オープンソースって何だろう

Ver 1.2

可知 豊　[http://www.catch.jp/]

Last update:2003-10-08

「オープンソース」って聞いたことがありますか。

インターネットをするだけ、ワープロや表計算を使うだけという普通のパソコンユーザーにとって、オープンソースなプログラムを使うことは、今までほとんどありませんでした。でも、オフィスソフトの OpenOffice.org とブラウザの Mozilla という 2 つのオープンソースソフトウェアがメジャーリリースされたことで、それが大きく変わろうとしています。

ここでは、普通のパソコンユーザーに向けてオープンソースについて解説します。

1.オープンソースの基礎知識

『ソースコードを公開して、プログラムを自由に使用・修正・配布できるようにする』

この考え方につけられた名前が、オープンソースです。

いきなり「ソースコード」なんて言葉が登場しても、さっぱり意味がわかりませんね。「ソースコード」は、プログラムの作り方に関する用語です。だから、プログラムがどんなふうに作られるか知っておけば、オープンソースのことがもっとよく分かります。

●*プログラムの作り方*

コンピュータは、単純な処理をものすごいスピードでこなす機械です。一見複雑に見える仕事も、単純な処理を高速で繰り返すことで片づけています。

プログラムを作るということは、この単純な処理を組み合わせて、複雑な処理手順を定義することです。

一番単純なプログラムとして、文字を 1 行だけ表示する例を考えましょう。

プログラムを作るには、次のよう動作を記述します。

```
#include <stdio.h>
   main()
   {
      printf("Hello World!\n");       <-文字を表示
   }
```

これは、"Hello World"という文字を画面に表示するプログラムです。4 行目が文字を表示させる部分で、"printf"が文字を表示する指示になります。プログラムを作ったことがない人でも、「Hello World という文字を、画面に出すんだな」ということは、なんとなく想像がつくでしょう。

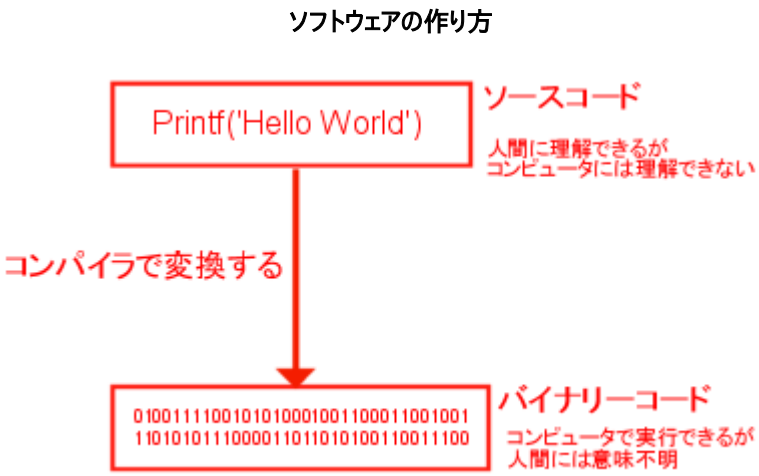
このように、プログラムを作るときには、まずプログラマがプログラム言語で記述します。この記述したプログラムが「ソースコード」です。

ここではたった 5 行だけですが、実用的なプログラムでは、これが数千行になります。1 万行を越えるプログラムも珍しくありません。

しかし、人間に理解できるソースコードでも、コンピュータは、そのまま処理できません。コンピュータが理解できるプログラムは、0 と 1 の羅列だけなのです。

そこで、このソースコードを変換します。変換にはいくつかの方式があります。ここでは、コンパイラというプログラムを使って、一括して変換しておく方法を紹介しましょう。

コンパイラは、人間が書いたソースコードをコンピュータが理解できる形式に変換します。"printf('Hello World');"を変換すると、コンピュータが理解できる処理の手順になります。これは、0 と 1 の組み合わせで表されます。このように変換されたプログラムを「バイナリーコード」と呼びます。



●バイナリーコードの特徴

「printf('Hello World');」というソースコードは、コンパイラによってバイナリーコードに変換されます。そのバイナリーコードを実行すると、画面に"Hello World"と表示されます。

一般のユーザーが利用しているソフトウェアは、ほとんどがバイナリーコードです。Microsoft Office も Windows もバイナリーコードです。

バイナリーコードは、次のような特徴を持っています

- バイナリーコードがあれば、そのプログラムを利用できる
- 異なる実行環境では、それに合わせたバイナリーコードが必要
- バイナリーコードだけ見ても、どう動いているか理解するのは大変

プログラムの実行時には、使うのはバイナリーコードです。ソースコードは必要ありません。ソースコードを一括して変換する方式では、バイナリーコードだけがあれば良いのです。

でも、バイナリーコードだけでプログラムが動作する訳ではありません。Windows のアプリケーションを実行するには、Windows が動いている必要があります。

プログラムを実行する時、必ずそのプログラムのターゲットになる実行環境が必要です。Winows アプリケーションなら、Windows が実行環境です。Windows 自身がプログラムですから、Windows 用の実行環境も必要になります。Windows 互換のパソコンが、その実行環境になります。

そして、同じバイナリーコードを実行する時には、この実行環境が共通でなければなりません。この種類が違うと、同じバイナリーコードは動きません。例えば、Windows と Macintosh は違う種類の実行環境なので、同じプログラムは動きません。

そのために、実行環境の数だけバイナリーコードを用意することになります(1-1)。

このようなバイナリーコードの動作を、人間が理解するのは大変です。

ソースコードは、プログラマが読めば、その動作を理解できます。そもそも人間が記述したのですから、必要なら不具合を修正したり、新しい機能を追加することもできます(1-2)。

一方、バイナリーコードを読んでも、プログラムの動きを理解するのはさらに大変です。例えば、Windows 版の OpenOffice.org をインストールするには、HDD に 250Mbyte 以上の空き容量が必要です。このかなりの部分がバイナリーコードになってます。ましてや、不具合を修正したり、新しい機能を追加するのは至難の業です。

バイナリーコードの特徴

● *ソフトウェアのビジネス*

では、あなたが使っているプログラムとソースコード/バイナリーコードは、どんな関係にあるんでしょう。

パソコンショップやオンラインショップでは、いろいろなプログラムを販売しています。ユーザーは、それを買ってきて、自分のパソコンにインストールして使います。また、買ってきたパソコンに付属していることもありますね。

さて、プログラムを開発したら、ソースコードをバイナリーコードに変換するのでした。企業は、優れたプログラムを開発したら、ソースコードを企業秘密にして、バイナリーコードだけ販売します。あなたが買ったプログラムは、このバイナリーコードなのです。ですから、たとえ不具合を見つけても誰も改造できません。修正版が出るのを待つか、バージョンアップに付き合うしかありません。いわば、企業はソースコードを独占しているのです。

じつは、ユーザーはバイナリーコードを買っているではありません。バイナリーコードを利用する権利を買っているのです。この権利は「ライセンス」と呼ばれます。ライセンスには、ユーザーがどんな条件でバイナリーコードを利用できるのか書いてあります。ユーザーがお金を払うと、そのライセンスの条件内で使用できるようになります。例えば、パソコン 1 台だけとか、複数のパソコンで利用するなら、同時に利用するなら何台まで、パソコンに最初から付属していた場合は、そのパソコンでだけ使って良いといった具合です。

このような条件は、ソフトウェアパッケージを買うと、パッケージの外側やマニュアルに書いてあります。インストールの際の使用許諾にも表示されます。この使用許諾に同意するから、利用する権利が得られるのです。

ユーザーは、使用許諾の範囲を超えて、バイナリーコードを使用してはいけません。複数のパソコンにプログラムをインストールしたり、他の人にコピーさせるのは、ライセンス違反=不正コピーになります。

これまでのソフトウェアビジネスは、このようにソースコードを独占することで成立してきました。企業は、そのソースコードを押さえることで、ビジネスの主導権を握っています。

2.オープンソースの考え方

ソースコードが誰でも入手できたらどうでしょうか。そこからバイナリーコードを作るのは簡単だし、それをコピーするのはもっと簡単ですね。それに、異なるコンピュータの間でも、ソースコードがあれば比較的簡単にプログラムを移植できます。

ここでは、ソースコードの扱い方を中心にして、オープンソースの考え方を解説します。

●*バイナリーコードが自由にコピーできるとしたら*

バイナリーコードが無料で配布されたらどうなるか、ちょっと考えてみましょう。バイナリーコードが必要な人は、自由に無料でコピーできるのです。

当然ですが、もう誰もソフトウェアなんか買いませんよね。ソフトウェア開発企業は大損です。ということは、開発にお金がかけられません。せっかく無料で手に入っても、高機能なソフトウェアはなくなってしまうかも知れません。

実はこれ、無料で配布されているオンラインソフトと同じですよね。一般には「フリーソフト」と呼ばれています。

でも、バイナリーコードが自由にコピーできるだけでは、プログラムの開発には、あまり影響がありません。もしも、そのプログラムの不具合を直したい、機能を追加したいと思っても、バイナリーコードだけでは大変です。

●*ソースコードを公開する*

では、ソースコードが公開されていたらどうでしょう。そこからバイナリーコードを作ってそれを配っても、費用はほとんどかかりません。必要なら、自由に改良できるでしょう。

ソースコードを公開することは、コンピュータの黎明期から、主に大学の研究室を中心に行われてきました。

研究者の間では、自分の研究成果を非公開にすることはありません。それでは、他の研究者に評価してもらえないからです。コンピュータに関する研究では、その時にソースコードも公開するのが一般的でした。研究室同士で、同じコンピュータを持っているとは限りません。そこで、ソースコードを公開することで、コンピュータの種類が違っていてもプログラムを動かせるようにしたのです。

ソースコードが公開されて自由に使えると、プログラムの改良がある程度解消します。もしも、問題や改良点があるなら、それを自分でプログラムすればいいのです。異なるコンピュータに移植しても良いでしょうし、もっと高機能なプログラムをど

んどん開発しても良いでしょう。

●ソースコードの公開方法

ソースコードを公開する方法には、いくつかの種類があります。ただ公開するだけでなく、どんな場合に使って良いか、公開する人が条件を定めるのです。このような使用条件もライセンスと呼ばれました。

例えば、アメリカには「パブリックドメイン」という公開方式があります。これは、ソースコードの著作権を放棄したもので、そのプログラムを誰でも自由に使用・配布・改良できました。

しかし、この考え方には欠点がありました。著作権を放棄したために、それが企業の製品に組み込まれ、その改良版を自社の著作物として主張できたのです。これでは、せっかくソースコードを公開していても、無駄になってしまいます。

また、日本の国内法では、そもそも著作権の一部である著作人格権を放棄できないので、パブリックドメインは実現できません。

●ソースコードのコピーレフト

リチャード・ストールマンは、コンピュータの世界では知らない人はいないという伝説的な人物です。MIT(マサチューセッツ工科大学)の AI ラボという有名なコンピュータ研究所で活躍し、Emacs という高性能なエディタを開発したことで知られています。

このストールマンが、パブリックドメインに変わる新しいソースコードの公開方法「コピーレフト」を考え出しました。

> **コピーレフト**
>
> プログラムを作ったら、ソースコードを必ず公開する
>
> このソースコードの著作権は放棄しない
>
> でも、次の条件を守る限り、自由に使用・改良・配布して良い
>
> 条件：改良したソースコードに、まったく同じ条件を付けること

コピーレフトでは、パブリックドメインと違って、著作権は放棄しません。

でも、ソースコードを公開します。そのソースコードは、自由に使用・配布・改良ができます。そして、その改良版も、同じように使用・改良配布が自由で「ただし改良したソースコードは、同じ条件を付けること」とするのです。だから、この改良版の改良版も、同じ条件を付けなくてはなりません。これがずーっと続くのです(2-1)。

「コピーレフト」(CopyLeft)という名前は、著作権のコピーライト(CopyRight)をもじったものです。Right(右)と Left(左)をかけた駄洒落になっているのです(2-2)。

ストールマンは、この作戦を本当に実行しました。

これを GNU プロジェクトと名付けて、それを支援する団体「フリーソフトウェア財団」(FSF)を設立しました。また、自分が開発した Emacs に加えて、gcc(GNU C コンパイラ)を開発し、広くユーザーに公開しました。この gcc は、コンパイラの代名詞とも言える存在になっています。さらに、コピーレフトの定義を弁護士に読んでもらい、正式の法律文書にした上で「GNU 一般許諾ライセンス」として公開しました。このライセンスは「GPL」と呼ばれています。

そして、この運動が多くの成果を上げたのです。

●Linux とコピーレフト

現在、数多くのプログラムが、GPL の元で公開されています。中でも、Linux はその最大の成果でしょう。

Linux は、ソースコードが GPL の元で公開されている UNIX 互換の OS です。本来「Linux」は、カーネル(kernel)と呼ばれる OS の核を示す言葉ですが、その Linux カーネル上で動作するシステム全体も同じ名前で呼ばれています。1991 年、当時ヘルシンキ大学の大学院生だったリーナス・トーバルズによって開発がスタートし、現在でも全世界のボランティア開発者によって改良が続けられています。何より、非常に安定して動作するという特長を持っており、インターネットの基幹業務で高い人気を集めています。

Linux カーネルは、開発中のソースコードがインターネットで公開されています。誰でも、自由に使用・改良・配布ができます。もしも、不具合や改良点があるなら、それを自分でプログラムすればいいのです。もちろん、改良したソースコードは GPL に基づき公開する必要があります。このフィードバックを集約することで、Linux の開発は、猛スピードで進んできたのです。

誰でも自由に改良できると聞くと、Linux はバラバラに開発されていると思われがちです。でも、実際にはそうではありません。インターネットで連絡を取りながら、緩やかな組織を形成して、技術を統合しています。このような組織は、開発コミュニティと呼ばれます。

プログラムの開発で大きな時間をとられるのは、動作チェックとその修正です。これを効率よく片づけるには、できるだけたくさんのコンピュータで動作チェックします。これは企業でも大変なことです。しかし、インターネットを利用して同時に作業を進めれば、この動作チェックはすぐに片づけられます。コストを気にする企業では不可能なほど、膨大なフィードバックが得られるのです。

また、動作チェックをして不具合を見つけた本人が、公開されているソースコードを調べて、その不具合を自分で修正してくれたら、作業はもっと早くなります。

Linux は、ソースコードの公開とインターネットを上手に組み合わせることで、驚くようなスピードで開発が進み、高い安定性を得たのです。

●オープンソースというコンセプト

Linux の成功を受けて、ソースコードを公開するというやり方が注目されました。

特に、エリック・レイモンドの論文「伽藍とバザール」で、Linux の開発手法と効果が詳しく解説されました(2-3)。

『ソースコードを公開して、プログラムを自由に使用・修正・配布できるようにする』

この考え方につけられた名前が、オープンソースです。そのためコピーレフトだけでなく、パブリックドメインや BSD ライセンスも含まれていることになります。

「ソースコード」の意味がわかれば、今度は理解できますよね。でも、コピーレフトとは、どう違うんでしょう。なぜ「オープンソース」なんて呼ぶんでしょう。

ソースコードを公開するソフトウェアのライセンスは、GPL だけではありません。この他にも、色々なソフトウェアがソースコードと共に独自のライセンスで公開されてきました。

そこで「オープンソース」という呼び名が、オープンソース・ソフトウェア・イニシアティブ(OSI)によって提唱されました。OSI は、前述の「伽藍とバザール」を書いたエリック・レイモンドが中心になって設立されたオープンソースの推進団体です。OSI は、これまでの色々なソースコード公開ライセンスに統一的な呼び名を付け、その考え方を明確に定義したのです。そして、ビジネスでのオープンソースソフトウェアの普及を目指しました(2-4)。

オープンソースであるプログラムの配布条件は、いくつかのの基準を満たしていなければなりません。これを簡単に説明すると次のようになります。

- ソースコードを公開する
- 自由に再配布できる
- ソースコードは修正して配布してもいい
- ソースコードの出所を明示する
- 個人や集団によって差別しない
- 使用する分野によって差別しない
- 他の契約との組み合わせで、条件を変えない
- 特定製品でのみ有効なライセンスの禁止
- ライセンスは、ほかのソフトウェアのライセンスに干渉しない

GPL と比べて、条件がずいぶんややこしくなりましたが、これは GPL の条件を緩くしたものなのです。オープンソースの条件では、改良版も同じ条件で配布する必要がありません。元になるソースコードのライセンスの範囲内で、独自に決められるのです。例えば、オープンソースなライセンスの一つである「BSD ライセンス」では、改良したソースコードを再配布する必要がありません(2-5)。

オープンソースの定義では独自のライセンスを定義している訳ではありません。代わりに、GPL などのライセンスが、この条件に合っているかを判断して、オープンソースの認定マークを発行しています。

Linux の成功が注目されると共に、オープンソースの考え方も広がりました。また、その品質の高さから、他のオープンソースソフトウェアも普及していきました。現在では、多くの企業がオープンソースソフトウェアを開発したり、利用しています。

オフィスソフトの OpenOffice.org とブラウザの Mozilla は、このようなオープンソースソフトウェアです。

これまで、オープンソースソフトウェアは、一般ユーザーが意識して使うことはありませんでした。Linux も、いわば縁の下の力持ちとして普及してきたのです。

それが、OpenOffice.org と Mozilla の登場で大きく変わろうとしています。Linux を配布している人たちも、一般のユーザーが使える Linux に力を注ぎ始めています。

3.一般ユーザーのためのオープンソース

オープンソースは、コンピュータをもっと良い物にするのに役立ちます。これは、一般ユーザーにとっても、大きなメリットをもたらします。

●*フリーソフトとオープンソースの違い*

無料で自由に使えるプログラムというと、パソコンの一般ユーザーは、無料で使える「フリーソフト」や「フリーウェア」を思い浮かべるでしょう。この言葉は、無料ソフトの代名詞としてすっかり定着してしまいました。

また、インターネットでダウンロードできる「オンラインソフト」も有名ですね。継続的に使用する場合に料金を払う「シェアウェア」というのもあります。また、フリーソフトでオンラインソフトというのも、たくさん存在します。

でも、オープンソースソフトウェアは、これとはずいぶん違います。フリーソフトもシェアウェアも、ソースコードは公開されるとは限りません。これに対して、オープンソースでは、必ずソースコードが公開されています。

オープンソースソフトウェアは、無料で自由にダウンロードできますから、フリーソフトでもあり、オンラインソフトでもあります。でも、その根本の発想は違います。

●*オープンソースで開発が進む*

オープンソースでソースコードが公開されていると、開発者本人が改良できなくても、誰か別の人が手を付けられます。オープンソースで公開されているということは、他の人の改良を奨励しているという意思表示でもあります。改良版を配布しても構いませんから、改良者は、自分の作業を無駄にすることがありません。

すでにプログラムが改良されているなら、オリジナルのプログラムと改良版を統合するのも、決して難しくはありません。そんな開発者が集まれば、プログラムのバージョンアップだって、組織的に進むでしょう。

一般に開発コミュニティでは、ソースコードの改良が奨励されます。ユーザーが、「あの機能が欲しい」「この機能が便利だと思う」と言うだけでは、プログラムは良い物になっていきません。その機能を誰かがソースコードに盛り込むことが必要だからです。そして、改良版のソースコードがひとたび採用されたら、その人の名前は永遠に記録に残されるのです。

●*オープンソースは無料？*

オープンソースの配布条件は、「自由に再配布できる」と決められているだけです。

つまり、無料でも良いし、有料でも良いのです。オープンソースのソフトウェアは配布に際して手数料を取っても構いません。手数料はいくらでも構いませんが、高すぎれば、他の人から入手するでしょう。買った相手がそれを再配布しても良いからです。

現在、オープンソースでソフトウェアを開発している企業がたくさん存在します。このような企業は、ソフトウェアの販売ではなく、有償サポートで利益を上げる場合が多いようです。また、他のプラットフォームへの移植を有償で請け負う企業や、自社で利用するプログラムをオープンソース化する企業もあります(3-1)。

オープンソースで開発すると言うことは、プログラマが無償で働くと言うことではありません。オープンソースのソフトウェアを販売している会社は、プログラマに給料を払って開発に参加させています。その成果でソフトウェアが良い物になれば、さらに販売がのびるからです。

もちろん、腕に覚えのある人が無償で働く場合もあるでしょう。でも、そんな優秀なプログラマなら、世界中から求人が殺到するはずです。

●*一般ユーザーにとってのオープンソース*

では、一般のユーザーにとってオープンソースはどんなメリットがあるのでしょう。

- 自由に使える
- コミュニティで開発が進み高性能になる > 安心して使える
- 無料で入手可能

ソースコードが公開されないソフトウェアの場合は、その情報を独占している人の状況に縛られることになります。それが企業なら、もしかしたら、来年はライセンス料を値上げされるかも知れません。突然倒産してサポートを受けられなくなるかも知れません。でも、オープンソースになっていれば、ソースコードが公開されています。だから、誰かが改良できます。問題があれば調べることもできます。オープンソースは、ソフトウェアを情報の独占から解放する手段なのです。

一般のユーザーにとっても安心ですね。

もちろん、ソフトウェアが無料で入手できるというのも大きな魅力です。

ただし、サポートは有料かも知れません。

できるだけお金をかけずにサポートを受けたいなら、一番良いのは、あなたがサポートする側に回ることでしょう。安定してサポートが受けられれば、もっとたくさんの人が使うようになって、さらに安心して使えるようになるはずです。

プログラムを改良するのは、誰でもできることではありません。でも、質問に答えたり、情報を整理するのなら、あなたにもできることがあるはずです。

サポートする人が存在すれば、さらに多くの人が安心してオープンソースソフトウェアを使えるようになるでしょう。たくさんの人が集まれば、問題を調べたり、質問に答えるのも、それほど手間ではありません。インターネットを介すれば、大した費用をかけずにユーザーをサポートできるはずです。例えば、OpenOffice.org 日本ユーザー会が、そういう存在です(3-2)。

●*ソースコードがあってのオープンソース*

一般のパソコンユーザーにとって、本当に欲しいのは無料のバイナリーコードでしょう。そして、そのプログラムが自由に使えて高性能なら言うことはありません。ですから、ソースコードが公開されるかどうかは、あまり気にならないと思います。

でも、そのプログラムの質を高めるためには、ソースコードは欠かせない存在です。あなたが、質の高いプログラムを自由に使いたいなら、ソースコードの公開にこだわるべきです。オープンソースになっていれば、そのプログラムはさらに進化する可能性を秘めているからです。

それに、もうひとつ理解して欲しいことがあります。

オープンソースにおいては、ユーザーが「あの機能が欲しい」「この機能が便利だと思う」と言うだけでは、プログラムは良い物になっていきません。その機能を誰かがソースコードに盛り込むことが必要だからです。開発コミュニティは、オープンソースソフトウェアをそんなふうに進化させるために存在しています。

あなたにも、このコミュニティに貢献できることはないでしょうか。もちろん、誰でもプログラムを作れるわけではありません。でも、オープンソースソフトウェアを良くするために、あなたにも手伝って欲しいのです。

4.オープンソースの現状と問題点

オープンソースにも決して問題がない訳ではありません。ここでは、もうちょっと詳しくなりたい人のために、オープンソースの現状と問題点を解説します。

●*色々なライセンス*

代表的なオープンソースソフトウェアには、次のものがあります。

代表的なオープンソースソフトウェア

ソフトウェア名	機能	ライセンス
Linux	OS	GPL
FreeBSD	OS	BSD ライセンス
Mozilla	ブラウザ	GPL+LGPL+MPL

ソフトウェア名	機能	ライセンス
OpenOffice.org	オフィスソフト	LGPL+SISSL
Apache	WEB サーバー	Apache software License
Perl	スクリプト言語	Artistic ライセンス

ライセンスにずいぶん種類がありますね。

実は、ライセンスによって、改良したソースコードを公開する条件が少しずつ違っているのです。

OpenOffice.org をはじめ、いくつかのプログラムでは、複数のライセンスが適用されています。これらは「デュアルライセンス」と呼ばれます。デュアルライセンスでは、プログラムを 2 種類以上のライセンスで配布します。それらのライセンスは、それぞれ条件や制限が異なっています。ユーザーは、どちらか都合の良い方を選んで、使用・改良・配布することになります。

開発コミュニティに改良版を付け加えるには、その改良版もデュアルライセンスにする必要があります。

●*GNU ライセンス*

数あるオープンソースなライセンスの中で、もっとも普及しているが GPL です。「GNU 一般公衆使用許諾」(GPL:GNU General Public License)が正式な名前です。これは、次のサイトで読むことができます。

GNU General Public License

http://www.gnu.org/licenses/gpl.html

GNU 一般公衆利用許諾契約書の日本語訳

http://www.opensource.jp/gpl/gpl.ja.html

GPL は、コピーレフトを明文化して、実際の使用許諾文書にしたものです(4-1)。実物を読んでみると、抜け穴がないようにと厳密な書き方になっているため、意味をつかみにくいかも知れません。

また GPL に関連して、LGPL と GFDL というライセンスがあります。

GNU Lesser General Public License

http://www.gnu.org/copyleft/lesser.html

GNU 劣等一般公衆利用許諾契約書の日本語訳

http://www.gnu.org/copyleft/lesser.ja.html

劣等使用許諾は、略して LGPL と呼ばれています。これは、ソースコードの部品集(ライブラリと呼ばれます)のソースコードを公開するライセンスとして利用されてきました。部品集は、他のプログラムに組み込まれて初めて役に立ちます。だからと言って、組み込んだプログラムもオープンソースになってしまうのは、ちょっとやり過ぎですよね。そこで、組み込むだけならオープンソースにならないように、効力を弱くしたのが LGPL です(ただし、現在ではライブラリに適用することは奨励されていません)。

●*OpenOffice.org のオープンソース*

OpenOffice.org は、LGPL と SunIndustry Standards Source License(SISSL)のデュアルライセンスで開発・配布されています。

LGPL を適用すると、それを組み合わせたソフトウェアを商用ソフトウェアとして利用できるようになります。OpenOffice.org と組み合わせた商用ソフトウェアを可能にするため、このライセンスが使われています。

一方の SISSL では、オープンソースの条件に適合するライセンスで、互換性を維持する条項が特徴です。

Sun Industry Standards Source License

http://www.opensource.org/licenses/sisslpl.html

SISSL 日本語訳(参考)

http://ja.openoffice.org/sissl_ja_01.html

●*GPL とBSD ライセンスの違い*

オープンソース条件の違いの中で、良く論争の的になる条項があります。それは、再配布時のライセンスをどうするかという問題です。

オープンソースでは、ソースコードを修正し、再配布することが許されています。

この時、再配布するソースコードの利用許諾は、どうするべきでしょうか。

BSD ライセンスでは、この点で制限がありません。そのために、修正したソースコードを独自ライセンスで公開できます。その際に、オリジナル版の著作権者名を表示すれば良いのです。極端な例では、特定の企業が自社製品に組み込んで、独自の使用許諾で販売できます。

このようなことがおこると、オープンソースで開発されてきた成果が、独自ライセンスのソフトウェアの中に取り込まれてしまいます。以後、そのソフトウェアに企業が付け加えた成果は、外部からは利用できません。

BSD ライセンスのソフトウェアは、利用許諾の条件が少ないので、良い意味でも悪い意味でも気軽に利用できるのです。

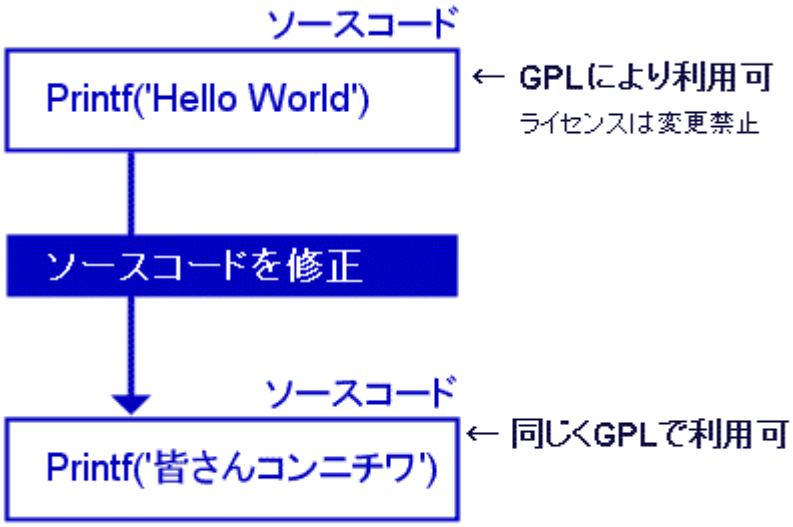
BSD ライセンスの場合

これに対して GPL では、修正版のソースコードを公開する時には、それも GPL にしなければならないとなっています。GPL なソフトウェアを組込んだソフトウェアも、GPL にする必要があります。例えば、GPL なワープロソフトのソースコードがあったら、それを基に開発した年賀状ソフトも、GPL にしなければならないのです。このように GPL なソフトウェアを基にしたソフトウェアを、そのソフトウェアの"派生物"と、GPL では呼んでいます。

ただし、この条件は、GPL なソフトウェアを使用して作られたデータに適用される訳ではありません。つまり、GPL なワープロソフトで作成した文書ファイルを GPL にする必要なありません。このように作られたデータを、GPL なソフトウェアの"出力結果"と、GPL では呼んでいます。

ソフトウェアにおいて、なにが"派生物"でなにが"出力結果"か、そのソフトウェアの種類によって違ってきます。特定の GPL なソフトウェアの使用許諾を読むときには、その点にも注意しましょう。

GPL の場合



GPL の場合は、改良したソースコードも必ず GPL で公開しなければなりません。

一方、FreeBSD で採用されている BSD ライセンスでは、必ずしも同じ条件で公開する必要がありません。そのため、このライセンスで公開されているソースコードは、改良した内容を企業の独占的なソフトウェアに取り込めます。例えば、それをマイクロソフトが製品に組み込むことが可能です。これは、オープンソースの成果を利用したい企業にとって、魅力的なライセンスです。

●オープンソースは一枚岩ではない

「オープンソース」という言葉は、ソースコードを公開するという考え方を一般に広め、それを企業に利用させることに成功しました。でも、ソースコードの公開方式がいくつもある中で、各ライセンスの違いを際だたせることにもなりました。

たとえば、オープンソースと認定されている BSD ライセンスのプログラムでは改良版を商業ソフトウェアに変更可能です。ソースコードをオープンにしないで、その成果を独占できるのです。

そのため、コピーレフトに賛成しているプログラマは、BSD ライセンスの開発コミュニティには参加できないことになります。そのプログラムを改良しても、将来、独占的なソフトウェアに使われてしまう可能性があるからです。改良版を GPL にすることもできますが、その改良版は、オリジナルに取り込むことができません。

結局、BSD ライセンスのソフトウェアを改良するプログラマは、改良版のライセンスをどうするか悩む必要があります。自分の改良版が優れているという自信があるなら、それを独占的なソフトウェアに切り替えるという誘惑が存在します。一方で、自分の改良版を GPL として公開すれば、その先は悩む必要はありませんが、他の改良版と組み合わせることはできなくなるかも知れません。

このように、オープンソースソフトウェアは、そのライセンスによって少しずつ考え方が違っています。GPL を推進している人たちからは、この点が批判の対象になります。また、次に説明するように、オープンソース自体も批判の対象になっています。開発コミュニティの中で起こるこのような意見の不一致は、宗教論争と揶揄されることもあります。

●フリーソフトウェアと呼んで欲しい

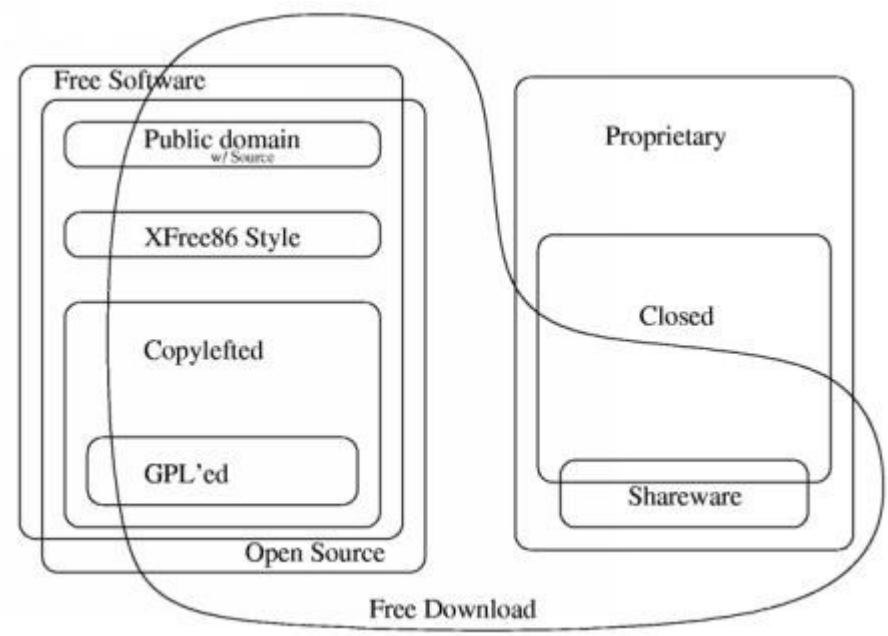
GPL を考え出したストールマンのフリーソフトウェア財団(FSF)は、ソースコードを公開して自由に使用・修正・配布したソフトウェアを、「オープンソースソフトウェア」ではなく「フリーソフトウェア(Free Software)」と呼ぶことを提唱しています。

英語のフリー(Free)という言葉は、自由と無料という二つの意味を持っています。"フリーソフトウェア"の"フリー"は、自由という意味のフリーです。フリーソフトウェアは、いつでもどこでも誰でも自由に利用・修正・配布できるのです。

次の図は、フリーソフトウェア財団によるソフトウェアの分類です。

この場合のフリーソフトウェアは、無料のオンラインソフトではなく、ソースコードが公開されて、誰でも自由に使用・修正・配布ができるソフトウェアを示しています。

ソフトウェアの分類



フリーソフトウェア財団「フリーおよびフリーではないソフトウェアの分類」より

Chao-Kuei による図解

http://www.gnu.org/philosophy/categories.ja.html

この図でも分かるように、オープンソースソフトウェアとフリーソフトウェアは、微妙に違うものです。これは、基本となる考え方が違うからです。オープンソースはソースコードを公開することにこだわるのに対し、フリーソフトウェアはソフトウェアに対する自由にこだわっているのです。

これについて、リチャード・ストールマンは次の文書を発表しています。

Why ``Free Software" is better than ``Open Source"

http://www.gnu.org/philosophy/free-software-for-freedom.html

「フリーソフトウェア」が「オープンソース」より好ましい理由

日本語訳： yomoyomo

http://www1.neweb.ne.jp/wa/yamdas/column/technique/fsffj.html

この文書では、「「オープンソース」という用語を用いたら何も問題の解決にならず、それどころか問題を生み出してしまう」ことを説明しています。

この他にも、フリーソフトウェア財団はフリーソフトウェアについての多数の解説を発表しています。あまりに解説が多すぎて、煙たがられることもない訳ではありません。

残念ながら、フリーソフトやフリーウェアという紛らわしい言葉が、無料ソフトの代名詞として定着しています。オープンソースという言葉も市民権を得つつあるようです。でも、フリーソフトウェアという言葉が、どこまで受け入れられるか定かではありません。

とはいえ、コピーレフトの考え方を表現している GPL は、自由なプログラムをつくるためのソースコードの公開方法として、もっとも強力なライセンスです。「自由なソフトウェア」という考え方は、少しずつ浸透していくかも知れません。

●結局、オープンソースって何だろう

オープンソースという名前は、いわばマーケティングのために付けられたキャンペーン名に過ぎません。

「ソースコードを公開する」という運動を、ビジネスに関わる人々に理解してもらうため、開発コミュニティにマーケティングと言う発想を持ち込んだ結果、こんな名前がつきました。実際に、このキャンペーンのおかげで、企業は Linux や色々なプログラムのソースコードを利用するようになり、オープンソースという呼び名を使うようになりました。

とはいえ、開発コミュニティは単一の組織ではありません。コンピュータに関わる人たちの緩やかな連合体です。当然、そこには色々な考え方の持ち主がいますし、その考え方が統一されることもないでしょう。

緩やかな連合体ですが、開発コミュニティは現実の存在です。インターネットを通じて、多くの人が情報を交換し、プログラムを開発しています。そして、このコミュニティがある限り、オープンソースソフトウェアはこれからも進化していくでしょう。

5.注釈

●1-1：そのために、実行環境の数だけバイナリーコードを用意する事になります

実は、Macintosh 上で Windows のアプリケーションを実行する VirtualPC というプログラムが存在します。これは、Windows の実行環境を Macintosh 上で再現するプログラムです。このようなプログラムはエミュレータと呼ばれます。また、Java というプログラム言語で開発されたプログラムは、Windows でも Macintosh でも Linux でも、どんなパソコンでも動くようにできます。こちらの場合は、Java VM(バーチャルマシン)という実行環境を各コンピュータで用意しています。この Java VM によって、コンピュータの違いを吸収してしまうのです。

●1-2：必要なら不具合を修正したり、新しい機能を追加することもできます

そうは言っても、これはそれほど簡単なことではありません。数千行に渡るソースコードを追っかけるのは、大仕事です。普通は、人間が読みやすいように、その働きを説明したコメントをソースコードに大量に記入しておきます。

●2-1:これがずーっと続くのです

こんなふうに、同じ性質が引き継がれていくのは、「継承」というプログラム上のテクニックです。

ストールマンは、この問題をプログラマらしく定義したのです。

●2-2:駄洒落になっているのです

『個人的には、left は leave の過去分詞でもあるので、著作権を保持している（Public Domain ではない）と言う意味もかけているという点の方が、右・左よりもシャレてると解釈してきましたが・・・考え過ぎかもしれない・・・』というコメントを樋口さんから頂きました。

また『そのほかに、いわゆる右翼・左翼の左 -革命- の意味も持っているという説もあります。（RMS 自身がそう発言したという話は聞きませんが） 事実、GNU の活動は文字通りに革命を起こしたのはご存知のとおりです。』というコメントを無津呂さんから頂きました。

●2-3:Linux の開発手法と効果が詳しく解説されました

「伽藍とバザール」エリック・S・レイモンド著・山形浩生訳

http://cruel.org/freeware/cathedral.html

●2-4 : ビジネスでのオープンソースソフトウェアの普及を目指しました

オープンソースという名前を採用するに至った経緯は、次のページが詳しく書かれています。

「ハッカーの逆襲」　エリック・S・レイモンド著・稲葉祐之訳

http://www.cus.cam.ac.uk/~yi205/revenge-j.html

●2-5 : ビジネスでのオープンソースソフトウェアの普及を目指しました

OSI によるオープンソースの定義は、次のサイトで読めます。

オープンソースの定義(The Open Source Definition)

http://www.opensource.org/docs/definition.html

OSDN Japan にある日本語訳

http://osdn.jp/article.pl?sid=01/08/17/0849221

●3-1:オープンソースでソフトウェアを開発している企業がたくさん存在します

利益をあげる手段としては、オープンソースソフトウェア それ自体の有償サポート以外に、自社ソリューションを 実現する手段の一つとしてオープンソースソフトウェアを 開発するというものがあります。

たとえば Zope Corp. における Web アプリケーションサーバ Zope がそれに該当します。PostgreSQL や MySQL などでも類似した事例があります。

他には「他プラットフォームへ移植する」という事を 請け負うケースがあります。

RedHat に買収された Cygnus は組み込み系メーカの依頼を受けて各種プロセッサ向けに GCC や GNUPro などのツール, eCos を 移植していました。NetBSD の組み込み系改造や移植を請負う企業もあります。

「プリンタは幾つか採用しているところがあります: RICOH IPSiO(NetBSD/mips)。

パチンコはかなりの部分で NetBSD が組み込まれているらしい。これは GPL では具合が悪いので(公開の義務) 、BSD ライセンスだと有利という理由に依ります。」

「ルータなどのネットワーク関係の製品とかも NetBSD を内蔵 しているものが幾つかありますね。製品情報として公開して いるものもちらほら。マルチメディアサーバ mmEye は 以前に雑誌記事でも見かけました。SH3 プロセッサへの移植をやった所ですよね、確か。http://www.brains.co.jp/mmeye/mmeye.html http://www.jp.netbsd.org/ja/Ports/sh3/」

> Thanks　無津呂さん 、なかたさん、はらおかさん

●3-2:OpenOffice.org 日本ユーザー会が、そういう存在です

オープンソースなオフィスソフト OpenOffice.org の日本のユーザーの集まりです。

ドキュメントの整備・翻訳、日本語版設定ツールなどを開発しています。

http://ja.openoffice.org/

●4-1:コピーレフトを明文化して、実際の使用許諾文書にしたものです

GPL を自分のプログラムのライセンスとして利用するなら、内容を良く読んでおきましょう。

さらに、そのプログラムで何が派生物なのか、出力結果なのか明確にしておくと良いでしょう。

例えば、Linux カーネルでは、システムコールのインターフェースはライセンスから外してあるそうです。

「Linuxの場合、システムコールのインターフェイスは、カーネルそのものには含まれない部分とみなしている（というか、私がそう宣言したということになる）。」（Linux の強みより）

これがが GPL の適用範囲内だと、システムコールを利用しているソフトウェア(つまり、全ての Linux アプリケーション)が派生品とみなされ、GPL になってしまうからです。

参考　「Linux の強味」　リーナス・トーバルズ著・倉骨彰 訳

「オープンソースソフトウェア 彼らはいかにしてビジネススタンダードになったのか」より

http://www.law.co.jp/okamura/OpenSource_Web_Version/Web_version991206.html