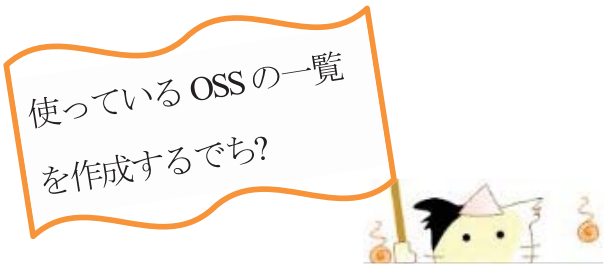
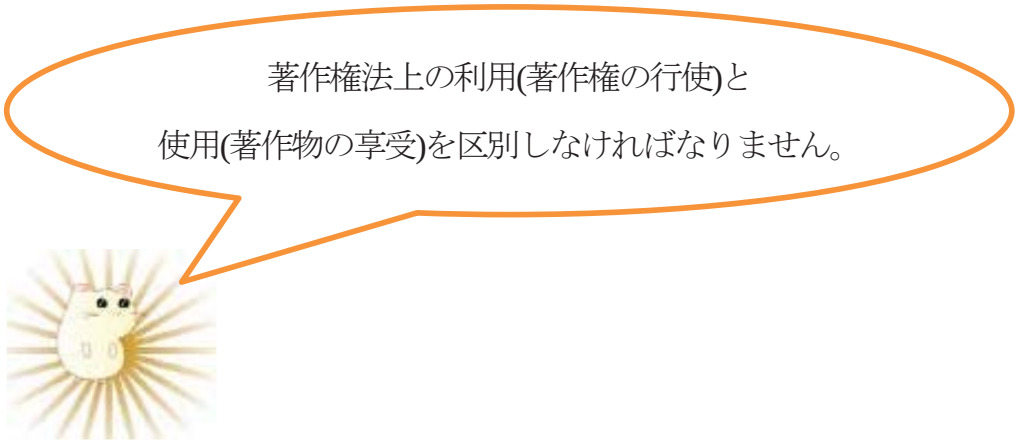
第5章 トラブル回避のための基本的な施策案

OSS を利用する際にライセンス違反・著作権侵害などのトラブルにならないようにするために、筆者は、「OSS ライセンスと著作権法」講義をはじめ、ガイドラインの作成支援や開発管理プロセスの改善支援などの有償サービスを提供している。その中で紹介している基本的な施策案をいくつか示す。





5.1 利用 OSS の一覧表の作成

OSS のライセンス条件を気にする前に、まずは現状を把握することが肝要である。自分が扱う対象にどんな OSS が含まれるのか、つまり、自分が誰のどの権利を行使(無断なら侵害)しようとしているのかを自覚することである。そのために、対象に含まれる OSS の一覧表を作成する必要がある。

(1) 一覧表の内容

利用 OSS の一覧表には以下の 5 項目を含むようにする。

- 利用 OSS
  - ① OSS の名称
  - ② OSS のバージョン
- 利用 OSS ライセンス
  - ③ OSS ライセンス名
  - ④ OSS ライセンスのバージョン
  - ⑤ OSS ライセンスタイプ(2.4 節参照)

筆者の提案する利用 OSS の一覧表の形式案を以下に示す(表 5-1)。つまり、利用 OSS とそのバージョン、OSS を利用する際に選択する OSS ライセンス名とそのバージョン、わかれば、その OSS ライセンスタイプを確認しておくのである。

表 5-1 利用 OSS 一覧表例

	利用 OSS		利用 OSS ライセンス		
	①OSS 名	②バージョン	③OSS ライセンス名	④バージョン	⑤OSS ライセンスタイプ
1	Apache Velocity	2.0	Apache License	2.0	BSD タイプ
2					
3	Samba	3.0.x	GPL	2	GPL タイプ
4	Samba	3.2.x	GPL	3	GPL タイプ
5					
6	Apache HTTPD	2.0.48	Apache Software License	1.1	BSD タイプ
7	Apache HTTPD	2.0.65	Apache License	2.0	BSD タイプ
8					
9	MyODBC	2.50.39	public domain	-	-
10	MyODBC	3.51.01	GPL	2	GPL タイプ
11					

バージョンが必要なのは、稀なことではあるが、OSS はバージョンにより、ライセンスも変わることがあるからである。10 年以上も前の話で恐縮だが、例えば表中の 3 行目～4 行目の Samba は、バージョン 3.0.x 系までは GPL のバージョン 2 だったが、バージョン 3.2.x 系以降は GPL のバージョン 3 である。OSS のバージョン②と OSS ライセンスのバージョン④が並ぶので混乱しないように注意してほしい。表中 6 行目～7 行目の Apache HTTPD のライセンスも 2004 年に Apache License 2.0 ができてから、Apache Software License 1.1 から変更になっている。これらは、同じライセンスのバージョンの違いであるから、あまり扱い上の違いはないが条文の文章は大きく異なる。一方、表中 9 行目～10 行目の MySQL のアクセスライブラリである MyODBC は、3.51.x 系バージョンから、public domain や LGPL から現在の GPL バージョン 2 に変わった。この場合、OSS の扱いが大きく異なるので、注意しなければならない。

このように OSS のバージョンによって OSS ライセンスが大きく変わることもあるので、出荷などで再頒布するたびに、OSS の一覧表を作成し、ライセンスを確認しておく必要がある。

(2) 利用と使用の区別

一覧表を作成する際に、気をつけなければならないのは、記載しようとしている OSS が「利用」なのか「使用」なのかということである。

OSS の「利用」の例としては、以下のものが挙げられる。

- ✓ OSS を含む商品を販売・頒布する場合
- ✓ OSS を含むサービス品を頒布する場合
- ✓ AGPL の OSS を改変して用い、インターネット経由のサービスを提供する場合※。

  ※著作権法上、利用といえるか疑問だが、利用扱いの条件が指定されている。

OSS の「使用」の例としては、以下のものが挙げられる。

- ✓ OSS のツールでプログラムを開発・デバッグする場合

- ✓ OSS のツールで性能を測定する場合
- ✓ OSS のツールで保管庫に格納する場合
- ✓ OSS を用いて、インターネット経由のサービスを提供する場合。ただし、改変した AGPL の OSS の場合を除く。

図を用いて、この「使用」と「利用」の違いを説明してみる。

自分が扱う対象は、OSS A を流用して作成する、または、プログラム A そのものだとする。

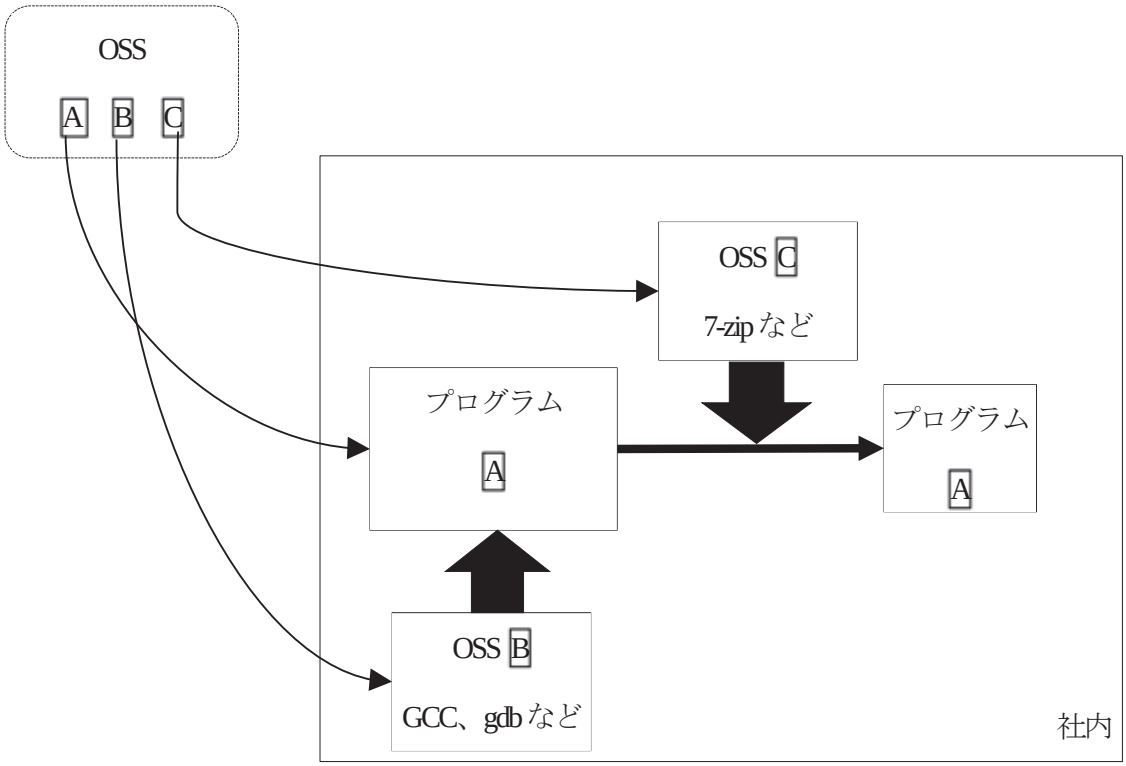


図 5-1 扱う対象プログラム A の開発-「使用」

プログラム A をコンパイル・リンク・デバッグに、GCC や gdb といった OSS B を使う。さらに、プログラム A を 7-zip などの OSS C でアーカイブし頒布しやすいようにした。この一連の社内での作業(図 5-1)のなかで、OSS A B C はどれも頒布されていないので、著作権法でいうところの「使用」であり、他人(OSS 開発者)の著作権を行使していない。

さて次に、社内での作業を終えた対象のプログラムは、社内で使用する他、商品として出荷・頒布する場合もあれば、サービス品として無料で頒布する場合もある。有償で

あれ無償であれ、プログラムの頒布は、著作権法上の「利用」に当たり、無断で利用できない。OSS の無断の利用とは、OSS ライセンスの条件を満たさないで頒布することである。無断で利用すれば著作権侵害(著作権法違反)になる(図 5-2)。

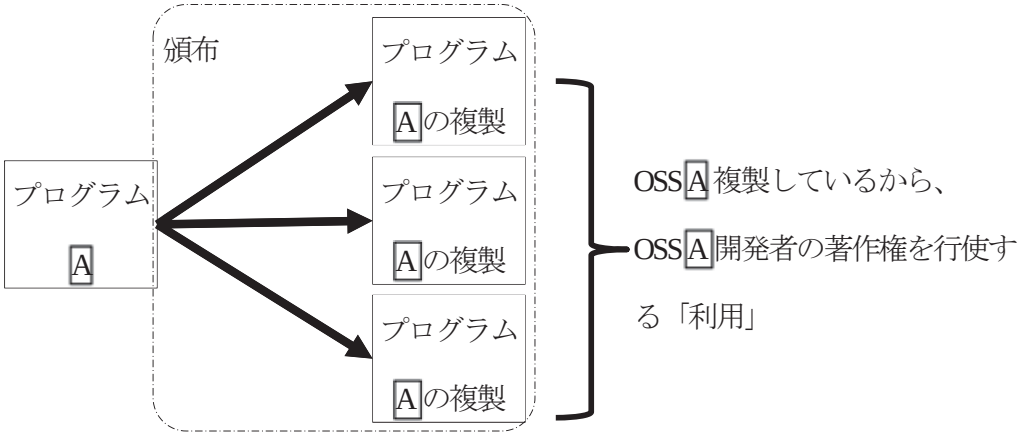


図 5-2 プログラムの頒布 - 「利用」

このように、ほとんどの OSS ライセンスは著作権の複製権の行使の許諾条件であるが、4.1 節で紹介したように改変(翻案)権に引っかけた AGPL(Affero GPL)v3 というライセンスがある[41]。プログラムの複製・頒布しないでローカルに実行すること、および、さらに改変して実行しても著作権法上の「使用」に当たる[42]が、AGPLv3 は改変権の行使に許諾条件を課した。AGPLv3 の条文は、GPLv3 の第 13 条を、改変したプログラムを実行して Web サービスなどインターネット上のサービスを提供する条件に置換したものである。Web サービス利用者(例えば、買い物利用者)は、バイナリを受領していないにも関わらず、サーバ側で動作している AGPLv3 の改変された OSS のソースコードを入手できる権利が与えられる(図 5-3)。

[41] 4.1 節で触れた MongoDB は OSD に反して OSS と言えないので、そのライセンス SSPL も OSS ライセンスと言えないので、ここでは除外する。

[42] 著作権法（翻訳、翻案等による利用）第四十七条の六により「著作権の制限」の対象になる。

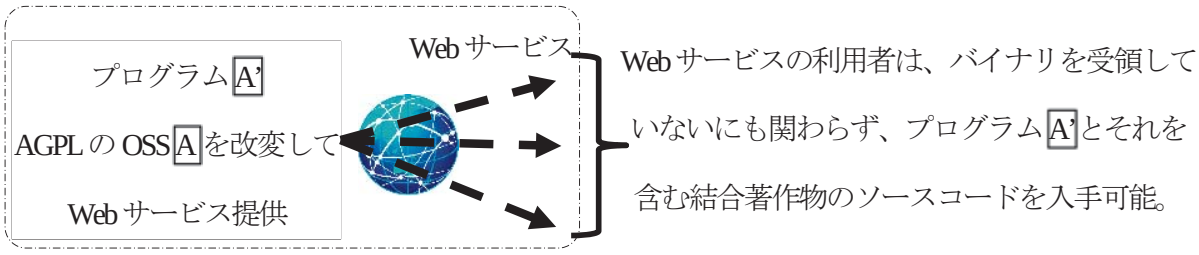


図 5-3　AGPL の OSS を改変して Web サービスに使用

AGPLv3 第 13 条で、サービス利用者にも、「対応ソースを受領する機会を無料で提供しなければならない」とあるからである。このライセンスは、筆者から見ると、以下の点で特殊である。

1. それまでに OSS ライセンスは、複製・頒布の際の許諾(ライセンス)であったが、頒布を伴わない行為に条件を付けている。
2. AGPL の主な OSS は企業製プログラムであり、AGPL で提供しているとともに、商用ライセンスでも提供している場合がほとんどである。そもそも、Affero という名称は、元となった AGPL を作成した企業名から来ている。

以前から、複製を伴わない ASP(アプリケーションサービスプロバイダー)が改変した OSS を大量に使用しているにも関わらず、OSS コミュニティに還元していないことを「ただ乗り(free rider)」と称して不満に思う者が少なからずいた。GPLv2 でこれを防げないことを GPLv2 の「抜け穴(loop hole)」と呼ぶ者がいる。GPLv3 策定時には、GPLv3 自体に取り込むことを強く主張した者も少なくなかったが、取り込まれず、別ライセンスとなった。条件が追加になって変わるのであるから妥当な判断だと筆者は考えている[43]。

しかし、AGPL のプログラムの著作者は、商用ライセンスでも販売する企業がほとんどであり、あたかも、お試し版のライセンスとして AGPL でプログラムを公開している状況とも受け取れる。その状況から、商用サービスで利用する際には、商用ライセンスを購入するのがビジネス上、妥当であろうと筆者は考える。そのため一覧表では、このケースも便宜的に OSS の「利用」の行為として扱うものとする。

[43] 逆をいえば、GPLv2 と GPLv3 の条件は見た目と違って、ほとんど変わっていないのである

5.2 OSS 利用ガイドラインの作成

OSS を上手に活用するために、社内ガイドラインを作成する企業が多い。しかし、OSS 利用のガイドラインを作成する際には、以下の 3 点に注意すべきである。

(1) 誤解を招く表現は使用しないこと

(2) ポリシーのみで終わらないこと

(3) プログラム構造のみで GPL を回避しようとしないこと

(1) 誤解を招く表現は使用しないこと

第 3 章で紹介したようにインターネット上にある OSS ライセンス、特に GPL の解説は誤解または誤解を招きやすい表現が多い。表 5-2 に誤解を招く表現の例を示す。

表 5-2 誤解を招く表現の例

	誤解を招く表現
a	ソースコードの公開
b	GPL の伝播性(ウイルス性)
c	GPL が適用される
d	ソース開示義務が発生する
e	自由に利用できるソフトウェアライセンスである

社内の OSS ライセンスのコンプライアンスを任された者が意外に無神経にこのような表現を使用する。OSS ライセンスについてよく調べている者ほど、このような表現を用いたインターネット上の不適切な解説を多く読んでいるため、不適切な表現に違和感を持たずに流用してしまうのである。始末が悪いことである。ネット上の他、IPA や SOFTTIC の報告書には、このような誤解を招く表現が満ちているため、その表現で理解したつもりになっているらしい。

少々、根拠を確認し、論理的に考えれば、誤解を招く不適切な表現でしかないことがわかるであろう。

(a) 「ソースコードの公開」という誤解を招く表現

この表現は、「ソースコードを Web で公開しなければならない」ような誤解を招く。

そのような要求は GPL にはない。

元々は、MIT の Emacs のコミュニティでの「ソースコードを共有する代わりに、改変したら、そのソースコードもコミュニティ内に公開し共通する」というルールと思われる。それをベースに、「バイナリの受領者だれにでも、ソースコードを入手できるようにする」という意味で「公開」という用語が使われていたのであろう。

しかし、それも、1985 年、GNU Emacs General Public License でリリースする際に、そういう特権的なことは間違っていたとリチャード・ストールマン氏は主張を変更している[44]。

そのような当初のルールが一人歩きしているのであろう。

(b) 「GPL の伝播性(ウイルス性)」という誤解を招く表現

この表現は、「GPL のプログラムに接触すると、あたかも GPL が感染する」かのような誤解を招く。

例えば、自身が開発したアプリケーションプログラム(AP)が一度でも GPL ライブラリをリンクした過去があるならば、AP のライセンスが GPL になるという誤解である。

著作権法上、著作者以外が頒布(複製)の許諾条件を指定する権利はないので、そのようなことは起こり得ない。GPL にもそのような記載はない。

しかし、ネット上では「GPL のプログラムを含む全体のプログラム(結合著作物)を頒布する際の GPL の条件である」ことを「感染する」と表現している情報が多い。そのため、GPL を調べている者ほど、安易に「感染する」「伝播する」という表現を使ってしまっている。

[44]自由としてのフリー(2.0)リチャード・ストールマンと自由ソフトウェア革命第九章 GPL より

条文上は、GPL ライブラリをリンクしたアプリケーションが GPL の条件を受け入れてソース開示できないならば、GPL ライブラリのプログラムを頒布できない、つまり、GPL ライブラリをリンクしたアプリケーションも頒布できないだけである。

(c) 「GPL が適用される」という誤解を招く表現

この表現は、再頒布者に「ライセンスを適用する権利があるが、GPL により強制されるルールがある」かのような誤解を招く。

著作権法上、著作者以外が頒布(複製)の許諾条件を指定する権利は無く、再頒布者にライセンスを適用する権利は無い。

世の中に誤解している人が多いが、BSD ライセンスの OSS も、受領した者が著作者に無断で、商用ライセンスや GPL に変更などできない。

GPL が適用されるのではなく、著作者が指定した GPL の条件が再頒布の条件であって、その条件を満たせないならば、再頒布できないという権利関係の話である。

(d) 「ソース開示義務が発生する」という誤解を招く表現

この表現は、製品出荷後でも「要求されれば、粛々と義務を履行して、ソース開示すればよい」という誤解を招く。

製品出荷時点で、OSS を複製・頒布してしまっているので、既に著作権侵害を犯している。「出荷すると義務が発生する」と誤解していると手遅れである。OSS ライセンスは複製・頒布となる出荷前に満たさなければならない再頒布の許諾条件である。

(e) 「自由に利用できるソフトウェアライセンスである」という誤解を招く表現

この表現は、「OSS ライセンスは(マイクロソフト社製品の EULA のような Agreement(合意)する)ライセンス契約」であるかのような誤解を招く。

特に、OSS を使ったことがない者は、シュリンクラップやクリックオン(クリッククラップ)のような合意行為により契約するものと誤解する。つまり、OSS にもイ

ンストール時や最初の起動時にプログラム使用許諾書に合意を求めるダイアログが出て、そこで合意するものと誤解しているのである。

さらに、そうやって表示されるプログラム使用許諾書の内容が、GPL などに置き換わったものが OSS ライセンスと誤解しているのである。

このようなネット上にはびこっている表現を拾って作成されたガイドラインでは、社内に誤解を広めることになる。ガイドラインを作成する際には、このような誤解を招く表現で書いてしまわないように特に注意して記述しなければならない。

(2) ポリシーのみで終わらないこと

OSS ライセンスの概要紹介とポリシーのみを記載したガイドラインは、全社方針として良いかもしれないが、開発現場で役に立つにはほど遠い。開発現場で必要なガイドラインは、表 5-3 のように、1 概要紹介、2 ポリシー(対処方針)に加えて、3 具体的な利用法は条文のどこに対応するのかの解説と、4 条文に基づく対策の例示が必要であると筆者は考える。

表 5-3 開発現場で必要なガイドラインの構成例

1	具体的な製品形態に合わせた	OSS ライセンスの概要紹介
2	具体的な OSS の利用方法での	対処方針
3	利用 OSS ライセンスの条文の解説は	具体的な利用方法に合わせて詳細に
4	対策ガイドは	(誤解を招く表現ではなく) 根拠(条文など)に基づいた対策

これを記述するためには、少なくとも、以下の 2 点を調査・整理する必要がある。

① 商品の頒布方法(流通形態)：どこで複製し頒布するのか

② 開発(予定)物件に含まれる OSS：どのような OSS を使うか

前者①は整理することができても、後者②を調査することが難しい場合がある。そういう場合には、OSS 検出ツール Black Duck などを使うのも一つの手である。日本では、NEC の他、いくつかの企業で扱っているが、ツールの出力結果の分析を支援できる企業

は少ない。NEC 以外の取り扱い企業の多くでは、ツールが適切に動作することしかサポートされないようなので注意する必要がある。

人手または OSS 検出ツール結果の分析により、利用 OSS と OSS ライセンスを特定したら、ガイドラインには、以下のような詳細な分析が記載されることが望ましい。

- どういう出荷(頒布)の仕方で
- どういうスタンスのコミュニティの
- どの OSS をどのように使うのか
- OSS のライセンス条文のどの条項に対して、どのように対処するのか

このような調査分析報告書のようなガイドラインは、出荷形態や利用 OSS の種類に依存するので汎用性は乏しい。そのため、事業部などの単位、できれば製品群ごとに作成する必要がある。同じ製品でもビジネスモデルが変われば、出荷形態つまり OSS の複製・頒布の形態が変わる。例えば、サービス提供に「使用」していた OSS であっても、そのサービス提供のシステムを外販するビジネス(いわゆる「横展開」)を始めると OSS の「利用」になる。「OSS ライセンスは、システム開発時に気を付ければ良い」と誤解している人は、この点に気がついていないことが多い。つまり、使用時は OSS ライセンス違反でないが、条件を満たさずに利用した途端に OSS ライセンス違反となることである。このようなビジネスモデルの変更に気がつく程度に製品と近いところに、OSS ライセンスを確認できる者と対応したガイドラインが必要である。

(3) プログラム構造のみで GPL を回避しようとしないこと

ポリシーとして「ソースコードの開示はしない」という企業はあると思う。その対応として、「GPL タイプのライセンスの OSS を利用しない」、「OSS 検出ツールで非使用を確認する」ならば、妥当な対応と思う。

しかし、GPL の OSS は利用するが、ソース開示しないために、以下のような根拠のない対応策を羅列したガイドラインを見かけることがある。

× GPL のライブラリでも標準インタフェースならば大丈夫。

× GPL のプログラムを呼び出すときは、中継プログラムを挟めば(ラッパーを被せれば)伝播しない(図 2-14 参照)。

× Linux 上のアプリケーションプログラムに LGPL のライブラリを挟めば、(Linux の)GPL は伝播しない。

これらは、開発者がプログラム構造をなんとか工夫すれば、GPL によるソース開示の条件を回避できると誤解したガイドラインである。インターネット上の誤解を招く表現を鵜呑みにした上で、このような根拠のない対応策に腐心したガイドラインを見かけることがある。OSS ライセンスが著作権行使の許諾の条件が記載されているという基本を理解していないのである。このようなガイドラインのレビューを依頼されることがあったが、根本的な理解が誤っているのだから改善することはほぼ困難であった。OSS ライセンスは著作権から正しく理解することが必要である。筆者が提供する「OSS 利用ガイドライン作成支援サービス」でも、「OSS ライセンスと著作権法講義」を受講済みの人を対象としているのは、そのような理由のためである。理解の共有化ができなければ、書いた意味が誤解され、延々と無意味なやりとりが繰り返されるケースが多いためである。

5.3 品質管理プロセスの改善

OSS 利用ガイドラインを作成したら、その内容に従って開発を行うわけであるが、開発部隊が大人数の場合、なかなか全員に浸透しない場合がある。その場合、内容に従って開発が行われているか確認し定着化を図る必要がある。

機能の開発という立場から見れば、OSS ライセンスの条件を満たすという行為は、余計な作業に見られがちである。展開した当初は意識していても、しだいに忘れ去られ、何もしなくなるケースもあるだろう。そうならないためには、OSS ライセンスを確認するステップを品質管理プロセスに組込み、確認が実施されていなければ出荷できないとする標準規程に改訂するのが一つの手である。

その場合、

- OSS を利用しているモジュールにおいて「OSS ライセンスの条件を満たしていることを確認する」

は当然であるが、他に 2 つのケースを考慮し、確認漏れがないように対応すべきである。

一つ目は

- OSS を利用していないモジュールであるならば「OSS 検出ツールで OSS が 0%の検出結果であることを確認する」

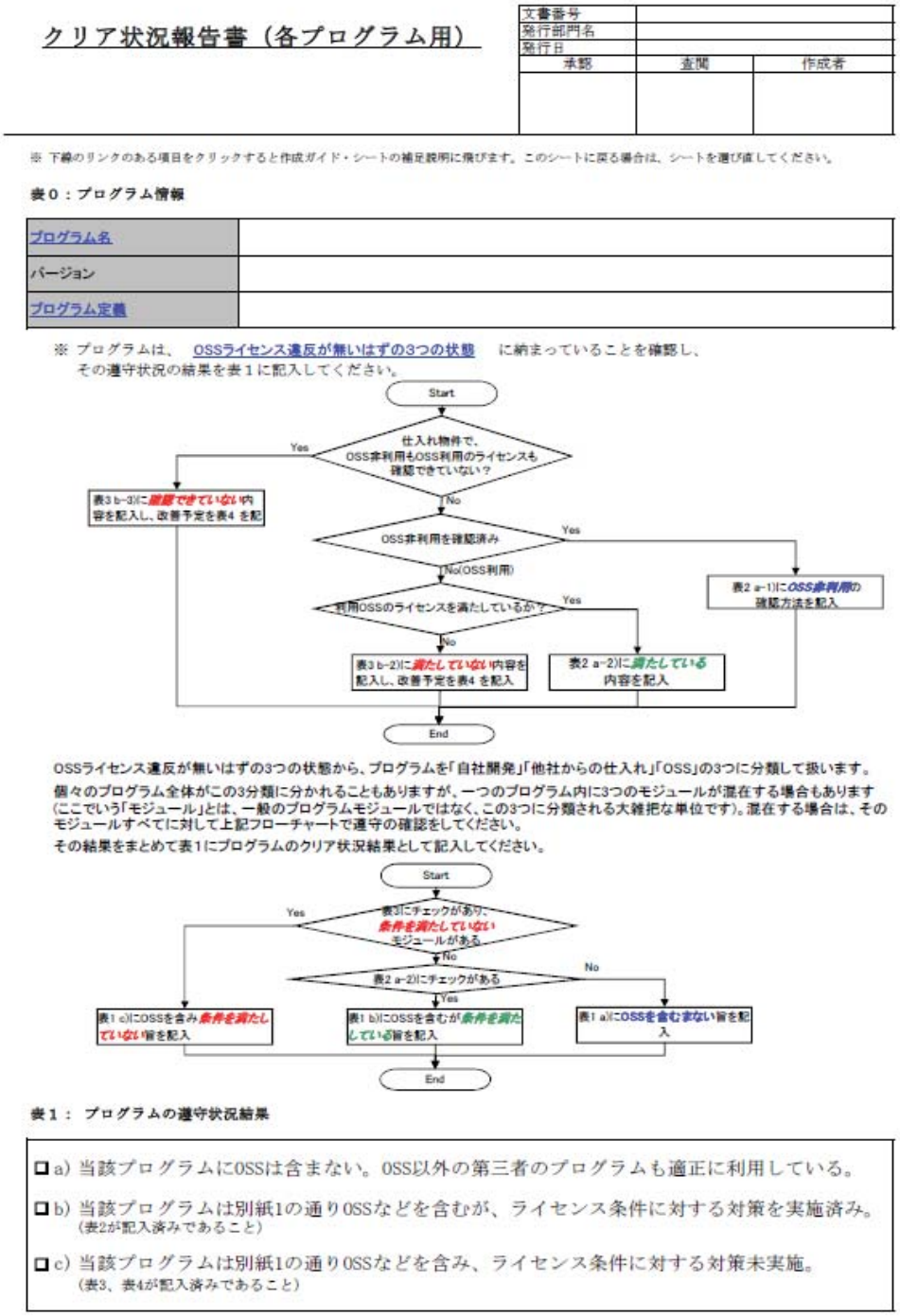
ことである。開発リーダーが「開発プログラムはすべて目を通しているから OSS は全く利用していない」と主張したとしても、確証として記録できる物を残して置くべきである。納品先のメーカによっては、確証がなければ納品を受け付けないところもある。

二つ目は、

- 外注先からバイナリで納品されたため、ソースコードで OSS の流用を確認できないならば、「外注先から上記 2 ケースどちらかの確認結果を文書で入手し確認する」

ことである。納品物件を含む製品を販売・再頒布する場合、「納品物件にソースコードが無いから確認できない」ことが、OSS ライセンス条件を満たさないことの免罪符にはならないからである。

その結果を各モジュールで OSS ライセンスのコンプライアンス問題をクリアしているという確証として「クリア状況報告書」(図 5-4)というものを記載することを提唱している。



クリア状況報告書（各プログラム用）

文書番号		
発行部門名		
発行日		
承認	査閲	作成者

※ 下線のリンクのある項目をクリックすると作成ガイド・シートの補足説明に飛びます。このシートに戻る場合は、シートを選び直してください。

表0：プログラム情報

プログラム名	
バージョン	
プログラム定義	

※ プログラムは、 OSSライセンス違反が無いはずの3つの状態 に納まっていることを確認し、
その遵守状況の結果を表１に記入してください。

OSSライセンス違反が無いはずの3つの状態から、プログラムを「自社開発」「他社からの仕入れ」「OSS」の3つに分類して扱います。
個々のプログラム全体がこの3分類に分かれることもありますが、一つのプログラム内に3つのモジュールが混在する場合もあります（ここでいう「モジュール」とは、一般のプログラムモジュールではなく、この3つに分類される大雑把な単位です）。混在する場合は、そのモジュールすべてに対して上記フローチャートで遵守の確認をしてください。
その結果をまとめて表１にプログラムのクリア状況結果として記入してください。

表１： プログラムの遵守状況結果

- ❑ a) 当該プログラムにOSSは含まない。OSS以外の第三者のプログラムも適正に利用している。
- ❑ b) 当該プログラムは別紙1の通りOSSなどを含むが、ライセンス条件に対する対策を実施済み。
  （表2が記入済みであること）
- ❑ c) 当該プログラムは別紙1の通りOSSなどを含み、ライセンス条件に対する対策未実施。
  （表3、表4が記入済みであること）

図5-4クリア状況報告書(各プログラム用)の一枚目例

さらに、各「モジュール」でクリアしていたら、「製品」としてクリアしたという「クリア状況報告書」を作成し、出荷判定会議などの資料として確証を残すという管理形態である。これら一連の流れをまとめると図5-5となる。

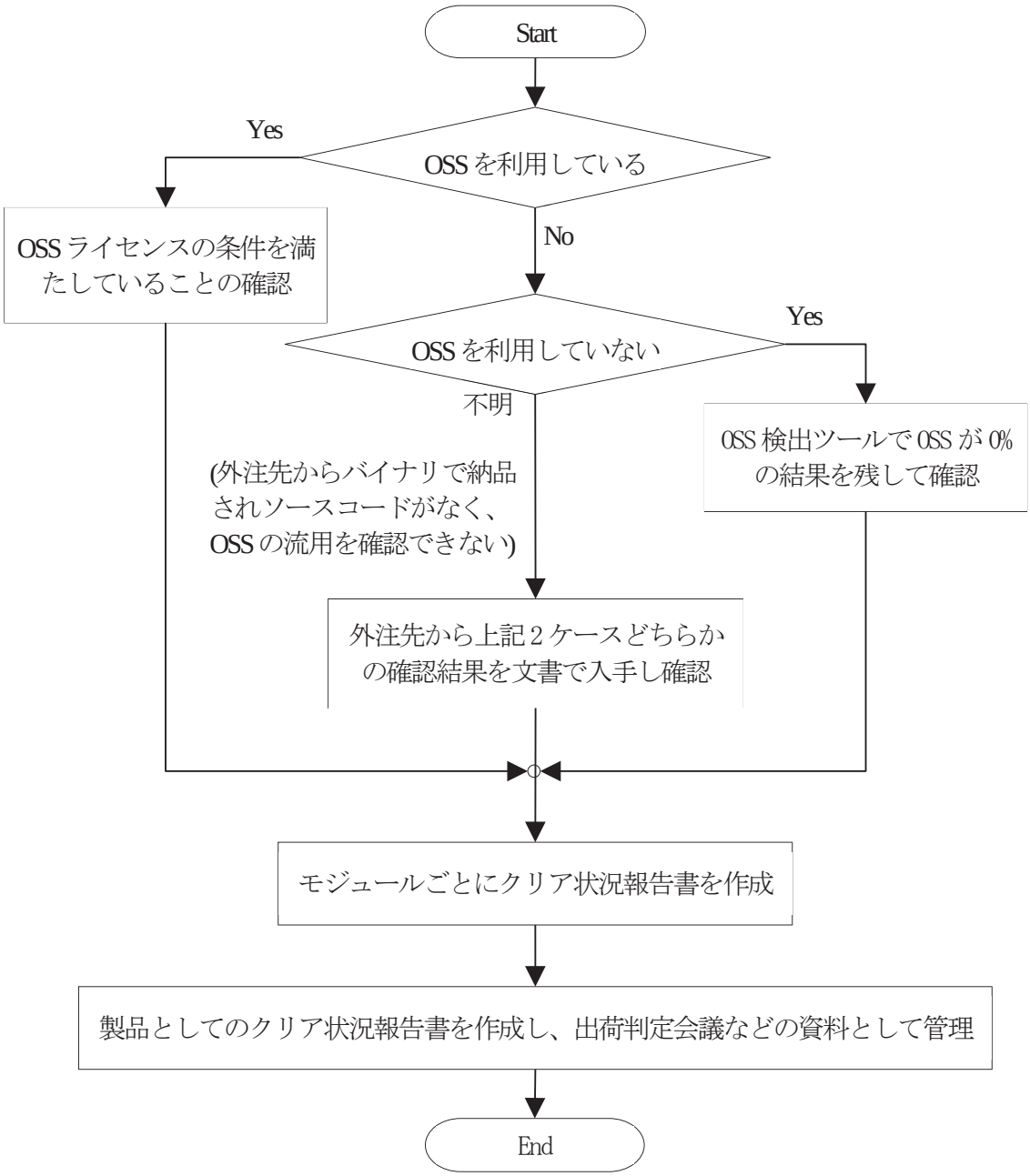


図5-5 クリア状況報告書を用いた OSS のライセンスの確認手順

例えば、このような帳票管理を取り入れた品質管理プロセス標準に改訂するのである。

ただし、余分な作業は少ないほうがよく、上記の管理フローを無条件に全社に導入するのは不適切かもしれない。商品を広く頒布する消費者向け商品は、OSS ライセンス違

反を指摘されやすいため、作業が重くても、是非、クリア状況報告書とOSS検出ツールとの併用による管理を検討すべきである。また、消費者向け製品でなくても、訴訟やライセンス違反の指摘の多い海外での利用の多い製品では、消費者向け製品に準じて管理をする企業が少なくない。

このように、商品のリスクの度合いによって、管理のさじ加減を変えることが望ましいのではないかと筆者は考える。

(1) OSSライセンスの条件を満たしていることの確認のために

OSSライセンスの条件を満たしていることの確認のために、チェックシートならぬ「問診票」(図5-6)を提供している。私がチェックシートとしていない理由が２点ある。

① 数が多い場合、毎回実施することは現実的ではないこと

② ヒアリングをしないと質問を正しく理解されないことがあること

実際に、①については、デッドコピーした回答が増え意味が無くなり、②については、例えば、GPLのOSSについて「ソース開示しているか？」という問いに「Yes」と回答していても、ヒアリングしてみると「言われれば、ソース開示するつもりだった」とソース開示条件であることを理解していないことがあったからである。このように、一度か二度、OSSライセンスの条件を正しく理解しているか確認するヒアリングのために使用するという意味で問診票と呼んでいる。

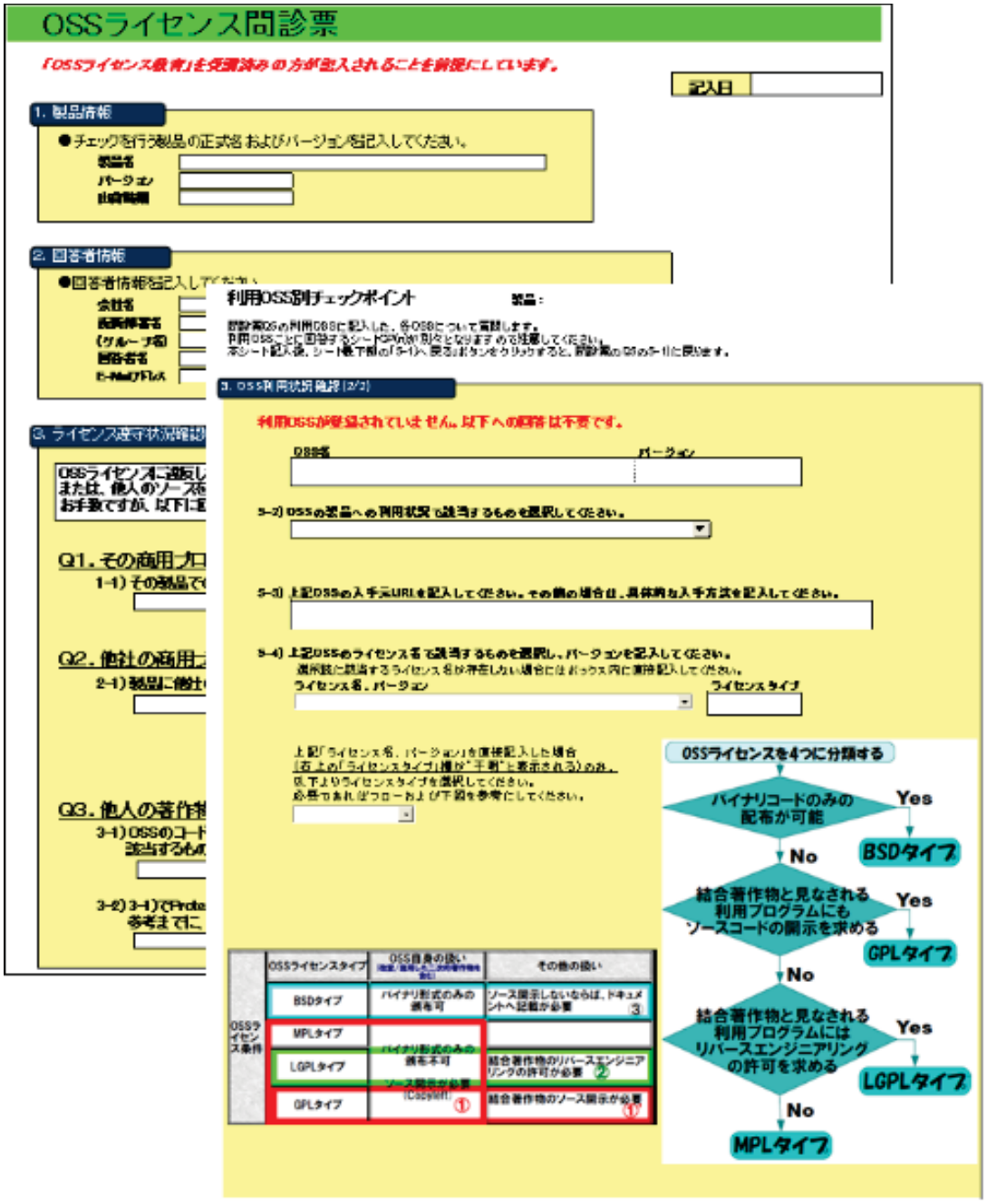


図5-6 問診票の例

(2) OSS を利用していないモジュールであることを確認するために

「OSS 検出ツールで OSS が 0%の検出結果であることを確認する」ことができれば上々であるが、以下のように検出されても利用していないと言えるケースがあり、これを考慮する必要がある。

① 偶然の一致

② 著作権で保護されない部分の流用

偶然の一致についてだが、著作権侵害には、類似性と依拠性の両方が必要である。類似していただけでは著作権侵害とならない。侵害されたとされる著作物を見たり聞いたりして真似た事実がなければ著作権侵害とならない。そのため、ツールで類似ソースコードが検出されただけでは著作権侵害とならない。あまり発生しないケースではあるが、一応、作成者に独自に創作したものか確認する必要がある。

また、類似性と依拠性の両方があったとしても、著作権で保護されない部分を流用したのであれば、著作権侵害とならない。著作権で保護されない部分、つまり、著作物に該当しないプログラムの例として以下のような物が挙げられる[45]。

- 誰が創作しても同じものとなるプログラム
- 簡単な内容をごく短い表記法によって記述したもの
- ごくありふれたもの

このことからも、「GPL の OSS から一行でも流用したら GPL にしなければならない」という流言は、著作権を正しく理解していない表現と言える。

5.4 ライセンスの設計の必要性

単純な機能設計の例として、ある機能とある機能を組み合わせて、新たな機能を実現する場合がある。OSS を利用して、そのような機能設計をする際には、ライセンス設計も必要である。つまり、機能の組合せを検討すると同じように、ライセンスの組合せも検討する必要がある。そうしないと、どのような不具合があるか以下に説明する。

(1) GPLv2 と Apache License 2.0 の関係

例えば、このような開発を想定しよう。Web 性能の強化のために、Apache HTTPD のソースコードを流用して、Linux カーネルモジュールを作成したとする。それぞれのライセンスは以下の通りである。

---

[45] パテント 2007 Vol. 60 No. 6 特集《平成 18 年度著作権委員会》井上正「プログラムの著作物性」

- Apache HTTPD：Apache License 2.0 (BSD タイプのライセンス)
- Linux カーネル：GPLv2 (GPL タイプのライセンス)

BSD タイプのライセンスの多くは、2.1 節で紹介したように、GPL タイプのライセンスで条件が包含される。しかし、Apache License 2.0 には、それまでの BSD タイプのライセンスにはない、いわゆる「特許報復条項」と呼ばれる条件などが追加されている。そのため、Apache License 2.0 の条件は、GPLv2 の条件で包含されず、図 5-7 のような関係にある。

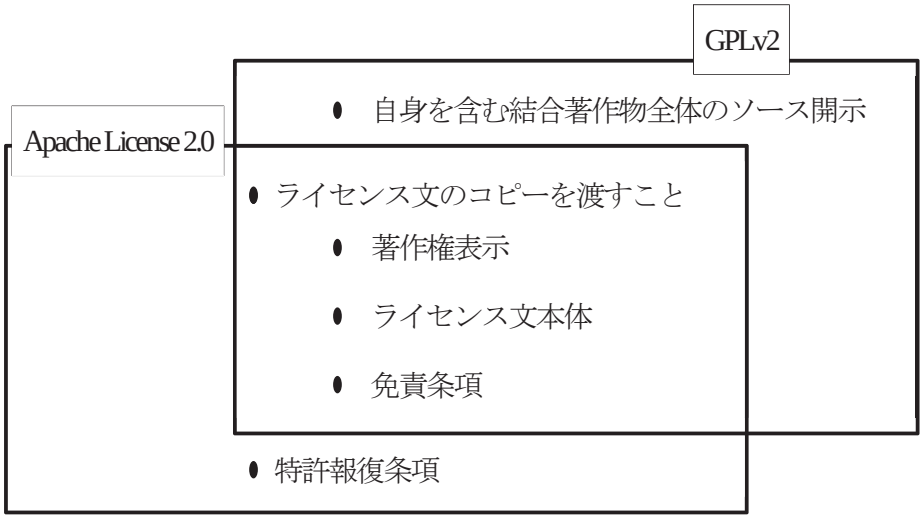


図 5-7 Apache License 2.0 の条件と GPLv2 の条件の互い包含されない関係

「特許報復条項」の内容はここでは触れないが、重要なのは、GPLv2 にはない条件が Apache Licesne 2.0 には存在し、条件が包含関係に無いということである。

このように Apache HTTPD 由来のカーネル Web モジュールを含む Linux カーネルを開発した場合、「Linux カーネル + Web モジュール」全体のライセンスは、何になるのかを次に考える。

(2) 二つのライセンスを含む全体のライセンス

「Linux カーネル + Web モジュール」全体のライセンスは、[GPLv2+特許報復条項] とする案と [GPLv2]とする案が考えられる。それぞれについて、考えてみよう。

① [GPLv2+特許報復条項] とする案の問題

GPLv2 には、Apache License 2.0 にある「特許報復条項」なるものが存在しない[46]。そのため、GPLv2 に「特許報復条項」を加えた条件で再頒布するという案である。

この条件で「Linux カーネル+Web モジュール」を再頒布した場合、Web モジュールの Apache HTTD については、Apache License 2.0 の条件が含まれているので問題ない。一方、Linux カーネルについては、GPLv2 の「2.b)その全体をこの許諾書の条件に従って」や「6.これ以上他のいかなる制限も課してはならない」という条件を満たしていない。「特許報復条項」が加えられた分、GPLv2 の条件からはみ出るし、追加の条件を課していることにもなるからである。

したがって、この条件で再頒布すると GPL の条件を満たしておらず、Linux カーネルの著作者の著作権を侵害することになる。

② [GPLv2]とする案の問題点

多くの人が「GPL とリンクすると GPL になる」と誤解していることから、よく見かける方法で、全体を GPLv2 の条件で再頒布する案である。

この条件で「Linux カーネル+Web モジュール」を再頒布した場合、Linux カーネルについては、もちろん問題ない。一方、Web モジュールの Apache HTTD については、Apache License 2.0 の条件の一つである「特許報復条項」がない条件で再頒布することになる。つまり、条件を満たせていない。

したがって、この条件で再頒布すると Apache License の条件を満たしておらず、Apache HTTD の著作者の著作権を侵害することになる。

このように、異なる OSS ライセンスの OSS を結合し、再頒布しようとする際、それぞれの OSS ライセンス条件をどうしても満たせない組み合わせがある。どうしても矛盾する。これを、2 つの OSS ライセンスが両立しない(incompatible)という。

互換性か両立性か

[46] GPLv3 には追加された

‘compatible’を「互換性」と訳すことが多いが、辞書を引くと、もう一つ、「両立性」という意味がある。この異なる OSS ライセンスの OSS の結合の問題の場合、意味からして、ライセンス条件を差し替え可能かの問題ではない。したがって、一般に広まっている「互換性がない」という表現は誤訳である。

例えば、GPLv2 と「両立性のある」FreeBSD Copyright のライセンスを「互換性がある」と表現すると、「FreeBSD Copyright のライセンスは、GPLv2 に差し替えることができる」という意味になる。そのため実際に、「FreeBSD もソースコードの開示が必須なのか」と誤解する人がいた。

両立性は、別々のものが混在してもそれぞれ成立することであり、差し替え可能なことではない。

結局、「Linux カーネル＋Web モジュール」の組合せの開発をしたとしても、全体に設定できるライセンス条件が存在しない。どちらかの条件で複製・頒布する販売をすると、たとえ GPL の条件ですべてのソースコードを開示しても著作権侵害となる。ライセンス設計をせずに開発を進めると、販売できない製品を開発することになり、無駄な開発をしてしまうことになる。

このような事態を避けるために、機能設計とともに、それらのライセンスの組合せが可能なのか(両立するか)を検討するライセンスの設計が必要なのである。