1.1 Introduction

Free and open--source software (F/OSS, FOSS) or Free/Libre/Open--Source Software (FLOSS) is software that is both free and open source. It is liberally licensed to grant users the right to use, copy, study, change, and improve its design through the availability of its source code.

In the context of free and open--source software, free refers to the freedom to copy and re--use the software, rather than to the price of the software. The Free Software Foundation, an organization that advocates the free software model, suggests that, to understand the concept, one should "think of free as in free speech, not as in free beer".

Free Software:

- means free as in “free speech” (vs. as in “free beer”)
- freedom to use as desire, copy, distribute, modify
- often also free as in “free beer” (zero cost)
- preferred term of Richard Stallman

Open Source Software:

- source code (original program) is available to user
- to be able to modify, must have source code
- alternative term to Stallman's “free software”

FOSS: Free and/or Open Source Software

FLOSS: Free/Libre/Open Source Software(to emphasize the meaning of “free”)

Not all people like these acronyms---particularly FLOSS (because of dental interpretation).

Note that software can be mostly free but not open source—e.g., Java (in the beginning).

Software can also be open source but not free—e.g., source may be made available to paying customers (usually at additional cost).

"*Because open source software features open code, more programmers are able to view the code, create new functionality, and fix bugs. This follows the same natural way that science has developed over time*." - Taoism of Open Source; Chen Nan Yang

All FOSS licenses require that the software remain available to use, modify, and distribute at no cost. Most FOSS licenses also include kind of a lawyer's version of the Golden Rule, requiring that all software modifications, such as bug fixes and enhancements, must also be made available under the same license -- with permission to use, modify, and distribute at no cost. This naturally creates living software, continually growing the value for all. These licenses are good at generating the trust that leads to large and sustainable communities.

FOSS is increasingly the go--to standard for operating systems to user applications, for individuals to large enterprises. It helps reduce costs, avoid lock--in, increase productivity, enhance security, and improve standards compliance. With the best long--term investment protection, FOSS is the lowest risk choice for software systems today.
FOSS Examples

- Linux (operating system)
- KDE, GNOME, Xfce (desktop environments)
- Android (phone operating system/environment)
- Apache (web server)
- MySQL, PostgreSQL (DBMS's/servers)
- Perl, PHP, Python (scripting languages)
- OpenOffice (office software suite)
- GCC (GNU compiler collection)

FOSS Characteristics

- Often built collaboratively via Internet.
- Support for FOSS is typically provided via maillists, newsgroups, and web forums.
- Most FOSS is free of cost (“free beer”), so no continual cost for “upgrades.”
- Not limited to running single instance or having to guarantee license provisions are being met.
- Many companies exist to provide support or customization for businesses using FOSS

1.2 The FOSS Philosophy

The first known use of the phrase free open source software on Usenet was in a posting on 18 March 1998, just a month after the term open source itself as coined. In February 2002, FOSS appeared on Usenet newsgroup. In early 2002, MITRE used the term FOSS in what would later be their 2003 report Use of Free and Open Source Software (FOSS) in the U.S. Department of Defense.

What does society need?
It needs information that is truly available to its citizens -- for example, programs that people can read, fix, adapt, and improve, not just operate. But what software owners typically deliver is a black box that we can't study or change.
Society needs to encourage the spirit of voluntary cooperation in its citizens. When software owners tell us that helping our neighbors in a natural way is “piracy”, they pollute our society's civic spirit. This is why we say that free software is a matter of freedom, not price.
You deserve to be able to cooperate openly and freely with other people who use software.
You deserve to be able to learn how the software works, and to teach your students with it.

You deserve to be able to hire your favorite programmer to fix it when it breaks. You deserve free software.

The basic idea behind open source is “ When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, and people fix bugs. The FOSS is focused on the technical values of making powerful, reliable software, and is more business--friendly. It also focuses on the moral issues of Free Software and on the practical advantage of the FOSS distributed development method.

The popular myth surrounding Free/Open Source Software is that it is always “free”— that is, “free of charge.” To a certain degree this is true. No true FOSS application charges a licensing fee for usage. Most FOSS distributions (Red Hat, SuSE, Debian, Fedora, Ubuntu etc.) can be obtained at no charge off the Internet. On a licensing cost based, FOSS applications almost always cheaper than proprietary software. However, licensing costs are not the only costs of a software package or infrastructure. It is also necessary to consider personnel costs, hardware requirements, opportunity costs and training costs.

- The availability of the source code.
- The right of code modification, improvement and redistribution.
- Large base of developers and users.
- Have an alternative.
- Developing local capacity/industry
- Reducing imports/conserving foreign exchange
- Reducing copyright infringements
- Enabling localization
- Increasing competition
- Reducing Total Cost of Ownership
- Enhancing security
- Achieving vendor independence
- Increasing access to information

Development philosophy

In 1997 essay The Cathedral and the Bazaar, Eric S. Raymond suggests a model for developing OSS known as bazaar model Raymond likens the development software by traditional methodologies to building a cathedral, "carefully crafted by individual wizards or small bands of mages working in splendid isolation". He suggests that all software should be developed using the bazaar style, which he described as "a great babbling bazaar of differing agendas and approaches."

In the traditional model of development, which he called the cathedral model, development takes place in a centralized way. Roles are clearly defined. Roles include people dedicated to designing, people responsible for managing the project, and people

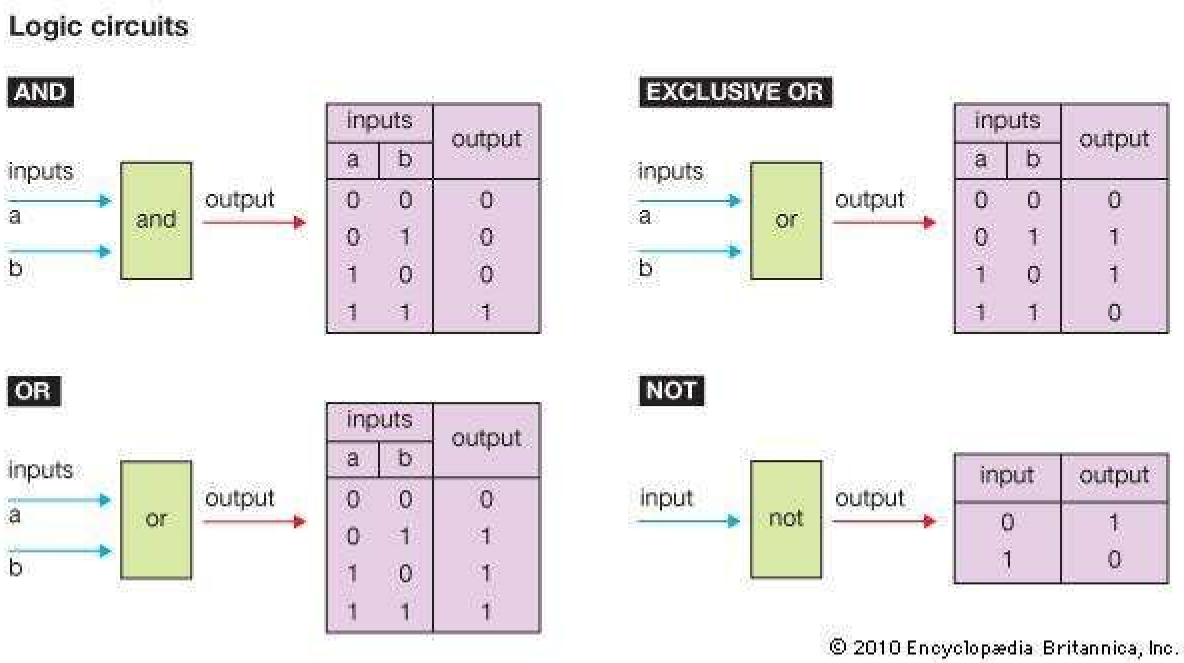
responsible for implementation. Traditional software engineering follows the cathedral model.
The bazzar model, however is different in this model, roles are not clear defined software developed using the bazaar model should exhibit the following patterns:
-- Users should be treated as co--developers
The users are treated like co--developers and so they should have access to the source code of the software. Furthermore users are encouraged to submit additions to the software, code fixes for the software, bug reports, documentation etc. Having more co--developers increases the rate at which the software evolves. If many users view the source code, they will eventually find all bugs and suggest how to fix them.
-- Early releases
The first version of the software should be released as early as possible so as to increase one's chances of finding co--developers early.
-- Frequent integration
Code changes should be integrated as often as possible so as to avoid the overhead of fixing a large number of bugs at the end of the project life cycle.
-- Several versions
There should be at least two versions of the software. There should be a buggier version with more features and a more stable version with fewer features. The buggy version is for users who want the immediate use of the latest features, and are willing to accept the risk of using code that is not yet thoroughly tested. The users can then act as co--developers, reporting bugs and providing bug fixes.
-- High modularization
The general structure of the software should be modular allowing for parallel development on independent components.
-- Dynamic decision--making structure
There is a need for a decision making structure, whether formal or informal, that makes strategic decisions depending on changing user requirements and other factors

Design Logic, Source Code, Binary Code

Logic Design:

The basic organization of the circuitry of a digital computer. All digital computers are based on a two-valued logic system—1/0, on/off, yes/no (see binary code). Computers perform calculations using components called logic gates, which are made up of integrated circuits that receive an input signal, process it, and change it into an output signal. The components of the gates pass or block a clock pulse as it travels through them, and the output bits of the gates control other gates or output the result. There are three basic kinds of logic gates, called “and,” “or,” and “not.” By connecting logic gates together, a device can be constructed that can perform basic arithmetic functions.

logic circuit



inputs		output
a	b	
0	0	0
0	1	0
1	0	0
1	1	1

inputs		output
a	b	
0	0	0
0	1	1
1	0	1
1	1	0

inputs		output
a	b	
0	0	0
0	1	1
1	0	1
1	1	1

input	output
0	1
1	0

© 2010 Encyclopædia Britannica, Inc.

Source Code:

A text listing of commands to be compiled or assembled into an executable computer program is called source code.

Initially, a programmer writes a program in a particular programming language. This form of the program is called the source program, or more generically, source code.

To execute the program, the programmer must translate it into machine language, the language that the computer understands. A compiler usually performs the first step of this translation process.

The compiler translates the source code into a form called object code. Sometimes the object code is the same as machine code; sometimes it needs to be translated into machine language by a utility called an assembler.

Source code is the only format that is readable by humans. When you purchase a proprietary program, you receive them in their machine--language format. This means that you can execute them directly, but you cannot modify them but in case of open source programs, you will get the entire program source.

To execute the program, the programmer must translate it into machine language, the language that the computer understands. A compiler usually performs the first step of this translation process.

The compiler translates the source code into a form called object code. Sometimes the object code is the same as machine code; sometimes it needs to be translated into machine language by a utility called an assembler.

Source code is the only format that is readable by humans. When you purchase a proprietary program, you receive them in their machine--language format. This means that you can execute them directly, but you cannot modify them but in case of open source programs, you will get the entire program source.

The source code consists of the programming statements that are created by a programmer in any programming tool and then saved in a file.

Binary Code

A coding system using the binary digits 0 and 1 to represent a letter, digit, or other characters in a computer or other electronic device is known as binary code. All computer languages are based on binary code. It is the back end of all computer functioning. Binary means that there is a code of either 0 or 1 for a computer to toggle between. All computer functions will rapidly toggle between 00 or 01 at an incomprehensible speed.

This is how computers have come to assist humans in tasks that would take so much longer to complete. The human brain functions holistically at much more rapid speeds than a computer in doing other types of very complicated tasks, such as reasoning and analytical thought processes. In computing, binary codes are used for any of a variety of methods of encoding data, such as character strings, into bit strings.

Those methods may be fixed--width or variable--width. In a fixed-- width binary code, each letter, digit, or other characters, is represented by a bit string of the same length; that bit string, interpreted as a binary number, is usually displayed in code tables in octal, decimal or hexadecimal notation. A bit string, interpreted as a binary number, can be translated into a decimal number.

For example, the lowercase a as represented by the bit string 01100001, can also be represented as the decimal number 97.

Examples of popular open source products

Whatever software business needs - eg for word processing, accounting or running a website - it is easy to find an open source version. Many websites such as SourceForge host open source projects and are good places to look for new software.

Types of open source software

Some popular open source software for business include:

Open source office software

- Abiword - word processing tool
- Open Office.org - business productivity suite
- Libre Office - business productivity suite
- CiviCRM - customer relationship management
- SugarCRM - customer relationship management
- vTiger - customer relationship management
- MySQL - database
- Ingres - database

Open source accounting software

- GNU cash - small business accounting and personal finance software
- SQL Ledger - web-based enterprise resource planning system
- Front Accounting - accounting and enterprise resource planning software
- PostBooks - accounting and business management system
- Compiere - accounting, CRM and ERP

Open source operating systems

- GNU/Linux (various versions or distributions include Debian, Fedora, Gentoo, Ubuntu and Red Hat) - operating system
- OpenSolaris - operating system
- FreeBSD - operating system
- Android - mobile phone platform

Open source website software

- Wordpress - blog hosting platform
- Alfresco - enterprise content management system
- Joomla - enterprise content management system
- Drupal - enterprise content management system
- Magento - ecommerce website platform
- PrestaShop - ecommerce website platform
- Apache - web server software
- NginX - web server software
- PHP - web application scripting engine

Open source browsers and communication applications

- Juice - podcasting
- Mozilla Firefox - web browser
- Mozilla Thunderbird - email client
- Pidgin - instant messaging
- Zimbra - email and collaboration server
- FileZilla - FTP client
- MediaWiki - information sharing platform

Open source IT security

- Smoothwall - firewall and security tools
- Wireshark (aka Ethereal) - security application
- KeePass - password management
- Clam AV - antivirus software

Images/multi-media

- GIMP - image processing/graphics editing
- VLC - multimedia file playback
- Ogg - open video and audio codecs
- Blender - animation and video editor

Open source development tools

- Ruby on Rails - rapid web application development
- Eclipse - integrated development environment

Future of FOSS

PCs:

GNU/Linux is gaining ground, rising quickly in usability in the last few years (which mainly was its drawback).

Companies jumped in (Google Chrome OS).

Embedded systems:

Gaining ground as currently writing an OS especially for an embedded system is not worth it in most of cases, simply install linux in lots of current systems.

Mobile devices:

The next (or current?) hype for linux.

Android is linux based. But Maemo/Meego is the really interesting thing: Linux+GNU+Debian stack+X+Gnome+QT