



OPENCHAIN
カリキュラム

FOSSトレーニング参考資料 第2版 最終ドラフト
[OpenChain仕様書 第1.0版対応](#)

本資料は [Creative Commons CC0 1.0 Universal](#) ライセンスの下でリリースされています。

本スライドは米国法令に準じています。米国外では法的要件事項が異なる場合がありますので
コンプライアンス トレーニング プログラムで本スライドを使う際にはこの点を考慮する必要があります。

本スライドは法的助言を提供するものではありません。These slides do not contain legal advice

OpenChain カリキュラムのスライドへようこそ。本スライドは、組織内チームに向けたFOSSコンプライアンスについてトレーニングを実施する際に、トレーニングの補助として使い、OpenChainの仕様への適合を達成するために用いることができます。

このスライドは、半日のトレーニング セッションとして提供することができます。また、各章を分割し、個別のモジュールとして提供することも可能です。各章には質問形式の「理解度チェック」のスライドを設けており、回答はスライドのノート欄に記載されています。これらの質問と回答はFOSSコンプライアンスの組織内テストの素材として使うことができます。

Welcome to the OpenChain Curriculum Slides. These slides can be used to help train internal teams about FOSS compliance issues and to conform with the OpenChain Specification.

You can deliver these slides as one half-day training session or you can deliver each chapter as a separate module. Please note that each chapter has “Check Your Understanding” slides with questions and answers in the slide notes. These can be used as the basis for in-house tests for FOSS compliance.

Disclaimer（免責事項）

- ・本文書は、The Linux Foundation におけるOpenChain プロジェクトの英文ドキュメント「OpenChain Curriculum Release 2」の公式翻訳版となります。ただし、翻訳版と英語版との間で何らかの意味の違いがある場合には、英語版が優先されます。
- ・また、OpenChain は世界のメンバー企業が参加しているプロジェクトですが、資料の細部について必ずしも各国の法令に対応していない可能性があります。本翻訳版を日本で活用する際には、各企業の法務部門を加えた検討が不可欠です。
- ・This is an official translation from the OpenChain Project. It has been translated from the original English text. In the event there is confusion between a translation and the English version, The English text shall take precedence.

コンテンツ

1. 知的財産とは何か？
2. FOSSライセンス概論
3. FOSSコンプライアンス概論
4. FOSSレビューにおけるソフトウェアの重要概念
5. FOSSレビューの実施
6. コンプライアンスマネジメントの始めから終わりまで（プロセス例）
7. コンプライアンスでの落とし穴とその回避

このスライドは、単発での3時間トレーニング セッション、もしくは短めのセッションに分け章単位で重点を置いたトレーニングとして実施する場合において、その進め方の説明に用います。

This slide is relevant to providing either a single three hour training session or explaining how a series of shorter sessions focused on “per chapter” training will work.

FOSS ポリシー

- ・<<本スライドは、FOSSポリシーが企業内のどこに置かれているかを周知するためにご使用ください（OpenChain仕様書1.0の1.1.1項）>>

このスライドで、企業が、社内文書として内部FOSSポリシーがどこにあるかを示すことができるようになっています。

This slide is intended to help a company identify where their internal FOSS policy is located in the company documentation.

第1章

知的財産とは何か？

本章では知的財産の「全体像」に焦点を当てていきます。著作権法、特許法、商標法の基礎について明確に理解していない可能性のあるマネージャーや開発者に有用でしょう。

This chapter is focused on the “big picture” of Intellectual Property. This chapter is probably most useful for managers or developers who might not understand clearly the fundamentals of copyright, patent and trademark law.

“知的財産”とは何か？

- ・著作権（コピーライト）：作者が著した原作を保護する
 - （根底のアイデアではなく）表現を保護
 - ソフトウェア、書物、音響・映像作品、半導体マスク（回路パターンの原版）
- ・特許権（パテント）：新規性、有用性、非自明性を持つ発明
 - イノベーションを奨励するための限定的な独占権
- ・営業秘密：価値ある機密情報を保護する
- ・商標：（言葉、ロゴ、標語、色などの）製品の出所を識別する標識を保護する
 - 消費者とブランドを保護；消費者の混乱やブランドの希薄化を回避する

本章では、FOSSコンプライアンスに最も関係する、
著作権と特許権に焦点を当てる

ここにある概説で知的財産のすべての側面を網羅することは意図していません。ここで意図しているのは、「全体像」の観点から、当カリキュラムで議論するのがFOSSコンプライアンスに最も関係する著作権と特許権だということを確認してもらうことです。

This overview is not intended to cover all aspects of Intellectual Property. It is intended to provide context for the “big picture” and to establish that today we are only discussing copyright and patents, the areas most relevant to FOSS compliance.

ソフトウェアにおける著作権の概念

- ・基本ルール＝著作権は独創的作品を保護する
- ・一般的に著作権は、書物、動画、絵画、音楽、地図などの著作物に適用される
- ・ソフトウェアは、著作権によって保護される。（特許権で保護される）
「機能」ではなく、「表現」（実装の細部における独創性）が保護される
- ・その作品の著作権保有者は、自らが創作した作品だけをコントロールでき、他の誰かの独立した創作物はコントロールできない

このスライドでは、ソフトウェアの著作権についての“全体像”を説明しています。

This slide explains the “big picture” of copyright in software.

著作権の中でソフトウェアに最も関係する「権利」

- ・ソフトウェアを「複製」する権利 - コピーを作成することができる
- ・「派生的著作物※」を作る権利 - 修正を加えることができる
 - 派生的著作物という用語は、原作に基づいてはいるものの、その新しい著作物が原作のコピーではなく、原作に対し独自に創造的な作業が加えられたと主張できるレベルの作品を言う（この用語は米国法令に基づいているの要留意）
- ・「頒布」する権利
 - 頒布とは、一般的に、ソフトウェア部品のコピーをバイナリまたはソースコードの形態で他のエンティティ（個人や外部の企業・組織）に提供する行為とみなされる

注：何をもって「派生的著作物」、「頒布」とするかの解釈はFOSSコミュニティの中ににおいても、関連した法務関係者の間においても議論の対象となっている

※日本の著作権法の「二次的著作物」に該当

このスライドでは、ソフトウェアに対する著作権法の最重要部分を明確にしています。

This slide clarifies the most important parts of copyright law to software.

ソフトウェアにおける特許の概念

- ・特許は、機能を保護する—これには、コンピューター プログラムのような演算方法が含まれる
 - 抽象的なアイデアや自然法則は保護しない
- ・特許保有者は、他者の独立した創作であっても、あらゆる人に対しその機能の使用を停止させることができる
- ・他者がその技術を使いたい場合、特許ライセンス（その技術の使用、製造、製造委託、販売、販売の提示、および輸入に関する権利※の許諾）を求めることができる

※それぞれ英文で、rights to 「use」, 「make」, 「have made」, 「sell」, 「offer for sale」, and 「import」

このスライドでは、ソフトウェアに関連する 特許の概念を説明しています。

This slide explains patent concepts relevant to software.

ライセンス

- ・「ライセンス」は、著作権や特許の保有者が他者に対し許諾や権利を与える手法
- ・ライセンスは以下に対し制約を課すことができる
 - 許可される使用形態（頒布、派生的著作物の作成、製造、製造委託、大量生産）
 - 独占的、または非独占的な許諾条件
 - 地理的な範囲
 - 無期限か、期限付きか
- ・ライセンスはその許諾に条件を持たせることができる。すなわち何らかの義務を満たした場合にのみ、そのライセンスを得る
 - 例）帰属情報を提供する、互恵的ライセンスを供与する
- ・保証、免責、サポート、アップグレード、保守に関する契約事項も含まれる場合がある

この スライドは、「ライセンス」とは何かを説明しています。それは米国法令下の「契約」とは異なっています。ここではライセンスの中にどういったものがあるか、その境界を説明しています。

This slide explains what is a “license.” This is different to a contract under US law. This slide explains the boundaries of what can be in a license.

理解度チェック

- ・著作権法はどのようなものを保護しますか？
- ・ソフトウェアにとって最も重要なのは著作権のどのような権利ですか？
- ・ソフトウェアは特許の対象になりますか？
- ・特許はその保有者に対しどういった権利を付与しますか？
- ・単独で自分のソフトウェアを開発した場合でも、そのソフトウェアについて第三者から著作権ライセンスを受ける必要がある可能性がありますか？特許の場合は？

著作権は原作者の独創的な作品を保護します。著作権がアイデアの表現を保護するのに対し、特許は根底にあるアイデアそのものを保護している点で異なります。原作者の作品としては、写真、歌、コンピューターのプログラムコードなどが含まれます。

ソフトウェアの著作権で重要なのは：複製する権利、派生的著作物を作成する（もしくは改変する）権利、および頒布する権利です。

ソフトウェアは特許を受けることができます。特許はコンピューター プログラムのような演算方法を保護します。ただし、特許は機能を保護するもので、抽象的なアイデアは保護しません。

特許保有者は他者の製品が独立に創出されたかどうかに関係なく、他者が特許を実施することを排除できます。

ソフトウェアを独立して開発した場合、それが独立開発であり、問題とされている著作権付きソフトウェア コードにアクセスしなかったことを示すことができれば、著作権のライセンスは不要である可能性が高いと考えられます。ただし、誰もがその著作権付きのソフトウェア コードを知っており、あなたがアクセスできたと考えることが合理的だとすると、これを証明すること

は困難となります。ソフトウェアについて特定の特許が主張する機能を読み取れる場合には、そのソフトウェアが独立開発かどうかには関係なく特許ライセンスが必要となるでしょう。そのような例の1つとしてFFMpeg があります。これはビデオの符号化／復号化のためのコーデック機能を提供するフリー ソフトウェア プロジェクトです。しかしながら、何らかのフォーマットを符号化／復号化をするには特許ライセンスが必要となります。

Copyright protects original works of authorship. It's different than patent in that copyright protects the expression of an idea, whereas patent protects the underlying idea itself. Examples of works of authorship include photographs, songs, and computer code.

Most important copyright concepts for software are: right to reproduce, right to make creative works (or right to modify), and right to distribute.

Software can be subject to a patent. Patent protects method of operation, such as computer program. However, patent protects functionality, and not abstract ideas.

Patent holder can exclude others from practicing the patent, regardless of whether the others have independently created the product.

If you have independently developed your own software, then you may not need a copyright license if you can show the independent development and you had no access to the copyrighted work in question. This is difficult if the copyrighted work is popular such that it'd be reasonable to assume that you had access. If your software reads on a patent, then you will need a patent license regardless of whether you've independently developed the software. An example of this would be FFMpeg, which is a free software project that provides the codecs for encoding and decoding videos. However, you would still need a patent license to encode and decode a certain format.

第2章

FOSSライセンス概論

本章は、FOSSライセンスに馴染みが薄い法務関係者やマネージャー、もしくは開発の方にとって有用です。

This chapter is useful for lawyers, managers or developers who may not be familiar with FOSS licenses.

FOSS（フリー／オープンソース ソフトウェア）ライセンス



- FOSSソフトウェアのライセンスは、一般的に改変と再頒布を許容する条件の下で、ソースコードの入手が可能となっている
- FOSSライセンスには、帰属情報の提供や著作権宣言文の保持、もしくはソースコードの入手を書面で申し出ること^{*}に関する条件を有する場合がある
- 代表的なライセンスは、オープンソース イニシアチブ (OSI) がそのFOSS定義 (OSD)に基づいて承認した一連のライセンス。OSIが承認したライセンスの全リストは、以下のページを参照：
<http://www.opensource.org/licenses/>

※「書面による申し出(Written offer)」と言われる

このスライドでは、FOSSライセンスがどういったことをするかの「全体像」を提供します。またここでは、FOSSライセンスについてさらに多くを調べるために情報源についても説明しています。

This slide provides the “big picture” about what FOSS licenses do. It also explains a resource where you can find out more about some FOSS licenses

パーミッシブ（寛容）なFOSSライセンス

- ・パーミッシブなFOSSライセンス – 制約が最も少ないFOSSライセンスについて言及する時に用いられる用語
- ・例：3条項BSDライセンス
 - BSDライセンスは、著作権表示と同ライセンスの保証に関する免責事項が維持される限り、いかなる目的においても制限ない再頒布を許容するパーミッシブなライセンスの一例
 - このライセンスは派生製品の宣伝に許可なく貢献者の名前を使用することを制限する条項を含んでいる
- ・その他の例：MITライセンス、Apache-2.0ライセンス

本スライドでは、FOSSライセンスの最も基本的なタイプであり、ライセンス上の要求が最も少ない「パーミッシブ」FOSSライセンスについて説明しています。最も基本的な要求は 著作権表示を含めることです。

This slide explains "permissive" FOSS licenses, the most basic type of FOSS license, which usually have minimal requirements. The most basic requirement is to include a copyright notice.

ライセンスの互恵性とコピーレフトライセンス



- ・ライセンスの中には、派生的著作物（同じファイル、同じプログラム、あるいは他のバウンダリにあるソフトウェア）を原作と同一の条件で再頒布することを要求するものがある
- ・これは、「コピーレフト」、「互恵的」、あるいは「遺伝的」効果と言及される
- ・GPL version 2.0よりライセンス互恵性の例：
「『プログラム』またはその一部を含む著作物、あるいは『プログラム』かその一部から派生した著作物を頒布あるいは発表する場合には、その 全体をこの契約書の条件に従って第三者へ無償で利用許諾しなければならない。」
- ・互恵性やコピーレフトの条項を組み入れたライセンスとして、GPL、LGPL、AGPL、MPL、およびCDDLのすべてのバージョンが挙げられる
- ・コピーレフト ライセンスは、ソースコードが入手できる状態にあることを義務付ける場合がある

このスライドでは、パーミッシブ ライセンスよりも強い要求事項を持つ、より複雑なタイプのFOSSライセンスとして、互恵性と「コピーレフト」について説明しています。これらは、「原作」と「派生的著作物」を原作と同じ条件の下で頒布することを要求します。

This slide explains reciprocity and Copyleft, a more complex type of FOSS license that have additional requirements above permissive licenses. They require distribution of the original work and derivative works under the same terms as the original work.

プロプライエタリライセンス、 もしくはクローズド ソース ライセンス

- ・プロプライエタリ ソフトウェア ライセンス（もしくは商用ライセンス、もしくはEULA）は、ソフトウェアの使用、改変、もしくは再頒布についての制約を有する
- ・プロプライエタリ ライセンスは、多くの場合、金銭の支払いやライセンス料を伴う
- ・プロプライエタリ ライセンスは、ベンダーごとの独自性がある — 存在するベンダー数と同じバリエーションのプロプライエタリ ライセンスがあり、それぞれを個別に評価しなければならない
- ・FOSSの開発者たちは、通常、「プロプライエタリ」という用語をFOSSでない商用のライセンスを言い表す際に用いるが、FOSSライセンスもプロプライエタリ ライセンスも、知的財産をベースにしたものであり、どちらもそのソフトウェア資産にライセンスを付与したもの

このスライドでは、プロプライエタリ ライセンスもしくはクローズド ソース ライセンスについて説明しています。これらのライセンスをFOSSライセンスと比較すると、多くの場合、要件やルールに大きな相違があります。

This slide explains proprietary or closed source licenses. These licenses often have very different requirements and rules compared to FOSS licenses.

その他のライセンス

- ・フリーウェアープロプライエタリ ライセンスの下で、無料または非常に低いコストで頒布されるソフトウェア
 - ソースコードが入手できるものもあれば、できないものもあり、派生的著作物の作成について、一般的には制限される
 - フリーウェアのソフトウェアは、通常すべての機能が使え（機能制約がない）、制限なく使える（使用日数の制約がない）
 - フリーウェアのソフトウェアは、使用タイプ（個人使用、商業目的、学術目的など）についての制約や、ソフトウェアのコピー、頒布、派生的著作物の作成についての制約を課す
- ・シェアウェアー基本的に試用を前提に、無料で、期間・機能を限定して使用者に提供されるプロプライエタリ ソフトウェア
 - シェアウェアの目的は、将来の購買者がその有用性を評価できるよう、完全版ライセンスの購入前にプログラムを試用する機会を提供すること
 - 大半の企業は、シェアウェアを非常に警戒する。なぜならシェアウェア ベンダーは、そのソフトウェアが組織内で自由に広まってしまった後で、高額なライセンス料の支払いを迫ることがしばしばあるため
- ・フリーウェアとシェアウェアは、FOSSではない

他のタイプのライセンスも使われます。これらは時としてFOSSと混同されることがあります、その要求事項は実質的に異なります。フリーウェアおよびシェアウェアのライセンスは、FOSSライセンスと同じもの、もしくは互換性があるものとみなすべきではありません。

There are other types of license used. Sometimes these are confused with FOSS but their requirements are actually different. Freeware or Shareware licensing should not be regarded as the same or compatible with FOSS licensing.

パブリック ドメイン

- ・パブリック ドメインという用語は、法令で保護されない知的財産を意味する。したがって、パブリック ドメインのものについては、ライセンスを求めずに誰でも使用できる
- ・開発者は自身のソフトウェアに対し「パブリック ドメイン宣言」を行うことができる
 - ▶例) 「本ソフトウェアのすべてのコードと文書類は著作者によりパブリック ドメインに供されました」
 - ▶パブリック ドメイン宣言は、FOSSライセンスと同じものではない
- ・パブリック ドメイン宣言とは、開発者がそのソフトウェアに対し保有できるあらゆる知的財産権を放棄もしくは消滅させ、制約なくそのソフトウェアが使用できることを明示する試みだが、この宣言の執行可能性については、FOSSコミュニティにおいて議論の対象となる
- ・パブリック ドメイン宣言は、保証免責条項のような他の条項を伴うことが多い。その場合、そのソフトウェアは、パブリック ドメインというより、あるライセンスの下にあるとみなすことができる

このスライドではパブリック ドメインについて説明しています。ソフトウェア作品に対しそれがいかなる制約もないことを意味する公開方法の1つと言えます。米国ではパブリック ドメイン ソフトウェアもFOSSに含まれる可能性がありますが、すべての国々がその存在を認識したり、パブリック ドメインの下に原作者であることを放棄したりすることを許容するわけではないことに留意しなければなりません。ドイツがその一例です。

This slide explains public domain, a type of release that means the work is released without any restrictions whatsoever by the authors. In the US public domain software can be included in FOSS code, but it should be noted that not all legal jurisdictions recognize the existence or permit the release of authorship under public domain. Germany is one example.

ライセンスの両立性（互換性）※

- ・ライセンス両立性（互換性）は、（異なるライセンス間で）ライセンス条項に矛盾がないことを確かなものにするプロセス
- ・1つのライセンスが何かすることを要求し、他方のライセンスがそうすることを禁じている場合、それらは矛盾する。その2つのソフトウェア モジュールの組み合わせがライセンスの下での義務を発動させる場合には、2つのライセンスは両立しない（互換ではない）
 - GPL-2.0とEPL-1.0はそれぞれ、頒布される「派生的著作物」に対し義務を拡張している
 - GPL-2.0のモジュールが、EPL-1.0のモジュールに結合（Combine）され、統合されたモジュールが頒布される場合、そのモジュールは：
 - ✓ (GPL-2.0によれば) GPL-2.0のみで頒布されなければならないことになる、さらに
 - ✓ (EPL-1.0によれば) EPL-1.0のみで頒布されなければならないことになる。
 - ✓ 頒布者は2つの条件を同時に満足することはできないので、このモジュールは頒布できない
 - ✓ 上記はライセンスが両立しない1つの例

「派生的著作物」の定義はFOSSコミュニティでもその見解が分かれる傾向にある

※FOSSライセンスに係る「Compatibility」の日本語訳として「両立性」、「互換性」2つの方向性があるため併記し

このスライドではライセンスの両立性（互換性）について説明しています。両立性（互換性）は、どのライセンスが一緒に使用できるかを理解するまでの考え方です。FOSSにはお互いに両立（互換）できるもの、できないものがあります。コードやライセンスを選択する際にこれは重要な検討事項となります。

--
This slide explains license compatibility, the way of understanding what licenses can be used together. Some FOSS licenses are compatible with each other. Some are incompatible. This is an important consideration when choosing code and choosing licenses.

告知／表示

告知／表示（Notice）は、しばしば著作者やライセンスに関する情報を提供する。たとえばファイル先頭のコメント行文字列などの形がある。また、FOSSライセンスでは、ソースコードや文書類の一定の場所に告知／表示を設定することを要求する場合がある。これは著者の功績を称えたり（帰属情報）、そのソフトウェアが改変されたことを明確にさせたりするためである。

- ・著作権表示（Copyright notice）－その著作物の著作権保有者を世に知らしめるべく、ソフトウェアの複写物に掲載される識別子のこと。
例：Copyright © A. Person (2016).
- ・ライセンス告知（License notice）－その製品に含まれるFOSSのライセンス条項や条件を知らせる表示。
- ・帰属表示（Attribution notice）－出荷製品に含まれる表示であり、製品内のFOSSの原作者が誰であるかを知らせる。
- ・改変告知（Modification notice）－ファイルのソースコードに対して改変を実施したという告知。たとえばファイルの上部に著作権表示を加える、など。

このスライドでは、告知／表示（Notice）について説明しています。これは、（FOSSソースコード）ファイル内のコメント文字列（テキスト）によって、著作者やライセンスについて説明するもので、多くの場合、（FOSSソースコード）ファイルに適用されるライセンスを知る最も重要な方法として認識されています。

--
This slide explains license compatibility, the way of understanding what licenses can be used together. Some FOSS licenses are compatible with each other. Some are incompatible. This is an important consideration when choosing code and choosing licenses.

マルチライセンス

- ・マルチライセンスとは、複数の異なるライセンス条件の下でソフトウェアを頒布する手法
 - ▶ 例：ソフトウェアが「デュアルライセンス」である場合、受領者はそのソフトウェアの使用や頒布に際し、2つのライセンスのどちらかを選択できる
- ・注：ライセンサ（ライセンス供与者）が複数のライセンスを課す手法と混同しないこと。そのような場合には、すべてのライセンス要求を満たさなければならぬ

このスライドは マルチライセンスについて説明しています。これは、2つ以上のライセンス条件がソフトウェアに適用される状況です。

結合的 (Conjunctive) = 複数のライセンスを適用します。

GPL-2.0 プロジェクトはBSD三条項ライセンス下のコードも含みます。

この状況においては両方の条項を満たさなければいけません。

離接的 (Disjunctive) = 複数のオープンソース ライセンスから1つのライセンスを選択します。

Mozilla 3ライセンス (tri-license)

Jetty

Ruby

離接的なライセンスは、FOSSポリシーを策定する際により深く調査すべき重要な事柄となることがあります。

離接的なライセンスの下では、ライセンスを選択することができます。たとえば、GPLとよりパーミッシブなライセンスが選択肢にあった場合、ライセンスの両立性と要件を十分検討した上で、どちらのライセンスで頒布するかを選択できます。

プロジェクトが離接的なライセンスを設定しても、時として、あなたが利用しようとしたコードには1つのライセンスだけが設定されている場合もあります。おそらく、そのコードの作成者が、この選択をすでに実施してしまっているのかもしれません。使いたくないライセンスが選択されていた場合は、原作品の著作権保有者が誰かを明確にし、そこから直接コード入手すべきかどうかを検討しなければなりません。

例)

MPL 1.1/GPL 2.0/LGPL 2.1 --

「本ファイルの内容は Mozilla Public License Version 1.1（「本ライセンス」）の適用を受けます。本ライセンスに従わない限り本ファイルを使用することはできません。

このファイルの内容は、上記に代えて、GNU General Public License Version 2 以降のライセンス（「GPL」ライセンス）、もしくは - GNU Lesser General Public License Version 2.1以降のライセンス（「LGPL」ライセンス）の条件に従って使用することも可能です。この場合、このファイルの使用には上記の条項ではなく GPLもしくはLGPL ライセンスの条項が適用されます。

このファイルの他者による使用をGPLもしくはLGPLライセンスの条件によってのみ許可し、MPLによる使用を許可したくない対象者は、上記の条項を削除することでその意思を示し、上記条項をGPLもしくはLGPLライセンスで義務付けられている告知およびその他の条項に置き換えてください。対象者が上記の条項を削除しない場合、受領者はMPLまたはGPLもしくはLGPLライセンスのいずれによってもこのファイルを使用することができます。」

「デュアル（Dual）」=ここで記述したすべての状況で使われうる、混乱を招く用語ですが、通常この用語はOSSライセンスもしくは商用ライセンスの選択に関するビジネスモデルについて言及しています。ビジネスモデルとしてのデュアル ライセンスについての詳細は、こちらを参照してください：
<http://oss-watch.ac.uk/resources/duallicensing>

This slide explains multi-licensing. This is the situation where more than one set of license terms can apply to a piece of software.

Conjunctive = Multiple licenses apply

GPL-2.0 project also includes code under BSD-3-Clause

In this situation you have to comply with both sets of license terms

Disjunctive = Choice of one open source license or another

Mozilla tri-license

Jetty

Ruby

Disjunctive licensing may be something important to explore more deeply when creating a FOSS policy.

Under disjunctive licensing you have a choice of licensing, i.e. GPL and a more permissive license option, you may choose which license you are going to distribute under depending on license compatibility, license requirements. Sometimes a project has a disjunctive licensing situation, but only one license is included in your code – so perhaps the person you got the code from already made this choice. If they choose the license you weren't going to use, now you might have to consider if you should figure out who the original © holder is and get the code directly from them

Example:

MPL 1.1/GPL 2.0/LGPL 2.1 - -

"The contents of this file are subject to the Mozilla Public License Version - 1.1 (the "License"); you may not use this file except in compliance with - the License.

...

Alternatively, the contents of this file may be used under the terms of - either the GNU General Public License Version 2 or later (the "GPL"), or - the GNU Lesser General Public License Version 2.1 or later (the "LGPL"), - in which case the provisions of the GPL or the LGPL are applicable instead - of those above.

If you wish to allow use of your version of this file only - under the terms of either the GPL or the LGPL, and not to allow others to - use your version of this file under the terms of the MPL, indicate your - decision by deleting the provisions above and replace them with the notice - and other provisions required by the LGPL or the GPL. If you do not delete - the provisions above, a recipient may use your version of this file under - the terms of any one of the MPL, the GPL or the LGPL. "

"dual" = confusing term that may be used for any of these situations, but usually refers to business model of OSS license or commercial license choice
For more on dual-licensing as a business model: <http://oss-watch.ac.uk/resources/duallicence2>

理解度チェック

- ・FOSSライセンスとはどういったものでしょうか？
- ・パーミッシブなFOSSライセンスの典型的な義務としてどういったものがありますか？
- ・パーミッシブなライセンスの名前をいくつか挙げてください。
- ・ライセンスの互恵性とはどういったことを意味していますか？
- ・コピーレフトの形態をとるライセンスの名称をいくつか挙げてください。
- ・コピーレフト ライセンスの下で使用されるコードについては何が頒布される必要がありますか？
- ・フリーソフトウェアとシェアウェアはFOSSとみなされますか？
- ・マルチライセンスとはどういったものでしょうか
- ・FOSSの告知／表示にはどういった情報がありますか？またそれらはどのように使われますか？

FOSSライセンスは、一般に改変と再頒布を許容する条件の下でソースコードを入手可能にするFOSSソフトウェアのライセンスです。

パーミッシブなライセンスの典型的な義務は、著作権表示と保証免責条項がソフトウェアに含まれることです。多くの場合、当該ライセンスでは許可なく著作者の名前を使用することを明に禁止しています。

パーミッシブなFOSSライセンスの例としてはMIT、BSD、Apacheライセンスがあります。

ライセンスの互恵性は、著作権のあるソフトウェアの派生的著作物が同じライセンスの下で入手できなければならないことを意味しています。その他の言い方として、「遺伝的」、「コピーレフト」、「共用（Share-alike）」、さらには、非難的な意味で「ウィルス性」といったものがあります。

コピーレフト スタイルのライセンスにはGPL、LGPLといったものがあります。

コピーレフト スタイルのライセンスには、多くの場合、ソース入手についての義務が規定されており、プログラムやライブラリのバイナリ版を頒布する

場合に、そのバイナリに対応したソースコードを提供することを求めます。ソースコードは同じ版名のものでなくてはならず、内容は頒布するバイナリ版に対応していなくてはいけません。

フリーウェアとシェアウェアはFOSSではありません。フリーウェアもシェアウェアもコストなしに入手可能だとしても、使用者に対しソフトウェアの改変を許容していないことがこの理由です。実際には、多くのフリーウェアとソフトウェアがプロプライエタリソフトウェアに共通するライセンス上の制約を含んでいます。

マルチライセンスはソフトウェアを複数のライセンスの下で使うことができる手法のことを言います。たとえば、あるオープンソース ソフトウェアは MITとGPLv2の2つのライセンスで供与することができます。そのようなケースでは、使用者がニーズに合わせてライセンスを自由に選択できます。

FOSSの告知／表示には、著作権保有者を識別し、そのソフトウェアをコントロールするライセンスについての情報を含む場合があります。FOSSの告知／表示が改変について告知を提供する場合もあります。FOSSの告知／表示を帰属告知の目的で、保持、再生成することを求めるライセンスもあります。

FOSS licenses are Free and FOSS Software licenses generally make source code available under terms that allow for modification and redistribution.

Typical obligations of a permissive FOSS license are that the copyright notice and warranty disclaimer are included with the software. Very often, the license would expressly prohibits users from using the author's name without permission.

Examples of permissive FOSS licenses include MIT, BSD, and Apache.

License reciprocity means that the derivative work of the copyrighted work must be made available under the same license. Other names being used include "hereditary", "copyleft", "share-alike", and pejoratively "viral."

Examples of copyleft-style licenses include GPL and LGPL.

Copyleft-style licenses often have source availability obligations, which require you to provide accompanying source code when you distribute a binary version of a program or library. The source code should be of the same version and content that corresponds to the binary version you distribute.

Freeware and Shareware are not FOSS. The reason is that even though freeware and shareware are available without cost, they don't allow the users to make modifications to the software. In fact, many of the freeware and shareware contain similar license restrictions common in proprietary software.

Multi-license refers to the practice where software is made available under multiple licenses. For example, an open source software can be dual-licensed under MIT and GPLv2. In that case, you are free to choose the license that suits your need.

FOSS Notices may include information about the identity of the copyright holders and the license governing the software. FOSS Notices may provide notice about modifications. Some licenses require that FOSS Notices be retained or reproduced for attribution purposes.

第3章

FOSSコンプライアンス概論

本章では FOSSコンプライアンスについての全体像を取り扱います。コンプライアンスがどのように機能するか基本原則から説明します。

This chapter covers the big picture of FOSS compliance. It explains how compliance works from first principles.

FOSSコンプライアンスのゴール

- ・自らの義務（FOSSの使用を検出し、追跡する）を認識すること。自身のソフトウェアを構成するすべてのFOSSコンポーネント（および、それぞれで確認されたライセンス）を特定、追跡し、そのリストを保管するためのプロセスを持つ必要がある
- ・使用されるFOSSに対しすべてのライセンス義務を果たすこと。組織のコンプライアンス プログラムは、業務遂行上生じる代表的なFOSSユースケースを認識し、これに対応する必要がある

このスライドではFOSSコンプライアンスには 目的が2つあることを説明しています。1つは、自身の義務（FOSSを検出し、追跡する）を認識し、そこで得た情報を維持するプロセスを持つことです。もう1つは、ライセンスの義務を果たすことです。

This slide explains that FOSS compliance is really a two-part goal. The first is to know your obligations and have a process to support this knowledge. The second is to satisfy the obligations.

履行すべきコンプライアンスの義務には どんなものがあるか？



関与するライセンスにもよるが、義務としては以下のようなものがある。

- ・**帰属やその他告知。** 下流のユーザーがソフトウェアの起源やライセンスによって認められた権利を知ることができるように、ソースコードや製品の関連文書、もしくはユーザインターフェース上に著作権やライセンスに係る文言を含めること。
- ・**ソースコードの提供。** 原作ソフトウェア、組み込んだソフトウェアや改変部分、およびビルド用のスクリプトも含んだソースコードを提供すること。

以下を契機としてこれらの義務が発動する場合がある：

- ・外部への頒布
- ・改変を加えたかどうか

このスライドでは、代表的なFOSSライセンスにおいてどのようなコンプライス義務を履行しなければならないかについて話を展開しています。

This slide expands on what compliance obligations must be satisfied in typical FOSS licenses.

FOSSにおける条件と制約

使用するFOSSによっては以下の条件や制約のうち1つもしくはそれ以上に従う必要がある。

- ・著作権表示（および、その他の告知）を保持すること
- ・ライセンスの写しを提供すること
- ・改変告知を提供すること
- ・混乱を避けるために、改変版の名前を異なる名前とすること
 - ・（改変の有無を問わず）ソースコードへのアクセス先を提供すること
- ・改変版（派生的著作物）を同じライセンス下に置くこと
- ・帰属告知を提供すること
- ・プロジェクト名、著作権保有者名、商標を使用しないこと
- ・原作のライセンスの下で供与された権利を他者に制限すること
- ・解除条項（違反すれば、ライセンスを失うこと）

このスライドではFOSSライセンスの使用する際に遭遇する、共通的な条件や制約のいくつかについて説明しています。ライセンスが異なればその義務も変わってくることを覚えておいてください。

This slide explains some of the conditions or restrictions commonly encountered when using FOSS licenses. Remember, different licenses have different obligations.

FOSSコンプライアンスのトリガー：頒布

- ・外部に対するマテリアル（バイナリ、ソースコードなど）の配布
 - ユーザー機器やモバイル デバイスにダウンロードされるアプリケーション
 - JavaScript、Web クライアント、ユーザー機器にダウンロードされるコード など
- ・いくつかのFOSSライセンスについては、コンピューター ネットワークを通じたアクセスが「トリガー イベント」となりうる。その際のトリガーとは「コンピューター ネットワークを通じユーザーがリモートで当該 FOSSと相互に作用すること」。
 - いくつかのライセンスがサーバー上で実行されるソフトウェアへのアクセスを可能にすることを含めたトリガー イベントを定義している。（例：Afferro GPLのすべての版についてソフトウェアを改変した場合）

このスライドでは、いつFOSSライセンスの義務が「発動(trigger)される」のかについて説明しています。FOSSライセンスは著作権ライセンスであり、基本的なコンプライアンスのトリガーはコードを他の法人（legal entity）に頒布する時です。

This slide explains when FOSS obligations are “triggered.” FOSS licenses are copyright licenses and the basic compliance trigger is when you distribute code to another legal entity.

FOSSコンプライスのトリガー：改変

- ・既存プログラムに対する変更（例：ファイル中のコードの追加、削除、コンポーネントを組み合わせる行為）
- ・改変が派生的著作物を生み出し、FOSS の著作者が改変に対し義務を課したり制限したりすることもある
- ・改変をトリガーとして発動されるFOSSの義務の例：
 - 改変の告知
 - 製品のバイナリに対応したソースコードの提供

このスライドでは、コードの改変がFOSSライセンス下の義務を課すものとなりうることを説明しています。また、派生的著作物についても若干触っています。

This slide explains that modifying code can impose obligations under FOSS licenses. It explains a little bit about derivative works.

FOSSコンプライアンス プログラム

FOSSコンプライアンスを成功させてきた組織は（ポリシー、プロセス、トレーニングやツールなどから成る）独自のFOSSコンプライアンス プログラムを作り上げている。それには以下のような意図がある。

1. 商用製品におけるFOSSの効果的使用を促進する
2. FOSS開発者の権利を尊重し、ライセンス義務を果たす
3. オープンコミュニティに参加し、コントリビュートする

このスライドでは、コンプライアンス プログラムがどのように機能するかについて大まかに（基本的概要として）説明しています。

This slide explains how FOSS compliance programs work in “broad strokes” (a basic overview).

コンプライアンスを実践する

以下対応のためビジネスプロセスおよび十分な数のスタッフを準備する：

- ・FOSSソフトウェアの起源とライセンスの確認
- ・開発プロセスにおけるFOSSソフトウェアの追跡
- ・FOSSレビューの実施と、ライセンス義務の確認
- ・製品出荷時におけるライセンス義務の履行
- ・FOSSコンプライアンス プログラムの監督、ポリシーの策定、およびコンプライアンスに関する意思決定
- ・トレーニング

このスライドでは、FOSSコンプライアンス実務が組織内でどのように機能するかについて詳しく説明しています。

This slide explains more about how FOSS compliance practices can work in an organization.

コンプライアンスのメリット

ロバストなFOSSコンプライアンス プログラムがもたらすメリット：

- ・FOSSのメリットや、FOSSが組織に与える影響についての理解が深まる
- ・FOSSの使用に伴うコストとリスクについての理解が深まる
- ・FOSSコミュニティやFOSS関連組織とより良い関係が生まれる
- ・有効なFOSSソリューションについての知識が高まる

このスライドではコンプライアンス がライセンスの法的義務の履行という域を超え、組織にもたらすメリットについて述べています。

This slide describes some of the benefits that compliance brings to an organization beyond the fact of fulfilling the legal obligations of the license.

理解度チェック

- ・FOSSコンプライアンスとは何を意味しますか？
- ・FOSSコンプライアンス プログラムの2つの主要なゴールとは何ですか？
- ・FOSSコンプライアンスプログラムを実践する上で重要なものを挙げ、その内容を述べてください。
- ・FOSSコンプライアンスプログラムのメリットとしてどんなものがありますか？

FOSSコンプライアンスとは、FOSSのライセンス条項に従うことを意味します。これは、ライセンスについての理解、ライセンス条項を支えるプロセスの具備、見落としや誤りに対処するプロセスの具備といったことを伴います。

FOSSコンプライアンスプログラムの2つの主要なゴールとは、**自身の義務を知ることと義務を果たすこと**です。

FOSSコンプライアンス プログラムでの重要な業務には以下が含まれます：

- ・FOSSソフトウェアの起源とライセンスの確認
- ・開発プロセスにおけるFOSSソフトウェアの追跡
- ・FOSSレビューの実施と、ライセンス義務の確認
- ・製品出荷時のライセンス義務の履行
- ・FOSSコンプライアンス プログラムに対する監督、ポリシーの策定およびコンプライアンスに関わる意思決定
- ・トレーニング

FOSSコンプライアンス プログラムは、さまざまなメリットを提供します。たとえばFOSSが組織にどう影響を与えるかという点や、FOSSに関連づけられるコストやリスクについての理解の向上、またFOSSコミュニティとのより良い関係、有効なFOSSソリューションについての知識の向上といった点があり

ます。

FOSS compliance means following the licensing terms of FOSS licenses. It involves understanding the licenses, having processes to support the license terms, and having processes to address any oversights or errors.

The two main goals of a FOSS compliance program are **know your obligations** and to **satisfy your obligations**.

The important business practices of a FOSS compliance program include:

- Identification of the origin and license of FOSS software
- Tracking FOSS software within the development process
- Performing FOSS review and identifying license obligations
- Fulfillment of license obligations when product ships
- Oversight for FOSS Compliance Program, creation of policy, and compliance decisions
- Training

A FOSS compliance program provides various benefits such as an increased understanding of how FOSS impacts your organization, an increased understanding of the costs and risks associated with FOSS, better relations with the FOSS community and increased knowledge of available FOSS solutions.

第4章

FOSSレビューにおけるソフトウェアの重要概念

本章ではFOSSの使用を理解する際に重要な、根本的なソフトウェア概念のいくつかについて述べていきます。

This chapter describes some fundamental concepts in understanding FOSS usage

そのコンポーネントをどう使うのか？

共通するシナリオに含まれるもの：

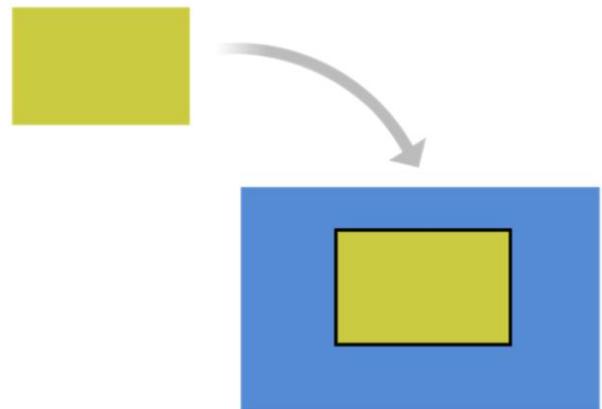
- ・ 取り込む (Incorporation)
- ・ リンクする (Linking)
- ・ 改変する (Modification)
- ・ 翻訳する (Translation)

このスライドは コンプライアンスにおいてFOSSコンポーネントの使用でどういったことを考慮すべきかという点について触っています。ユースケースが異なれば法的効果も違ってきます。次の数枚のスライドでこれらのコンセプトを具体的に説明していきます。

This slide is about how the use of FOSS components is a consideration for your compliance. Different use cases will have different legal effects. The next few slides explain these concepts in more detail.

取り込む (Incorporation)

開発者はFOSSコンポーネントの一部を自身のソフトウェア製品にコピーできる。



関連する用語 :

- ・ 統合する (Integrating)
- ・ 結合する (Merging)
- ・ 貼り付ける (Pasting)
- ・ 適応させる (Adapting)
- ・ 挿入する (Inserting)

このスライドでは、FOSSを使う際の「取り込む」の意味について概説しています。

This slide outlines what incorporation means when using FOSS.

リンクする (Linking)

開発者はFOSSコンポーネントを自身のソフトウェア製品とリンクもしくは接合(join) することができる。

関連する用語 :

- ・ 静的／動的リンクする (Static/Dynamic Linking)
- ・ 対応する (Pairing)
- ・ 結合する (Combining)
- ・ 活用する (Utilizing)
- ・ パッケージ化する (Packaging)
- ・ 相互依存性を生成する (Creating interdependency)



このスライドでは、FOSSを使う際の「リンク」の意味について概説します。

This slide outlines what linking means when using FOSS.

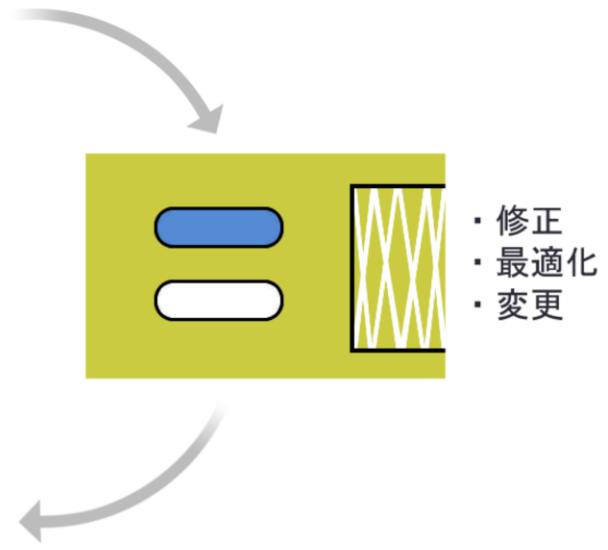
改変する (Modification)

開発者はFOSSコンポーネントに対して、次のように変更を加えることができる：

- FOSSコンポーネントに新たなコードを追加／注入する (Adding/injecting)
- FOSSコンポーネントを修正する (Fixing)、最適化する (Optimizing) または変更する (Making change)
- コードを削除する (Deleting) または除去する (Removing)

・追加
・注入

・削除



このスライドでは、FOSSを使う際の「改変」の意味について概説しています

。

This slide outlines what modification means when using FOSS.

翻訳する (Translation)

開発者は、コードをある状態から異なる状態に変換することができる。

例として以下のようなものがある：

- ・ 中国語から英語への翻訳
- ・ C++ からJavaへの変換
- ・ VHDLのマスクパターンやネットリストへのコンパイル
- ・ バイナリへのコンパイル



このスライドでは、FOSSを使う際の「翻訳」の意味について概説しています

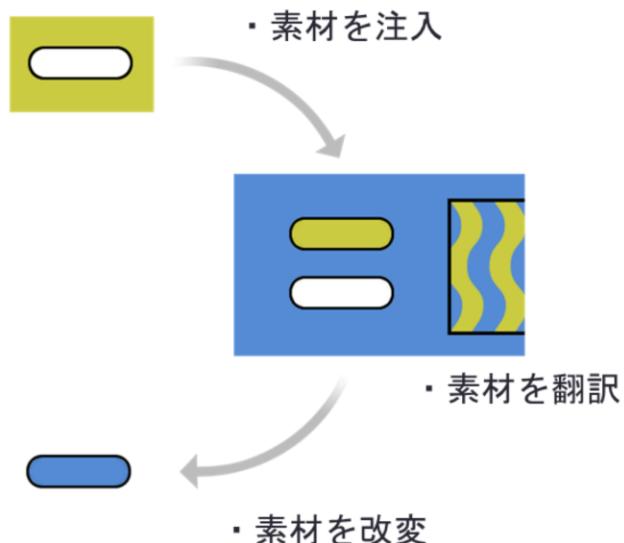
◦

This slide outlines what translation means when using FOSS.

開発ツール

開発ツールがこれらの操作のいくつかをバックグラウンドで実行してくれる場合がある。

たとえば、開発ツールのコード部分を出力ファイルに挿入してくれるものがある



このスライドでは、開発ツールが「裏方となって」これらのアクションを実施する場合があることを説明しています。この内容は企業によく知っておいていただきたいところです。

This slide explains that development tools may do some of these actions “behind the scene”, and this is an area that companies should be aware of.

FOSSコンポーネントをどのように頒布するか？



- ・誰がソフトウェアを受け取るのか？

- 顧客／パートナー
- コミュニティ プロジェクト

- ・頒布用のフォーマットは何か？

- ソースコードでの頒布
- バイナリでの頒布
- ハードウェアにプリインストール

このスライドでは、頒布することの背景にあるいくつかの考え方を説明しています。これはFOSSライセンスは通常、頒布の期間内に適用されるものであるためです。この点はコンプライアンスプログラムで考慮すべき重要なポイントです。

This slide explains some of the concepts behind distribution. Because FOSS licenses usually apply during distribution, this is a key point to consider in a compliance program.

理解度チェック

- ・取り込むとはどういうことですか？
- ・リンクするとはどういうことですか？
- ・改変するとはどういうことですか？
- ・翻訳するとはどういうことですか？
- ・頒布を検討する上で重要な要素は何ですか？

取り込みとはFOSSコンポーネントの一部を自身のソフトウェア製品にコピーすることです。

リンクとは自身のソフトウェア製品とFOSSコンポーネントをリンク（Link）もしくは接合（Join）することです。

改変とはFOSSコンポーネントに変更を加えることです。

翻訳とはコードのある状態から別の状態に変換することです。

オープンソースを頒布することを考える際には、以下の2つのことを考える必要があります。

そのソフトウェアを受け取るのは誰か？

- ・顧客／パートナー
- ・コミュニティプロジェクト

頒布フォーマットは何か？

- ・ソースコードによる頒布
- ・バイナリによる頒布
- ・ハードウェアにプレインストール

Incorporation is when you copy portions of a FOSS component into your software product.

Linking is when you link or join a FOSS component with your software product.

Modification is when you make changes to a FOSS component.

Translation is when you transform the code from one state to another.

When thinking about distribution of Open Source you should consider two things:

Who receives the software?

- Customer/Partner
- Community project

What is the format for delivery?

- Source code delivery
- Binary delivery
- Pre-loaded onto hardware

第5章

FOSSレビューの実施

本章では「FOSSレビュー」について述べていきます。FOSSの使用方法が分析され、関連する義務が決定されます。

This chapter describes a “FOSS Review” process in which FOSS usage is analyzed and the relevant obligations are determined

FOSSレビュー

- FOSSコンプライアンス プログラムにとって鍵となる要素がFOSS レビューのプロセスであり、これにより企業はFOSSに関する義務を分析し決定することができる
- FOSSレビューのプロセスには以下のステップがある：
 - ・関連情報の収集
 - ・ライセンスの義務の分析と決定
 - ・企業のポリシーや事業目標の観点からの指導

FOSSレビューはFOSSコンプライアンス プログラムの基本的構成要素です。

FOSSレビューはエンジニアリング チーム、ビジネス チーム、および法務チームが集まる場となりえます。より大規模に首尾よく行うために、計画や組織化を必要とする場合があります。

- 関連情報収集においてエンジニアリング チームもしくは開発チームが参加することもあります。
- 法務チームはライセンスの義務について分析、決定を下し、指導を行います。
- ビジネスおよびエンジニアリング チームは指導を受けて、実装します。

The FOSS Review is a basic building block of a FOSS Compliance Program.

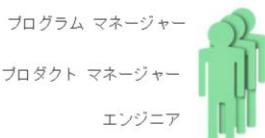
A FOSS Review can be the meeting point for engineering, business and legal teams, and can require planning and organization to successfully conduct on a large scale.

- Engineering or developer teams may participate in gathering relevant information
- Legal teams analyze and determine license obligations and provide guidance
- Business and engineering teams may receive and implement guidance

FOSSレビューの開始



FOSSレビューを
開始する



プログラム マネージャー

プロダクト マネージャー

エンジニア

FOSSレビューのプロセスは、FOSSを取り扱うプログラム マネージャー、プロダクト マネージャー、エンジニアなどの参加が必要。

注：このプロセスは外部ベンダーからFOSSベースのソフトウェアを受領した時に開始される場合もある。

最初のステップはFOSSレビューを開始するために適切な参加者を特定することです。

以下のような問い合わせが重要です：

- FOSSの使用について誰が意思決定者なのか（マネージャー、アーキテクト、個々の技術者など）？
- FOSSの使用について彼らはどのように質問・疑問を上げることができるのか？
- 開発プロセスの中にFOSSレビューが開始できる定まったチェックポイントがあるか？

The first step is to identify the proper parties to initiate a FOSS Review

Important questions to ask include:

- Who are the decision makers about FOSS usage (managers, architects, individual engineers, etc.)?
- How can they raise questions about FOSS usage?
- Is there a regular point in your development process where FOSS Reviews can begin?

どのような情報を集める必要があるか？

FOSSの使用分析にあたり、FOSSコンポーネントの属性、起源、使用方法などの情報を集める。たとえば以下のようなものがある。

- ・ パッケージ名
- ・ 版名（バージョン番号）
- ・ オリジナルのダウンロード元URL
- ・ ライセンスおよびライセンスのURL
- ・ 説明
- ・ 改変に関する記述
- ・ 依存関係のリスト
- ・ 製品で意図している使用方法
- ・ そのパッケージを内包する製品のファースト リリース（最初の公開・販売）
- ・ ソースコードを入手できるか
- ・ ソースコードがどこでメンテナンスされるか
- ・ そのパッケージが他の経緯で以前に承認されたことがあるか？
- ・ 輸出管理対象となる技術が含まれているか
- ・ 外部ベンダーからの提供物の場合：
 - 開発チームのコンタクト ポイント
 - 著作権表示、帰属表示、およびライセンスの義務履行に必要なベンダー改変ソースコード

注目すべきは、この情報のリストが 非常に多く見えることです。しかし、必要とされる情報量はFOSSコードを取り扱おうとする企業の規模、および、FOSSをどのように取り扱うかに依存します。大規模な組織体は小規模なものよりも多くの情報を必要とする傾向があります。

外部ベンダーを利用した場合は、いくつか付加的な論点があります。まず、将来的にFOSSに関する問題が生じた場合、そのベンダーを追跡調査する必要があるかもしれません。その際に信頼できるコンタクト先が重要となります。またベンダーから引き渡されたFOSSに対しライセンスの義務を果たす必要があるかもしれません。そういう義務を果たすべく必要性に応じて告知／表示やソースコードがあることを確かめましょう。

It should be noted that this list of information looks quite large. However, the amount of information required depends on the size of your company and what you intend to do with the FOSS code. Large entities tend to require more information than small entities.

There are a couple additional issues in the case of external vendors. First, you may need to follow up with the vendor if FOSS issues arise in the future, and having a

reliable point of contact is important. You may also need to meet FOSS license obligations for FOSS delivered from the vendor. Ensure you have the notices and source code as needed to meet these obligations.

FOSSレビューチーム



FOSS レビューには複数の支援グループが参加し、FOSSの使用に関する支援、指導、とりまとめ、およびレビューを協力して行う。レビュー チームには、以下の複数のチームが含まれる。

- ・ライセンスの義務を特定し、評価する法務チーム
- ・FOSSの使用の確認と追跡を支援するスキャン・ツール サポート チーム
- ・事業企画、商用ライセンス、輸出コンプライアンスなどを取り扱い、FOSSの使用によって影響を受ける可能性のある専門家

FOSSレビュー チームは異なった分野にまたがって構成されます。

法務チームは、社内もしくは外部の弁護士を含めることができます。ライセンスの義務に応じたFOSS使用をレビューし、評価します。

法務チームは、次のように他のチームからサポートされる場合もあります。

- ・FOSSの使用を特定し、追跡する調査・分析チーム。このチームはコードベース（ソースコードの集積場所）に存在するFOSSコンポーネントを特定するためのコードスキャンツールやフォレンジクス（法的確証収集）ツールを駆使した支援などを行います。また本チームは、後続コンプライアンスプロセスを支援するべく、FOSSの使用について収集した情報を整理し、追跡することも実施します。
- ・その他に、商用ライセンスや輸出コンプライアンスおよび事業企画チームなど、FOSSに関連する論点で影響を受けうる専門家、代表者も想定されます。

The FOSS Review team may consist of an interdisciplinary team

The legal team, which may include in-house or outside attorneys, reviews and

evaluates the FOSS usage for license obligations

The legal team may be supported by others, including:

- Scanning and tooling teams that identify and track FOSS usage. These teams may provide support using code scanning or forensics tools to identify FOSS components in a codebase. The teams may also organize and track information gathered regarding FOSS usage to assist with later compliance processes.
- Other specialists or representatives that may be impacted by FOSS-related issues, such as commercial licensing, compliance or business planning teams.

提案されたFOSSの使用を分析する



FOSSレビューチームは指導を行う前に、たとえば以下のような論点に対し、収集した情報を査定する必要がある。

- ・完全性、一貫性、正確性（FOSSの明らかでない使用を精査するためにコードスキャナツールが使われることがある）
- ・宣言されたライセンスがコードファイルにある内容と合致しているか？
- ・そのソフトウェアの使用案を ライセンスが本当に許容しているか？

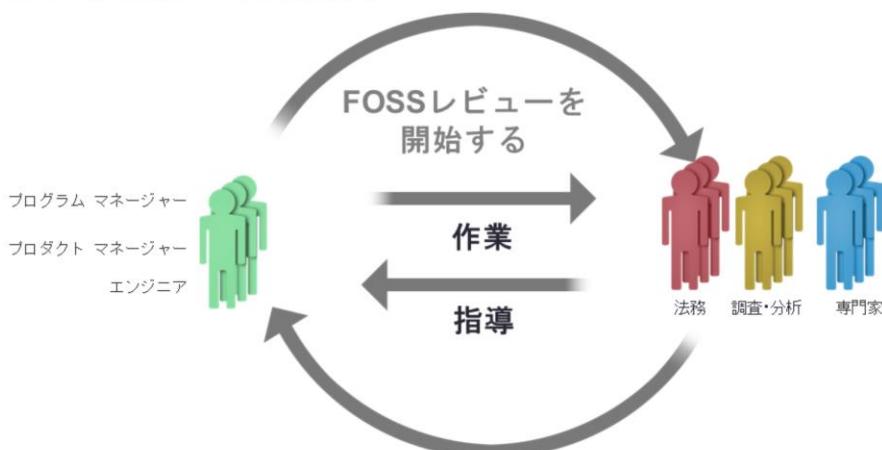
FOSSレビューチームは、FOSSの使用を適切に評価するための専門知識を有する必要があります。FOSSの使用案について法務チームやビジネスチームを教育するためにエンジニアリングチームの支援が必要となることもあります。たとえば、FOSSの明らかになつてない使用を見つけるためにコードスキャナツールが使われることがあります。

FOSSの使用案が十分査定されると法務チームは判断を下す際に必要な情報を得たことになります。

The FOSS Review team should have the expertise to properly assess the FOSS usage. This may require support from engineering teams to educate legal and business teams about the proposed FOSS usage. For example, code scanning may be used to locate undisclosed FOSS usage.

Once the proposed FOSS usage has been fully assessed, the legal team will then have the necessary information on which to make its judgments.

FOSSレビューの遂行



FOSSレビュープロセスは、インタラクティブに取り組むものとなる。この作業ではエンジニアリング チーム、ビジネス チーム、法務チームなど分野をまたぐ形となるため、フォローアップでの議論では内在する問題をすべての参加者が理解することが求められる。最終的に本プロセスではFOSSの使用についての確実な指導を行う。

FOSSレビューのプロセスは、利害関係のある参加者が協力できるように、柔軟なものである必要があります。時としてFOSSの使用シナリオがFOSSレビュー チームにとって明確でないこともあります。エンジニアリング チームは、より深くインプットを提供するための技量が必要となるでしょう。同様に、エンジニアリング チームは、FOSSレビュー チームからの指導を実行に移す際に支援を必要とするかもしれません。

--
The FOSS Review process should be flexible enough to allow the interested parties to collaborate. Sometimes a FOSS usage scenario may not be clear to the FOSS review team. The engineering team will need the ability to provide further input. Likewise, the engineering team may need assistance in implementing guidance from the FOSS review team.

FOSS レビューの監督



FOSSレビューのプロセスにおいては、関係者間での意見の相違があつたり、ある決定が特別に重要だったりする場合を想定し、十分な監督機能が必要となる。

FOSSレビュープロセスは監督機能を持つ必要があります（たとえば、この図では幹部レベルのレビュー委員会）。このような監督委員会は、重要な方針決定や、レビュープロセスでの関係者の意見の不一致の解決などを行います。

-

The FOSS Review process should have oversight (for example, an Executive Review Committee in this diagram). The oversight committee may make important policy decisions or resolve disagreements between parties in the review process.

理解度チェック

- ・FOSSレビューの目的は何ですか？
- ・FOSSコンポーネントを使いたい時に最初に行うべきアクションは何ですか？
- ・FOSSの使用に関する質問や疑問がある場合、何をするべきですか？
- ・FOSSレビューのためにどのような種類の情報を集めますか？
- ・誰がそのソフトウェアのライセンスを供与しているのかを確認するには、どのような情報が役立ちますか？
- ・外部ベンダーから受領したコンポーネントをレビューする際に追加的な情報として重要なものは何ですか？
- ・FOSSレビューで収集された情報の質を評価するためにどのようなステップを取ることができますか？

FOSSの使用に関する情報を収集し、分析するため、および適切な指導を行うためです。

FOSSレビュープロセスを開始します。このプロセスを開始する手法は企業によって異なりますが、開発でOSSの使用に関わる人たちにはオープンにするべきです。

FOSSレビュー プロセスを開始するか、FOSSレビュー チームにコンタクトを取ります。組織においてFOSSの使用者が指導を受け入れができるよう、そのプロセスは柔軟であるべきです。

第一歩としては、パッケージ名、版名（バージョン番号）、ダウンロード元URL、ライセンス、説明、製品内で意図される使用法などがあるでしょう。組織や意図したユースケースに依存して正確かつ詳細な情報が必要となるでしょう。

通常、著作権表示、帰属情報およびソースコードによって誰がそのFOSSソフトウェアをライセンスしているかを特定することができます。

将来発生しうるFOSSの問題を追跡するために必要な開発チームのコンタクト

ポイントです。外部ベンダーのソフトウェアをコントロールするFOSSライセンスの義務を履行するために、著作権表示、帰属表示、およびベンダーの改変に対応したソースコードを入手する必要があるかもしれません。

完全性、一貫性、正確性について情報をチェックすることです。このプロセスは、開示されていないFOSS使用に対してコード スキャン ツールで精査することも含めて支援チームの助けを借りることができます。

To gather and analyze information regarding FOSS usage and to produce appropriate guidance.

Initiate a FOSS review process. The method for initiating this process may vary by company, but should be open to those who are involved in using FOSS in development.

Initiate a FOSS review process or contact the FOSS review team. The process should be flexible enough so that FOSS users in your organization have access to guidance.

The package name, version, download URL, license, description and intended use in your product is a good starting point. The precisely level of detail you will need depends on your organization and intended use case.

The copyright notices, attribution and source code normally helps to identify who is licensing the FOSS software.

Development team's point of contact in case you need to follow up with future FOSS issues. You may also want to obtain copyright and attribution notices, and source code for vendor modifications if these are needed to satisfy license obligations for FOSS licenses governing the third party software.

Check information for completeness, consistency and accuracy. This process may be assisted by support teams, including teams that run code scanning tools to scan for undisclosed FOSS usage.

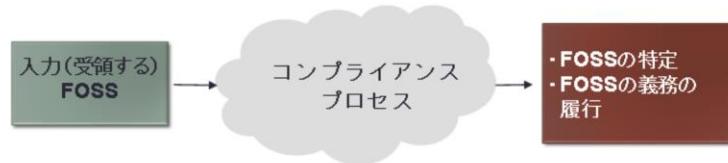
第6章

コンプライアンス マネジメントの始めから終わりまで（プロセスの例）

This chapter contains an example of a detailed end to end compliance management process.

概要

- ・コンプライアンス マネジメントは、製品（もしくはOpenChain 仕様書で定義の「供給ソフトウェア」）の中で使われるFOSSの取り込みと頒布をコントロールする一連のアクションで構成される
- ・コンプライアンスの適正努力（Compliance due diligence）の結果として、供給ソフトウェアで使用されているすべてのFOSSが特定できる。これにより、すべてのFOSSライセンスの義務が履行され、将来にわたり履行されることを確かなものにする
- ・大企業が詳細なプロセスを保有する一方で、小規模の企業では単にチェックリストを使うだけの場合がある。本章では大企業のプロセスの一例を紹介する



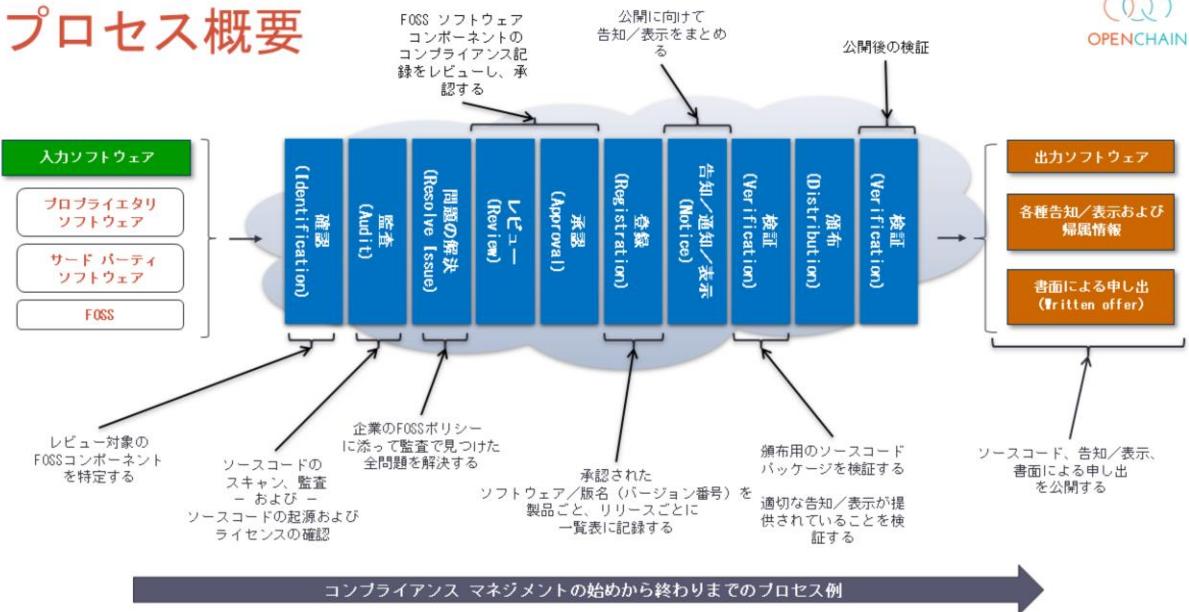
このスライドは、コンプライアンスマネジメントの定義と最終目標について述べています。

本章は大企業で実施される可能性のある具体的な例を提供します。小規模な企業では、より簡素化したプロセスで取り組むことが望まれるでしょう。

--
 This slide describes the definition of compliance management and its end goals.

Note that this section provides a detailed example of what may take place in a large enterprise. Smaller companies may wish to approach the process in a more streamlined way.

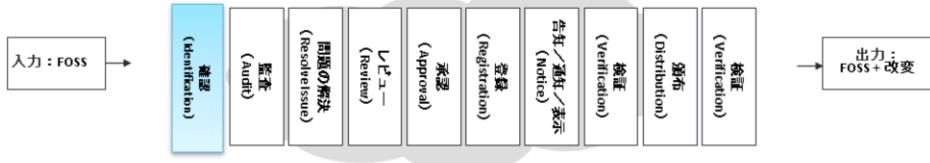
プロセス概要



本スライドは、本章で述べる各ステップの全体像です。

This slide is an overview of the steps that will be described in this chapter.

FOSSの使用を確認し、追跡する



すべてのソースに含まれるFOSSを確認し、追跡を開始する

- 前提条件 :**

- このプロセスは以下のイベントのうちの1つで開始される :
 - 開発チームがFOSSコンポーネントのレビュー や外部向けのリリースを要望する
 - 適切な承認がなく使用されているFOSSを発見する
 - サード パーティのソフトウェアの一部に使用されているFOSSを発見する

- ステップ :**

- 入力リクエストが登録される
- 全プラットフォームのスキャンが実施される
- サードパーティ 提供のソフトウェアに対する精査を実施する
- ソース リポジトリに追加されているが、入力リクエストのないすべてのFOSSコンポーネントを識別し、レビューを実施する

- 成果 :**

- そのFOSSについてコンプライアンス の記録が作成（またはアップデート）される
- ソースコードのスキャンまたはレビューのための（次のステップとなる）監査が要請される

ここでの例における最初のステップは、FOSSの使用を確認することです。

このステップは、「前提条件」で挙げたイベントのうちの1つによって始動されます。たとえば開発チームがリクエストを上げた（またはFOSSレビューを開始した）場合などです。またこのステップは、出荷ソフトウェアにFOSSが使用されている、または企業が使用するサード パーティ ソフトウェアにFOSSが使用されていること、そしてそのために適正なレビューの実施が必要であることをレビュー チームが発見した場合や、通知された場合にも開始します。

この例では、FOSSレビュー チームはエンジニアたちからのレビュー リクエストを通じて、内部開発・サード パーティのソフトウェアヘスキャンを実施することによって、あるいは、開発のブランチにチェックインされたコードのレビューによってFOSSの使用を確認します。その後レビュー チームはレビュー記録を生成し、次の「監査」ステップに進みます。

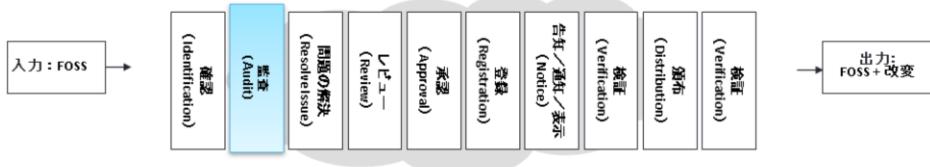
The first step in our example process is to identify FOSS usage.

This step may have been initiated by one of the events listed in “prerequisites.” For

example, a development team may have initiated a request (or initiated a FOSS Review). The step may also begin if the review team discovers or is notified that FOSS is being used in a software release or in third party software used by the company, and that a proper review needs to take place.

In this example, the FOSS review team may identify FOSS usage through review requests from engineers, from performing scans of internally-developed and third-party software, or reviewing code checked into development branches. The review team will then create a record of the review, then move to the next step (“Audit”).

ソースコードを監査する



FOSSコンポーネント、およびその起源とライセンスを確認する

- **前提条件 :**

- ・ 開発チームがコンプライアンスの記録をFOSSの使用方法に関する情報と一緒に提供する
- ・ 開発チームから提供される記録がない場合、FOSSコンポーネント発見時に記録が生成される

- **ステップ :**

- ・ 監査のためのソースコードが特定される
- ・ ソフトウェアツールによってソースがスキャンされる
- ・ 監査やスキャンによって「ヒット」したものがレビューされ、コードの起源が適正かどうかが検証される
- ・ ソフトウェアの開発／リリースのライフサイクルをベースに監査もしくはスキャンが繰り返し実施される

- **成果 :**

- ・ ソースコードの起源とライセンスを確認した監査レポートが生成される

次のステップは、前のステップで確認されたソースコードの監査です。

ここでの例では、企業は確認されたFOSSコンポーネントについて調査を実施しています（たとえば、宣言されているライセンスのレビューや、FOSSコンポーネントの起源の調査など）。また企業はソースコードの起源や構成を検証するためにスキャンも実施します。

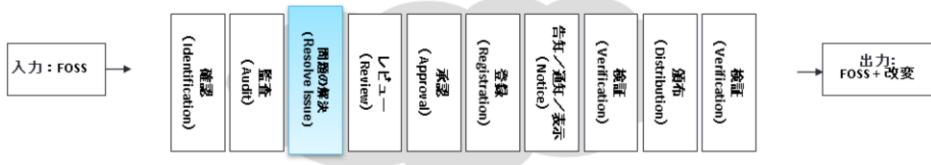
その後レビュー チームは、ソースコードの起源とライセンスに関して結論づけた監査レポートを作成します。

The next step is auditing source code identified in the previous step.

In our example, the company may conduct research into the identified FOSS component (e.g., review declared licenses, research origins of the FOSS component). The company may also scan the source code to verify the origin and composition of the code.

The review team may then produce an audit report with its conclusions regarding the origin and licensing of the source code.

問題を解決する



監査で確認されたすべての問題を解決する

- **前提条件 :**
 - ・ ソースコードの監査やスキャンが完了している
 - ・ 監査レポートがソースコードの起源とライセンスを特定し、さらなる調査が必要なファイルにフラグが立てられている
- **ステップ :**
 - ・ 監査レポートで指摘されたFOSSポリシーに反する問題を解決するために、適切なエンジニアにフィードバックを提供する
 - ・ 問題が解決されたことをエンジニアとともに確認する
- **成果 :**
 - ・ レポートでフラグを立てられたそれぞれのファイルに対する問題の解消、およびフラグの立てられたすべてのライセンスに関する矛盾の解決

ソースコードの起源とライセンスを確認した監査レポートが作成されると、レビュー チームは企業のFOSSポリシーに従い、すべての問題にフラグをつけ、レビューをする必要があります。たとえば、以前のステップで両立しないライセンス下にある異なるFOSSのコードを含んだFOSSコンポーネントを特定いたします。レビューチームはこの問題を解決するためにエンジニアリングチームに適切なフィードバックを提供する必要があります。

Once an audit report is produced that confirms the origin and licensing of source code, the review team should flag and review any issues under the company FOSS policy. For example, the earlier steps may have identified a FOSS component that contains other FOSS code under an incompatible license. The review team should provide appropriate feedback to the engineering team to resolve the issues.

レビューを実施する



監査レポートをレビューし、発見されたすべての問題が解決していることを確認する

・前提条件：

- ・ソースコードが監査されている
- ・すべての指摘された問題が解決されている

・ステップ：

- ・レビュー スタッフに適切な職権レベルを含める
- ・監査されたソースコード、ソフトウェア アーキテクチャ、およびFOSSの利用方法についてFOSSレビューを実施する（次スライドのテンプレート参照）
- ・FOSSライセンス下の義務を確認する

・成果：

- ・監査レポートにあるソフトウェアがFOSSポリシーと合致することを確かなものとする
- ・監査レポートで発見されたことを保存し、解決された問題を次のステップへの準備ができた（つまり承認された）ものとして示される

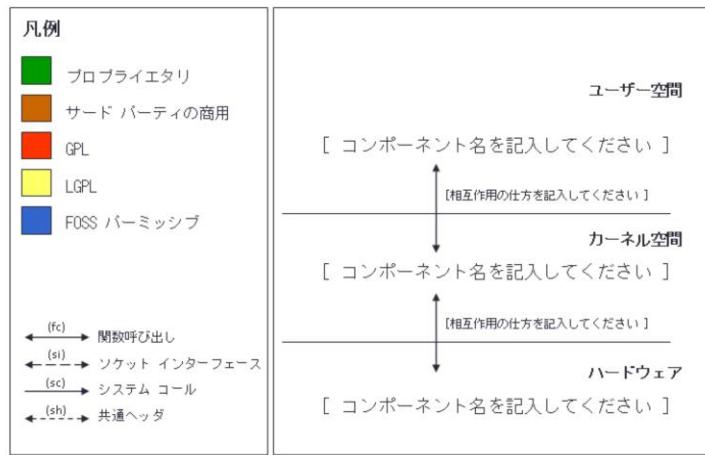
このステップでは、FOSSレビュー チームが直前のステップで収集された事実をレビューし、FOSSライセンス下で企業が負うべき義務を確認します。

このステップは直前のステップ（監査での問題を解決する）と密接に関係しています。直前のステップでは企業のポリシーと合致しないFOSSの使用を取り除きました。このステップでは使用していくことになったFOSSのライセンス義務を評価し、確認します。

In this step, the FOSS review team reviews the facts collected in the previous steps and identifies the company's obligations under the FOSS licenses.

This step may be closely linked with the previous step (Resolving Audit Issues). In the previous step we removed FOSS usage that did not conform to company policy. In this step, we evaluate and identify the license obligations for FOSS usage that is retained.

アーキテクチャ レビュー（テンプレートの例）

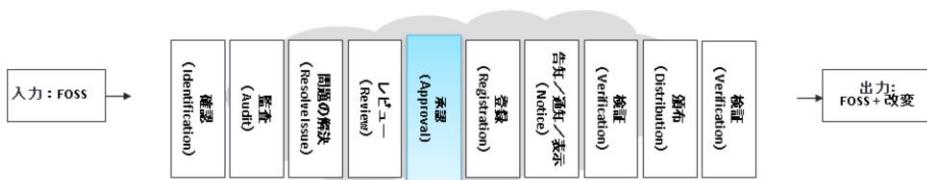


このスライドでは、FOSSの使用と企業のソフトウェアとの関係を説明するために使うテンプレートを掲載しています。たとえば、FOSSと企業のコンポーネントは一緒にリンクされるのか？といったことです。このようなテンプレートは、計画されたFOSSの使用についてFOSSレビューチームの理解を助けるためにエンジニアリング チームによって作成されることもあります。

This slide contains a template that may be used to illustrate FOSS usage and its relationship with company software. For example, how are FOSS and company components linked together? Templates such as these may be created by engineering teams to help educate the FOSS review team about planned FOSS usage.

承認

- 前ステップのソースコード監査およびレビューの結果に基づき、ソフトウェアの使用が承認、却下される
- この承認で、承認対象のFOSSコンポーネントのバージョン、使用方法、およびFOSSライセンス下で適用されるその他すべての義務などを明確にする
- 承認は適切な職権レベルで行われる必要がある



ここでの例における承認ステップでは、レビュー チームは問題のFOSSの使用を、それに伴う条件や義務に添って承認するかどうかを明らかにします。この承認では、FOSSコンポーネントの版名や承認される使用シナリオなどの重要な詳細情報を盛り込む必要があります。

In the approval step of our example process, the review team communicates whether it approves of the FOSS usage in question, along with any associated conditions or obligations. The approval should also include important details such as version numbers of FOSS components and the approved usage scenario.

登録／承認の追跡

- ・ 製品内での使用についてFOSSコンポーネントが承認された場合、それがその製品のソフトウェア一覧表に追加される
- ・ 承認内容とその条件が追跡システムに登録される
- ・ 新しいバージョンのFOSSコンポーネントや新しい使用方法が提案された場合には、新たな承認が必要となることを追跡システムで明確にする



前ステップの承認情報は、そのソフトウェアのリリースに関与するすべての人々が理解し、関連するライセンスの義務を履行できるように、登録され、追跡される必要があります。

 Approval information from the previous step should be tracked or registered so that anyone releasing the software can understand and comply with the relevant license obligations.

告知／通知／表示



製品リリース時に用いる適切な告知／表示を準備する

- ・著作権表示と帰属表示のすべてを提供することで、FOSSが使用されていることを表明する
- ・製品のエンドユーザーにFOSSソースコードの写しの入手方法に関する情報を提供する（GPLやLGPLのケースのように、その必要がある場合）
- ・必要に応じ製品に含まれるFOSSについてライセンス同意書全文のコピーを用意する

FOSSライセンスで求められる場合、適切な告知／表示文が準備されなければなりません（多くの場合、製品に添付されるテキストファイルで）。告知／表示には帰属表示や改変告知、あるいは、ソースコード提供の申し出が含まれます。いくつかのライセンスについては、ライセンス全文の写しを含める必要があります。

--
If required by a FOSS license, appropriate notices should be prepared (often in a text file that accompanies the release). Notices may include attribution notices, modification notices, or offers for source code. For some licenses, you may also need to include a full copy of the license text.

頒布前の検証



頒布されるソフトウェアがレビューされ承認されたことを検証する

・前提条件：

- FOSSコンポーネントの使用が承認されている
- FOSSコンポーネントがそのリリースのソフトウェア一覧表に登録されている
- 適切な告知／表示が準備されたている

・ステップ：

- 頒布用のFOSSパッケージが明確になっていて、承認されていることを検証する
- レビューされたソースコードが製品として出荷されるバイナリ形態の同等物と一致していることを検証する
- エンドユーザー向けに当該FOSSのソースコードをリクエストできる権利について情報提供するための適切な告知文がすべて用意されていることを検証する
- 確認されたその他義務の履行を検証する

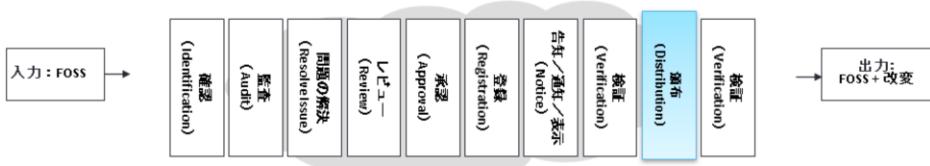
・成果：

- 頒布パッケージには、レビューされ承認されたソフトウェアだけが含まれている
- (OpenChain仕様書で定義される)「頒布コンプライアンス関連資料」として、頒布パッケージやその他頒布形態に適切な告知／表示が盛り込まれている

例として挙げたここでのプロセスについて、このスライドでは企業がリリース前にFOSSライセンスの義務を履行したことを探証していきます。ソースコード入手可能としなければならない場合、企業はソースコードが頒布されるバイナリファイルと合致していることを検証します。また企業は告知文が適切に生成され、頒布パッケージに含まれていることを必要に応じて検証します。

In this slide of our example process, the company verifies that it has met its FOSS license obligations before release. In cases where source code must be made available, the company verifies that the source code matches the binary files being distributed. The company also verifies that notices are properly produced and included in distribution packages as needed.

添付ソースコード※を頒布する



添付ソースコードを要求された形で提供する

・前提条件：

- すべての頒布前検証が完了し、問題が発見されていない

・ステップ：

- 製品に対応したソースコードを関連ビルトツールや文書類とともに提供する（例：頒布Webサイトへアップロードする、頒布パッケージに含める）
- ソースコードが、製品とバージョンに対応したラベルで識別される

・成果：

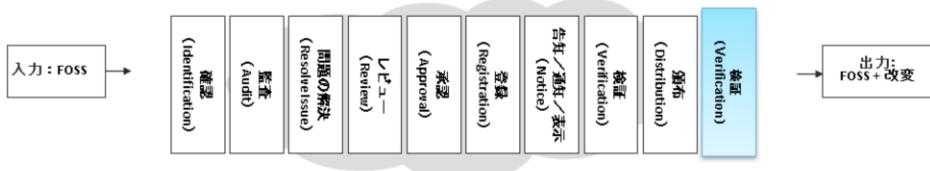
- ソースコードを提供する義務が履行される

※製品に対応したソースコードのこと

ソースコード入手可能にする際、企業は製品に対応したソースコードをFOSSライセンスが許可する仕組みで提供します。このことは、ソースコードをソフトウェアの頒布とともに提供、またはそれを書面による申し出を通じ入手可能とすること、もしくはWebサイトでソースコードのアーカイブを公開することを意味します。

In cases where source code must be made available, the company provides the accompanying source code through the mechanisms permitted under the FOSS license. This may mean providing the source code along with the software distribution, making it available through a written offer, or posting a source code archive on a website.

最終検証



ライセンス義務のコンプライアンスを検証する

・前提条件：

- 添付ソースコードが要求された通りに提供されている
- 適切な告知文が準備された

・ステップ：

- 添付ソースコードが（あるならば）適切にアップロードされたか、または頒布されたかを検証する
- アップロードされた、または頒布されたソースコードが承認されたものと同じバージョンとなっていることを検証する
- 告知/通知/表示が適切に公開され、入手可能となっているかを検証する
- その他確認された義務が履行されているかを検証する

・成果：

- 検証済みの頒布コンプライアンス関連資料が適切に提供される

このステップでは、企業の頒布行為がFOSSライセンスの義務を履行していることを検証します。このステップは一組織体としてFOSSレビュープロセス全体を監督する機能になります。

In this step, the company verifies that its distribution complies with its FOSS license obligations. This step could be a function of an entity providing oversight for the overall FOSS review process.

理解度チェック

- ・コンプライアンスの適正努力 (Compliance due diligence) としてどのようなものが関係しますか？（本カリキュラムのプロセス例に挙げた各ステップについて概要を述べてください）
 - 確認
 - ソースコードの監査
 - 問題の解決
 - レビューの実施
 - 承認
 - 登録／承認の追跡
 - 告知／通知／表示
 - 頒布前の検証
 - 添付ソースコードの頒布
 - 検証
- ・アーキテクチャ レビューではどういったことを期待しますか？

本カリキュラムのプロセス例では以下のステップがありました。

- ・ 確認 (Identification) – FOSSの使用を確認し追跡します。この作業はエンジニアからの要求、サード パーティによる開示、もしくはコード スキャンを通じて発生します。
- ・ ソースコードの監査 – 確認されたFOSSコンポーネントをライセンスと起源についてレビューします。
- ・ 問題を解決する – FOSSポリシーに反したFOSSの使用を除去します。
- ・ レビューの実施 – FOSSの使用に対する義務を査定し決定します。
- ・ 承認 – 承認の条件とライセンスの義務を明らかにします。
- ・ 登録／承認の追跡 – その後のステップのために承認の条件とライセンス義務を追跡します。
- ・ 告知／通知／表示 – FOSSライセンスで求められる形で告知文を準備します。
- ・ 頒布前の検証 – 頒布物のリリース前にコンプライアンスをレビューします。
- ・ 添付ソースコードの頒布 – ソースコードを必要に応じて入手可能にします。
- ・ 検証 – コンプライアンス プロセスの監督を実施します。

アーキテクチャ レビューではFOSSコンポーネントと企業のソフトウェア間の

関係を検査します。たとえば、FOSSと企業のコンポーネントがどのように互いにリンクするかといったことを検査します。

For our example process, the steps include:

- Identification - Identify and track FOSS usage. This may take place through engineer requests, third party disclosures, or code scanning.
- Auditing source code - Review identified FOSS components for license and origin information.
- Resolving issues - Remove FOSS usage that is incompatible with FOSS policies.
- Performing reviews - Assess and determine obligations for FOSS usage.
- Approvals - Communicate approval conditions and license obligations.
- Registration/approval tracking – Track approval conditions and license obligations for later compliance steps.
- Notices - Prepare notices as required by FOSS licenses.
- Pre-distribution verifications – Review distributions for compliance before release.
- Accompanying Source Code Distribution – Make source code available as needed.
- Verification – Provide oversight for compliance process.

Architecture reviews examine the relationships between FOSS components and company software. For example, how are FOSS and company components linked together?

第7章

コンプライアンスでの落とし穴とその回避

本章では、FOSSコンプライアンス プロセスで共通的な落とし穴について説明します。併せてそういった落とし穴を回避するアプローチについて考察していきます。

This chapter describes some common pitfalls in FOSS compliance processes, and discusses approaches to avoiding these pitfalls

コンプライアンスの落とし穴

本章は、コンプライアンス プロセスで回避すべき潜在的な落とし穴について説明する

1. 知的財産 (IP) に関する落とし穴
2. ライセンス コンプライアンスに関する落とし穴
3. コンプライアンス プロセスに関する落とし穴

本章ではFOSSコンプライアンスプロセスで避けるべき、共通的な落とし穴について説明します。

In this chapter, we will describe some common pitfalls to avoid in the FOSS compliance process.

知的財産に関する落とし穴

タイプと説明	発見のされ方	回避策
<p>コピーレフト型のFOSSがプロプライエタリ コードやサード パーティのコードに意図せずに取り込まれてしまう：</p> <p>このタイプの失敗は、開発プロセスにおいて、エンジニアがFOSSポリシーに反して、（自社にとって、もしくはサード パーティにとって）プロプライエタリなソースのコードにFOSSコードを追加（またはカット & ペースト）する時に起こる。</p>	<p>このタイプの失敗は、ソースコードをスキャンや監査実施の結果として、以下と合致可能性があるものとして発見される：</p> <ul style="list-style-type: none"> • FOSSのソースコード • 著作権表示 <p>ソースコード自動スキャン ツールはこの目的のために使用することができる</p>	<p>このタイプの失敗は以下の対策によって回避できる：</p> <ul style="list-style-type: none"> • コンプライアンスでの問題、各種タイプ／カテゴリーのFOSSライセンス、およびプロプライエタリ ソースコードにFOSSソースコードを取り込むことの意味を意識されるように、技術スタッフにトレーニングを提供する • ビルド環境においてすべてのソースコード（プロプライエタリ、サード パーティ、FOSS）に対し、定期的にソースコード スキャンや監査を実施する

このスライドで挙げている最初の落とし穴は、コピーレフト型のライセンスのFOSSが気づかれずにプロプライエタリのコードと混在してしまうところで生じます。

この状況はライセンスの告知／通知／表示に関してソースコードを監査することや、コード スキャン ツールの使用を通じて発見されることがあります。

予防策として、エンジニアリング スタッフへのトレーニング提供、および開発プロセスにおける監査やスキャンの定期的な実施などがあります。

The first pitfall described in this slide arises where copyleft-style licensed FOSS is inadvertently mixed with proprietary code.

This may be discovered through auditing source code for license notices or using code scanning tools.

Preventative measures include training of engineering staff, and building regular audits or scans into the development process.

知的財産に関する落とし穴

タイプと説明	発見のされ方	回避策
コピーレフト型のFOSSがプロプライエタリなソフトウェアに意図せずにリンクされてしまう（逆もまた同様） このタイプの失敗は、ライセンスが相互に矛盾するか両立しないソフトウェア（FOSS、プロプライエタリ、サードパーティ）をリンクした結果起こる。リンクの法的効果についてはFOSSコミュニティで議論の対象となる。	このタイプの失敗は異なるソフトウェアコンポーネント間のリンクに対し依存性追跡ツールを使うことで発見できる。	このタイプの失敗は以下の対策によって回避できる： 1. エンジニアリングスタッフをトレーニングし、FOSSポリシーの法的見解に反したライセンスを持つソフトウェアコンポーネントへリンクすることを回避する 2. ビルド環境全体に対し、継続的に依存性追跡ツールを実行する
ソースコードの改変を通じてプロプライエタリのコードがコピーレフト型のFOSSに組み込まれてしまう	このタイプの失敗は、FOSSコンポーネントに組み入れたソースコードを確認・分析するための監査やスキャンによって発見されることがある。	このタイプの失敗は以下の対策によって回避できる： 1. エンジニアリングスタッフへのトレーニング 2. 定期的なコード監査の実施

このスライドで挙げている最初の落とし穴は、コピーレフト型のライセンスのFOSSが気づかれることなくプロプライエタリソフトウェアにリンクされてしまうところで生じます。

このタイプの失敗は、依存性追跡ツールの使用や、アーキテクチャのレビューによって検出できます。

予防策は、エンジニアリングスタッフのトレーニングや、開発プロセスへのアーキテクチャレビューの組み込みなどです。

2つ目の落とし穴は、プロプライエタリコードがコピーレフト型ライセンスのFOSSに組み込まれることで生じます。たとえば、エンジニアリングチームがFOSSコンポーネントに対して行った改変により、プロプライエタリコードが含まれてしまうようなケースです。

このタイプの失敗は、FOSSコンポーネントに組み込まれたソースコードを監査することで発見できます。

予防策としては、エンジニアリングスタッフのトレーニングや、開発プロセスへの定期的な監査を組み込まれることなどがあります。

The first pitfall in this slide arises where copyleft-style licensed FOSS is inadvertently linked to proprietary code.

This type of failure may be detected using dependency tracking tools or reviews of architecture.

Preventative measures include training of engineering staff, and building architectural reviews into the development process.

The second pitfall arises where proprietary code is included in copyleft-style licensed FOSS. For example, an engineering team making modifications to a FOSS component may include proprietary code in the modifications.

This type of failure may be discovered through auditing source code introduced into the FOSS component.

Preventative measures include training of engineering staff and building regular audits into the development process.

ライセンス コンプライアンスに関する落とし穴

タイプと説明	回避策
添付ソースコードを提供しない	このタイプの失敗は、製品を市場に出す前の段階で、ソースコードの全体像を捕捉し、製品のリリース サイクルごとのチェックリスト項目を公開することで回避できる。
間違ったバージョンのソースコードを提供してしまう	このタイプの失敗は、バイナリのバージョンに対応した ソースコードが確実に公開されるようコンプライアンス プロセスに検証ステップを加えることで回避できる。
FOSSコンポーネントの改変に対応したソースコードを提供しない	このタイプの失敗は、FOSSコンポーネントに対応した原作のソースコードに加え改変に対応したソースコードが確実に公開されるようコンプライアンス プロセスに検証ステップを加えることで回避できる。

このスライドで挙げている最初の落とし穴は、企業が製品のバイナリに対応したソースコードを提供する義務を負っている一方で、その履行ができていないところで生じます。

2つ目の落とし穴は、企業がソースコードを提供していても、頒布したバイナリと合致する正しい版名の提供ができていないところで生じます。

3つ目の落とし穴は、企業がFOSSコンポーネントを改変したにもかかわらず、改変した版のソースコードを公開できていないところで生じます。企業は、代わりに原作版のFOSSコンポーネントを公開してしまうことがあります。

いずれのケースにおいても、失敗はコンプライアンス プロセスに適切なステップを実行することで回避できます。たとえば、リリースされたバイナリに対応するソースコードは、バイナリ版と併せてソースコードの全体像を捕捉し、保存されることが必要です。バイナリのリリースに合ったソースコードが確実に提供されるように、リリースに先立った検証作業でもチェックすべきでしょう。

--
The first pitfall in this slide arises where a company has an obligation to provide

accompanying source code, but fails to do so.

The second pitfall arises where a company provides accompanying source code, but fails to provide the correct version that matches the distributed binary version.

The third pitfall arises where a company modifies a FOSS component, but fails to publish the modified version of the source code. The company instead publishes the source code for the original version of the FOSS component.

In each case, the failures may be prevented by properly applying steps in the compliance process. For example, source code for released binaries should be captured and stored along with the binary version. Verifications prior to release should check to ensure the proper source code is provided with the binary release.

ライセンス コンプライアンスに関する落とし穴

タイプと説明	回避策
<p>FOSSソースコードの改変に印付けがされていない： 変更したFOSSのソースコードに、FOSSライセンスが要求する印付けがされていない</p>	<p>このタイプの失敗は、以下の対策によって回避できる：</p> <ol style="list-style-type: none">ソースコード リリース前の検証ステップでソースコード改変の印付けを行うエンジニアリング スタッフにトレーニングを実施し、公開されるすべてのFOSSソフトウェアやプロプライエタリ ソフトウェアの著作権表示やライセンス情報をエンジニアリング スタッフが確実に更新できるようにする

このスライドで挙げている落とし穴は、企業がFOSSコンポーネントを改変する際、FOSSライセンスが求める改変への印付けをしていないところで生じます。この落とし穴は、コードに印付けするプロセスを実装したり、検証ステップの中で印付けしたりすることで回避できます。

The pitfall in this slide arises where a company modifies a FOSS component, then fails to mark its modifications when required by the FOSS license. This pitfall may be prevented through implementing processes for marking code or within verification steps.

コンプライアンス プロセスにおける失敗

説明	回避策	予防策
開発者がFOSSの使用について承認を求めない	このタイプの失敗はその企業の FOSS ポリシーやプロセスに従事するエンジニアリング スタッフへの トレーニング の提供によって 回避できる。	このタイプの失敗は、以下の対策によって予防できる： 1. ソフトウェア プラットフォーム全体に対する定期的なスキャンを実施し、「宣言されていない」FOSS の使用を検出する 2. 企業のFOSSポリシーやプロセスに従事するエンジニアリング スタッフにトレーニングを提供する 3. 従業員の人事考課にコンプライアンスを含める
FOSSトレーニングが受講されない	このタイプの失敗はFOSSトレーニングの修了を 従業員の専門性開発計画の一部とし、人事考課の管理対象にすることで回避できる。	このタイプの失敗は、指定期日までの FOSS トレーニング受講をエンジニアリング スタッフに義務付けることで予防できる。

このスライドの落とし穴は、FOSSコンプライアンス プロセスがエンジニアリング チームに融合できないところから生じます。ここでは、エンジニアリング チームがFOSSの使用をレビュー プロセスに上げない、もしくはFOSSの使用に取り組む方法についてトレーニングを受けないケースを挙げています。

予防策としては、エンジニアリング トレーニングをモニタリングしたり、コンプライアンス プロセスをエンジニアリング チームに利用しやすいものにするといったことがあります。

The pitfalls in this slide arise from a failure to integrate the FOSS compliance process with the engineering team. In these cases, the engineering team does not raise FOSS usage to the review process, or does not receive the training on how to handle FOSS usage.

Preventative measures include monitoring of engineering training, and also making the compliance process easily accessible to the engineering team.

コンプライアンス プロセスにおける失敗

説明	回避策	予防策
ソースコードの 監査が実施されない	<p>このタイプの失敗は、以下の対策によって回避できる：</p> <ol style="list-style-type: none"> 周期的なソースコード スキャン／監査の実施 定常的に監査を反復的開発プロセスにおけるマイルストーンと位置付ける 	<p>このタイプの失敗は、以下の対策によって予防できる：</p> <ol style="list-style-type: none"> スケジュール遅延とならないよう適切なスタッフを配置する 定期的な監査を確実に実行する
監査で発見された 問題（スキャン ツールや監査レポートで「ヒット」したもの）を解決できない	このタイプの失敗は、監査レポートが未完了の場合にコンプライアンス チケットの解決（つまりクローズ）を許可しないことで回避できる。	このタイプの失敗は FOSSコンプライアンス プロセスの承認ステップにブロック機能を実装することで予防できる。
FOSSレビューがタイムリーに求められない	このタイプの失敗は、エンジニアリングチームがFOSSソースコードの採用を決定していない場合でも、それより早期にFOSSレビュー リクエストを開始することで回避できる。	このタイプの失敗は教育を通じて予防できる。

このスライドでは、コンプライアンス プロセスの失敗によって発生する結果について述べています。最初は、FOSSコードベースが開発の中で使用され、適切なレビューなしでリリースされるケースです。2つ目は、FOSSの使用は周知されていても、ライセンスの義務がレビュー・決定されていないケースです。最後は、コンプライアンス プロセスがリリース期限のプレッシャーに直面し、タスクを実行する時間が限られているケースです。

This slide describes potential consequences of compliance process failures. In the first case, a code base may be used in development and releases without proper review. In the second case, FOSS usage may be known, but license obligations are not reviewed or determined. In the last case, the compliance process may face release deadline pressures and have limited time to perform its tasks.

製品出荷前にコンプライアンスを確認する

- ・企業は製品が（どのような形態であれ）出荷される前にコンプライアンスを優先して実行しなければならない
- ・コンプライアンスを優先することで以下が促進される：
 - ・組織内でのFOSSの効果的な使用
 - ・FOSSコミュニティやFOSS関連組織とのより良い関係

本章で述べた落とし穴を避けるには、リソースと努力が必要になりますが、FOSSコンプライアンス プロセスを優先することは重要なことです。そうすることで、開発プロセスにおけるFOSSの使用を効果的なものにし、またFOSSコミュニティにおける良好な協働関係の維持にも役立つことになります。

While avoiding the pitfalls described in this chapter may take resources and effort, prioritizing the FOSS compliance process is important. It can help you more effectively use FOSS in your development process, and also help maintain good working relationships within the FOSS community.

コミュニティとの関係を確立する

FOSSを商用製品に使用する企業として、FOSSコミュニティと良好な関係を創出し、維持することは非常によいことです。自身が使用し、商用製品にデプロイしているFOSSプロジェクトに関連する特定のコミュニティについては特にそうでしょう。

さらに、FOSS関連組織や団体との良好な関係は、コンプライアンスを履行する最良の方法について助言を得る上で、大いに助けになるでしょう。また、コンプライアンス上の問題についても助けてくれるでしょう。

ソフトウェア コミュニティとの良好な関係もまた、双方向コミュニケーションに役立つことでしょう。（たとえばソフトウェアの改良をアップストリームに提供し、コミュニティのソフトウェア開発者からサポートを受けるといったこと）

FOSSコンプライアンス プロセスは、FOSSコミュニティにおける良好な協働関係を確立するための重要な要素です。

Your FOSS compliance process is a building block to establishing good working relationships within the FOSS community.

理解度チェック

- ・FOSSコンプライアンスではどのようなタイプの落とし穴がありますか？
- ・知的財産に関する失敗例を1つ挙げてください。
- ・ライセンス コンプライアンスでの失敗例を1つ挙げてください。
- ・コンプライアンス プロセスでの失敗例を1つ挙げてください。
- ・コンプライアンスを優先することのメリットにはどのようなものがありますか？
- ・コミュニティとの良好な関係を維持するメリットにはどのようなものがありますか？

落とし穴は、次に大別されます： 知的財産（IP）における失敗、ライセンス コンプライアンスでの失敗、コンプライアンス プロセスでの失敗

IPでの失敗の例は、プロプライエタリ コードとオープンソース コードの混 合です。これは企業が望まない形でプロプライエタリ ソフトウェアを一般公 開させる結果になりかねません。

ライセンス コンプライアンスでの失敗の例としては、オープンソース ソフ トウェアの改変部に印付けすることを怠る、そのソフトウェアに含まれるオ ープンソース ソフトウェア コンポーネントを適切に記載していない、もし くはそのソフトウェアに対応するすべてのソースコードを入手可能にしてい ない、などがあります。

コンプライアンス プロセスでの失敗例として、オープンソース ソフトウェアの監査、レビュー、承認に関わるプロセスの失敗があります。監査者が、レポート中の全警告アイテムを「放棄した（Waived）」、またはレビューや 承認プロセスに時間がかかりすぎた、などです。

コンプライアンスを優先することのメリットには、FOSSの使用をより効果的 なものにできることや、 オープンソース コミュニティと良好な関係を構築

できるといったことがあります。

コミュニティとの良好な関係を維持するメリットには、FOSSライセンスの要求への対応をよりよく評価できるようになること、FOSSの使用とコントリビューションについてより良い双方向コミュニケーションが得られること、などがあります。

Pitfalls can occur under the following categories: IP failure, license compliance failure, and compliance process failure.

An example of IP failure would be commingling of proprietary code and open source code, which may result in making proprietary software available to general public despite company's preference.

An example of license compliance failure would be a failure to mark an open source software after modification or to properly list the open source software components in the software or to make the complete and corresponding source code available.

An example of compliance process failure would be a failure in the process related to audit, review, or approving the open source software. Auditors "waived" all the red-flagged items in a report, or that the review and approval process takes too long.

The benefits of prioritizing compliance are that you become more efficient in your use of FOSS, and that you build a better relationship with the open source community.

The benefits of maintaining a good community relationship are that you can better assess how you can comply with the FOSS license requirements, and you have a better two-way communication with regard to contribution and use of the FOSS.