



ロボット革命イニシアティブ協議会
Robot Revolution & Industrial IoT Initiative

オープンソースを活用した ロボット開発のための ライセンス・特許ガイドライン

2019 年 5 月

Version 1.0

オープンソースソフトウェアを自社ロボット開発に利用したり、自社ロボット用のソフトウェアをオープンソースとして公開する際のライセンスや特許の取り扱いに関するガイドライン

ロボット革命イニシアティブ協議会
ロボットイノベーション WG
ロボットソフトウェアライセンス・特許調査検討委員会

目次

1. はじめに.....	4
1.1. ロボット開発における OSS に利用の現状	4
1.2. OSS に利用におけるライセンス・特許に関する問題点	4
1.3. 本ガイドラインにて提示する情報	5
1.4. 対象読者	5
1.5. 委員名簿	8
2. ソフトウェアライセンス	9
2.1. ソフトウェアの著作権	9
2.1.1. 著作権法により保護されないもの	11
2.2. 著作権により保護される権利.....	11
2.2.1. 著作人格権.....	12
2.2.2. 著作権.....	14
2.2.3. ソフトウェアの使用と利用の違い	17
2.2.4. オープンソースソフトウェアの利用許諾	18
2.3. オープンソースライセンス	19
2.3.1. ライセンスの種類・制約の種類	21
2.4. OSS ライセンスにかかわる係争事例と対策	23
2.4.1. GPL 違反事例	23

3. その他の著作権とライセンス	26
3.1. ドキュメントの著作権	26
3.1.1. クリエイティブ・コモンズ・ライセンス	26
3.1.2. GFDL (GNU Free Documentation License)	29
3.2. 形状データに関する著作権とライセンス	30
CAD データの著作権	30
3.3. AI にかかわる知的財産権	32
3.3.1. AI 学習に用いるデータの著作権	34
3.3.2. 学習済みモデルが生成するコンテンツの知的財産権	35
3.3.3. ソフトウェア・AI を用いたシステムに関する知的財産権	36
4. ロボットのソフトウェアと特許	38
4.1. 特許庁による特許動向調査	39
4.2. ソフトウェア特許	41
4.3. ソフトウェア特許と係争事例	43
4.3.1. 一太郎・花子特許侵害事件	43
4.3.2. GIF 特許事件	44
5. ロボットシステム開発における OSS の利用と公開	46
5.1. OSS ポリシー策定	48
5.1.1. OSS 利用時のメリット・リスクの同定	48
5.1.2. OSS 公開時のメリット・リスクの同定	54
5.1.3. OSS 利用ポリシーの策定	57
5.1.4. OSS ガイドラインの作成	59
5.2. ポリシーに基づく OSS の利用	61

5.2.1.	体制の構築と運用	62
5.2.2.	教育の実施	62
5.2.3.	社内からの問い合わせ対応	63
5.2.4.	OSS 利用の申請・審査・承認	63
5.2.5.	脆弱性リスクへの対応	64
5.2.6.	特許の把握	64
5.2.7.	開発委託時の注意	65
5.2.8.	意図しない OSS 混入の検査	65
5.2.9.	利用している OSS の把握	67
5.2.10.	出荷前チェック	69
5.2.11.	外部からの問い合わせ対応	69
5.2.12.	保守更新作業	69
5.3.	ポリシーに基づく OSS としての公開	70
5.3.1.	OSS としての公開の申請・審査・承認	70
5.3.2.	公開する著作物の範囲(ソースコード、モデル、ドキュメント)	70
5.3.3.	ライセンスの設定	71
5.3.4.	著作権表示	71
5.3.5.	公開のための手順	72
5.3.6.	公開後のメンテナンス・サポート体制	73
5.3.7.	外部からの貢献に対する対応(ソース取り込み、注意点)	74
5.3.8.	開発、サポートの終了時の対応	74
6.	終わりに	75
7.	参考文献	76

1. はじめに

1.1. ロボット開発における OSS に利用の現状

近年、ロボットシステムの複雑化や応用範囲の拡大に伴い、ロボット開発においてもオープンソース・ソフトウェア（以下 OSS: Open Source Software）が用いられることが増えつつある。この流れは、ROS（Robot Operating System）や RT-Middleware（Robot Technology Middleware）など、ソフトウェアのモジュール化と再利用を促すソフトウェアプラットフォームの利用拡大に伴い、一層加速している。海外では、こうした OSS を利用したロボットシステムの開発とアプリケーションの開拓およびビジネス化が拡大している。日本国内でも、2018 年に SONY から発売された aibo では、500 以上のオープンソースが利用されるなど、ロボットシステムを開発するうえで OSS を利用することは不可欠となっている。

1.2. OSS に利用におけるライセンス・特許に関する問題点

一方で、ロボットメーカーがロボットを製品化するうえで、あるいはシステムインテグレータがシステムを開発するうえで、OSS の品質や責任の所在の問題とともに、ライセンスや特許に関する取扱いが難しいといった声が聞かれる。実際のところ、OSS を利用するうえで、ライセンス条項に従い適切に利用することは、自社で開発し著作権を保有するソフトウェアを利用することに比べると、潜在的リスクを抱えることになるのは事実である。一方で、複雑化するロボットシステムの開発コストを削減し、いち早くリリースすることが求められる昨今において、OSS は適切に利用すればそうしたリスク以上のベネフィットが得られることは、TV や AV 機器・家電・オフィス機器などが近年では OSS や Linux 等オープンソース OS を活用している現状からも理解できる。こうした先行事例では、OSS のライセンスや特許の問題を体系的に整理し、社内における利用ポリシーを定め、開発プロセスと連動したチェック・承認体制を構築することで、こうした潜在的リスクに対処している。また、こうした対処方法に関しては IPA（独立行政法人 情報処理推進機構）などでもガイドライン[1]としてまとめられている。

1.3. 本ガイドラインにて提示する情報

ロボットのソフトウェアの部分における OSS 利用時のライセンスや特許に関する課題とその対策については、先行事例や上記[1]のガイドラインが参考になる。一方で、ロボットというハードと密接に関係する製品・システムの性質上、ロボットハードウェアの CAD モデルデータや各種物理パラメータ情報の二次利用や、システムの内部情報にかかわるドキュメント情報の公開といった、ソフトウェア一般では考慮されてこなかった課題に直面するケースが今回明らかとなった。特に、これまで OSS をダウンロードし利用する際のライセンスや特許の問題については、数多く議論されガイドライン化されているが、上述のように、ソースコード、CAD モデル等、自社開発したものをどのように公開するかについてはあまり整理されていない。また、国内では OSS に対する貢献、すなわち既存の OSS のバグの修正や、改善提案やそのためのソースコードの本家開発元への提供、についての体系的議論は少ない。

本ガイドラインでは、こうしたロボットシステムに特有のオープンソースの取り扱いについて主に整理するとともに、OSS に対する貢献に関する諸問題についても取り扱う。

1.4. 対象読者

オープンソースを製品開発に利用することは、近年の TV をはじめとするデジタル家電など、様々な分野で行われており、そうした先行領域において、大企業などではすでにオープンソースの利用の仕方について、主にライセンスや特許問題に対応するためのガイドライン化やポリシー制定などが行われている場合もあります。一方で、中堅・中小企業や、ハードウェア開発がメインで、ソフトウェアの開発をそれほど得意としていないロボット開発企業等においては、オープンソースの活用は未知の領域であるところも多いはずです。

本ガイドラインは、そのようなオープンソースを活用した開発に慣れていないが、これからロボットシステム開発にオープンソースを活用したいと考えているメーカーやシステムインテグレータの技術者を対象に、オープンソースを活用するために必要なライセンスや特許に関する基礎的な知識を提供することを目的としています。したがって、主な対象は、オープンソースをあまり利用したことがない、あるいは利用したことはあるけど社内にガイドラインや利用体制などが十分構築されていない、メーカーやシステムインテグレータ等の技術者・研究者を想定しています。さらに、オープンソースに

はすでに公開されているソフトウェアを利用するだけでなく、既存のオープンソースプロジェクトに対して、バグレポートや機能拡張のソースコードの提供、あるいは開発者個人や企業として新たにオープンソースプロジェクトを立ち上げるなどの貢献を行う、いわゆる公開の側面もあります(図 1)。本ガイドラインでは、オープンソースの利用から一歩踏み込んで、公開の面に関してもガイドラインを提供します。

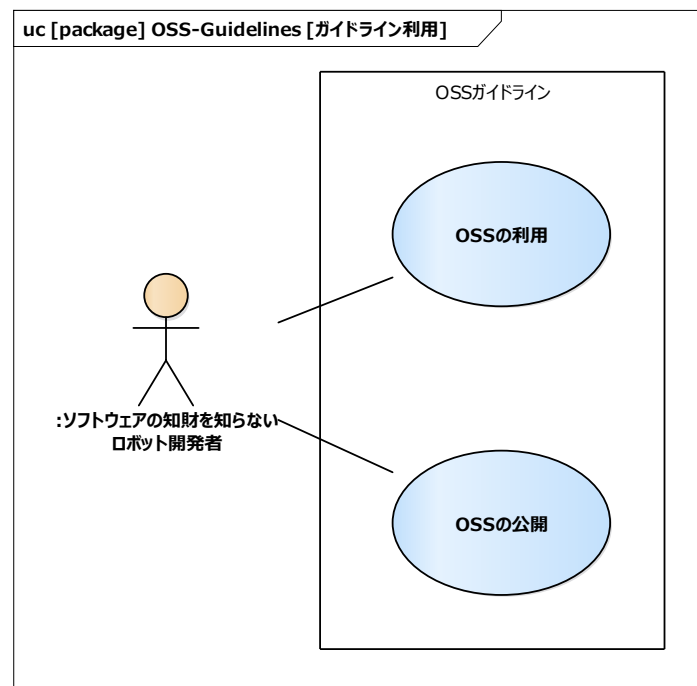


図 1 オープンソースにおける利用と貢献のケース

書籍「OSS ライセンスの教科書」では、企業においてオープンソースをどのように扱うかについて、ライセンスの問題を中心にかなり詳細に解説が述べられています。この「OSS ライセンスの教科書」では、組織におけるオープンソースの導入・活用レベルには、以下のレベル 0～レベル 4 まであるとしているのでここで引用します。

本ガイドラインでは、主にレベル 0～レベル 3 程度の社内体制が十分にできていない企業やその技術者を対象にしているといえます。加えて、オープンソースを適切に利用するには、社内の他の技術者や他の部門(知財・法務部門、品質管理部門等)、あるいは経営層とも連携することが重要です。したがって、もしそれらの部門においてもオープンソースの適切な利用に関する知識が十分に

ない場合には、関係部門を巻き込んで体制を徐々に構築していくことが肝要です。本ガイドラインは、その際の入り口になる情報を提供するものと考えてください。

表 1 OSS への対応レベル(引用:OSS ライセンスの教科書 p.213)

体制レベル	段階
レベル 0	OSS の利用は一切禁止されている
レベル 1	ごく少数のメンバーに限られた OSS を適切に使っている
レベル 2	主に使われている OSS については適切な対応ができる段階
レベル 3	多くの OSS について適切な対応ができる段階
↓	
組織的体制の構築・OSS 利活用の本質をとらえる	
↓	
レベル 4	(OSS 対応チームが組織されたうえで)多くの OSS について適切な対応ができる段階、加えてコミュニティへの貢献もできるレベル

1.5. 委員名簿

【委員長】 (国研)産業技術総合研究所 ロボットイノベーションセンター ロボットソフトウェアプラットフォーム研究チーム長	安藤 慶昭
【副委員長】 (国研)産業技術総合研究所 ロボットイノベーションセンター ロボットソフトウェア研究ラボ長	原 功
【委員メンバー】	
(学)会津大学 情報システム学部門 ロボット工学講座 教授	成瀬 継太郎
(学)会津大学 復興支援センター 特任教授	屋代 眞
(株)セック 開発本部 第四開発部 テクニカルマネジャー	中本 啓之
THK(株) シニアクリエイティブプロデューサー	永塚 正樹
THK(株) 事業開発統括部	三好 崇生
(株)東芝 研究開発センター 機械システムラボラトリー 主任研究員	山本 大介
トヨタ自動車(株) Tーフロンティア部	高橋 太郎
(一財)日本品質保証機構 認証制度開発普及室 主幹	櫛引 豪
パナソニック(株) プロダクト解析センター 電気ソリューション部 課長	岡本 球夫
富士ソフト(株) エリア事業本部 中部支社 マニュファクチャリングソリューション部 プロジェクトマネージャー	石川 貴雄
(株)安川電機 開発研究所 つくば研究所 課長代理	平田 亮吉
【オブザーバ】	
(国研)新エネルギー・産業技術総合開発機構 ロボット・AI 部 主査	増田 昌庸
富士ソフト(株) 技術管理統括部 技術開発部 技術管理室 リーダー	越野 愛美
(敬称略)	

2. ソフトウェアライセンス

2.1. ソフトウェアの著作権

ソフトウェアライセンスについて考える上で、「ソフトウェアは著作物」であり、かつ著作権の対象となる知的財産であることを理解する必要があります。

日本の著作権法の定義によれば、著作物とは「思想又は感情を創作的に表現したものであって、文芸、学術、美術又は音楽の範囲に属するもの」(2条1項1号)と定められており、この定義からは、ソフトウェアは著作物に含まれないように見えます。しかしながら、著作権法第2条では、プログラムとは、「電子計算機を機能させて一の結果を得ることができるようにこれに対する指令を組み合わせたものとして表現したもの」と定義され、その上で第10条における「著作物の例示」において「プログラムの著作物」として示されており、ソフトウェアは著作権法により保護される著作物の一種となっています。

日本の著作権法では、国際条約(ベルヌ条約)に基づき、**無方式主義**を採用しており、特段の登録手続きを経ることなく、著作者が著作物を創作した時点で、著作権法で定める知的財産的権利が認められます。さらに、この著作権は、日本国内のみならず、ベルヌ条約締結国においても同様の保護を受けることができます(図2)。

逆に言えば、日本国外で開発されたソフトウェアについても、ベルヌ条約締結国間で同様に保護されるため、インターネット上で入手可能なオープンソースソフトウェアについても、ほとんどの場合開発者の国籍によらず著作物として保護されます。こうしたオープンソースソフトウェアは取扱いを誤ると企業においてはコンプライアンス上のリスクとなるため、ソフトウェアが著作権法等の法律により保護される知的財産であることを念頭に置き、その法律を遵守したうえでの適切な取扱いを行う必要があります。

【POINT】

ソフトウェアは国を超えて著作物として保護され、適切な取扱いが求められる。

ベルヌ条約：正式名称は「文学的及び美術的著作物の保護に関するベルヌ条約」であり、160 か国余りが加盟する著作権に関する基本条約。著作者による明示的な主張・宣言がなくとも自動的に発生すること（無方式主義）、著作権が他の加盟国においても、その国の著作者と同等の保護を受けることができる（内国民待遇）等を謳った国際条約。

参考: WIPO (World Intellectual Property Organization),
<https://www.wipo.int/treaties/en/ip/berne/index.html>

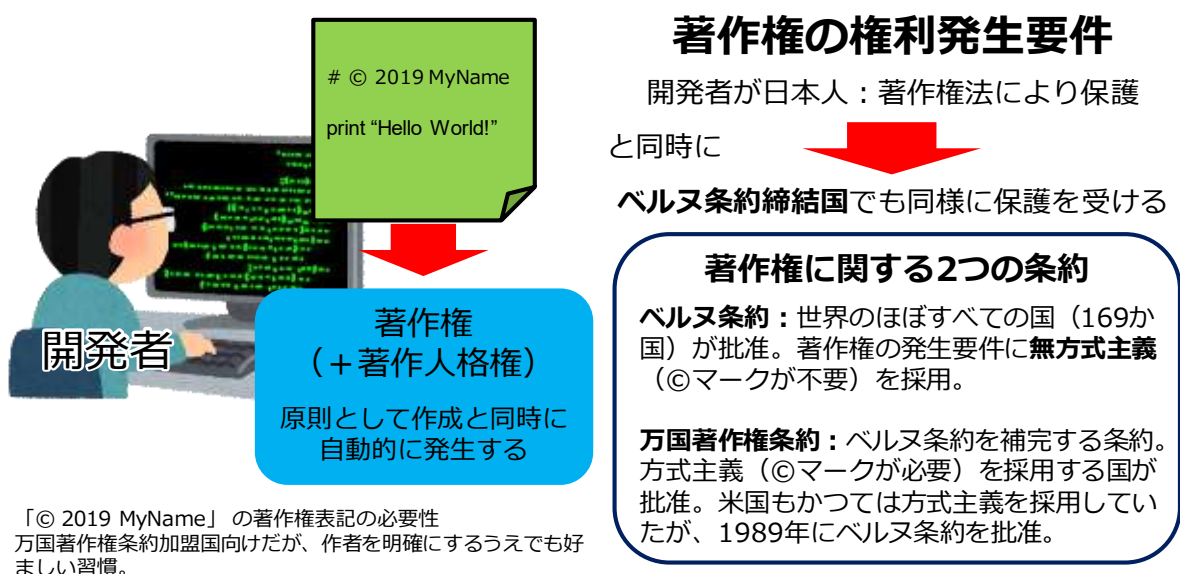


図 2 ソフトウェアの著作権の発生要件

しかし、だからと言ってオープンソースソフトウェアの利用を怖がる必要はありません。オープンソースソフトウェアには、世界中の優秀なソフトウェア開発者が協力して開発し、製品として販売されるものより高性能・高機能なものも多数あります。これらの多くは、一定の条件下では自由に利用、二次配布、つまり製品などに組み込んで出荷することも認められており、この条件を的確に把握し対応することで、こうしたソフトウェアを無償かつ自由に使用することができます。

本ガイドラインでは、オープンソースソフトウェアを活用するために必要な事項と注意点をわかりやすくまとめています。

2.1.1. 著作権法により保護されないもの

上で述べたように、著作権法においてプログラムとは「電子計算機を機能させて一の結果を得ることができるようこれに対する指令を組み合わせたものとして表現したもの」であり、これにかかわる様々な権利を保護することを目的としています。つまり、著作権法はソフトウェアの「表現」を保護するためのものであり、ソフトウェアに内在するアイデアやノウハウは著作権法によっては保護されないのです。こうしたアイデアやノウハウは、その他の知的財産権（特許権、実用新案などの工業所有権）として保護されるべきであるという考え方に基づいています。

なお、著作権法では第 10 条の九第 3 項において、著作権法により保護は「プログラム言語、規約、解法」には及ばないと明記されています。プログラム言語は、ソフトウェアを記述する言語そのもので、表現ではなく表現の手段であるとの観点から保護の対象外となっています。また、規約とはインターフェースや API、あるいは通信におけるプロトコルなどを指し、これらも著作権法の保護の対象外となっています。最後に解法ですが、これはアルゴリズムや手順を示したフローチャートであり、このようなアイデア自体は著作権保護の対象外となっています。また、ソフトウェアが実現する機能自体も保護が及びません。あくまで著作権はこうしたものを、プログラムとして表現したものを保護する法律であることを留意する必要があります。

【POINT】

著作権法はプログラムの表現を保護する法律。言語、インターフェースやプロトコル、アルゴリズム、あるいは機能などは著作権法で保護されない。

2.2. 著作権により保護される権利

ソフトウェアも著作物として国内外でその権利が保護されることは上に述べたとおりですが、その権利にはどのようなものがあるのでしょうか？

日本の著作権法においては、著作者がその著作物に対して有する人格的利益を保護する権利として著作人格権が、また、著作物を活用して収益や名声などを得ることができる財産的権利としての著作財産権が規定されています。

2.2.1. 著作人格権

著作人格権（第 18 条～第 20 条）は、著作権者が著作物を創作した直後からその著作者に紐づけられる権利（一身専属性）であり、**他人には譲渡でない権利**とされています（著作権法 59 条）。ベルヌ条約においても同様に、たとえ著作権を他人に譲渡した後であっても、著作人格権は著作者が保持する権利として規定されています。逆に言えば、著作権は第三者に譲渡することが可能であり、著作権を譲渡された著作権者が後述する著作権を行使することができるのです。一方、著作人格権により規定される権利は、著作権者が変わったとしても、原著作者により保持され続けます。

なお、著作者が死亡した場合（法人の著作者の場合は解散した場合）には、日本の国内法では著作人格権は消滅する（相続することもできない）とされますが、ベルヌ条約においては、著作者の死後も権利の保護を求めているなど若干に違いがあります。

著作人格権には以下の 4 つ（ベルヌ条約的解釈では 3 つ）の権利があります（図 3）。

- **公表権**：著作権法第 18 条。未発表の著作物を公表する権利。公表・発表と同時に公表権は消失する。なお、ベルヌ条約にはこの規定はない。
- **氏名表示権**：著作権法第 19 条。著作物の公表に際し、著作者の実名や変名を著作者名として表示する、またはしない権利。
- **同一性保持権**：著作権法第 20 条 1 項。著作者の意に反して、著作物の変更・切除・その他の改変を禁止する権利。
- **（名誉声望保持権）**：著作権法 113 条 6 項。著作者の名誉又は声望を害する方法によりその著作物を利用する行為を禁止する権利。ベルヌ条約においては、同一性保持権と一体の条文で規定されているが、国内法では改変を伴わない利用を考慮し独立した規定となっている。

ただし、著作権法において、ソフトウェアの著作権は比較的新しく追加された概念であり、また他の著作物にはない特性を持つため、ソフトウェアすなわち「プログラムの著作物」について、他の著作物と異なる取り扱いが規定されている部分があります。



著作人格権

- 公表権
- 氏名表示権
- **同一性保持権**（改変の制限）

著作権

- **複製権**（コピーの制限）
- 上演権および演奏権（演劇等）
- 放送権、有線送信権
- 口述権（講義、朗読等）
- 展示権（美術・写真等）
- 上映権および頒布権（映画）
- 貸与権
- **翻訳権、翻案権等**（改変、拡張、移植等の制限）
- **二次的著作物の利用に関する現著作者の権利**

図 3 著作権により保護される権利

第 20 条の 3 同一性保持権

第 20 条では、著作物に対する現著作者が保有する著作人格権の一つである同一性保持権が規定されています。ただし、プログラムの著作物については、その特性上、特定の計算機上で利用できない場合や、そのソフトウェアをより効果的に使用するためには、改変を許す規定が別途定められています。つまり、著作者のものとは異なる環境上で動作させるうえで必要な改変、あるいは移植のための変更は、同一性保持権を侵害しないとみなされるのです。

このように、プログラムの著作権においては、ソフトウェアの特性に配慮して、同一性保持権が若干緩められています。

【POINT】

著作人格権は譲渡できない著作権者に固有の権利。

2.2.2.著作権

著作権は、上述したように、著作者が創作した著作物に対して有する知的財産権(知的所有権)の一種であり、上記著作人格権以外の著作権は著作財産権ともいわれます。著作人格権とは異なり、著作権は他人に譲渡することが可能です。

著作権自体は、美術、音楽、文芸、学術などに関する著作物一般について、以下の権利が定められています。したがって、ソフトウェア、とくにオープンソースソフトウェア一般の利用形態において適さない条項も存在します。以下に、著作権法において保護される知的財産権を示します(図 3)。

- 複製権(コピーの制限)
- 上演権および演奏権(演劇等)
- 放送権、有線送信権
- 口述権(講義、朗読等)
- 展示権(美術・写真等)
- 上映権および頒布権(映画)
- 貸与権
- 翻訳権、翻案権等(改変、拡張、移植等の制限)
- 二次的著作物の利用に関する現著作者の権利

個々の権利についての解説は省きますが、著作人格権同様、著作権で規定される種々の権利についても、条文上「プログラムの著作物は除く」等の但し書きで他の著作物とは区別され、別途プログラムの著作物に関して定めている条文があり、これを以下に示します。

第 15 条 職務上作成する著作物の著作者

第 15 条においては、法人等の従事者が職務上作成する著作物については特段の定めがない限り、その著作者は作成した個人ではなく法人となることが定められています(職務著作)。職務著作に対する著作権が成立する要件は、

- ① 法人その他使用者(法人等)の発意に基づくものであること
- ② 法人等の業務に従事する者が職務上作成するものであること

③ 法人等が自己の著作の名義の下に公表するものであること

④ 作成の時ににおける契約、勤務規則その他に別段の定めがないこと

となっていますが、プログラムの著作物に限っては、③の「法人等が自己の著作の名義の下に公表するものであること」が除かれています。これは、他の著作物が一般に公表されることを前提として創作されるのに対し、ソフトウェアは企業（等法人）が自社内で（外部に公表せずに）使用することを目的に作成される場合があることがあり、これを考慮した条項となっています。

つまり、外販しない、あるいはオープンソースにしない、企業内で開発されたソフトウェアに対しても、作成した時点でソフトウェアの作成者ではなく法人に対して著作権が認められるということです。

第 47 条の 2 プログラムの著作物の複製物の所有者による複製等

プログラムの著作物については、その複製物の所有者は、複製または翻案をすることができると定められています。つまり、ソフトウェアのバックアップ（複製に相当）を行ったり、ソフトウェアの拡張・流用や移植（翻案）を行ったりすることができると定められています。

さらに、翻案によって創作される二次著作物についても複製を行うことも認められています。つまり、元のプログラムを改変・拡張等した二次著作物としてのソフトウェアについても、バックアップ程度の複製ならば許可されるということです。ただし、その場合、二次創作物に対しても原作者の貢献部分については現著作者の権利が及ぶことは考慮しないといけません。つまり、あるプログラムを改変・拡張等したプログラムあくまで二次著作物であり、バックアップ以上の複製や配布等は許されないということになります。

さらに、複製物の所有者が、所有権を第三者に譲渡する、あるいは著作者が許諾する条件を満たさなくなるなどして、その所有権を失った場合には、複製物を保存してはいけない、つまり保持しているソフトウェアを破棄しなければいけないこととなっています。

第 49 条 複製物の目的外使用

上記の 47 条に規定に従って複製または翻案したソフトウェアについては、その複製物や二次著作物を頒布または公衆に提示、つまりオープンにすることは、原作者が占有する複製権を侵害する

行為とみなされます。また、所有権を有しないソフトウェアに対して改変・拡張等を行うことも、同様に原作者が占有する翻訳権、翻案権等を侵害する行為とみなされます。

つまり、人のソフトウェアやその二次著作物を勝手にオープンにしてはいけない、また、所有権を持たないソフトウェアを勝手に改変してはいけないということになります。

第 78 条の 2 プログラムの著作物の登録に関する特例

日本の著作権は無方式主義を採用しており、特別の登録手続きなしに著作権が認められることを上で述べましたが、著作権に関する係争が発生した場合、事実関係の証明を容易にするために、著作権の登録制度が存在します。通常の著作物については、文化庁に対して著作物の情報を登録するのに対して、ソフトウェアについてはその特性上名称などだけでは著作物を特定できないため、「プログラムの著作物に係る登録の特例に関する法律」に定められるように、ソフトウェアのコピーとともに、財団法人ソフトウェア情報センターに対して登録を行うことができます。

ただし、オープンソースの世界では別途そのような登録を行うことはまれであり、一般には Source Forge や github など、オープンソースプロジェクトのホスティングサイト上で開発を進め、オープンソースのコミュニティに認知されることで、原作者の権利が暗黙的に守られているといえるでしょう。

第 113 条 侵害とみなす行為

業務で使用するソフトウェアの中に、仮に著作権を侵害するソフトウェアが含まれていた場合、そのソフトウェアを取得した時にそのことを知っていた場合に限り、使用者が著作権を侵害したとみなされます。つまり、買ってきたソフトウェアやダウンロードしてきたソフトウェアが著作権を侵害して作成されたソフトウェアの場合、入手した時点でそのことを知らなかった場合には、使用者は著作権を侵害していることにはならず、かつ、今後も使用し続けることもできます。ただし、複製や翻案は行うことはできません。

【POINT】

ソフトウェアのコピー、改変や再配布などは原作者に無断で行うことはできない。

なお、このようにソフトウェアに関する著作権の規定が特別扱いされているように見える理由は、もともと通産省がソフトウェアの著作権保護のための新規の法律を制定しようとする際に、当時の米国との貿易摩擦から、米国風の著作権法の拡張によるソフトウェアの著作権化という枠組みに従って、著作権法を拡張することでソフトウェアの知的財産保護を進めざるを得なかったという事情があります。

以上述べたように、著作者は著作物に対して、著作権法やベルヌ条約等で規定されている多くの権利が独占的に認められており、これを利用しようとする場合には、多くの制約があること、また「プログラムの著作物」として規定されるソフトウェアに関しては、他の著作物とは若干異なる運用がなされることを注意する必要があります。

2.2.3. ソフトウェアの使用と利用の違い

著作権法上、「使用 (use)」と「利用 (exploit)」は区別されています。「使用」は本を読む、絵を鑑賞する、そしてソフトウェアに関しては、コンピュータ上で動作させるといった、著作物本来の用途で用いることを指します。一方「利用」は、複製する、演劇を上演する、音楽を放送するといった本来の用途以上の著作権上の権利を行使することを指し、ソフトウェアにおいては、改良や機能拡張等の行為がこれに当たります。

なお、著作権法には使用权という権利は規定されていません。したがって、ソフトウェアを使用すること自体は、著作権を侵害するものではありません。唯一、使用による著作権の侵害とされる行為は、上記 113 条の「侵害とみなす行為」に規定されている方法、つまり、あるソフトウェアが著作権を侵害していることを知ったうえで使用すること、が当てはまります。ただし、実際にソフトウェアを使用するときには、CD/DVD-ROM、USB メモリ等のメディアや Web サイトから、利用しようとするコンピュータ上にインストール(複製)しますので、著作権をもつ著作者から、こうした権利(ソフトウェアの複製)を行使することの許可(許諾)が必要となります。

以上のように、ソフトウェアの著作権者は様々な著作権を独占的に持っているため、第三者がそのソフトウェアを使用するためには、その権利の一部または全部について原著作者から許諾を受ける必要があります。ソフトウェア企業等が販売するパッケージソフトウェアにおいては、通常インストール時に、予め定めた「規約に同意することを条件に使用を許諾する」といった同意を求める形で、売

買契約ないしは使用許諾契約を結んでいます。それ以外にも、記憶媒体のパッケージを開封（シュリンクラップ契約: Shrink-wrap contract）することを規約への同意とみなす方法、インストール時に画面上で同意を求めるクリック（クリックラップ/クリックスルー契約: click-wrap/click-through contract）を行うことで同意とみなす方法等がとられている。こうした売買契約や使用許諾契約には、通常複製数の制限や第三者への譲渡の禁止、リバースエンジニアリングの禁止、改変・改造の禁止等、著作権法で保護される事項以外により保護制限を強める条項を盛り込んでいる場合が多い。この場合、契約に基づき使用許諾を得ているため、たとえ著作権法に違反しなくとも、使用許諾を逸脱した場合は契約違反となるため注意が必要である。

2.2.4.オープンソースソフトウェアの利用許諾

一方、オープンソースソフトウェア（OSS）では、ソースコードがオープンにされているという性質上、一次利用者は単に「使用」するだけであれば特段の許諾等なしにソフトウェアを使用することはできる。使用とは上述のように単なる著作物を本来の用途で用いることであり、ソフトウェアの実行やソースコードの閲読・コンパイルが該当する。そのような意味で、OSS の場合、ソフトウェアの使用は自由です。

一方、利用（exploit）とは、複製や翻案の行為を指し、ソフトウェアを改変したり、拡張・機能追加を行う、あるいはそのようにして作成した二次著作物（オープンソースライセンスにおいては派生物（derived works）と呼ぶ）を再頒布したり、新たに作成したソフトウェアの一部として、対象のソフトウェアを同梱する行為を含みます。

これらの行為（ソフトウェアの利用）は著作権法によって制限されているが、OSS の場合は一定の条件に従うことにより、これらの権利を含むソフトウェアの利用を許諾する。こうした利用許諾を与えるものをライセンス（オープンソース・ライセンス）と呼びます（図 4）。つまり、著作権法などにより本来著作者に独占的に与えられる「利用」の権利を、ソフトウェアの再利用と技術の発展への貢献を意図して、広く第三者に許諾しているソフトウェアをオープンソースソフトウェアとよび、その規約としての利用許諾をオープンソースライセンスと呼んでいるのです。

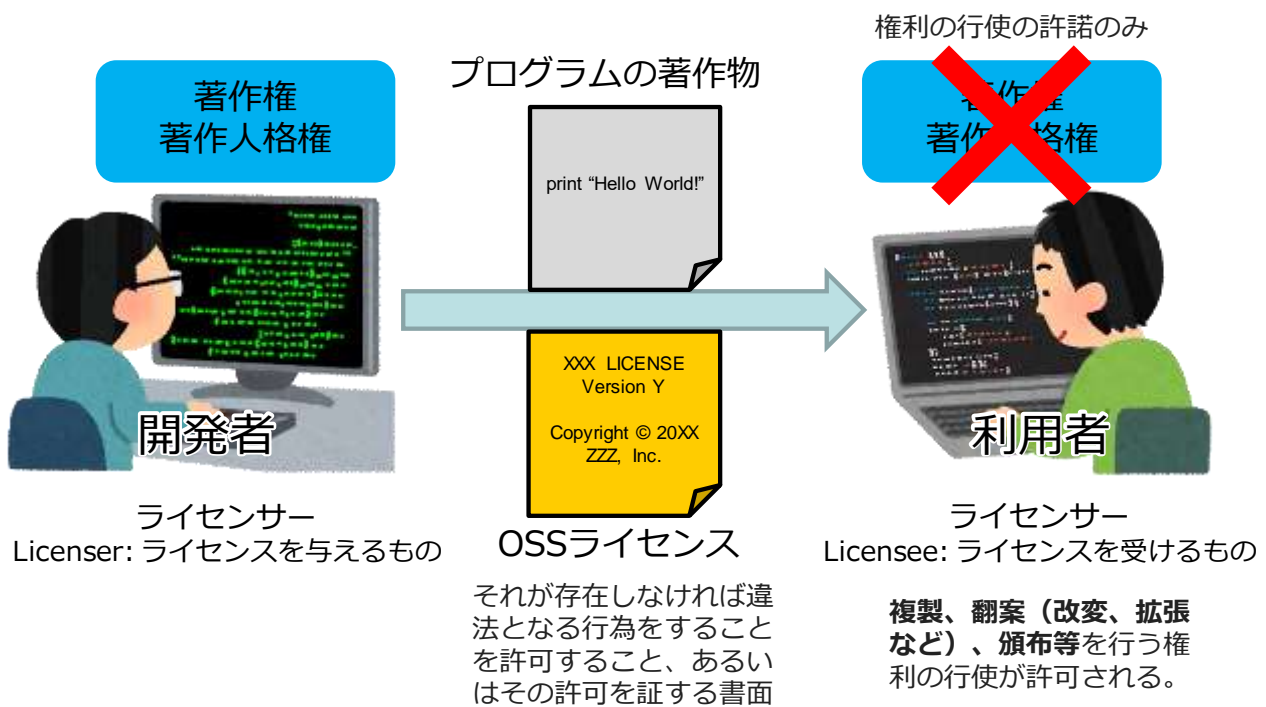


図 4 OSS ライセンスにより許諾される行為

2.3.オープンソースライセンス

上述のように、オープンソースライセンスは「一定の条件」のもとに、第三者にソフトウェアの利用を許諾するものです。ただし、オープンソースライセンスと言っても、代表的なものだけでも GPL (GNU Public License)・LSPL (Lesser GNU Public License)、BSD (Berkley Software Distribution) ライセンス、MIT (Massachusetts Institute of Technology) ライセンス、Apache ライセンスなどが知られており、これ以外にも非常に多くの種類のライセンスが存在します。たいていのオープンソースプロジェクトは、こうしたメジャーなライセンスのいずれかを採用し、その条件のもとに配布されています。利用を許諾するための「一定の条件」にも、ライセンスごとに実に様々な条件が設定されており、こうしたオープンソースを利用する際には、それぞれの条件や制約を理解したうえで利用する必要があります。

オープンソースライセンスの定義には特定の標準はありませんが、一般に OSI (Open Source Initiative、オープンソースイニシアチブ)、FSF (Free Software Foundation、フリーソフトウェア財

団)といったオープンソースの利用促進を図る団体がそれぞれ定義を定めており、これらの定義に合致しているライセンスが、オープンソースライセンスとして認知されています。

たとえば、OSIによるオープンソースライセンスの定義(OSD, Open Source Definition[7])には、1)自由な再頒布の許可、2)ソースコードの入手性・可読性の確保、3)派生物の自由な利用の許諾等、10項目のオープンソースソフトウェアの定義が定められており、これらの条件に合致した82のライセンスをオープンソースライセンスとして認めています(2019年4月現在)。一般に、オープンソースライセンスに共通する条項としては、

- 自由な使用、自由な利用の許諾
- 再頒布条件の表示
- 著作権者の免責条項(無保証であることの明示)
- 著作権表示
- 特定の個人・グループ、利用用途による差別・区別の禁止

などがあります、これ以外に、ライセンスによって異なる条項としては、

- 特許(報復)条項(GPLv3, EPL等)
- 商標、その他知的財産権に関する条項(Apache License 2.0等)
- コピーレフト条項(GPL等)

などが、ライセンスによっては定められていることがあります。

オープンソースのライセンスの詳細については、「OSSライセンスの教科書」[2]等、詳しい解説を参照するとともに、自分が使用するオープンソースのライセンスについては、一度ライセンスの原文を読んでみることをお勧めします。ここでは、オープンソースライセンスの大まかな種類・類型と、それぞれのライセンスが課している条件を簡単に説明するにとどめます。

2.3.1. ライセンスの種類・制約の種類

オープンソースには様々な種類のものがありますが、以下のようにいくつかで大別することができます。

コピーレフト型ライセンスと非コピーレフト型ライセンス

コピーレフト (Copyleft) とは、FSF (フリーソフトウェア財団) が主張する「ソフトウェアの自由な利用」を促進するためのコピーライト (著作権、Copyright) と相対する概念 [8] で、著作者が著作権を保持したまま、二次的著作物も含めて、すべての者に対して著作物の利用・再配布・改変を保証しようとする考え方です。代表的なライセンスとしては、GPL (GNU Public License) があり、GPL でライセンスされたソフトウェアは、その派生物もすべて GPL ライセンスが適用され、かつ、そのソースコードもオープンにしなければならないという非常に制約の強いものです。これは、GPL 汚染とも呼ばれることもあります。また、GPL では、GPL ライセンスと結合して (静的にしる、動的にしるライブラリをリンクして使用するなどの方法で) 使用するソフトウェアについては、GPL あるいは GPL と両立するライセンスであることを要求しています。こうした制約を少し緩め、動的リンクされるものに対しては必ずしも GPL と両立することを要求しない LGPL (Lesser GNU Public License) のような準コピーレフト型のライセンスもあります。また、コピーレフト型および準コピーレフト型ライセンスを総称して、互惠型ライセンスという区分とする考え方もあります [2]。このほか、準コピーレフト型ライセンスには MPL (Mozilla Public License) や EPL (Eclipse Public License) といったライセンスもあります。

一方で、BSD ライセンスや MIT ライセンスは、非コピーレフト型ライセンスとも呼ばれ、原則として著作権表示・ライセンス表示のみが条件として課され、元のソフトウェアを改変して作成した二次派生物については、ソースコードをオープンにする義務は課されていません。これは、利用者にとっては非常に自由度が大きく、たとえ元のソフトウェアを改変したものを自社製品に組み込んで販売しても、著作権表示・ライセンス表示のみを行えば、改変後のソースコードを開示する必要がありません。このように、ソースコードの二次利用者のコード開示に関して寛容であるという意味で、寛容型ライセンスと区分する考え方もあります [2]。これは、オープンソースを製品に利用しようとする企業にとっては、製品開発に当たって元のソースコードを改変した部分を秘密にできるため、大きなメリットのように見えます。一方で、改変部分をクローズにするということは、元のオープンソースプロジ

エクトにその変更がフィードバックされず、クローズにした企業は独自に改変部分を保守しなければならないということを意味します。表 2 にこれらのオープンソースライセンスによる制限や影響範囲、公開義務の一覧と、これらのライセンスを採用するロボット関連ミドルウェアやライブラリの一覧を示します。

FSF が主張するように、著作者や二次著作者あるいは利用者が、いつでも派生物のソースコードを入手・利用できることこそが、ソフトウェアにおける真の自由であり、ソフトウェア技術の普及・発展に貢献することである、という考え方がある一方で、非コピーレフト型ライセンスのように、商用利用時におけるソースコードをオープンにすることの難しさに配慮し、改変や二次利用に対して寛容な条件を認め、利用者を増やすことがオープンソースソフトウェアの発展に寄与するとの考え方もあります。

以上のように、どのようなライセンスを適用するかは、原著作者がそのソフトウェアやそれを利用する人に対して、ソフトウェアをどのように扱ってもらいたいのか、という意図が込められていることを理解しましょう。また、自分、自社がソフトウェアのソースコードをオープンにする際には、どのようなライセンスを設定するかによって、そのソフトウェアをどのように扱ってほしいかを表明することになることも理解しましょう。

表 2 代表的なオープンソースライセンスの制限と採用するロボット関連ソフトウェア

ライセンスタイプ	修正・拡張	改変影響範囲		公開義務		ロボット関連ミドルウェアやライブラリ
		静的リンクの影響	動的リンクの影響	改変部分	改変部分以外	
GPL	可	有	有	有	有	Player/Stage, ORCA (一部 LGPL)
LGPL	可	有	無	有	有	OpenRTM-aist, Choreonoid
BSD	制限なし	制限なし	制限なし	無	無	ROS, MoveIt!, OpenCV, YARP, URBI

2.4. OSS ライセンスにかかわる係争事例と対策

オープンソースライセンスを遵守しない場合には実際にどのようなことが起こるのでしょうか？ここでは、オープンソースライセンスにかかわる係争事例と対策を見ていきたいと思います。

OSS ライセンス違反のうち、GPL (GNU Public License)の条件を守っていないケースが「GPL 違反」と呼ばれるものです。GPLv2 は Linux kernel に採用されており、元となるオープンソースソフトウェアを複製、改変した著作物にも GPL を適用しなければならない、ソースコードを開示しなければならない等、派生物を作製した人に対しても様々な義務が課せられているのが特徴です。ここで、GPL 違反の事例を紹介します。

2.4.1. GPL 違反事例

FANTEC GPL 違反事例[11]

2013 年 6 月、FANTEC 社が配布したメディアプレーヤーに GPLv2 の OSS が含まれているファームウェアを利用していましたが、開示されたソースコードにはその OSS のコードが含まれておらず、また製品とソースコードのバージョンが異なっていることが、2012 年 5 月、フリーソフトウェア財団ヨーロッパ (Free Software Foundation Europe ; FSFE) 主催のコンプライアンスハッキングワークショップで発覚しました。ドイツのプログラマーで、GPL 遵守を監視するプロジェクト“gpl-violations.org”を立ち上げている Welte 氏が、これを GPL 違反であるとして提訴しました。訴えられた FANTEC 社は、開示したソースコードは、ファームウェアを提供した中国のサプライヤーから受け取ったものであり、サプライヤーによってソースコードの内容は保証されている、と主張しましたが、裁判の結果 Welte 氏が勝訴し、FANTEC 社に対して罰金、弁護士費用の支払い、ソースコードの開示が命じられました。これにより、サプライヤーのコンプライアンスに依存するのは不完全で、製品を製造した FANTEC 社に最終的な責任があることが明示されました。

この事例では以前にも Welte 氏が FANTEC 社に対して GPL 違反を指摘し、話し合いで GPL の条件を遵守することに合意したという背景があり、にもかかわらず再び GPL 違反が発覚したため訴訟に至ったケースです。この事例を見てわかるように、GPL 違反が発覚したとしてもいきなり提訴されることはまれで、たいていの場合は事前に指摘・警告があり、この段階で指摘を真摯に受け止め、OSS コミュニティ側とコミュニケーションを密に取り、迅速に適切な対応を行えば訴訟を回避で

きたでしょう。また、このケースでは、オープンソースライセンス条件の遵守責任が、サプライヤーのみならず、最終製品製造者にあることが示されています。

スキャナプリンタドライバにおける係争事例

国内でスキャナ・プリンタを製造販売する E 社は、Linux 用のスキャナ・プリンタ用ドライバにおいて、多言語対応するために gettext パッケージ（国際化と地域化に対応するライブラリ群）のソースコードの一部を使用し、そのドライバを 2002 年から無料で広く配布していました。このソースコードのライセンスは GPL だったので、これを含むドライバのソースコードも GPL ライセンスを適用する必要がありましたが、E 社はドライバのソースコードを GPL とはしておらず、ソースコードの公開義務もはたしていませんでした。

ユーザがソフトウェアを解析して GPL 違反を発見、ネット上の掲示板に書き込むなどして発覚したことを受け、同社はすぐに Web サイト上で謝罪、試用しているオープンソースのライセンスを明確にするとともに、使用する gettext パッケージを LGPL 準拠のものに差し替え、非公開のコンポーネントについては、LGPL 2.1 に明記されている目的に限りリバースエンジニアリングを許可するよう使用許諾を変更するなど適切な対応をとった結果、これまでの OSS への貢献もあり、OSS コミュニティから逆に賞賛を浴びる結果となったとのことでした。

aibo における対策事例

SONY が 2018 年 1 月に発売した犬型ロボット「aibo」には ROS をはじめとするオープンソースソフトウェアが 500 以上も使われています[15]。主な利用ソフトウェアとしては、ROS (Kinetic) や PCL のようなロボットに特有なものや、SDL2, libogg などマルチメディア系、nginx, jQuery 等のような Web サービス系のオープンソースが利用されています。それぞれのオープンソースが採用しているライセンスも GPL, LGPL, BSD, MIT, Apache など様々であり、それぞれに適切な対処が求められます。SONY のソフトウェアライセンス情報サイト[15]には aibo で利用されているオープンソース毎のライセンスが列挙されています。

製品としてオープンソースを再配布する際に、その製品にどのようなオープンソースが利用されているかをこのような形で明記するのは最低限の対応です。（ライセンスによっては必ずしも利用を明記する必要はない。）例えば、SONY は ROS の最低限のフレームワークを利用しつつ、

rostopicproxy のような ROS パッケージには独自の修正を加えるなど、改変も行っていました。しかしながら、ROS の大半のソフトウェアのライセンスは BSD であり、派生物の開示等は必要なく、クローズドな運用が可能になっています。

一方で、他のライセンスのオープンソースの場合には気を付けなければならない部分もあります。日経 Robotics の記事[16]によれば、aibo 発売の翌日にユーザから、利用オープンソースライセンス一覧に GPLv3 のものが含まれることを根拠に、aibo へのソフトウェアのインストール方法の開示要求がありました。GPLv3 では GPLv2 までではなかった、ユーザが自身の製品にソフトウェアをインストールする手段を用意することを求める条項が追加されていました。

実際には、当初 SONY が提示した利用オープンソースとそのライセンス一覧が誤りであり、実際には組み込まれていない GPLv3 のソフトが含まれていたり、すでに GPLv3 ではないオープンソースが含まれていたものの、ライセンス表記が更新されていなかったなど、ライセンス開示上のミスと、構成管理上のミスとして利用ライセンスのリストを修正しました。

3. その他の著作権とライセンス

3.1. ドキュメントの著作権

ソフトウェアに付随するドキュメント事態も著作物としてみなされるため、著作権保護の対象となります。したがって、その利用にあたっては、著作権の侵害とならないよう取り扱いに留意する必要があります。オープンソースライセンスはソフトウェアに対する利用許諾条件を示すものであり、オープンソースソフトウェアに付随するドキュメントについては、ソフトウェアのソースコードとは個別に扱われることを留意してください。つまり、対象のオープンソースソフトウェアにいついて、改変・再頒布などが許されているからと言って、必ずしも付随ドキュメントについても、改変・再配布が同じように許されているとは限りません。

逆に、自分で開発したソフトウェアをオープンソースとして公開する際に、当然ドキュメントとともに公開されると思われますが、その際、ソースコードとは別にドキュメントの取り扱いについても考慮しておく必要があります。

文章等の著作物の適正な再利用を促進することを目的とし、改変や営利目的利用の可否ごとに、あらかじめいくつかのパターンを定め、著作者が手軽に意思表示を図れるように策定された、クリエイティブ・コモンズ・ライセンス[17]と呼ばれるライセンス群があります。クリエイティブ・コモンズ・ライセンスは、オープンソースの世界のみならず、様々な文書などの著作物をオープンにしたい人々に利用されています。

もし、自分で作成した文書を、再利用を意図して公開する場合、クリエイティブ・コモンズ・ライセンスのいずれかを添えて公開することをお勧めします。

3.1.1. クリエイティブ・コモンズ・ライセンス

クリエイティブ・コモンズ・ライセンスは、法律の専門家でなくともライセンスの意図を正しく理解できるよう、リーガルコード、コモンズ証、デジタルコードの3層からなるレイヤで構成されています[17]。

リーガルコード(図 5)は、従来からある法律用語で記述された法的に有効な条文から構成される表記であり、これを法律の専門家でなくとも理解できるよう表記にしたものをコモンズ証(図 6)と呼びます。コモンズ証にはライセンスごとにアイコン(図 7)が定められており、一目でどのライセンスを採用しているのかがわかるようになっています。加えて、検索やツールなどのソフトウェアがライセンスを識別できるよう、RDF 等のファイルフォーマットで表記したものをデジタルコードと呼びます。

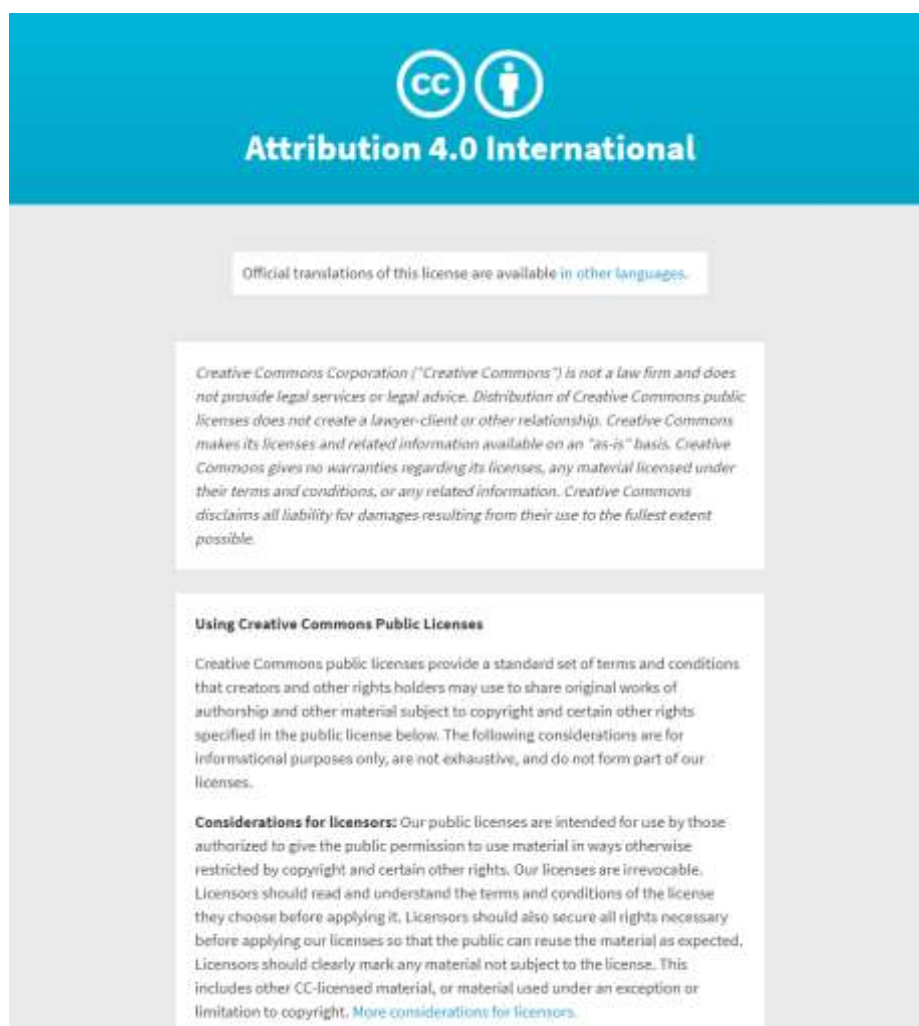


図 5 クリエイティブ・コモンズのリーガルコードの例
(Creative Commons Web ページより)



図 6 クリエイティブ・コモンズのコモンズ証の例
(Creative Commons Web ページより)



図 7 コモンズ証アイコンの例

クリエイティブ・コモンズ・ライセンスには、著作物の改変の可否（許可する、許可するがライセンス条件は継承、許可しない）、および商用利用の可否（許可する、許可しない）の組み合わせにより 6 種類のライセンスが定められています（表 3）。

表 3 6 種類のクリエイティブ・コモンズ・ライセンス (引用: wikipedia「クリエイティブ・コモンズ・ライセンス」)

		作品の商用利用を許可するか	
		許可する	許可しない (NC)
	許可する	 CC BY (表示)	 CC BY-NC (表示-非営利)
	許可するが ライセンス条件は継承 (SA)	 CC BY-SA (表示-継承)	 CC BY-NC-SA (表示-非営利-継承)
	許可しない (ND)	 CC BY-ND (表示-改変禁止)	 CC BY-NC-ND (表示-非営利-改変禁止)

ライセンス条項はそれぞれ独立した 4 つの条項: BY (著作権者の表示を要求)、NC (Noncommercial、非営利目的での利用に限定)、ND (No Derivative Works、改変の禁止)、SA (ShareAlike、二次著作物に対しても同じライセンスの継承を要求)、があります。クリエイティブ・コモンズ・ライセンスの著作物を利用する際には、これらの条項に違反しないよう取り扱う必要があります。また、作成した著作物を公開する際には、著作物の特性や著作者の配布意図に応じてこれらの中から適切な条項を選ぶ必要があります。

3.1.2. GFDL (GNU Free Documentation License)

なお、文書などの著作物の自由な利用を意図したライセンスには、上記のクリエイティブ・コモンズ・ライセンス以外にも、GPL 等のオープンソースライセンスを定めた GNU プロジェクトが定める、GNU Free Documentation License (GFDL) [18][19] もあります。GFDL では、クリエイティブ・コモ

ンズ・ライセンスのように、営利利用禁止や改変禁止等の条項はなく、GPL 同様、無断複製・改変・頒布および販売を許諾し、二次配布以降も同一条件を課すものとなっています。

3.2.形状データに関する著作権とライセンス

ロボットの開発や利用をするうえで、ロボット(ハードウェア)とそれを動作させるソフトウェアのみならず、シミュレーションなどで3次元形状データなどが利用することがあります。シミュレーションなどで利用されるCADモデルデータの知財の取り扱いはどのように考えればよいのでしょうか？

3Dプリンタを使えば、だれでも手軽にCADデータから実際の物体を作れるようになり、AIのような新しい技術の登場で、AIが作り出す文章や画像などのデータ、また様々なデータを与えることにより学習される学習済みデータなど、これまでの法律や考え方で整理がつかないデータや情報が増えています。そこで政府では、平成28年に「次世代知財システム検討委員会」を立ち上げ、こうした新しいデータなどに対する知財の在り方を検討し報告書としてまとめています。

CADデータの著作権

著作権法第10条1項六では著作物の例示として、「地図又は学術的な性質を有する図面、図表、模型その他の図形の著作物」という項目があり、ロボットなどの機械の設計図の一種であるCADによる3次元モデルデータなどは、一見著作物として保護されるように思われます。

一方で、これまでの判例から、日本の司法では、機械などの技術思想の保護は、特許などの工業所有権の制度にゆだねられるべきであり、設計図に具現された技術的思想がいかに優れたものであっても、その思想自体が著作権法の保護を享受しうるものではない[22]、という考え方が一般的なようです。つまり、設計図や図面は、作成者の意図を製作者に過不足なく伝えるため画一的に決められた設計図の表記法に従って表現されたもので、その表現そのものには作成者の独創性が入り込む余地はない、という考え方が根底にあるようです。

一方で、例えばすでに著作物である漫画やアニメのキャラクター(二次元)を3次元化しCADモデルとして作成した場合、原著作物の派生物とみなされ、それ(CADモデル)にも著作権が認められる可能性があります。また、特許・意匠登録されているロボット等のCADモデルの場合、モデルデ

一タ自体が特許法上、意匠法上の「物(プログラムを含む)」であるかどうかにより判断が分かります。つまり、特許権、意匠権で保護されている元の物を 3D プリンタなどで簡単に再生産することができるような G コードのようなプログラムに類するものや、3D プリンタの入力によく利用される STL (Stereolithography) データであれば、「物(プログラムを含む)」に該当し、その複製や譲渡などは特許権・意匠権の侵害になる可能性があります。

他方、特段の知的財産保護がなされていないものを計測して作成した 3D データについては、一般にはデータ自体が知的財産保護の対象とならないと考えられますが、3D データ作成に当たって工夫を加えた場合は、データ作成の過程で付加価値が生じていると考えられ、著作物として保護される可能性もあります。ただし、現状の法律の枠組みで明確に保護されるとは限らず、解釈の問題となり、法律の専門家でも意見が定まらない可能性があります[20]。

【POINT】

3D データ・CAD データの権利保護は法的に未整備で様々な解釈がある。

いずれにしろ、ロボット等第三者が作成した CAD データ等は、著作権やその他の工業所有権を有するものとして、その権利を侵害しないよう注意して取り扱うことが得策でしょう。CAD データなどがオープンにされている場合は、もし利用許諾が明確に定められている場合はそれに従い、そうでない場合でも、CAD データ作成者等に権利があるとし、改変・二次配布等は避けて取り扱う必要があるでしょう。

逆に、自社のロボットモデルデータをシミュレーション等で利用してもらいたい場合や、より積極的に改変・二次配布をしてもらいたい場合には、利用者に対して明確な利用条件を示したうえで利用許諾を与えるべきでしょう。

【POINT】

**CAD データの利用は利用許諾に従い、それが明確でなければ注意が必要。
公開時には、利用許諾を明確にすることがユーザへの利用を促す。**

ミスミの例

ミスミ(株式会社ミスミグループ本社)は機械加工製品の販売を行う企業であるが、オンライン販売では多くの機械部品についてその CAD データを配布しています。利用者は設計時にこれら CAD データを使用し機械設計を行い、最終的にミスミに部品発注を行うことが想定されています。ミスミでは CAD データ利用規定以下のように明示しています[24]。

1. データの利用目的

株式会社ミスミ(以下「当社」といいます)が本サイトに掲載する CAD データ(3D CAD データ、3D CAD 中間データおよび 2D CAD データをいいます)は、CAD を用いたレイアウト設計において、当社または当社取り扱いメーカーの製品(以下、製品といいます)の干渉・形状等を簡便に確認のためにご利用いただくことを目的として提供しております。

(中略)

4. 著作権

CAD データに含まれる一切の情報の著作権は、当社または当社取り扱いメーカーに帰属します。当該著作権は、著作権法および国際条約により保護されています。当社の事前の許可を得ることなく、上述の利用目的以外に CAD データを利用(複製、改変、アップロード、掲示、送信、頒布、ライセンス、販売、出版等を含む)することはできません。

トヨタ HSR の例

トヨタ自動車株式会社が開発する HSR(Human Support Robot)は、ROS インターフェースが提供されるとともに、rviz で利用可能な URDF やメッシュデータについても github 上で公開しています。URDF については、BSD ライセンス[25]で、メッシュデータについてはクリエイティブ・コモンズ・ライセンスの BY-NC-ND(表示-非営利-改変禁止)[26]で、それぞれ公開されています。

3.3.AI にかかわる知的財産権

上記 3D データ以外にも、著作権およびその他知的財産権が追いついていないロボットに関する分野としては AI にかかわる様々なデータや出力結果の取り扱いがあります。AI についても上述の「次世代知財システム検討委員会」[20]において議論されており、こうした AI にかかわるデータ・出力データの取り扱いを考えるうえでの一定の視点を与えています。

AIについては、まず、学習時にかかわる諸データについての著作権や知的財産権について考慮する必要があります(図 8)。学習のために利用される入力としての生データ、これらデータの集合体としてのデータベース、さらにデータベース中のデータを前処理して学習器に入力するためのデータセット、および学習により得られた学習済みモデルなど、それぞれについて著作権などについて考慮する必要があります。

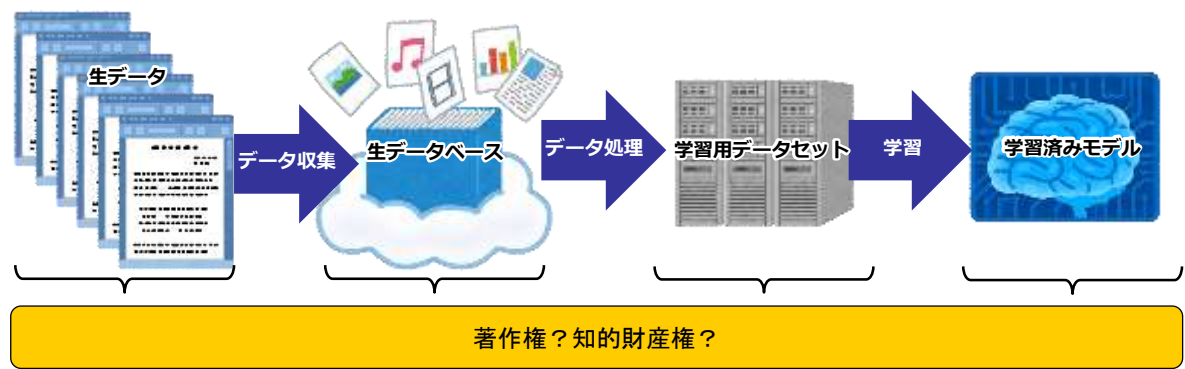


図 8 AI 学習フェーズにかかわる著作権等

加えて、学習済みモデルを利用して、AI に仕事をさせるフェーズ(実行フェーズ)で AI が生成するコンテンツについての著作権等についても考慮する必要がありますし、また AI を利用して入力から出力に至るシステム全体についての知的財産権(主に特許)についても考慮する必要があります(図 9)。

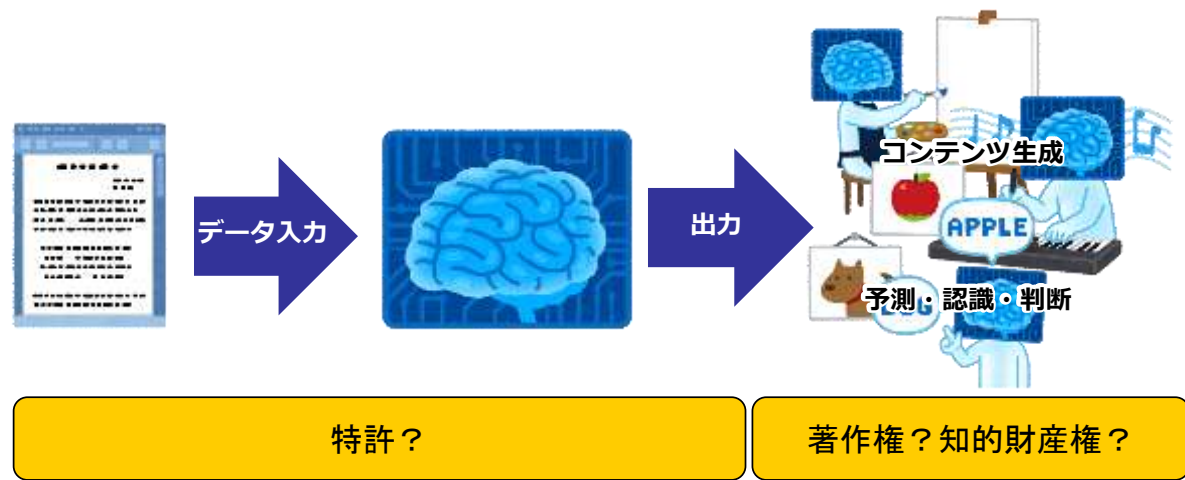


図 9 AI 実行フェーズにかかわる著作権等

3.3.1. AI 学習に用いるデータの著作権

AI 学習に既存の文書や写真、動画など生データを用いる場合、通常の著作権の考え方からすると、それらの著作物を何らかの形で複製しコンピュータ上に取り込む必要があるため、著作者の許諾を得る必要があると考えられます。、しかしながら、著作権法の 47 条の七では、以下のように、コンピュータによるデータ解析に用いる場合について、著作物の複製・翻案を許可しています[27]。

コンピュータ等を用いて情報解析(※)を行うことを目的とする場合には、必要と認められる限度において記録媒体に著作物を複製・翻案することができる。ただし、情報解析用に広く提供されているデータベースの著作物については、この制限規定は適用されない。

※情報解析とは、大量の情報から言語、音、映像等を抽出し、比較、分類等の統計的な解析を行うことをいう。

従って、図 8 の生データベースと学習用データセットはそれぞれ、データ解析に用いるために著作物を複製・翻案したものとして、著作権者の了解を得なくとも作成・利用することができます。続いて、学習用データセットを持ち出で、ディープラーニング等の学習を行わせる場合、学習済みモデルには通常は元の生データの痕跡はほぼ残らない状態でモデルデータが構成されます。したがって、この場合も著作権法で制限される複製・翻案、その他のいずれの操作にも当たらないと考えられ、著作権者の同意なく学習モデルを生成することができると考えられます。ただし、学習モデル内に、元のデータがそれと分かる形で残る場合は、著作物の改変とみなされる可能性もあるため注意が必要です。

また、上記著作権法第 47 条の七は、平成 30 年 5 月に改正、平成 31 年 1 月 1 日から以下の新たな条文(第 30 条の四)が施行され、AI などに用いるデータの利用に関する制限がより緩和されました[28]。

(著作物に表現された思想又は感情の享受を目的としない利用)

第三十条の四 著作物は、次に掲げる場合その他の当該著作物に表現された思想又は感情を自ら享受し又は他人に享受させることを目的としない場合には、その必要と認められる限度において、いずれの方法によるかを問わず、利用することができる。ただ

し、当該著作物の種類及び用途並びに当該利用の態様に照らし著作権者の利益を不当に害することとなる場合は、この限りでない。

中略

二 情報解析(多数の著作物その他の大量の情報から、当該情報を構成する言語、音、映像その他の要素に係る情報を抽出し、比較、分類その他の解析を行うことをいう。第四十七条の五第一項第二号において同じ。)の用に供する場合

三 前二号に掲げる場合のほか、著作物の表現についての人の知覚による認識を伴うことなく当該著作物を電子計算機による情報処理の過程における利用その他の利用(プログラムの著作物にあつては、当該著作物の電子計算機における実行を除く。)に供する場合

要約すると、上記 47 条の七の但し書きの、“ただし、情報解析用に広く提供されているデータベースの著作物については、この制限規定は適用されない。”の制限がなくなり、生データから作成した学習用データセットを販売・譲渡などを行うことができるようになっていきます。ただし、学習用データセット内の原著作物はあくまで学習用データとして利用することが前提で、利用者が無制限に利用できるわけではありません。

3.3.2. 学習済みモデルが生成するコンテンツの知的財産権

AI が生成したコンテンツの知的財産権については、まだ明確な決まりがない状態であり、上述の「次世代知財システム検討委員会」[20]においても議論されています。特に、著作権については無方式主義により生成と同時に著作権が発生し、かつ著作権は保護期間が長いため、保護が過剰となることが懸念されています。人間が AI を道具としてコンテンツを生成することで、新たなイノベーションや文化を生み出す可能性もある一方で、人間の関与の度合いを創作されたコンテンツから、どこまでの人による創作で、どこからが AI による創作であるかを区別することは難しく、それに伴いどこまでの権利を誰に与えるかを判断するのは困難です。

ロボットは AI の出力装置として利用されるケースが多々ありますが、AI の指令によりロボットが何らかのコンテンツを生成した場合にも、そのコンテンツにどの程度の知的財産権が認められるかは議論が待たれるところです。

3.3.3.ソフトウェア・AI を用いたシステムに関する知的財産権

AI を用いたシステム全体に関する知的財産としては、特許権について考慮する必要があります。特許庁は「IoT 関連技術の審査基準等について」[29]として、近年発展が目覚ましい AI 技術や IoT 技術に関連する特許や実用新案の考え方について整理しています。

特許法において「発明」とは「**自然法則を利用した技術的思想の創作**」であるとされており、システム全体として自然法則を利用していないものは「発明」に該当せず特許保護の対象外としています。AI やロボットも含め、コンピュータソフトウェアを利用するものについては、この「**自然法則を利用した技術的思想の創作**」に該当するか否かを慎重に検討する必要があるとしています。加えて、発明に該当しないものとして、「情報の単なる提示」（情報の提示を主たる目的とするもの）、すなわち単なるデータそのものは発明には該当しません。ただし、構造を有するデータやデータ構造はプログラムに準ずるものとして発明に該当する可能性があります。つまり、AI における学習モデルなども、単なるデータではなくプログラムとして扱われ、特許権保護の対象となる可能性があります。

このほか、特許として認められるためには、新規性・進歩性が必要になります。特に、ロボットシステム、AI システムは複数のサブシステムからシステムが構成され、個々のサブシステム（サブコンポーネーション）に対しても審査検討の対象となり、そこに新規性が認められれば特許が成立する可能性があります。つまり、A,B から構成されるシステムにおいて、A の存在が B の構造・機能を特定する場合、A に新規性が認められる場合には、A によって構造・機能が規定される B についても新規性が認められると解釈します。また、進歩性とは、例えば、特定の学習済みモデルから得られる特有の情報処理や出力情報による有利な効果が認められる場合、進歩性があると判断されます。

図 10 にリンゴの糖度データをもとに、糖度データを予測するシステムに関する事例を示します。このシステムでは、携帯型糖度センサにより取得した収穫前のリンゴの糖度データと、過去のリンゴの糖度データや気象条件から将来のリンゴの糖度データを予測して出力します。このうち、請求項 1 と 2 の収穫前のリンゴの糖度データとサーバに記録されたリンゴの糖度データ自体に対しては発明性は認められませんが、これらのデータとその他の条件から糖度条件を予測する部分については発明性が認められるとしています。請求項 3 については、ソフトウェアや AI 学習モデルに基づ

き、糖度予測データをリングにかかわる技術的性質に基づく情報処理から導き出すため、全体として自然法則を利用した技術的創作として発明と認められている。

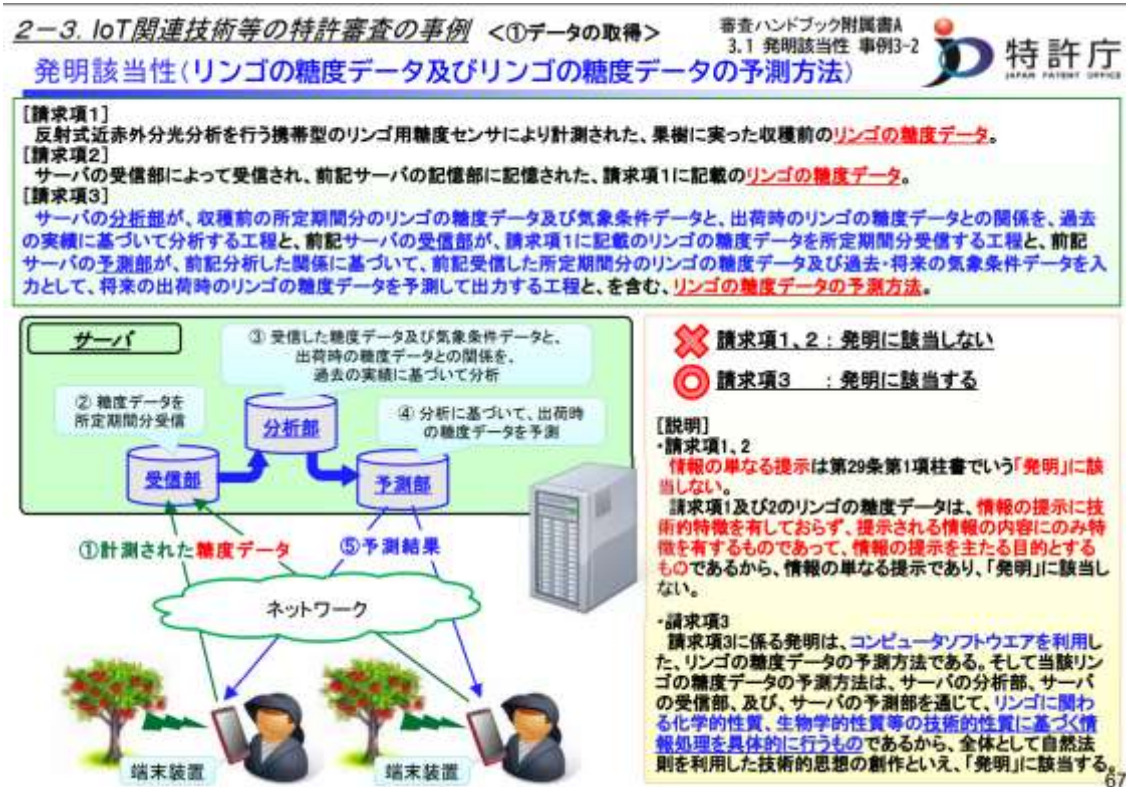


図 10 発明該当性の例 ([29]より引用)

以上のように、ソフトウェアや、AI 学習済みモデルについてもソフトウェア同様の働きをするものについては、自然法則を利用し、新規性・進歩性が認められるものについては特許が成立します。

4. ロボットのソフトウェアと特許

特許法第二条で示されている通り、特許法で保護される「発明」とは、「自然法則を利用した技術的思想の創作のうち高度なもの」と定義されています。

これまで述べてきたように、著作権は「表現」を保護するのに対して、特許は「アイデア」を保護します。したがって、「ある機能」を実現するプログラム A があるとき、そのプログラムを利用するには著作者の許諾が必要ですが、「ある機能」を実現する別のプログラム B をプログラム A とは全く別の書き方で作成すれば、プログラム A の著作権者の権利は及びませんのでプログラム B の作者は自由に使うことができます。

一方、特許はアイデアを保護するものであり、プログラム A による「ある機能」に特許が認められれば、特許権を有する人や企業はその権利を独占的に使う（実施する）ことができます。「ある機能」を実現する別のプログラム B を作ったとしても、元の特許権に基づく保護対象となるため、他者が利用しようとするれば、通常は有償で特許権者にライセンス（許諾）してもらう必要があります。このように、特許権は著作権よりも独占的かつ強力な権利です。

【POINT】

著作権は(プログラムの)表現を、特許権は(プログラムの)アイデアを保護する。
特許権の保護範囲は著作権に比べ広範かつ強力です。

OSS を使用してロボット製品を開発するうえで、特許は避けて通れない問題です。特許については、主に

- 開発したロボット製品が他社の特許を侵害していないか？
- 開発したロボット製品に（出願可能な）特許性のある技術が含まれていないか？

の二つの視点で見する必要があります。ただし、いずれの場合も自社のロボット製品で使われている技術と類似した技術を調べ（出願前調査や侵害予防調査と呼ばれる）、自社の技術がすでに出願さ

れている場合は特許侵害に対する何らかの対策をとる必要があり、一方自社の技術に特許性があり、かつ誰も出願していなければ特許出願することができることになります。

こうした調査は、知財の専門家が中心となつて行う必要があることと、対象とする技術によっては膨大な調査を行う必要があるため、時間に余裕をもって計画的に調査および出願を行う必要があるということです。もし、開発しようとするロボットシステムに特許性があるものが含まれている可能性がある場合には、早めに知財担当者や特許事務所等に相談することが肝要です。

技術者、開発者レベルでも、完全には特許調査を行うことはできませんが、関連する分野の特許について事前に全体像を把握しておくことは有用です。本ガイドラインでは、知財担当者や弁理士などが行う特許調査についての詳細には触れませんが、技術者にとって有用と考えられる特許に関するいくつかの情報を提示します。

4.1. 特許庁による特許動向調査

特許庁では毎年いくつかの分野に対して、世界における特許出願動向を調査した報告書を取りまとめています。ロボット技術は直近では平成 25 年度(図 11)に調査が行われており、過去には平成 18 年度、平成 13 年度にも行われています。ロボットおよび関連する分野に関して近年の特許動向調査対象を表 4 特許庁・特許技出願技術動向調査(ロボットに関連する技術)表 4 に示します。ロボット開発技術者がロボットを開発する過程で、このような調査報告書に目を通しておき、事前に技術動向や特許出願動向を知ることは重要です。なお、特許庁の Web ページには概要のみが掲載されていますが、報告書本体は特許庁図書館や国立国会図書館で閲覧できるとともに、発明推進協会から CD-ROM を購入することができます。

平成 21 年度	加速度センサ
平成 18 年度	□ロボット
平成 13 年度	□ロボット

利用しようとする OSS が特定の特許に抵触しているかどうかは、そのソフトウェアだけで判断することは困難です。というのも、次に述べるようにある特許とソフトウェアの関係はほとんどの場合一対一ではなく、そのソフトウェアやアルゴリズムを用いて実現されるシステムが、技術的に有用でかつ新規性・進歩性があるか否かにより特許性が判断されるためです。

4.2. ソフトウェア特許

「3.3 AI に関わる知的財産権」でも述べたように、システム全体として「自然法則を利用した技術的思想の創作」であり、かつ進歩性・新規性を有するものについては特許が認められることとなります。ただし、ソフトウェアで実現されたものについては、日本のみならず欧米においても、従来の特許制度で想定している特許の成立要件では取り扱うことが困難であることから、ソフトウェアそのものが保護対象になるかどうかについて、これまで様々な議論がなされてきたということを理解しておく必要があります。

1980 年以前は、ソフトウェアは機械語と密接に関係しており、アルゴリズムや数式などの抽象概念は一般的には特許と認められないという原則が存在する一方、コンピュータや周辺装置からなる機械を電氣的に駆動して処理を行うための機械語命令群であるプログラムは、それら機械を構成する一部としては特許として認められることがありました。

一例として、UNIX を開発した Dennis Ritchie は、setuid(特定のコマンドを実行者ではなく所有者(管理者等)の権限で実行するための仕組み)に関する特許(US patent #4135240、1972 年出願、1979 年成立)を取得していますが、当時はソフトウェアに対して特許が認められるかどうかがあいまいであったため、その仕組みをハードウェアで表現し出願しています。また、日本での例としては、暗号の発明である「欧文字単一電報隠語作成方法」は、装置を用いずかつ自然力を使用していないため工業的発明と認められなかった判例(昭和 28 年、最高裁)があります。

その後、1990 年以降は、ソフトウェア工学の進歩により、プログラミング言語で書かれたソフトウェアは、機械語的具体性のあるものから徐々に抽象的概念を記述するための言語へと変化してゆき、必ずしも装置・機械としてのコンピュータの存在を前提としなくなってきました。また、GUI などの発達により、ソフトウェアは単にデータ処理の道具としてだけでなく、ヒューマンインターフェースとしてコンピュータの主要な入出力手段となり、その表示手法自体が重要な意味を持つようになってきました。一例としては、後述する「一太郎・花子特許侵害事件」の背景となったアイコンに関する特許などがあります。

また、1990 年代後半以降、経済のグローバル化と国際競争力確保の観点から推進された米国を中心とするプロパテント政策下では、インターネットおよび電子商取引の拡大にともない、いわゆる「ビジネスモデル特許」の出願が増大し、それまで特許にならないと考えられてきた対象の多くに特許が認められました。代表的なものとしては、Amazon.com による「ワンクリック特許」(1クリックで商品のオンライン購入が完結する手法に関する特許)があり、これは米国、日本では特許が認められたものの、カナダ、欧州などでは拒絶されています。なお、ワンクリック特許自体は米国では 2017 年、日本では 2018 年に特許自体は失効しています。

一方で、ビジネスモデル特許やソフトウェア特許における法廷紛争の増加により、こうした特許を安易に認めることに対して、産業界や世論の批判が強まり、近年ではこれらの特許審査が厳しくなる傾向にあります。また、オープンソースの世界では、GPL ver3 や Apache License、Mozilla Public License のように、特許報復条項(ソフトウェアのライセンシーが著作者や上流の頒布者、下流の受領者を特許権侵害で提訴した場合、そのような訴訟を提起した時点でライセンスを停止するという罰則)を盛り込むライセンスも出現してきましたが、その効果は限定的であり特許訴訟リスクを完全になくすものではありません。

【POINT】

**特許報復条項をもつ OSS ライセンスにより特許訴訟リスクは軽減される。
ただし、油断は禁物。**

原則として、コンピュータや機械的装置等ハードウェアとソフトウェアが協働し、産業応用上有用かつ新規性・進歩性がある技術に対しては特許が認められる、ということに関しては、日本のみならず欧州や米国等海外でも同様なようです。その意味では、ハードウェアの存在が前提となるロボット技術に関しては、自社で開発したソフトや OSS を利用する場合のいずれでも、関係する特許についてよく調査する必要があるようです。

4.3. ソフトウェア特許と係争事例

上述のように、ソフトウェア特許に関しては、現在も明確な該当要件がなく、また特許の審査基準についても、時代によって変化し、今でも議論が続いています。ここでは、特許庁による「ソフトウェア特許入門[44]」に示されているソフトウェア特許の係争事例をいくつか紹介することで、ソフトウェア特許に対する考え方と、対処方法についての理解を深めます。

4.3.1. 一太郎・花子特許侵害事件

日本におけるソフトウェアに関わる特許係争事例としては、「一太郎・花子特許侵害事件」が有名です。

この係争事例は、松下電器産業株式会社（以下、「松下」とする）が保有する GUI 上のアイコン操作に関する特許を、株式会社ジャストシステム（以下、「ジャストシステム」とする）が販売するワープロソフト「一太郎」とグラフィックスソフト「花子」が侵害しているとして、松下がジャストシステムに対して、両ソフトの製造販売の中止と製品の廃棄を求め提訴したものです。

争点となったのは、一太郎と花子において、ソフトウェアの「ヘルプモードボタン」をクリックした後、別のボタンをクリックすると、その機能の説明を表示するバルーンヘルプが表示される機能が、松下が「情報処理装置及び情報処理方法」（特許番号第 2803236 号）として 1989 年 10 月に特許出願、1998 年 7 月に登録されていた「アイコン」を用いた同様の機能の特許を侵害しているか否かでした。

一審の東京地裁では、「ヘルプモード」ボタン等は「アイコン」に該当するか、松下の特許が有効であるかどうかについて争われましたが、いずれも原告の松下の主張が認められ、ジャストシステムに対して両ソフトの製造・販売の中止と在庫の廃棄が命じられ、ソフトウェア業界に衝撃を与えました。

この判決を不服として、ジャストシステムは知財高裁に控訴した結果、松下の特許が外国の公知の技術であることを示すジャストシステム側から提供された資料を基に、松下の特許がそもそも無効であるとの主張が認められ、ジャストシステム側の逆転勝訴となりました。ただし、判決ではソフトウェアをインストールしたパソコンに、そのソフトウェアによる間接侵害が成立することも示しており、そのような間接侵害に基づいて提訴できるとした点についても注目すべきです。

4.3.2. GIF 特許事件

画像ファイルフォーマットである GIF (Graphics Interchange Format) に対して、Unisys 社が特許権を主張した事件は、世界中で大きな問題となったソフトウェアに関する代表的な特許係争事例でしょう。

GIF フォーマットは CompuServe 社が開発し 1987 年に仕様を発表した画像ファイルフォーマットの一つで、辞書式圧縮アルゴリズムである LZW 圧縮法を利用しています。この LZW 圧縮法自体は、Unisys 社により 1985 年に米国で特許が成立していました。GIF の仕様を発表した時点で CompuServe 社は、「Terry A. Welch, "A Technique for High Performance Data Compression", IEEE Computer, vol 17 no 6 (June 1984)」の論文を参照しつつもその特許が成立していること自体は把握していなかったようで、LZW 圧縮法が自由に利用できる技術と考えていました。その後、1994 年に Unisys 社と CompuServe 社間では、フリーソフト等 (non-commercial, non-profit, GIF-based applications, including those for use on the on-line services) に関しては特許料を徴収しないというという声明とともにライセンス交渉で合意しました。このため、GIF 規格に関するフリーソフトウェアの開発が多くの開発者により行われていました。

ところが、その 2 年後の 1996 年以降、Unisys 社はフリーソフトウェアに対する方針を変更し、強硬な態度を取り始めました。これにより、GIF を利用するフリーソフトウェアが急速に無くなっていく一方、特許のしがらみのない新たな画像フォーマットとして PNG (Portable Network Graphics) が誕生し、多くのフリーソフトが出現し、GIF に代わり PNG への対応が進みました。LZW に関する特許

は米国では 2003 年に、日本でも 2004 年に失効したため、現在ではフリーソフトを含む多くのソフトウェアでは GIF のサポートが復活しています。

以上、見てきたように、特許については著作権とは異なり、その機能の新規性・進歩性について特許権が認められており、またソフトウェアと機能の関係は一對一ではないため、利用しようとしている OSS が特許侵害をしているか否かを軽々に判断することは難しいでしょう。ロボットに関しては、ハードウェアを前提としてシステムが構成されることが普通であり、ソフトウェアとハードウェアの組み合わせが特許を侵害しているかどうかや、特許性があり出願可能であるかどうかについて、知財部門とよく相談し適切な対応をとることが重要になります。

5. ロボットシステム開発における OSS の利用と公開

上記で説明してきたように、OSS を適切に利用しないと、企業は訴訟や OSS コミュニティと対立してしまうなど様々なリスクを負うことになります。一方で、OSS 適切に利用することができれば、単純に開発コストを削減できるだけでなく、世界中の優秀な開発者による開発力を自社の製品に活用したり、OSS コミュニティの支援を受けたりすることも可能になります。

また、OSS の利用は公開と表裏一体です。コピーレフト型ライセンスの OSS を利用し改変したものを製品に組み込む等、二次配布する場合には、改変部分についても公開義務が生じます。あるいは、製品の性質上、自社製のある部分のソフトウェアを公開しないと意味をなさないケースも考えられます。より積極的に、ユーザや OSS コミュニティに対するアピールのため、自社製ソフトウェアを OSS 化するという戦略をとる企業もあります。

特に、ロボットの世界では、ROS (Robot Operating System) や RT ミドルウェアのようなオープンソースのソフトウェアプラットフォームがあり、世界中の研究成果がこれらのプラットフォーム上に集積されつつあります。これらのミドルウェアと連携するロボットでは、制御のためのドライバモジュールや、ロボットのシミュレーションに利用するモデルデータ等、その性質上公開せざるを得ないソフトウェアやデータ、ドキュメントも多々あり、こうした情報をオープンに提供することで、ユーザやコミュニティに対してうまくアピールしている企業も存在します。

闇雲になんでも公開する必要はありませんが、こうした世界の流れに後れを取らないようにするためにも、OSS に対する正しい知識をもち、戦略を立ててうまく活用することが重要です。

【POINT】

OSS の利用と公開は表裏一体、戦略をもって OSS に向き合うことが重要

OSS の活用のためのガイドラインなどは、IPAをはじめ様々な機関や OSS 推進機関から公表 [1][32]されるだけでなく、コンサルテーションを提供する企業も存在します。いずれのガイドラインでも、技術者・開発部門単体で取り組むだけでなく、経営・企画部門、知財・法務部門、品質管理部門等、関係する部署も巻き込んで会社全体の意思統一することが重要であることが指摘されています。また、社内だけでなく、顧客やソフトウェアの開発委託先など、社外のステークホルダとも調整が必要な場合もあります。図 12 に OSS にかかわるステークホルダの一例を示します。

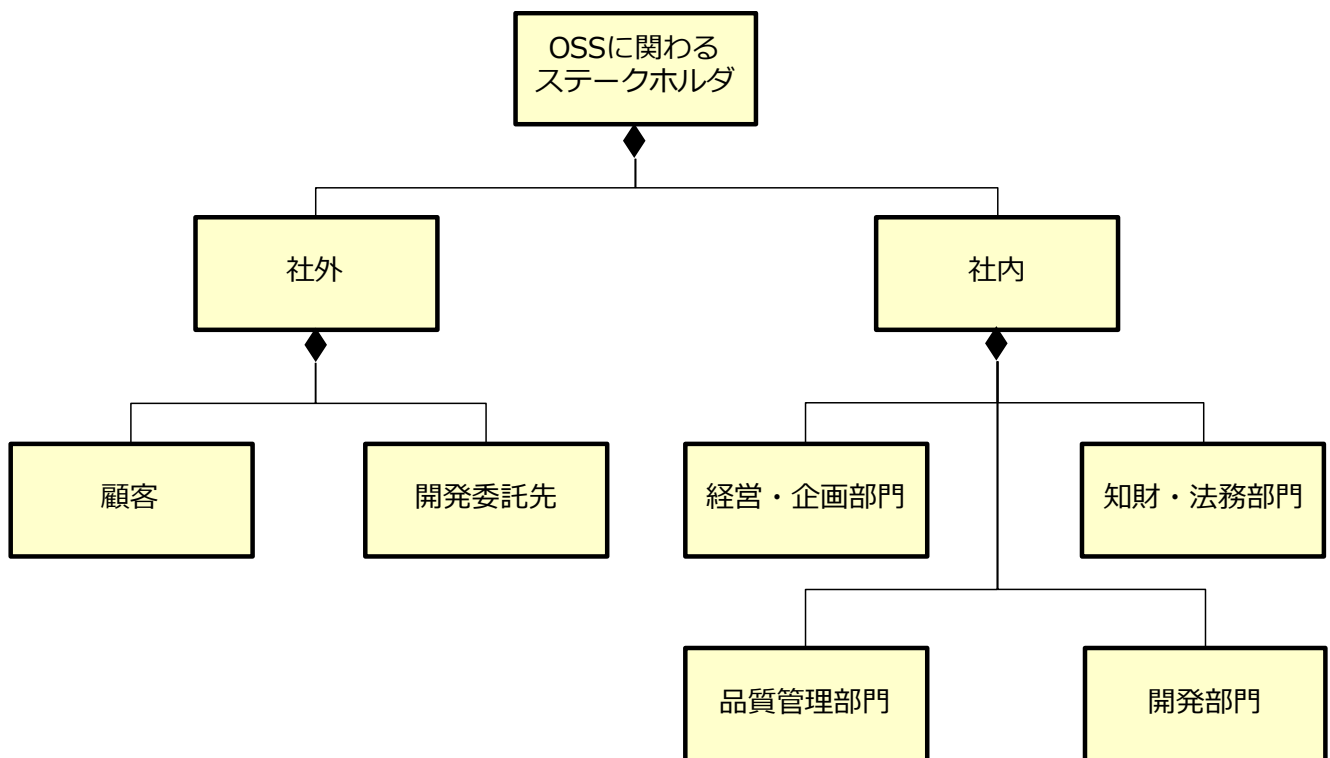


図 12 OSS 関わるステークホルダの整理(例)

本章では、主にロボット開発者が自社のロボット開発に OSS を活用したいと考えたとき、必要となる基本的な考え方を提示するとともに、社内の各部門を巻き込んで体制を構築するために必要な情報を提供します。

5.1. OSS ポリシー策定

OSS を利用したいと考えたとき、まず初めにそれが本当に自社の製品、あるいはビジネスにメリットをもたらすのかを検討する必要があります。そのためには、OSS を利用した際のメリットとリスクを冷静に分析し、OSS を会社として利用する目的・方針を明確に定める必要があります。これには、OSS を利用する技術的側面だけでなく、ビジネス戦略上の目的についても検討することが必要となります。また、公開についても、改変部分のソースコードの公開や、自社開発のソフトウェア、データなどの公開について、どのようなリスクとメリットがあるかについて、検討する必要があります。

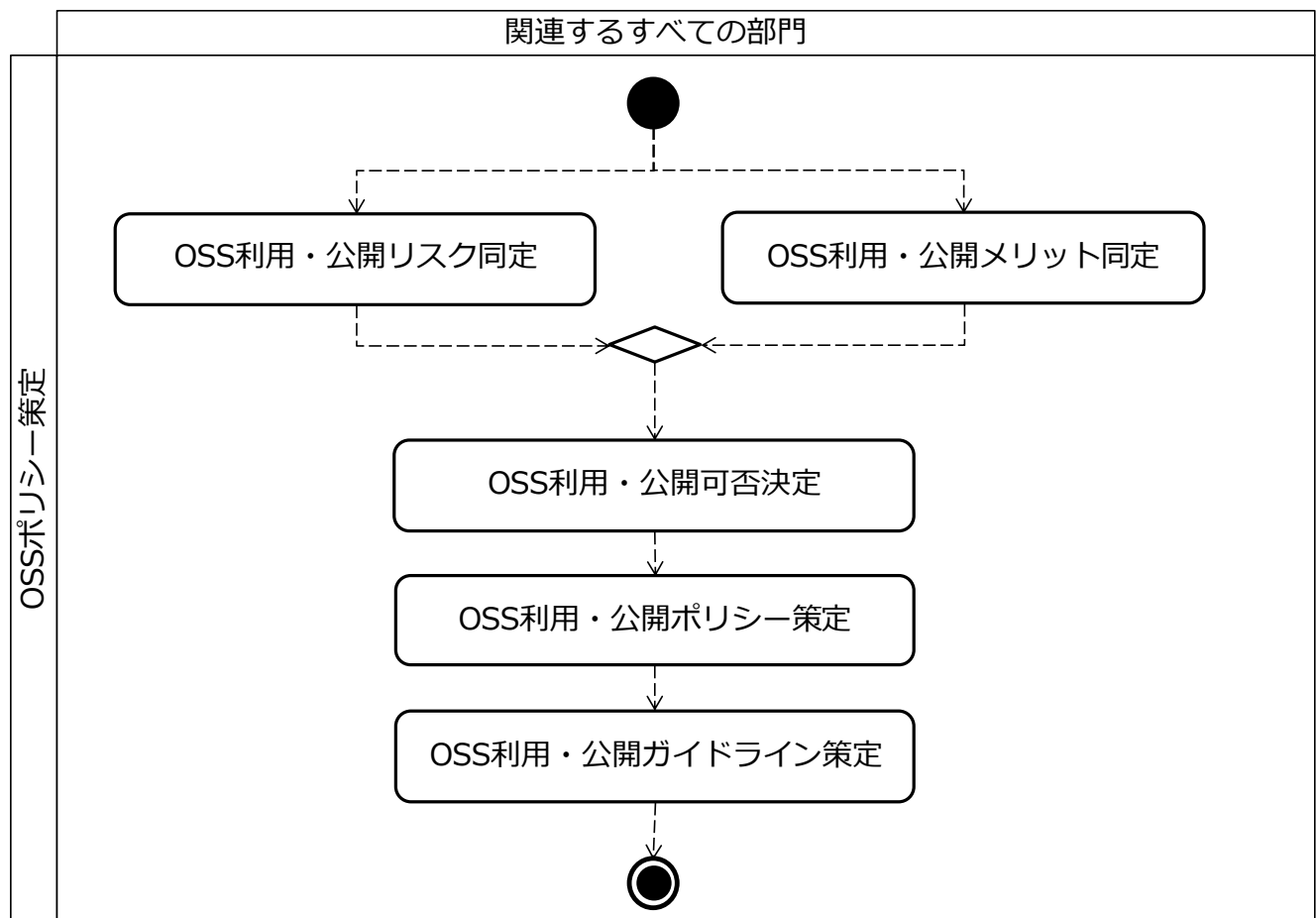


図 13 OSS 利用および公開のためのポリシー策定のためのアクティビティ

5.1.1. OSS 利用時のメリット・リスクの同定

OSS を利用するにあたり、そのメリットを明確にします。OSS を利用しようとするからには、何かしらのメリットがあるためそのような考えに至っているはずです。最初に利用しようと思った理由以

外にも、どのようなメリットがあるのかを検討し、列挙していきます。後述するリスクと比較し、メリットの方が勝るのであれば OSS を。一般に言われている OSS 利用のメリットとしては以下のようなものがあります。ただし、製品の種類や組み込み方、販売方法によってそれらがメリットとなるか否かも変わってきます。

- **コスト:** OSS は基本的には無償で利用できます。ただし、後述のリスクに対応するためにコストがかかることがあります。
- **開発速度:** すでにあるものを利用することで自社開発より素早く機能を実現できる。また、活発なオープンソースプロジェクトは、多数の開発者で日々更新され開発速度が速い。一方で、更新頻度は OSS のユーザコミュニティに依存するため、制御できない。
- **信頼性:** OSS はソースコードが公開されており、多くの人の目に触れまた利用されています。したがって、ユーザの多い OSS ではプロプライエタリなソフトウェアよりも高い信頼性を持つものもあります。ただし、これはすべての OSS に当てはまるとは限りません。
- **高機能・先端技術:** 多くのユーザを持つ OSS では、ユーザコミュニティからの多くのフィードバックにより、多くの機能が素早く実装される傾向にあります。バージョンアップも頻繁にあり、どんどん成長していく OSS もあります。また、最先端の技術がその研究者たちによって OSS 化されるケースもあり、最先端の技術をいち早く製品などに取り込むことも可能です。
- **ユーザコミュニティの存在:** 開発者やユーザから構成される大きなコミュニティが存在することが OSS の大きな特長です。そのソフトウェアの利用にかかわる多くの情報が存在する可能性があり、時にはプロプライエタリなソフトウェアのサポートよりも強力なこともあります。ただし、必ずしもサポートが保証されるものでもないというデメリットもあります。
- **ソースコード改変の自由:** OSS は基本的にはソースコードが公開されており、ライセンス条件を守れば、ソースコードを改変し、自社の製品に合わせて変更したり、拡張したりすることも可能です。また、バグがあった際にも、ソースコードを修正していち早く対応することが可能です。プロプライエタリソフトウェアではソースコードが提供されていなかったり、改変が許可されてい

かった利することが多いようです。ただし、OSS でこのようなメリットを享受するには、OSS の内部までよく理解している技術者が必要になります。

- **特定ベンダー非依存:** いわゆるベンダーロックインを回避することができる。
- **選択肢の拡大:** 同様の機能を実現する OSS が多数ある場合、その中から自社の要求やライセンス条件に合ったものを選択して利用することが可能です。
- **移植性・互換性:** 多くの利用者がいる OSS は、ユーザの様々な環境で実行されテストされるため、異なる OS や CPU アーキテクチャに対応しているケースも多いようです。また、OSS 自体がオープンであることを存在の根幹としているため、他のシステムとインターフェースしやすくなっていたり、互換性があったり等、囲い込みを主眼とするプロプライエタリソフトウェアとは、移植性・互換性の面で優位にある傾向にあります。
- **開発の継続性:** プロプライエタリなソフトウェアでは、開発企業が戦略転換などにより製品開発やサポートを取りやめたり、企業自体の倒産などにより開発が停止すれば、それを第三者が継続させるすべはありません。一方 OSS は、ソースが公開されているので、たとえ最初の開発者が開発から手を引いても、プロジェクト自体は他の開発者によって引き継ぐことができます。また、自社の製品にどうしても必要なソフトウェアであれば、自社内で保守を継続することも原理的には可能です。

以上のように、OSS 利用には単に無償で便利だから、という以外にも多くの利点があります。このような利点は、OSS を利用する周辺環境によっても変わってきます。上記に挙げた事項が必ずしもメリットにならない場合もありますし、これ以外のメリットも存在する場合がありますので、OSS の利用形態（ライブラリとして利用、OSS を修正して利用、ミドルウェアとして利用等）製品種別や OSS の利用の仕方、ビジネス環境を鑑み、自社で OSS を利用することにどのようなメリットがあるかを考えておきましょう。

【POINT】

自社の置かれた環境に鑑み、OSS を利用するメリットを列挙し冷静に判断しよう。

上では OSS 利用のメリットについて考えてみましたが、同時に OSS を利用する上でのリスクについても考える必要があります。一般的に言われている OSS 利用のリスクを以下に列挙します。

- **ライセンス違反:** OSS を利用する際に最もわかりにくい点として挙げられるのはそのライセンスに示された利用条件です。これに違反した場合には、企業は訴訟やレピュテーションリスクを負うことになります。OSS ライセンスには、大きく分けてコピーレフト型、準コピーレフト型、非コピーレフト型などのライセンスがあることはすでに述べましたが、同じ種類のライセンスでも細かな条項が微妙に異なっており、改変の有無、リンク方法、特許の有無などにより考慮する条件が変わります。ただし、非コピーレフト型ライセンスの物を選択すれば制限はかなり緩く、また（準）コピーレフト型であっても、自社開発ソフトウェアとの組み合わせ方を工夫することにより、複雑な条件を考慮しなくてもよくなるなど対応方法はあります。
- **品質・成熟度の問題:** メジャーな OSS は多くの利用者により使われることにより、いわゆる「枯れた（バグなどで尽くして安定）」状態になるため、ある程度の品質が担保されるが、自社で作成したソースコードとは異なり、テスト仕様書・テスト報告書等、一定の品質を保証するエビデンスがないことが一般的です。品質を担保するのであれば、自社で対象 OSS の成熟度に合わせてテストを実施する必要があるでしょう。
- **サポートが受けられない:** プロプライエタリなソフトウェアでは、一般的にはサポート費用を払うことによって、使用方法についての質問やトラブル発生時の対応をしてもらうことができます。OSS では、コミュニティや作者に質問をすることはできますが、返答が返ってくるかは必ずしも保証されません。OSS の利用は原則として AS IS（あるがまま）なので、使用に不安がある場合や、トラブルが発生した場合に自社で対応できない場合は使用をあきらめざるを得ないかもしれません。
- **言語の壁:** 多くの OSS プロジェクトが英語によってドキュメント化、コミュニティ内のコミュニケーションが行われています。英語のドキュメントを読みこなせない場合、英語でのコミュニケーションに不安がある場合は OSS の利用をためらうかもしれません。ただし、現在はオンラインの翻訳サービスでかなり高精度な翻訳が可能ですので、言語の壁は問題にならないかもしれません。

- **情報不足資料の少なさ:** OSS のメリットとして情報の多さを挙げましたが、あまりアクティブでない OSS プロジェクトでは、ドキュメントが貧弱、コミュニティが活発でない、など入手できる情報が少ない場合も考えられます。使用しようとしている OSS のドキュメントの多寡やコミュニティの活性度などを使用前に注意深くウォッチしておく必要があるかもしれません。
- **人的リソースの不足:** OSS は AS IS で提供されるのが原則であり、利用にあたっての諸問題は利用者側で解決することになります。このため時には、対象 OSS の内部の詳細まで理解する必要があります。また、バグなどが見つかった際に、ソースコードを自分で変更できるのが OSS の大きなメリットですので、そのためには利用する OSS の内部をよく知っているプログラマを確保する必要があるかもしれません。そうした人的リソースを確保できない場合は、何かトラブルがあった際に事態の解決に行き詰まることもあります。
- **コミュニティとの関係:** OSS の背後には、開発者とユーザから構成されるコミュニティが存在します。コミュニティにはその OSS に興味を持ち、利用し、またそのソフトウェアをより良くしたいと考えている人々が集まっています。わからないことがあって質問した時に、答えてくれるのはコミュニティの開発者やユーザなどのボランティアであり、サポートセンターではないことをよく理解する必要があります。メーリングリストやフォーラムなどではマナーを守り事前によく調べたうえで質問したり、他の質問者の質問に対して答えられる場合には自ら回答してあげるなど give and take の関係を保つことが必要です。コミュニティと良好な関係を築くよう注意する必要があります。
- **ソフトウェアの公開義務:** OSS のライセンスによっては、二次配布（製品へ組み込み、顧客の手に渡る場合も二次配布とみなされます）の際に、改変部分があればその改変部分の公開が求められることがあります。また、GPL のようなコピーレフト型ライセンスでは、GPL ライセンスのコードと自社開発のコードと一緒にリンクしただけで、自社コードも GPL（または GPL と互換性のあるライセンス）にし公開する必要がある、といった強い伝搬性を持つものもあります。公開する自社コードに自社製品の中核的な知財が含まれる場合は、公開したくない場合もあるでしょう。OSS を利用するにあたっては、ライセンスとその利用許諾条件をよく理解し、場合によっては利用をあきらめざるを得ない場合もあります。一方で、ロボット分野では ROS をはじめとし

て BSD ライセンスなど非コピーレフトライセンスで提供されるソフトウェアも増えており、利用制限がほとんどない OSS も多数あります。

- **選択肢が多すぎる:** OSS プロジェクトはだれでも始めることができ、継続するものやめるのも開発者の自由です。これにより、同じような機能を持つライブラリが多数存在し、利用する側から見ると、どれを利用したらよいか選択に迷う場合も往々にしてあります。また、複数存在する似た OSS のプロジェクトの中には、開発者が開発を途中でやめてしまったものもあり、そうした OSS をよく調査しないで利用すると、開発者やコミュニティの支援を受けられない場合もあります。OSS の利用を決定するにあたっては、機能面だけでなくこうした周辺環境についても事前に調査していたほうが良いでしょう。
- **顧客側の拒否感:** 受託開発でロボットシステムを開発する際、顧客側に OSS 利用に対して拒否感があり、利用を拒まれる場合もあります。また、社内の規定で OSS を利用したシステムとの接続を禁止するケースなども考えられます。こうした顧客側の要望で OSS の利用をあきらめざるを得ないケースもあります。
- **社内の他部署の意識レベルの差:** あなたが開発部門のエンジニアとして OSS を利用したいと考えたとします。ただし、製品に OSS を搭載してきちんと出荷するためには、上述のように経営層、知財・法務部門、品質管理部門等社内の様々な部署にも、OSS に対する正しい理解を持ってもらう必要があります。「どうせ誰にもわからない対応しなくたっていい」といった意識を持つものはいけませんし、逆に「OSS を使用して知財が流出したり、訴訟を起こされたりしては大変だ、OSS は利用禁止だ」というのも困ります。会社全体として OSS を正しく理解し、正しい対応をとれるよう、他の部署にも啓もうしていく必要があります。

上記は一般的な OSS にまつわるリスクですが、これ以外にも自社の製品の形態や OSS の利用の仕方、置かれているビジネス環境によっても上記にはないリスクが存在する可能性もありますし、上記のリスクの中にもほぼリスクとはならない事項もあります。まずは、自社の OSS を利用する上でのリスクをリストアップしてみましょう。

【POINT】

まずは自社の置かれた環境と OSS の利用によるリスクを見定めよう。

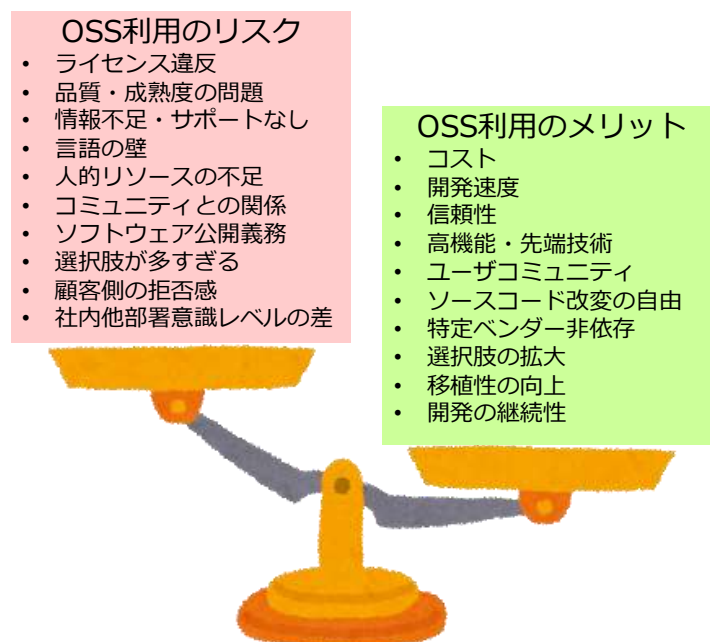


図 14 OSS 利用のメリットとリスクを同定したうえでの判断が求められる

5.1.2. OSS 公開時のメリット・リスクの同定

OSS として社内で開発されたソフトウェアを公開することは、これまで企業では行われることはまれでした。近年は、様々な戦略上の理由から、自社で開発したものを積極的にオープンソースで公開する企業が増えつつあります。自社の製品と市場における位置づけや、会社の戦略との関係から、OSS を公開することによるリスクやメリットについて検討するとよいでしょう。以下に、一般的に言われている OSS 公開のメリットを挙げます。

- **コミュニティによる開発力の利用**: OSS 化することにより、OSS コミュニティからのフィードバックを容易に得られるようになる可能性があります。ソースコードを公開していますので、場合によってはバグレポートだけでなくバグ修正パッチを送ってくれるユーザも現れるかもしれません。

あるいは、ユーザコミュニティが主導して、新しい OS や言語への対応、機能追加なども行ってくれる可能性があります。ただし、あくまでボランティアによる活動ですので、そうした開発者を引き付ける製品の魅力や、円滑なコミュニティ運営が無ければいけません。

- **OSS コミュニティへの貢献:** ソフトウェアを公開すること自体、OSS コミュニティに対して大きな貢献しているとコミュニティからはみなされます。そうした企業は OSS 活動に理解がある企業とみなされ、コミュニティからは好意的に受け取られるでしょう。OSS コミュニティの世界では、OSS に対する貢献の度合いが発言権の強さにつながることもあります。OSS コミュニティの中で振る舞うにあたって、OSS への貢献度合いは活動のやりやすさにつながることも多分にあることを理解しましょう。
- **企業・製品のアピール:** ソフトウェアを OSS として公開することで、その会社の製品が社会に対して好意的に受け取られ、製品そのものの宣伝となることが期待されます。OSS 化することは、企業の製品に対する自信の表れや、OSS への取り組みの姿勢を示すことになり、特に OSS を信奉するユーザに対しては大きなアピールポイントとなることがあります。また、既存の OSS を利用して製品を発表することだけでも、その OSS を支持するコミュニティに対して非常に強力な援護射撃となるため、OSS コミュニティに対する大きなアピールとなり得ます。
- **サポート、コンサル、SI ビジネスへの展開:** 自社製品を OSS として公開することで、まずは無償で利用してもらい、これを当該 OSS の保守・サポートや、コンサルテーションビジネス、あるいは当該 OSS を利用したインテグレーションビジネス (SI ビジネス) へつなげるという戦略があります。MySQL AB は GPL と商用ライセンスのデュアルライセンス方式でデータベースソフトウェアを無償公開するとともに、そのサポートをビジネスの主軸としていた代表的企業です。RedHat はオープンソースである Linux を含む OS 全体のオープンソースに対して大きな開発リソースを提供しつつ、これをサポートやインテグレーションビジネスとして展開しています。

同様に、OSS として自社のソースコードを公開することに関してのリスクについても検討します。知財、特許なども関係してきますので、知財・法務担当者なども交えて検討するのが良いでしょう。以下に、一般的に言われている OSS 公開のリスクの一例を示します。

- **機密の漏洩:**ソースコードをオープンにするということは、背景にある様々な知財やノウハウの公開につながるケースがあり得ます。本来表に出したくない社内の特許・その他の知財やノウハウを含む機密情報が、ソースコードと一緒に意図せずに表に出ることがないように注意する必要があります。
- **特許技術の無効化:**GPLver 3.0 や Apache License ver 2.0 など、いわゆる特許条項付きライセンスでは、これらのライセンスで利用が許諾されている人に対して、自身が持つ特許権に基づき特許侵害であると訴訟を起こした場合、自身に許諾されていたライセンスが終了するという条件が付されています。簡単に言えば、これら特許条項を持つライセンスの OSS では、コントリビュータは自身がもつ特許を OSS に含めた場合、その特許の利用を無償で許可する必要があります。誤って、自社の特許技術をそうしたライセンスの OSS と組み合わせてソースコードの改変部分をオープンにしてしまうと、自社の特許を無償で使われる(あるいは、訴訟を起こせば、その OSS を自社で利用できなくなる。)というリスクがありますので注意が必要です。
- **その他工業所有権の漏洩:** ロボットについては、ロボット自体のシミュレーションのために CAD モデルが必要とされることがあります。ROS の rviz や gazebo では、ロボットモデルを与えることで、動作状況の表示やシミュレーションなどが行えるため、様々なモデルがオープンソースで提供されています。設計上の詳細な CAD モデルをオープンにすることは、意匠権や様々な工業所有権の漏洩につながる場合も考えられます。たいていの場合、上記のロボットモデルは詳細なモデルからかなり間引きを行い、表示上、シミュレーション上差し支えのないくらい簡易なモデルとして提供されています。ロボットモデルを提供する場合も、どこまでの情報を公表するかを検討する必要があります。
- **不適切な公開:** OSS を利用・改変して自社製品を開発・販売した場合、非コピーレフト型ライセンス OSS を利用していれば、改変部分を公開しなければならないことは上で述べたとおりですが、公開の仕方を間違えると、OSS コミュニティから批判され評判を落とすこともあります。代表的な例としては、いわゆる”Tivoization (TiVo 化)” 問題といわれるもので、TiVo 社のハードディスクレコーダのソフトウェアが GPL 下で公開されていたものの、レコーダのハードウェアはデジタル署名済みのソフトウェアしか実行しないような仕掛けが組み込まれていたため、レコーダの所有者自身が自分で改変したレコーダのソフトウェアをインストールできないことを GNU のリチャ

ード・ストールマンが批判したことがありました。これにより、GPL ver3 以降では、これを防ぐための条項が含まれています。ただし、製品の品質保証上、ユーザに自由に改変されたソフトウェアをインストールされたくない場合もありますので、適切なライセンスを選択し、ユーザにも理解を求めることが肝要です。

- **レピュテーションリスク:** ソースコードを公開するということは、ある意味自社のソフトウェア開発力を公開することにもなります。あまりにバグの多いソフトウェアを安易に公開することで、OSS の貢献ととらえられるのではなく、会社の技術力に疑問を持たれるようでは本末転倒です。ソースコードを公開するということは、それを読む人がいるということです。読みやすいソースコードであるとともに、それを試してみることができるように、README やドキュメントについても、読む人のことを考えた作成と公開が必要になってきます。また、ソフトウェアに対して質問や、バグレポート、パッチの送付などがユーザからなされることも考えられますが、そうした貢献者に対して真摯に向き合うことも要求されます。そのような姿勢が、企業が OSS に対してどのように向き合っているのかという OSS コミュニティにおけるレピュテーション(評判)を形成することを理解する必要があります。

OSS により自社のソースコードを公開することのメリットは、対象のソフトウェアや自社製品の市場での位置づけによる変わってきます。一般的なリスク・メリットだけでなく、そうした個別の状況も考慮しつつ、自社が OSS を公開するリスクを冷静に分析することが肝要です。

以上のように、OSS 公開伴うリスクとメリットをよく検討したうえで、どのような方針で OSS やそのコミュニティと向き合うかを検討する必要があります。次は、その全体の方針となる OSS ポリシーの策定について説明します。

5.1.3. OSS 利用ポリシーの策定

OSS を利用する上でのメリットとリスクを同定したら、会社全体として OSS とどのように付き合うかについての基本方針「OSS 利用ポリシー」を策定します。OSS を利用する上で、関係各部署の意見を統一するためにも、基本方針としてのポリシーを定めておくことには大変重要な意味があります。ポリシーは会社全体の方針であり、様々なケースに対して柔軟に対応できなければなりません。

ので、基本方針のみ定めておきます。詳細なルールや個別の事項に対する対応については、後述するガイドラインで適宜定めるのが良いでしょう。

ポリシーでは、

- 会社として OSS に対する取り組みの基本方針
- 自社の著作物（ソースコード、データ、ドキュメント）の取り扱い方針
- 自社の著作物を OSS 化する際の基本方針や手続き
- 社外の OSS の取り扱いに対する基本方針や手続き
- OSS コミュニティに対する姿勢や、自社 OSS ユーザに対する基本方針
- OSS ライセンス違反発生時の対応方針

などを定めておくとい良いでしょう。

【POINT】

ポリシーでは基本方針を定め、詳細なルール化はガイドラインで定める。

OSS ポリシーの策定については、これを一般公開しているサイボウズ株式会社の例が参考になります[33]。サイボウズ株式会社では OSS ポリシーを定めそれを Web 上に公開するだけでなく、元となるファイル（Web 上に公開されているページは、sphinx にてテキストファイル进行处理して作成されている。）を github 上で CC0 (Creative Commons 0, パブリックドメイン≡著作権放棄) にて公開しています[34]。

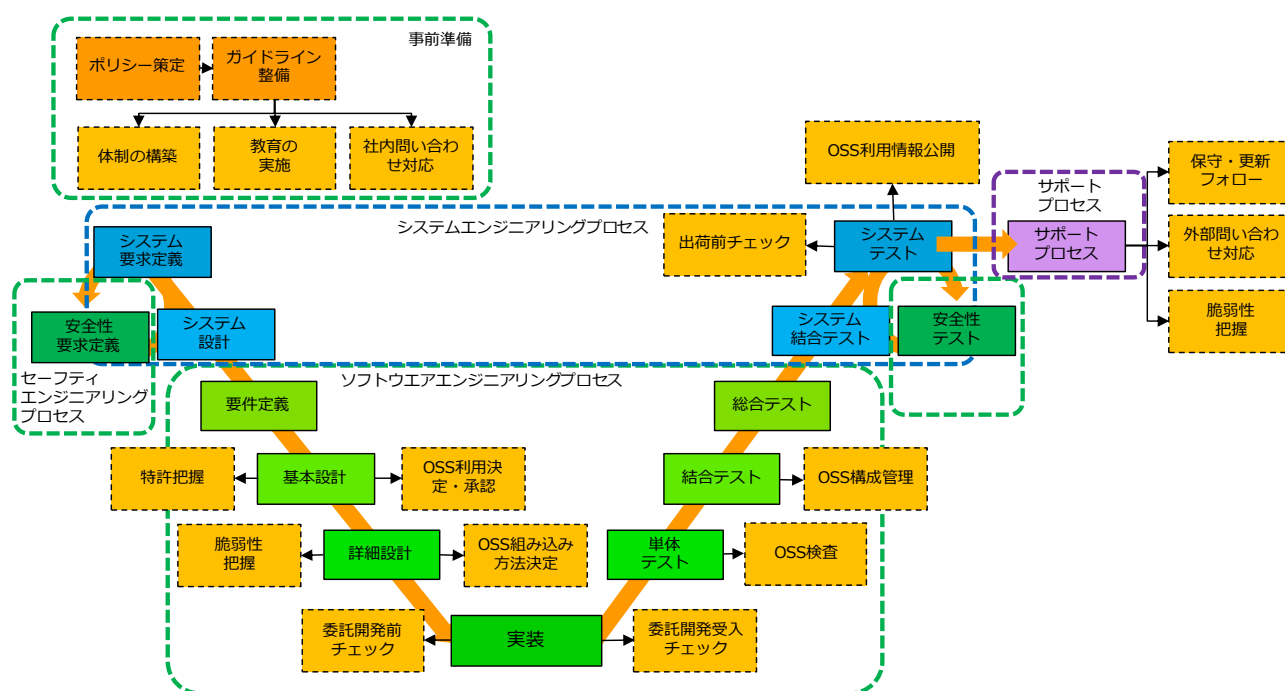
5.1.4. OSS ガイドラインの作成

OSS ポリシーで、OSS に対する基本方針を定めました。次に、ポリシーに基づき個別の事情も考慮したガイドラインを作製するのが良いでしょう。ポリシーで定めた基本方針に反しない範囲で、例えば OSS ライセンスごとに対する対応方法、詳細な手続き、開発時に考慮すべき事項やルール等、定めていきます。初めからすべてを明確にルール化するのは難しいでしょう。ガイドラインは自裁に OSS を利用する過程で、実情に合わせて改訂を繰り返していくことになるでしょう。重要なのは、ポリシーで定められた OSS に対する基本姿勢を考慮し、それに反しないように個別の事情を考慮しルール化することです。

ガイドラインの目的は、上記で挙げた様々なリスクへの対応策を用意することになります。一つのリスクに対して、複数の対応策が用意される場合もありますし、逆に一つの対応策で複数のリスクに対処できる場合もあります。いずれにしても、考えられるリスクに対して漏れの内容対応策を用意しておくことが重要です。対応策については、

- 開発のどの段階での対応が必要か？
- 誰(どの部署)がどのように対応するか？
- 対応を行ったエビデンスをどのように残すか？(申請書、データベース)

といった観点で検討するとよいでしょう。自社でソフトウェアの開発プロセスがある場合は、どの段階にどのような OSS 対応作業がマッピングできるか、といった観点で検討することもできます。これにより、開発の各段階で実施すべき対応策の関係が明確になるとともに、漏れもなくすることが可能です。図 15 に、ソフトウェア開発プロセスの一つである ESPR (Embedded System Development Process Reference) 2.0 の V 字モデルを例にとり、OSS 利用に対する対応策をマッピングした例を示します。



引用：ESPR (Embedded System development Process Reference) 2.0

図 15 開発プロセスと OSS 利用対応作業のマッピングの例

図 15 では、開発プロセスと OSS 利用対応作業のマッピングは明らかになりますが、上記の「誰が対応するのか？」や、後々トレースが可能なように「どのようなエビデンスを残すか？」についての観点からはまだ検討が不十分です。これに対応する例として、図 16 に示すような申請・承認フローについても検討するとよいでしょう。図 16 では、基本設計プロセスと詳細設計プロセスを抜き出して示していますが、基本設計プロセスで、開発部門が OSS の利用検討と特許について事前調査を行い、知財部門に特許の詳細調査と利用可否を申請しています。その結果を開発部門が受け取る際には、そのエビデンスを残しています。次の詳細設計プロセスでは、脆弱性調査について品質管理部門と同様のやり取りを行っており、ここでも申請と承認のエビデンスを残しています。こうすることで、どの段階でどの OSS の利用が決定されたかのエビデンスを残し、後々トレースできるようにしています。

これはあくまで一例で、会社の規模や組織体制によってはこのような対応が取れない場合もあるでしょう。その場合でも、製品出荷後トレース可能なエビデンスを文書で残すことは重要です。

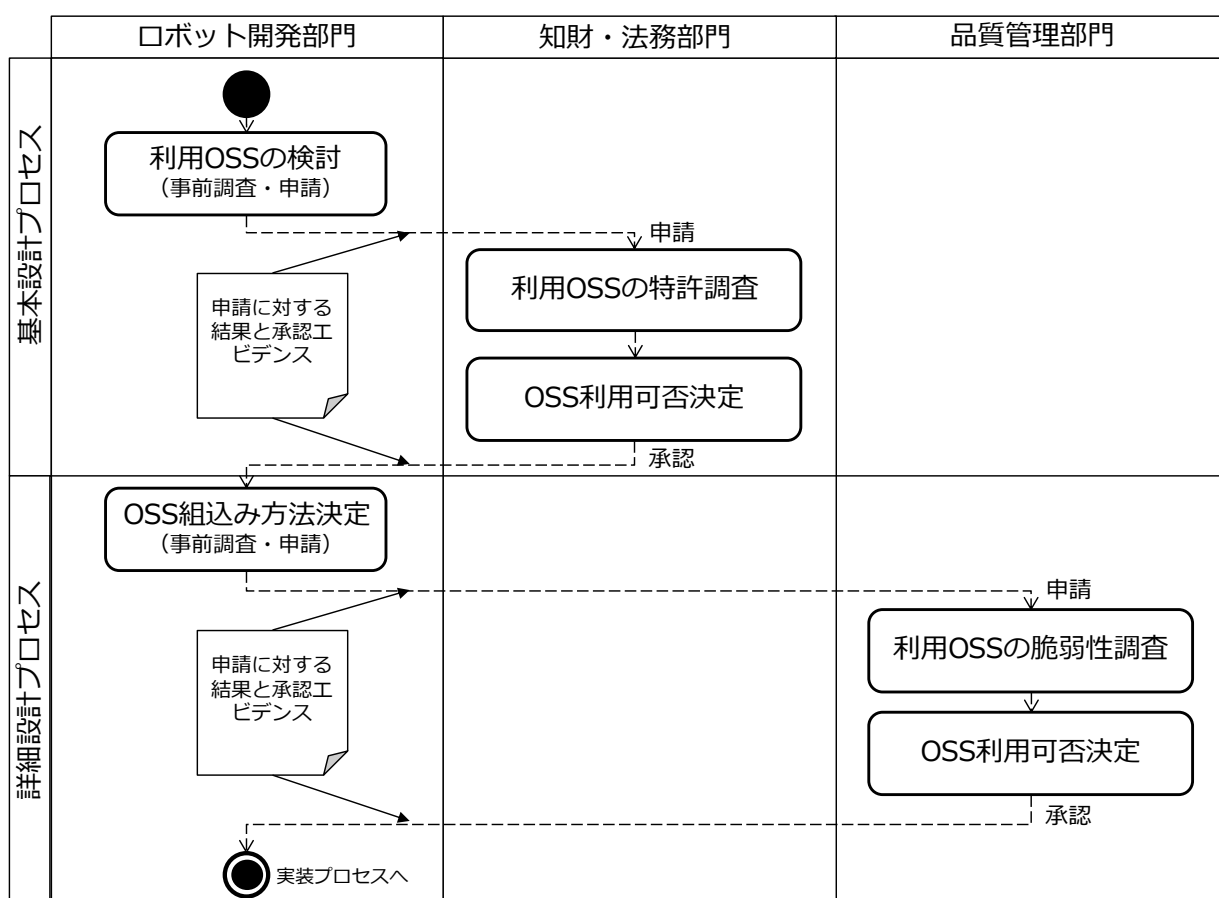


図 16 OSS 利用申請と承認プロセスにおけるデビデンスと役割分担の例

ここまででは、主に OSS の利用を例にとり、ガイドライン作成の概要について説明しましたが、OSS として派生物や自社開発のソフトウェアを公開するプロセスについても必要があれば検討します。それらをまとめて、製品開発におけるプロセスごとに、行うべき作業や手続きをガイドラインにまとめていきます。



5.2.ポリシーに基づく OSS の利用

ここでは、OSS の利用あたってのガイドライン作成や体制構築・運用を実際に行っていく上での注意点について説明します。

OSS を利用する上では、上記の訴訟等のリスクに対応するために、企業としての統一した方針（ポリシー）を定め、それに基づき関係する様々な部署が一貫した方針で行動することが肝要です。ポリ

シーを定めるにあたっては、OSS 利用にかかわる様々なリスクやそれを引き起こす原因の分析、および問題が発生した際に、企業としてどのように動くかなどを考慮する必要があります。また、ポリシーを定めても、これを社内の関係部署内や、開発委託先も含めて意識を共有し、一体として動くことができれば意味がありません。そのためには、社内・委託先を含めて連携する体制を構築する必要があります。以下では、OSS を活用するうえで必要なリスク軽減のための対策を示します。

5.2.1. 体制の構築と運用

OSS 利用ポリシーおよびガイドラインの運用を確実なものにするためには、体制と仕組みを構築し、継続して実施していくことが重要です。体制は、開発部門で OSS の利用について知識を有する人たちが始めることになるでしょうが、可能であれば専門知識を有する、品質管理部門、知財・法務部門なども巻き込んで、部門を横断し全社的な活動にしていくことが望ましいでしょう。

OSS ライセンス違反に関する訴訟は国内ではまだ見当たりませんが、うえて述べたように欧米では OSS ライセンス準拠を監視する団体などもあるため、訴訟にまで発展するケースもあり、いつ日本で起きても不思議ではありません。設計開発部門はそうした訴訟などの専門家である法務部門等と連携して、OSS ライセンス遵守のための体制を構築していく必要があるでしょう。

また、セキュリティに関しても、多くの事案が発生しており、OSS も含めてソフトウェアの脆弱性に対処する必要性が高まっています。ソフトウェアの脆弱性は開発時にすべてわかっていることはまれであり、製品出荷後も継続的に把握・対処する必要があります。そのためには品質管理部門などと連携して、体制を構築する必要があります。

5.2.2. 教育の実施

OSS 利用におけるリスクの大半は OSS ライセンス違反によるものであり、その多くはライセンスに対する理解不足に起因します。OSS ポリシーやガイドラインは定めただけでは意味がなく、従業員に対して継続的に教育を実施し OSS の知識を深め、それらポリシーやガイドラインを周知・徹底することが重要となります。

OSS には著作権があり、ライセンスで定められた利用条件があることを知ったうえで、開発の各プロセスにおいて所定の手続きを踏む必要があることを周知することで、インターネットで見つけたソースコードを安易に利用し製品などに組み込むことを減らすことができます。同時に、OSS は正しく対処すればその利用は奨励されるべきであるものでもあり、問題なのは OSS が利用されたことを誰も知らず、エビデンスとしても残っていない状態を避けるべきであることを皆が理解することが重要です。

上記のように、開発部門のみならず、知財・法務部門や品質管理部門に対してもガイドラインの周知・教育が重要です。加えて、開発の委託・受託を行うケースでは、営業部門についても OSS ライセンスの知識が欠かせません。開発を委託される場合、委託元から利用している OSS とその利用条件の提示を求められるケースが増えています。また開発を委託する場合は、委託先との契約時に OSS の利用についての自社のポリシーに合意してもらう必要があります。ガイドラインでは委託契約書にそうした条項について盛り込む旨を記載するとともに、契約書のひな型を示すのもよいでしょう。

5.2.3. 社内からの問い合わせ対応

ポリシー策定・ガイドラインの作成と教育による周知を行うだけでは、開発実施時の個別の事例に対応することは難しいでしょう。そうした、個別の事例に対して対応や利用の可否を回答・検討したり、そうした事例収集し How To 化やデータベース化したりするような窓口を設置することは、その後の OSS 利用効率を上げるためにも重要です。部署として設置するのが難しい場合は、OSS に詳しい担当者を充て、情報が集積されるようにするとよいでしょう。必要によっては、OSS 利用ポリシーやガイドラインの更新、FAQ の作成などを実施します。

5.2.4. OSS 利用の申請・審査・承認

上述したように、開発部門から利用する予定の OSS を申請し、それを他の部門や担当部署により審査・承認する仕組みを構築するのも効果的です。申請・審査・承認プロセスを設けることで、OSS の利用をエビデンスとして残すことができ、利用している OSS の把握やその後のトレーサビリティ確保に役立ちます。

設計段階から OSS ライセンスを意識することにより、開発の手戻りを最小限に抑えることができます。ソフトウェアの検査とともに、この仕組みを開発プロセスに取り込むとよいでしょう。

5.2.5. 脆弱性リスクへの対応

ソフトウェアには致命的なセキュリティ上の脆弱性が隠れている可能性があります。OSS は原則として無保証や損害賠償等についての免責をライセンスに規定しているものの、多くの利用者がいて影響力の大きい OSS については、脆弱性情報が公開されると比較的早期に修正が行われることが多いようです。ただし、これはすべての OSS に期待できるわけではなく、OSS の開発者が脆弱性を把握していても、必ずしも修正を行わない可能性もあります。そのような場合は、自力で不具合修正を行う必要があります。したがって、OSS を選定する時には、その OSS についてバージョンも含めて、事前に脆弱性情報を以下のようなサイトで確認しておくことを推奨します。

- National Vulnerability Database (NVD): <https://nvd.nist.gov/vuln/search>
- IPA 独立行政法人 情報処理推進機構: 重要なセキュリティ情報一覧:
<http://www.ipa.go.jp/security/announce/alert.html>
- JVN (Japan Vulnerability Notes) iPedia 脆弱性対策情報データベース: <http://jvndb.jvn.jp/>

また、委託開発を行際には、納品時までには発覚している脆弱性については、暗黙の了承として脆弱性の対策実施が契約に含まれることが多くあります。一方、受託案件の場合は顧客とどこまでの脆弱性に対応するか協議を行い、契約などで明確にしておくことが大切です。

5.2.6. 特許の把握

OSS の利用により、特許を含む知的財産権利侵害により訴訟される危険性についても事前に把握しておくことが重要です。OSS は第三者の権利侵害に対しても無保証です。そのため、OSS が他者の知的財産(特許や商標など)を侵害していた場合には、訴えられる可能性があります。

上述したように、GPLv ver3 や Apache License ver2 のように特許条項を含む OSS ライセンスも増えてきており、こうしたライセンスを採用している OSS では、特許による訴訟リスクは他のライセンス

の OSS に比べ幾分低いと考えられますが、特許権保有者がコントリビュータに含まれない場合にはこの特許条項もあまり意味をなさなくなります。また、依然として GPL ver2 等特許条項をもたないライセンスで公開されている OSS も数多くあり、OSS のバージョンと採用しているライセンスのバージョンも含めて、特許については知財・法務部門とも連携して調査する必要があるでしょう。

5.2.7. 開発委託時の注意

ソフトウェアの開発を外部に委託する場合は、契約時に委託先と OSS の利用について合意する必要があります。OSS ライセンスの遵守の義務は最終的には製品を配布する企業にあるため、企業はソフトウェアに含まれる OSS を把握する必要があります。それは委託先が開発したソフトウェアを受け取り委託元が製品に組み込む場合でも同様です。委託先からのソフトウェアの納品時に、ソフトウェアの中に含まれる OSS と、その OSS をどのような手段で検出したのかを確認する必要があります。

また、委託先が開発したソースコードが納品されない場合は注意が必要となります。委託先が開発したソフトウェアに関して外部から OSS の利用について問い合わせがあった場合は、ソースコードを調査することがあります。このような場合に委託先に協力を仰げるよう、契約に盛り込むことも必要になってくるでしょう。

5.2.8. 意図しない OSS 混入の検査

上述のように申請・承認による OSS 利用を徹底したとしても、ソフトウェア開発工程のすべてをチェックすることは不可能であり、開発したコードに意図しない OSS が混入する場合があります。開発を外部に委託する場合はさらにチェックが困難になります。このような意図しない OSS 混入にリスクを低減するため、膨大なソースコードの中から中に含まれる OSS を検出するツールが、いくつか販売あるいはオープンソースで開発されています。代表的なチェックツールとしては、以下のようなものがあります。

図 17 OSS 混入検出ツール例

ツール名 (開発販売企業)	商用・ OSS	概要
Black Duck79[35] (Black Duck Software 社・Synopsys 社)	商用	年間サブスクリプション形式の商用ライセンスのツール。大規模なソフトウェア情報データベース（4600 を超えるサイトからの OSS プロジェクトの情報の Knowledge Base）を提供する。ソースコード内の文字列を検索してコードスニペット単位で類似の部分を検索可能。セキュリティ脆弱性についても検知可能。
FlexNet Code Insight [36](旧 Paramida) (Flexera Software 社)	商用	対象のソースコードやバイナリ ファイルをスキャン・分析しレポートする。OSS を収集し特徴を抽出した DB とコードを比較しコードスニペット単位で OSS を検出。既知のセキュリティ脆弱性を含む OSS についても検出可能。
White Source[37] (WhiteSource 社)	商用	様々な言語に対応、様々なソースリポジトリ（Github 等）やビルドツール、CI ツール、issue トラッカーなどと連携可能なツール。基本的にはファイル単位で既存 OSS とのマッチングを行い検出する。NVD（National Vulnerability Database）や様々なバグトラッカーと連携して脆弱性検出も可能。
FOSSology[38] (Hewlett Packard 社)	OSS (GPLv2, LGPLv2.1)	HP が開発するオープンソースのライセンスチェックツール、ライセンスのスキャン、分類、著作権に重点を置いたツール。ライセンスデータベースは独自のものを構築する必要がある。ファイル単位でのマッチングおよび検出を行う。

こうしたツールを利用することで、膨大なソースコードの中から意図しない OSS の混入を検出することが可能になります。ツールはそれぞれ検出方法や適用範囲が異なります。こうしたツールの特性を理解したうえで使用する必要があります。

品質管理のために行う構造解析や静的解析などとともに、OSS 混入検査も、開発プロセスのできるだけ初期段階で一度は実施することが推奨されます。こうすることで、ポリシーに合致しない OSS の混入や意図せぬ OSS 混入が発覚した際の手戻りを最小限に抑えることができます。また、可能ならば、品質のためのコード解析とともに開発プロセス中の CI(継続的インテグレーション)に組み込み常に自動的にチェックする体制が構築できることが理想的です。

5.2.9. 利用している OSS の把握

意図しない OSS の混入だけでなく、製品で利用している OSS を把握することは重要です。例えば、そうした情報を蓄積することで、他の開発部門からの問い合わせに対して、利用実績のある OSS かどうか判断可能になります。また、外部からの問い合わせに対して迅速に対応するためにも、使用している OSS を把握しておくことは重要です。

使用しているソフトウェアおよびそのライセンスの状態を適切に把握するためには、製品や開発対象に組み込まれるソフトウェアの構成管理(Software Configuration Management, SCM)を適切に行う必要があります。ソフトウェア構成管理とは、狭義には単にバージョン管理を指すこともありますが、広義には変更依頼、要求の変更、機能追加・バグ改修、ビルドやパッケージ化による成果物の生成を行うとともに、任意のバージョンの製品を再現したり、不具合や故障発生時にトレース可能にするために、ソースコードや文書などの開発生成物の変更履歴を管理するとともに、製品のバージョンや、開発対象の納入先などに応じて、個々の開発生成物のどのバージョンが対応しているかを記録、管理することです。

OSS を利用するうえで、対象とするソフトウェアのバージョンとそのライセンスを正しく把握し、製品や開発対象に組み込まれたソフトウェア間でのライセンスの競合、ライセンスと適合しない方法での利用、同一の OSS でありながらバージョンにより異なるライセンスが設定されていることによる意図しないライセンス違反、等に適切に対処するためにも、適切な構成管理が必要となります。

こうした情報を記述する方式としては、Linux Foundation が定めた標準規格「Software Package Data Exchange 1.0 (SPDX 1.0) [39]」などがあります。SPDX は、フリー/オープンソースソフトウェアライセンス情報交換のための標準規格であり、ライセンス情報の共有を容易にしたり、企業や組織のコンプライアンス管理を容易にする目的で定められました。製品に使用する OSS のライセンス情報を適切に管理するためには、こうした標準に準拠した記述方式やそのためのツールで効率的に管理することも検討しましょう。

LINUX FOUNDATION COLLABORATIVE PROJECTS																																																																										
SPDX																																																																										
SPDX License List																																																																										
<p>The SPDX License List is a list of commonly found licenses and exceptions used in free and open source and other collaborative software or documentation. The purpose of the SPDX License List is to enable easy and efficient identification of such licenses and exceptions in an SPDX document, in source files or elsewhere. The SPDX License List includes a standardized short identifier, full name, vetted license text including matching guidelines markup as appropriate, and a canonical permanent URL for each license and exception.</p> <ul style="list-style-type: none"> License Exceptions: The list of commonly found exceptions to free and open source licenses, which can be used with the License Exceptions operator, "WITH" to create a license with an exception. Master Files: The HTML pages you see here are generated from the master files for the SPDX License List. Data Files: Machine readable files describing all of the licenses and license exceptions. Overview: General information about the SPDX License List, including principles for inclusion of a license and an explanation of the fields contained on the list. Matching Guidelines: Guidelines for what constitutes a license match to the SPDX License List. The license text on the HTML pages here will display verbatim text in blue and replaceable text in red (see Guideline A2 for more information). Request New License: Instructions for proposing a license or exception be added to the SPDX License List. <p>Version: 3.3 2019-10-24</p> <p>Note: You can sort by each column by clicking on the column header. By default, the table sorts by the identifier column.</p> <table> <tr> <th>Full name</th><th>Identifier</th><th>FOSS Free/Libre?</th><th>OSI Approved?</th><th>Text</th></tr> <tr> <td>0BSD (Zero Clause License)</td><td>0BSD</td><td></td><td></td><td>License Text</td></tr> <tr> <td>Ada Kernel/AdaCore License</td><td>AKL</td><td></td><td>Y</td><td>License Text</td></tr> <tr> <td>Adaptive License</td><td>Adaptive</td><td></td><td></td><td>License Text</td></tr> <tr> <td>Apache Software Foundation Source Code License Agreement</td><td>ASL-1.1</td><td></td><td></td><td>License Text</td></tr> <tr> <td>Apache-2.0 License</td><td>Apache-2.0</td><td></td><td></td><td>License Text</td></tr> <tr> <td>Amazon Digital Services License</td><td>ADSL</td><td></td><td></td><td>License Text</td></tr> <tr> <td>Apache-2.0 License v1.1</td><td>APL-1.1</td><td>Y</td><td>Y</td><td>License Text</td></tr> <tr> <td>Apache-2.0 License v1.2</td><td>APL-1.2</td><td>Y</td><td>Y</td><td>License Text</td></tr> <tr> <td>Apache-2.0 License v2.0</td><td>APL-2.0</td><td>Y</td><td>Y</td><td>License Text</td></tr> <tr> <td>Apache-2.0 License v2.1</td><td>APL-2.1</td><td>Y</td><td>Y</td><td>License Text</td></tr> <tr> <td>Apache-2.0 License v2.2</td><td>APL-2.2</td><td>Y</td><td>Y</td><td>License Text</td></tr> <tr> <td>Apache-2.0 License v2.3</td><td>APL-2.3</td><td>Y</td><td>Y</td><td>License Text</td></tr> <tr> <td>Attribution License</td><td>Attribution</td><td></td><td></td><td>License Text</td></tr> </table>					Full name	Identifier	FOSS Free/Libre?	OSI Approved?	Text	0BSD (Zero Clause License)	0BSD			License Text	Ada Kernel/AdaCore License	AKL		Y	License Text	Adaptive License	Adaptive			License Text	Apache Software Foundation Source Code License Agreement	ASL-1.1			License Text	Apache-2.0 License	Apache-2.0			License Text	Amazon Digital Services License	ADSL			License Text	Apache-2.0 License v1.1	APL-1.1	Y	Y	License Text	Apache-2.0 License v1.2	APL-1.2	Y	Y	License Text	Apache-2.0 License v2.0	APL-2.0	Y	Y	License Text	Apache-2.0 License v2.1	APL-2.1	Y	Y	License Text	Apache-2.0 License v2.2	APL-2.2	Y	Y	License Text	Apache-2.0 License v2.3	APL-2.3	Y	Y	License Text	Attribution License	Attribution			License Text
Full name	Identifier	FOSS Free/Libre?	OSI Approved?	Text																																																																						
0BSD (Zero Clause License)	0BSD			License Text																																																																						
Ada Kernel/AdaCore License	AKL		Y	License Text																																																																						
Adaptive License	Adaptive			License Text																																																																						
Apache Software Foundation Source Code License Agreement	ASL-1.1			License Text																																																																						
Apache-2.0 License	Apache-2.0			License Text																																																																						
Amazon Digital Services License	ADSL			License Text																																																																						
Apache-2.0 License v1.1	APL-1.1	Y	Y	License Text																																																																						
Apache-2.0 License v1.2	APL-1.2	Y	Y	License Text																																																																						
Apache-2.0 License v2.0	APL-2.0	Y	Y	License Text																																																																						
Apache-2.0 License v2.1	APL-2.1	Y	Y	License Text																																																																						
Apache-2.0 License v2.2	APL-2.2	Y	Y	License Text																																																																						
Apache-2.0 License v2.3	APL-2.3	Y	Y	License Text																																																																						
Attribution License	Attribution			License Text																																																																						

図 18 SPDX により記述されたソフトウェアのライセンスのリスト

```

FileName: ./sed-4.2/COPYING
SPDXID: SPDXRef-file-COPYING-8427-faa689eb
FileType: OTHER
FileChecksum: SHA1: 842745cb706f8f2126506f544492f7a80dbe29b3
LicenseConcluded: NOASSERTION
LicenseInfoInFile: GPL-3.0
LicenseComments: <text></text>
FileCopyrightText: NOASSERTION
FileComment: <text></text>
FileNotice: <text></text>
## Relationships
Relationship: SPDXRef-file-COPYING-8427-faa689eb CONTAINED_BY SPDXRef-package-sed_4_2_tar_gz-b0a9-7f98cbdc
Relationship: SPDXRef-file-COPYING-8427-faa689eb DESCRIBED_BY SPDXRef-DOCUMENT

```

図 19 SPDX で記述された sed 4.2 の情報

5.2.10. 出荷前チェック

製品の出荷前の最終的なチェックとして、組み込まれているソフトウェアがすべて OSS ライセンスを遵守できているかを、ソフトウェア構成管理情報を基に確認します。利用している OSS とそのライセンス、各ライセンスの利用許諾条件と組み込み方について確認するとともに、OSS ソフトウェアとライセンス・著作権や免責条項の表示、ソースコードの入手方法についての表示を行うなど製品出荷とソースコードの開示などに向けた準備を行います。Web サイト等を設けて、当該製品に利用されている OSS およびそのライセンスのリストや、ライセンスの定める条件に従って、元のソースコードや改変後のソースコード、パッチなどを Web サイトであらかじめ開示しておくといでしょう。

すでに、Linux をはじめとする OSS を積極的に利用してきた家電メーカーでは、Web サイトにて製品ごとの利用 OSS リストなどの公開をすることが一般的です[40][41]。

5.2.11. 外部からの問い合わせ対応

製品を出荷した後には、利用しているユーザや、OSS の適正利用を推進する団体等からの問い合わせがあるかもしれませんので、その対応のための窓口を設置します。会社の規模によっては、開発部門や品質管理部門がこの役割を担うことになりますが、いずれにしろ問い合わせに対しては、迅速に調査を行い、誠意を持って対応することが OSS コミュニティとの関係からも大切であることは、すでに述べたとおりです。

5.2.12. 保守更新作業

製品出荷後には、製品に対する保守・更新作業が継続される場合があります。自社製ソフトウェアに関してはバグフィックス、機能追加等が考えられますが、製品に利用された OSS は製品とは無関係にバグフィックスや機能追加がコミュニティによって行われますので、それらの動きを把握することが必要になります。セキュリティ上必要な修正や、明確なバグなどは OSS コミュニティおよび自社でともに行うことになります。自社で OSS のバグを発見、修正した場合は、極力修正パッチを OSS プロジェクトに対してフィードバックすることが肝要です。また、セキュリティ上深刻な脆弱性の発見やその修正がコミュニティ側で行われた場合、製品に対して影響がある場合には極力早く修正を反映させる必要があります。

5.3.ポリシーに基づく OSS としての公開

OSS 利用に伴う派生物の公開については、上述の OSS 利用においてライセンスを順守すればおのずと正しく実施されるでしょう。一方で、自社で開発したソフトウェア、データやドキュメントをオープンソースや Creative Commons ライセンスで公開する場合には、利用とは別の観点からルール化が必要でしょう。

ただし、基本的には上で決めた OSS ポリシーに従うものとして、利用と同様にガイドライン化をするとういでしょう。OSS としての公開のための体制構築、教育や社内からの問い合わせ対応については、上述の 5.2.1 体制の構築と運用、5.2.2 教育の実施、5.2.3 社内からの問い合わせ対応、と共通であるため割愛します。

5.3.1. OSS としての公開の申請・審査・承認

自社で開発したソフトウェアを OSS として公開するにあたっての全般的なメリットやリスクについては、上述した通りですが、実際に開発したソフトウェアを公開する場合には、公開当たって、公開のメリットや、実際に OSS コミュニティにどのように利用してほしいのか、個別に検討するとういでしょう。公開を決めたら開発部門から公開予定のソフトウェアを申請し、それを他の部門や担当部署により審査・承認する仕組みを構築しておきましょう。

公開に当たっての理由の妥当性に加えて、公開に伴うリスクに対応するための、機密情報や特許が含まれていないか等の、申請・審査・承認プロセスを設け、OSS の公開に当たってのエビデンスを残すことは重要です。

5.3.2. 公開する著作物の範囲（ソースコード、モデル、ドキュメント）

公開に当たっては、公開する著作物の対象範囲（ソースコード、モデルデータ、ドキュメント）を明確にします。また、それらがどのように利用されることを期待するか、どのような利用が想定されるかなどについても検討しましょう。

他の OSS に依存しているソフトウェア、他の著作物を引用したドキュメントなどもありますが、どこまでが自社の著作物なのかについて線引きをはっきりしておくことも重要です。

5.3.3. ライセンスの設定

ソフトウェアを OSS として公開する際に、どのようなライセンスを設定するかは大変重要な問題です。まず、ライセンスを設定していない OSS は、どのような利用許諾条件か不明であるために、多くの利用者から敬遠され、使われないことになります。OSS として公開するからには、多くの人々に利用してもらいたいと考えていると思いますので、どのように使われたいかによって、適切な利用許諾条件の OSS ライセンスを選択肢、公開するソフトウェアに明記しましょう。

公開するソフトウェアにライセンスを設定する場合には、独自のライセンスを作成し設定することは一般によくないことと考えられています。多くのソフトウェアライセンスが乱立すると、利用者からすると、個別のライセンスの許諾条件をいちいち精査しなければいけません。例えば、OSI (Open Source Initiative) 等が定める OSS ライセンス[42]の中から、自分が意図した利用許諾条件に合致するものを選択し設定するようにしましょう。

5.3.4. 著作権表示

開発されたソフトウェアやマニュアル等は開発された時点で著作物となり、著作権が発生します。上述したように、著作権自体はソースコードの記述がなされた時点で発生しベルヌ条約で保護されるため、明示的に著作権表示を行う必要はありませんが、ソフトウェアやドキュメントなどに著作権表示を明記することは良い習慣ですので、公開・非公開に関わらず行うとよいでしょう。ただし、会社としての規定や、受託開発時には契約に基づくルールがあるため、ルールに従う必要があります。

一方、開発したソフトウェアを OSS として公開する場合、付与する OSS ライセンスのほとんどは著作権情報の表示義務があるため、必ずソースコードに著作権表示をし、ライセンスとともに許諾条件を利用者に明示する必要があります。著作権は以下のように、「コピーライトを表す表記 (©, (C) など)+ 著作物の発行年+ 著作権者名」で表記します。

最低限の著作権表示記載例

```
// foobar_function
//   関数やファイルの説明等
//
//   (C) 2019, Robot Revolution & Industry IoT Initiative
//
//   場合によっては、以下に免責事項やライセンス本文を記載することもある。
//
#include <iostream>

int foobar_function (int arg1)
{
    : 以下ソースコード
```

図 20 ソースコードにおける著作権図示例

なお、ソースコードに記述されることから、“コピーライトを表す表記”は、環境依存文字や全角文字記号の「©」ではなく、ASCII 文字のみの「(c)」や「(C)」を利用し、著作権者も英字表記で記載することが望ましいでしょう。また、開発したソフトウェアにおいて、OSS を利用し改変した場合は、既存の OSS の著作権情報表記を記述した上で、今回の作成者の著作権表示を記述しておく必要があります。

5.3.5. 公開のための手順

公開に当たって、他の意図しない OSS の混入や特許の調査などについては、「5.2 ポリシーに基づく OSS の利用」同様に事前に調査しておくことが必要になります。詳細については省きますが、一連の作業について、同様にガイドラインとして定め、周知しておく必要があります。

公開に当たっては、チェックに漏れが無いようにチェックリストを作っておくとよいでしょう。以下に会津大学のロボット開発プロジェクト「RTC-Library-FUKUSHIMA」における「オープンソースソフトウェア登録確認ガイドライン」のチェックリストを一例として示します。

表 5 ガイドラインチェックリスト例

確認 No.	チェック内容	チェック結果
1	・ソフトウェアの開発時、OSS を利用した開発をおこなったか？	<ul style="list-style-type: none"> ・おこなった場合、No.2 の内容を確認 ・おこなっていない場合、意図しない OSS 利用確認のため、No.4 を確認
2	・身元不明なソースコード(注: 著作者不明、ライセンス不明、公開サイト不明等、ファイル単位、コードスニペット単位に依らず)を開発に使用していないか？	<ul style="list-style-type: none"> ・使用していない場合、No.3 の内容を確認 ・使用した場合、代替コードに置き換えも検討
3	・ソフトウェア開発で他の OSS を利用した場合、OSS ライセンス違反がないことを確認しているか？	<ul style="list-style-type: none"> ・確認していない場合、確認し適正なライセンスを付与 ・意図しない OSS 利用確認のため、No.4 を確認
4	<ul style="list-style-type: none"> ・OSS の適正な利用に配慮したとしても、意図せぬ OSS が混入するリスクがある。そのため、ツールを利用してソフトウェアを検査し、意図せぬ OSS 混入のリスクを軽減することを推奨。 <p>OSS 検査ツール: Palamida、FOSSology、BlackDuck など</p>	<ul style="list-style-type: none"> ・意図せぬ OSS の混入があった場合、代替コードに置き換え、適正なライセンスを付与

5.3.6. 公開後のメンテナンス・サポート体制

開発したソフトウェアを OSS として公開した場合、公開直後からそのソフトウェアの OSS としてのライフサイクルが始まります。つまり、ユーザによってダウンロードされ、使用されたり、時には二次利用されたりすることになります。その際に、ユーザから質問やバグレポート、あるいはパッチが送付されることがあります。こうしたコミュニティとのやり取りを円滑に行うために、github のようなソースコードリポジトリおよびユーザコミュニケーションをサポートする OSS ホスティングサービスを利用することを推奨します。Github をはじめとする OSS ホスティングサービスでは、ソースコードのバージョン

ョン管理だけでなく、ユーザからの質問や問題点を挙げるフォーラムや、ソースコードのパッチを送る機能等が提供され、コミュニティとの関係構築に役立ちます。ただし、単純にソースコードをアップロードするだけでなく、ユーザなどのからの質問に真摯に答えたり、バグレポートやパッチ送付に対しても可能な限り対応するなど、公開側の努力が求められます。

5.3.7. 外部からの貢献に対する対応（ソース取り込み、注意点）

OSS として公開する場合、ユーザからバグフィックスのためのパッチや、機能追加のためのソースの提供の申し出があるかもしれません。それらの外部からの貢献を取り込む際には注意が必要です。単純に提供されたソースコードを元のソースコードに取り込むと、そのソフトウェアは自分とソース提供者の共同著作物になります。この場合、OSS ライセンスを含む利用許諾の変更や、その他著作権により守られている自分・自社の権利の一部を、貢献者と共有することになり、貢献者全員の許可が必要になります。

もし、外部の貢献者との共有著作物となることが、会社の戦略上問題になる場合には、貢献者ライセンス同意書 (Contributor License Agreement: CLA) と呼ばれるものをあらかじめ作成しておき、貢献者に対してその貢献 (提供されたソースコード等) を自社あるいはその OSS プロジェクトが利用する権利を明示的に譲渡してもらう等して、後々問題とならないようにする必要があります。

たいていのコントリビュータは、その OSS プロジェクトの発展のために無償で貢献したいと思っており、CLA に同意することに対してはほとんど問題とならないでしょう。また、github のような OSS ホスティングサービスでは、ソースコードの差分の提供を受ける際に CLA への同意を確認する仕組みをフックで切る API が用意されており、こうした仕組みを利用して外部からの貢献を自動的に安全に取り込めるようにすることも可能です。

5.3.8. 開発、サポートの終了時の対応

ソフトウェアのライフサイクルでは、開発者の人的リソースが確保できなくなったり、ソフトウェアの役割が終了したりするとともに、OSS のプロジェクトが終了することがあります。

特に会社が主導してソフトウェアを OSS として公開した場合には、その会社がその OSS にたして人的リソースが割けなくなった時点で通常開発がストップします。ただし、OSS として公開しているので、ソースコードはオープンになっており、また多くのユーザが利用している場合もあるでしょう。

その場合、OSS ではユーザの一部の人々が勝手にプロジェクトを引き継ぐことも可能です。ただし、OSS を公開した企業としては、ユーザコミュニティに対して、開発が継続できなくなった旨を説明し、OSS プロジェクトを代わりに継続してくれる開発者をコミュニティの中から見つける努力をした方が良いでしょう。原著作者である企業から、開発を引き継いでくれる外部貢献者を明示的に指名することで、同じソースコードを引き継いだ OSS プロジェクトが乱立し、開発力が分散することを避けることができます。

6. おわりに

本ガイドラインでは、ソフトウェアの著作権、特許の取り扱いに関する基本的な考え方を説明するとともに、OSS を利用してロボットシステムを開発したり、開発したソフトウェアを OSS として公開したりする際に注意すべき点を解説しました。

OSS の利用にあたっては、様々なリスクもある一方で、上手に利用することができれば、開発効率・コスト、あるいは製品のマーケティングの面でも非常に大きな力を与えてくれる可能性があります。利用にあたってはリスクも見極め、正しい対応策を定め、一つ一つ着実に実行する必要があります。一つ一つの対応策はそれほど難しいものではありませんが、組織的かつ継続的に実施し続けることが肝要です。また、自社ソフトウェアを OSS として公開することも、リスクとともに大きなメリットを得ることができる可能性があります。

今後、ロボットシステムを開発における OSS の利用は、ますます広がっていくことは確実です。OSS に対する正しい考え方をもち、うまく付き合っていくことで、日本のロボット産業はまだまだ発展する可能性があります。このガイドラインが、OSS を利用したロボットシステム開発と新規ロボット市場開拓の一助となることを願います。

7. 参考文献

- [1] 独立行政法人 情報処理推進機構, “OSS ライセンス遵守活動のソフトウェアライフサイクルプロセスへの組み込み”, <https://www.ipa.go.jp/files/000029079.pdf>
- [2] 「OSS ライセンスの教科書」, 上田理著, 岩井久美子監修, 技術評論社, 2018
- [3] 独立行政法人 情報処理推進機構, “OSS ライセンスの比較および利用動向ならびに係争に関する調査”, <https://www.ipa.go.jp/files/000028335.pdf>
- [4] ThinkIT 連載: 今日から始める OSS ライセンス講座・コピーレフト型と非コピーレフト型 OSS ライセンスの違いとは?, <https://thinkit.co.jp/story/2014/02/03/4804?page=0%2C2>
- [5] GitHub.Inc Choose an open source license, <http://choosealicense.com/>
- [6] RTC-Library-FUKUSHIMA オープンソースソフトウェア 登録確認ガイドライン, <https://rtc-fukushima.jp/wp/wp-content/uploads/2017/12/891c642afbeaa0a0ee365be36d01430d.pdf>
- [7] Open Source Initiative, “The Open Source Definition”, <https://opensource.org/osd>
- [8] Free Software Foundation, “What is Copyleft?”, <https://www.gnu.org/copyleft/copyleft.html>
- [9] IPA OSS ライセンス関連情報: <https://www.ipa.go.jp/osc/osslegal.html>
- [10] オープンソフトウェア:OSS 人材育成:OSS モデルカリキュラム V2(法務分野に関する知識の項): https://www.ipa.go.jp/software/open/oss/oss_jinzai/curriculum_v2.html
- [11] オージス総研, “OSS ライセンス違反とその対策”, <https://www.ogis-ri.co.jp/otc/hiroba/technical/oss-license-violations/>
- [12] OSS×Cloud News, “OSS ライセンスをめぐる国内の係争事例”, <https://www.ossnews.jp/closeup/articles/?aid=201605-00005>

- [13]情報サービス産業協会企業ポリシー策定ガイドライン:
<https://www.jisa.or.jp/Portals/0/report/16-J013.pdf>
- [14]IPA「ビジネスユースにおけるオープンソースソフトウェアの法的リスクに関する調査」:
<https://www.ipa.go.jp/about/jigyoseika/04fy-pro/open/2004-741d.pdf>
- [15]SONY ERS-1000 (AL0.0.0.5.205) Software License Information,
<http://oss.sony.net/License/ERS-1000/>
- [16]“aibo に GPLv3 のソフトは入っているのか？ソニーが遭遇したオープンソースライセンスをめぐる顛末”, 日経 Robotics、p.16, 2018.3
- [17]クリエイティブ・コモンズ・ライセンス, <https://creativecommons.org/licenses/>
- [18]GNU Free Documentation License, <https://www.gnu.org/licenses/fdl-1.3.en.html>
- [19]GNU Free Documentation License (日本語訳), <https://opensource.jp/fdl/fdl.ja.html>
- [20]次世代知財システム検討委員会 報告書,
https://www.kantei.go.jp/jp/singi/titeki2/tyousakai/kensho_hyoka_kikaku/2016/jisedai_tizai/hokokusho.pdf
- [21]松村将生、「次世代知財システムの解説」、パテント、Vol.70、No.2、2017
- [22]西台満, “機械設計図の著作権”, 秋田大学教育文化学部研究紀要人, 文科学・社会科学部門 66, pp.1-6, 2011
- [23]水野 祐, “3D データが知的財産法に提起する課題～純粋／応用美術あるいは平面／立体の区別を超えて～”, パテント、Vol.70、No.2、pp.37-44, 2017
- [24]MISUMI-VONA “CAD データ利用規定”, https://jp.misumi-ec.com/contents/terms/cad_use.html

- [25]TRI HSR URDF データ, Github: ToyotaResearchInstitute / hsr_description,
https://github.com/ToyotaResearchInstitute/hsr_description
- [26]TRI HSR メッシュデータ, Github: ToyotaResearchInstitute / hsr_meshes,
https://github.com/ToyotaResearchInstitute/hsr_meshes
- [27]文化庁「著作物が自由に使える場合」、
http://www.bunka.go.jp/seisaku/chosakuken/seidokaisetsu/gaiyo/chosakubutsu_jiyu.html
- [28]文化庁「著作権法の一部を改正する法律(平成 30 年法律第 30 号)について」、
http://www.bunka.go.jp/seisaku/chosakuken/hokaisei/h30_hokaisei/
- [29]特許庁「IoT 関連技術の審査基準等について」、
https://www.jpo.go.jp/system/laws/rule/guideline/patent/iot_shinsa.html
- [30]特許庁、「特許出願技術動向調査」、<https://www.jpo.go.jp/resources/report/gidou-houkoku/tokkyo/index.html>
- [31]特許庁「平成 25 年度 特許出願技術動向調査報告書(ロボット)」「概要」、
https://www.jpo.go.jp/resources/report/gidou-houkoku/tokkyo/document/index/25_robot.pdf
- [32]社団法人情報サービス産業協会「オープンソースビジネスに取り組む SI 企業のための企業ポリシー策定ガイドライン」、<https://www.jisa.or.jp/publication/tabid/272/pdId/16-J013/Default.aspx>
- [33]サイボウズ株式会社「サイボウズの OSS ポリシー」、<https://cybozu-oss-policy.readthedocs.io/ja/latest/preface.html>
- [34]サイボウズ株式会社「サイボウズの OSS ポリシー」ソースドキュメント、
<https://github.com/cybozu/oss-policy>

- [35]Synopsys, “Black Duck ソフトウェア解析・コンポジション解析”,
<https://www.synopsys.com/ja-jp/software-integrity/security-testing/software-composition-analysis.html>
- [36]XLSoft (Flexera Software), “FlexNet Code Insight”,
<https://www.xlsoft.com/jp/products/palamida/index.html>
- [37]White Source, “White Dource”, <https://jp.whitesourcesoftware.com/>
- [38]FOSSology Workgroup a Linux Foundation Project, “FOSSology”,
<https://www.fossology.org/>
- [39]Linux Foundation, “Software Package Data Exchange® (SPDX®)”, <https://spdx.org/>
- [40]Panasonic, “Source Code Distribution Service”, <https://panasonic.net/cns/oss/index.html>
- [41]SONY, “Linux/OSS technical information”,
<http://oss.sony.net/Products/Linux/common/search.html>
- [42] Open Source Initiative, “Licenses & Standards “, <https://opensource.org/licenses>
- [43]会津大学, “RTC Library Fukushima: オープンソースソフトウェア登録確認ガイドライン”,
<https://rtc-fukushima.jp/document/>
- [44]特 許 庁, (社)発明協会アジア太平洋工業所有権センター「ソフトウェア特許入門」, 2009,
https://www.jpo.go.jp/news/kokusai/developing/training/textbook/document/index/introduction_to_software_patents_2009_jp.pdf

発行

2019 年 5 月 30 日

ロボット革命イニシアティブ協議会

ロボットイノベーション WG

ロボットソフトウェアプラットフォーム SWG

ロボット特許・ライセンス調査検討委員会