and Versioning)**
 - * **Structured storage**
 - * **MSMQ**
 - * **System management**
 - * **Ease of use**
 - * **Client integration**
 - * **Release / service pack process**
 - * **Long-term commitments**

Netscape / Mozilla

- **Referenced in CatB, Bezroukov, and HD**
- **Went open source as Netscape lost market share to Internet Explorer (IE)**
- **Raymond considered Mozilla critical test, Bezroukov considered it insignificant**
- **Valloppillil noted**
 - * **Capitalized on anti-Microsoft sentiment**
 - * **Added new credibility**
 - * **Scratched big itch**
 - * **Post-parity**
 - * **Small noosphere**

- **IE maintained ~90 - 95% market share from 98 to present**
- **Microsoft recently announced IE 6 last stand-alone browser**
 - * **boost to Netscape-Mozilla in Mac / Linux / UNIX communities**
 - * **May prod website vendors to move back towards W3C standard protocols**
- **Opera (commercial - adware product) also has established niche**

Recent developments

- **Microsoft now sharing source code** <http://www.microsoft.com/sharedsource>
 - * Licenses vary among recipients
 - * Only researchers may modify

- **Steve Ballmer memo targets Linux** <http://zdnet.com.com/2100-1104-1013124.html>
 - * “noncommercial software such and Linux and OpenOffice is seen as an interesting, ‘good enough’ or ‘free alternative’
 - * “Noncommercial software products in general, and Linux in particular, present a competitive challenge for us and for our entire industry, and they require our concentrated focus and attention”
 - * New concepts and integration key to Microsoft success
 - * Licensing plan was a blunder, security still a concern
 - * Adapting open source communication practices - e.g. online discussion of new system features and publishing regular builds