



エンタープライズ オープンソース： 実践的入門

By The Linux Foundation



あなたの所属する会社が、ソフトウェア エンジニアリングに関わる企業であれば、製品やサービスにオープンソース ソフトウェアを既に使用していると思います。そうであれば、潜在的リスクや不利益から身を守り、オープンソース ソフトウェアを最大限に活用することを確実にするためのオープンソース 戦略を備える必要があります。長年にわたり、新しい企業や業界が、オープンソースのエコシステムに流入してきました。そして、それぞれの企業が、オープンソース戦略を効率的に実践する一方で、独自の障害を克服しながら道を歩んできました。しかし、残念ながら多くの企業が、オープンソース エンジニアリングをフル活用する積極的な戦略を採用するよりも、オープンソース エコシステムに対応するだけ、という罫に陥っています。

本書は、企業のオープンソースの取り組み成果を効果的に増やすための活動の要点を説明することにより、オープンソース戦略を立案するための実践的アプローチについて解説したものであり、数百社の 20 年以上にわたるエンタープライズ オープンソースの経験に基づいています。本書は、企業のオープンソースに関わるソフトウェア エンジニアリングの管理者、開発マネージャ、コンプライアンスの専門家、上級エンジニア向けに書かれています。

目次

はじめに	5
なぜオープンソースなのか？	7
ツール ボックスの中の特別なツール	8
ソフトウェア、ソフトウェア、ソフトウェア	9
さまざまなビジネス モデルへの適応性	9
製品の依存性	10
イノベーションへの集中と近道	10
製品化の実現方法	11
オープンソース研究開発はイノベーションの成功要因	13
20年に及ぶエンタープライズ オープンソースの経験から得た教訓	14
オープンソース ソフトウェアの信頼性を確認する	15
オープンソースのスキル ポートフォリオの確認とコミュニティからの雇用	16
オープンソース戦略の開発	16
オープンソースの戦略ラダーにおける現在位置と目標位置の明確にする	18
オープンソース インフラの実装	23
Linux Foundation コンプライアンス イニシアチブへの参加	27
オープンソース チームのリーダーの雇用と昇進	28
オープンソース キャリアパスを形式化する	28
オープンソース トレーニングの開発またはアウトソーシング	29
進捗状況の追跡に役立つ測定基準の制定	30
オープンソース ファウンデーションとの関係を築く	32
プロプライエタリ ソースコードをオープンソース ライセンスでリリースするプランを立てる	32

社内コラボレーションを奨励する	33
柔軟な IT インフラを提供する	33
オープンソース イベントを主催する	34
オープンソース研究開発プロジェクトで大学とコラボレーション	34
社内プロジェクトのインナーソース プラクティスを調査する	35
なぜオープンソースの原則を取り入れるのか？	35
実際にはどのようなことか？	36
企業のための重要なオープンソースのワークフロー プラクティス	36
TODO グループへの参加	38
M&A 実践の更新	38
アウトソース開発契約の更新	38
直面する課題	39
結論	41
参考サイト	43

はじめに

エンタープライズ レベルのオープンソース ソフトウェアが使用できるようになってきたことで、企業の組織における製品開発、提供の方法が変わりつつあります。ホワイトボックスの開発コミュニティと公開ソースコードへのアクセスを組み合わせることで、ソフトウェアの調達、実装、テスト、展開、保守の方法について、各組織で異なった考え方をとることができます。これにより、開発コストの削減、製品開発の高速化、コード品質基準の向上など、さまざまなメリットを提供できる可能性があります。

ソフトウェア開発の大きな原動力として、オープンソース ソフトウェアを開発するためには、企業はビジネス レベルでの目標を定め、オープンソース ソフトウェアの使用の際に直面するすべての制約を明確にする必要があります。新しいポリシーの下でコンセンサスを形成し、

ビジネスの合理化を共有することが目標です。本書では、オープンソース ソフトウェアに対する防御的なアプローチから、強力なオープンソース エンジニアリングに支えられた攻撃的なマーケット リーダーシップへと変革する戦略の開発を手助けします。

なぜオープン
ソースなのか？

このプロセスの第一段階では、オープンソース エンジニアリングの成果、プロジェクト、または、業界のイニシアチブから何をgetたいのか、自問してください。オープンソース ソフトウェアの使用と開発から恩恵を受ける方法はたくさんありますが、万能のプランはありません。組織に合った戦略をカスタマイズし、オープンソース戦略の開発と実践から利益を得られるよう、企業のすべての局面を熟考することが重要です。

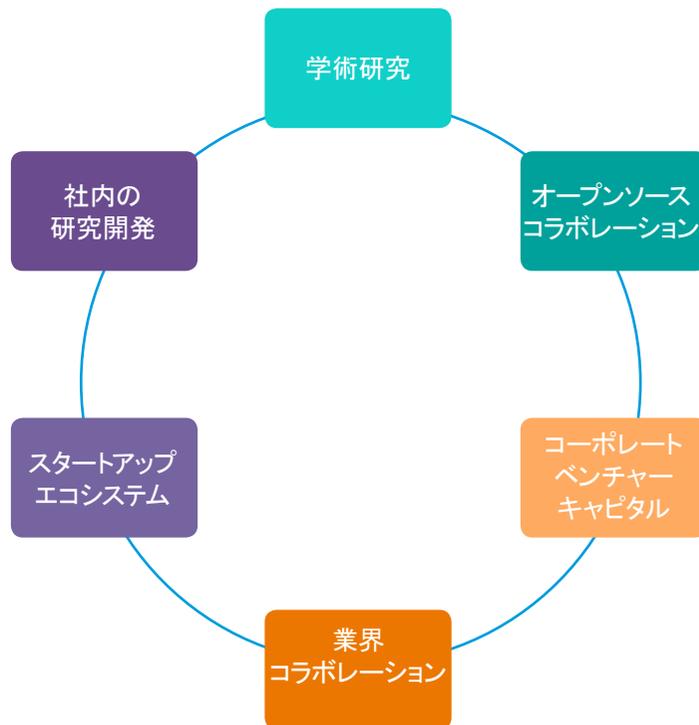


図 1：オープンソースは、企業がイノベーションを加速させることができる重要な要素です

ツール ボックスの中の特別なツール

まず、第一に、オープンソース ソフトウェアによって共同開発を可能にし、何十億ドルもの価値を持つオープンソース ソフトウェアの恩恵を受けることで、研究開発費を削減できます。特に、製品のニーズと要件を対象とするオープンソース ソフトウェアのプロジェクトのアップストリームと連携させることにより、製品開発をスピードアップし、開発期間を短縮することができます。オープンソース戦略にとって、このような調整を行うことがとても重要です。

オープンソース ソフトウェアの開発では、各種製品で使用される技術に強い影響を与えることで、業界のリーダーシップを推進することもできます。これにより、オープンソースがイノベーション エコシステムの重要なコンポーネントに取って代わり、競合する製品とサービスのコモディティ化を可能にします。また、オープンソースの開発力が強い企業は優秀なソフトウェア人材を引き付けるため、オープンソース開発に参加していることにより、人材確保において優位に立つことができます。

ソフトウェア、ソフトウェア、ソフトウェア

不動産の価値基準は立地だと言われるように、ソフトウェアは、事実上、すべての業界において、企業の価値を決める要因になっています。例えば、地元の電器店で売られている携帯電話について考えてみてください。携帯電話は、ハードウェア、ディスプレイ、カスタマイズできるメモリ、保存容量、防水加工または耐水性などを備えた素晴らしい機器です。本当の価値は、使ってみて初めてわかります。携帯電話を優れた機器にしているのはソフトウェアであり、消費者が次から次へと買い換える理由は、ソフトウェアの違いからなのです。

関連した話として、すべての業界の垂直スタックを見ると、オープンソースの普及率は、20%から85%と驚異的に増加しています。どのような業界も、どのような製品やソフトウェアの開発も、オープンソースソフトウェアに大きく依存していることでしょう。

さまざまなビジネスモデルへの適応性

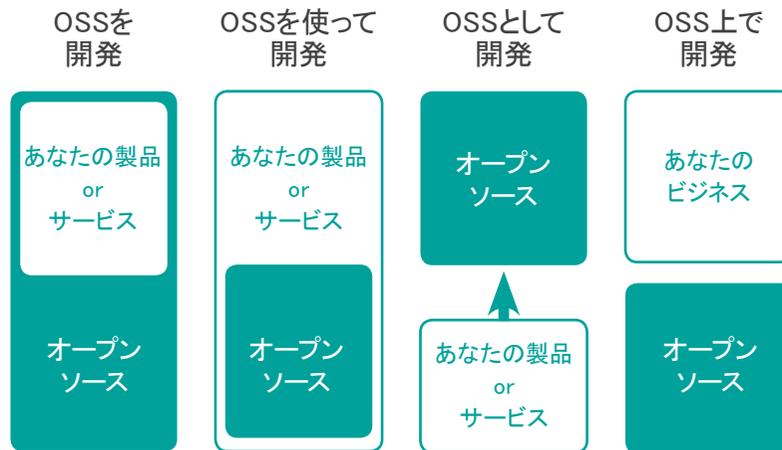


図2: ビジネスモデルをサポートするオープンソースソフトウェアのさまざまな使用方法

特定のライセンス要件はさておき、オープンソースソフトウェアは、図2で示すように、さまざまなビジネスモデルをサポートしています。この図で示すように、さまざまな方法でオープンソースを使用することができます。

これらのユースケースは、だいぶ前から実証されています。それぞれを説明するために、ビジネスモデルとユースケースは分類されていますが、これらのモデルのハイブリッドも存在し、それらは企業の製品戦略やサービス戦略によって決まります。一般的に、企業は、さまざまな製品やサービスを提供するために複数のモジュールを使用します。

オープンソースを開発する

オープンソースの開発は、最も基本的で、最もやりがいのあるモデルです。オープンソースソフトウェアの開発と商用化によって、直接的な利益の獲得が可能になります。

この目的のために、第一に企業は、オープンソース ソフトウェアを開発することに集中し、専門的なサービスと製品によってその価値を提供します。Red Hat は、このモデルを活用して大成功を収めた企業の最も良い例でしょう。

オープンソースを使って開発する

このモデルでは、オープンソースとともに、またはオープンソース上で稼働するプロプライエタリなソフトウェアやサービスを開発することができます。基本的な下位レベルのライブラリやとコンポーネントを提供するために、企業は、オープンソース ソフトウェアで開発するか、オープンソース ソフトウェアに依存します。これは幅広く使用されているモデルであり、オープンソース ソフトウェアが組み込まれていないソフトウェア製品はほとんどありません。

製品の依存性

組織が、オープンソースを使用しないで製品を開発することは、ほとんどありません。実際、我々の開発物は、すべて何らかの形でオープンソースに依存しており、これは商用製品の一部にソフトウェアを使用する企業にもあてはまります。自社の製品がオープンソース ソフトウェアに依存していて、その製品が高い市場価値を持っている場合、数十億ドルもの研究開発価値を無視しようとするのでしょうか？戦

イノベーションへの集中と近道

オープンソース ソフトウェアは組織の間で最も一般的な分野であるため、ソフトウェア スタックの下位レベルで頻繁に使用されます。下

オープンソースとして開発する

これは、ソフトウェアを開発してオープンソースとして提供し、付加価値を追加することで収益を得るというモデルです。つまり、企業は、製品やサービスを開発してオープンソース化し、それを利用したビジネスを構築しようとしています。ソースコードをオープンソース化する企業もこのモデルの一例です。

オープンソース上で開発する

この戦略では、企業はオープンソースを基盤とするオープンソース ソフトウェア上で製品を開発し、実質的な商業的価値が存在するスタックの上位レベルの部分を提供します。このモデルでは、プロプライエタリなソフトウェアやサービスはオープンソース ソフトウェアに強く依存しており、今日の新しいビジネスのほとんどは、この開発モデルとオープンソース エコシステムに大きく依存していくでしょう。

略的オープンソース プロジェクトに計画的に参加することで、社外の研究開発を活用して市場参入をスピードアップし、企業の最終収益を大きく引き上げます。

位レベルのコンポーネントをより有効に活用し、自社のリソースをソフトウェア スタックの上位レベルにおける差別化に集中させることで、

より特色のある製品を消費者に提供できます。これは、オープンソース ソフトウェアが可能にする基本的なビジネス上のメリットです。

考えてみましょう。製品を購入するときに特別なソフトウェア ライブラリが使用されているという理由で購入しますか？おそらく違います。そうではなく、エンドユーザの経験に基づいて製品を購入するのです。そしてそれは、競合製品よりも優れた機能や改良点によって決まる

のです。下位レベルのコンポーネントを開発する作業から解放されることにより、使用者が最も求めている価値を生み出すために貴重なリソースを使うことができます。

製品の実現方法

オープンソース ソフトウェアでより良い製品開発を実現する方法は、2 つあります (図 3)。

1. **直接的実現方法**：オープンソース開発関連の社内活動と会社が開発する製品またはサービスの間に直接の連携がある場合。
2. **間接的実現方法**：オープンソース活動に費やす努力が、企業が開発する製品とサービスに直接関係しない場合。

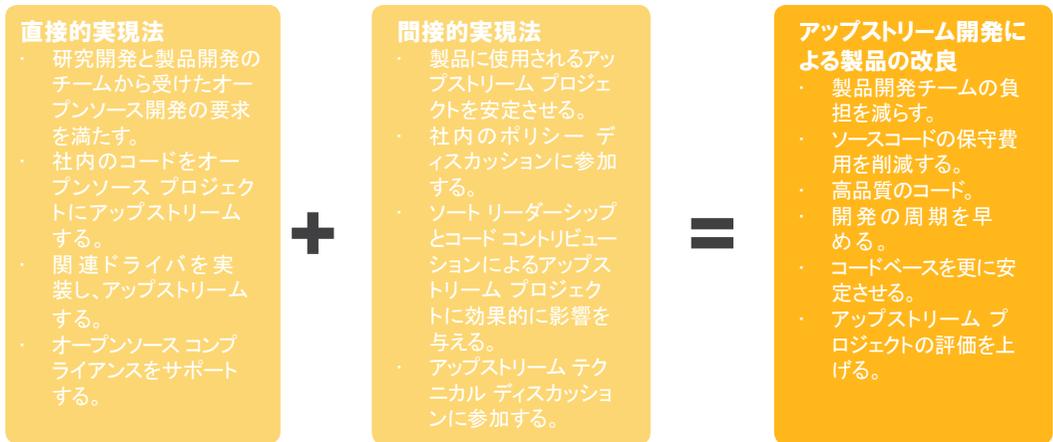


図3: オープンソース製品の實現方法

この影響は、オープンソース活動の一環で行われた特定の行動によって追跡することができます。

直接的な製品の實現方法

オープンソース プログラムは、製品とサービスに関連するコードをコントリビュートすることにより、組織のオープンソース コード開発に直接影響を与えます。オープンソース チームが製品開発を直接改善することができる方法は、下記のとおりです。

1. **研究開発と製品開発のチームから受けたオープンソース開発の要求を満たす。**製品開発チームには通常、オープンソース プロジェクトにコントリビュートする開発者がいますが、彼らは製品開発に束縛されていて、ほとんど自由がありません。多くの場合、製品開発チームに配置されているオープンソース開発者は、アップストリームの責務 (コミットまたはメンテナとして) と製品開発者としての役割の間でバランスを取るのが困難です。そこで、オープンソース グループが、「オープンソース プロジェクト Y で実装する機能 X が必要」という要求を受け、エンジニアリングチームがオープンソース プロジェクトに製品のコードを提供します。

2. **社内のコードをオープンソース プロジェクトにアップストリームする。**
ソースコードのコントリビューションは、オープンソース プロジェクトにおいて影響力を高め、企業の評価を高める最善の方法です。アップストリーム開発の主な目標は、製品やサービスに使用する重要なオープンソース コンポーネントに関連する技術的問題を減らすことです。つまり、企業がオープンソース ブランチと社内でのブランチの間の差分を最小化しようとしています。アップストリームへのコントリビュートが行われない場合、製品開発チームはアップストリームと連携しない大きなコードベースで身動きが取れなくなり、製品を向上するために時間を費やす代わりに、シンクロしないフォークに更新をバックポートするために膨大な時間を費やすことになるでしょう。
3. **コンプライアンス サポート。** プログラム マネージャは、コンプライアンス問題の解決支援を提供し、コンプライアンス チームに寄せられたオープンソース コンプライアンスに関する問い合わせをサポートします。

間接的な製品の実現方法

オープンソース プログラムの影響は、コード コントリビューションを超えて、さまざまなオープンソース プロジェクトに及びます。オープンソース プログラムは、PR、マーケティング、法的支援、開発者教育など、さまざまな方法で開発を可能にします。

1. **製品に使用されるアップストリーム コードを安定させる。**
オープンソースに専念するエンジニアリング リソースを持つ部署は、バグを発見し、修正し、テストすることによって、企業が使用しているオープンソース プロジェクトのコードを安定させます。自社を含むすべてのプロジェクト ユーザのために、コードの全体的な価値を向上させます。

2. **社内のポリシー ディスカッションと決議に参加する。**
オープンソース エンジニアリングは、従来のプロプライエタリ ソフトウェア開発とは異なる要件を持ちます。オープンソース チームの代表は、ポリシーがオープンソース開発の可能性を維持できるように、社内のポリシー ディスカッションに出席すべきです。
3. **リーダーシップとコード コントリビューションによって、アップストリーム プロジェクトに影響を与える。** オープンソース プロジェクトに影響を与える唯一の方法は、直接参加し、コードをコントリビュートすることです。開発の方向性に影響を与え、技術を成長させることでリーダーシップを発揮したいなら、必要なコードをコントリビュートするエンジニアが必要です。
4. **社外の技術ディスカッション。** オープンソースのエンジニアは、技術ディスカッションへの参加を通じて、コミュニティに影響を与えることができます。ディスカッションに参加するために、メーリングリストや IRC チャンネルを有効にし、最新のプロジェクト情報をアップデートするようにしましょう。公式なガバナンス構造を伴う大きなプロジェクトでは、質の高い人材には技術運営委員会への出席を要請する場合があります。
5. **社内の技術ディスカッション。** オープンソースの開発者は、組織の決定が、特定のプロジェクトのコミュニティの方向性と適合することを確実にするために、社内のポリシーとアーキテクチャ ディスカッションに参加できます。オープンソースの開発者は、オープンソース コードに依存する製品の長期的計画に関する戦略会議に出席すべきです。

オープンソース研究開発はイノベーションの成功要因



オープンソース研究開発は、企業の全体的なイノベーションのパイプラインにコントリビュートする複数の取り組みの1つです（図4）。学術研究、ベンチャー キャピタル、業界のコラボレーション、社内の研究開発、スタートアップ エコシステムなど、さまざまな活動と統合して、それぞれの効果を高めることができます。オープンソースの成果を複数のビジネス イニシアチブで共有する時、オープンソースの研究開発は最も効果を発揮します。

図4: オープンソースは多くのイノベーションの成功要因の1つです

20 年に及ぶ
エンタープライズ
オープンソースの
経験から得た
教訓

IBM が Linux の研究開発に 10 億ドルを投資すると発表した 2000 年 12 月は、オープンソースの歴史上、重要な節目となりました。ほとんどの企業がこれほどの規模の投資をしなかった時代、Linux とオープンソースに賭けた IBM は、業界における真の開拓者です。IBM は、オープンソース ソフトウェアとそのコミュニティについて、多くを学ぶ必要がありました。これは、企業のオープンソース導入のスタート ポイントであり、その後、数十の、そしてさらに数百の企業がこれに追随しました。

オープンソースがソフトウェア開発の新しい標準となった今、何千もの企業が、オープンソースのエコシステムに参加し、その一部となることを望んでいます。問題は、企業のラーニング カーブをいかに最小化し、正しいプロセスをスピードアップさせるかです。次節では、数百社の 20 年以上にわたるエンタープライズ オープンソースの経験から得た知識について探っていきます。

これらの実践に加えて、従来のソフトウェア開発の実践から、よりオープンでコラボレーティブな考え方に文化的なシフトを先導していく必要があることを言及することは重要です。企業内の力学は、オープンソースの取り組みに有利である必要があります。組織内のオープンソー

ス リーダーは、リソースに資金を供給し、ROI を正当化し、アップストリームに焦点を当てるなどの課題に直面するでしょう。このような課題に挑戦するには、考え方や知識の束縛から大きくシフトすることが求められます。以降の節で、これらの課題のいくつかについて探っていきます。

オープンソース ソフトウェアの信頼性を確認する

企業のオープンソース業務の改善への第一ステップは、企業がオープンソース ソフトウェアに依存している部分を明確にすることです。製品のポートフォリオに対するオープンソース コンポーネントの依存のレベルと重要性のレベルで、企業はいくつかの選択オプションを持っています。オプションの 1 つは、多くの事業単位に使用されているソフトウェアに集中することです。もう 1 つのオプションは、コンプライアンスのリスクがさらに高いソフトウェアに集中することです（モバイル アプリや組み込みハードウェアは、データセンタのコードよりもさらなるコンプライアンス問題をもたらすかもれません）。このような

手段によって、複数の事業単位やハイリスクな分野の投資利益率を示し、さらなる資金とサポートのチャンスを得ることができます。

企業の戦略や製品に直接利益をもたらすアップストリーム プロジェクトへのコントリビュートに集中しましょう。異なる興味深いプロジェクトを飛び回って夢中になることは簡単です。オープンソース エンジニアリングがコストセンターだと考える企業は、製品開発をサポートするプロジェクトに集中すべきです。

多くの製品で共通に使用されているオープンソース プロジェクトに参加するために、企業内の製品ポートフォリオを毎年見直すことは有用です。このリストは、その後、既存の限定されたリソースでトップ プロジェクトの取り組みに集中するため、いくつかの要因に基づいて優

先順位が付けられます。優先度によって決める方法論は、重要で正当性があり、資金投入可能なことに焦点を合わせ続けるための素晴らしい方法で効力があります。

オープンソースのスキル ポートフォリオの確認とコミュニティからの雇用

企業がオープンソースに依存する最も重要な方法を認識したら、オープンソースに必要な技能に適合した社内の優秀な人材を育てる必要があることを実感し始めるでしょう。社内のオープンソース専門家の育成には多くの時間を費やし、重要な開発者を雇用することは、企業が迅速に技術、認識、指導の能力を獲得するための重要なステップです。

まずは、社内の 2、3 人が Linux カーネルのような大きなプロジェクトに大きな影響を与えることでさらなる採用を呼び込むことにつなげつつ、既存の若手開発者を指導するのに十分なリソースを確保するのが理想です。

コミュニティで影響力がある、同僚たちに十分に認められる人材を見つけることが目標です。人材雇用において、技術ドメインの専門知識、オープンソースの技法と経験、仕事の実務、企業の利益と候補者の利益の一致、という考慮すべき 4 つの柱があります。特定のプロジェクトで、オープンソースの上級開発者の個人的な利益と技能が企業の利益と一致しない場合、彼らをやる気にさせることは非常に困難です。Linux のメモリ管理の専門家は、企業の優先事項であるファイルシステムには興味がないでしょう。よって、利益の一致を見つけることは、長年にわたる関係を持続するために重要です。

オープンソース戦略の開発

オープンソース戦略を決定する際に答えるべき多くの質問があり、それらには早い段階で答える必要があります。オープンソース戦略は、4 つの重要な要件に対処しなければなりません (図 5)。

- ・ 取り組みを目標とするオープンソース プロジェクト

- ・ 関与することを目標とするそれぞれのオープンソース プロジェクト コミュニティ
- ・ 社内のエンタープライズ オープンソース ガバナンス
- ・ 社内の企業文化とそれがオープンソースへの取り組みを可能にするか否か



図5: オープンソース戦略の4つの柱

次節では、オープンソース戦略を策定する前に考慮する必要がある3つの主要な質問について取り上げます。

いくつかの戦略目標から選択することが可能ですが、オープンソースソフトウェアを使用し開発する多くの企業に共有の目標があります。

- ・ 開発費の削減
- ・ 製品の品質と柔軟性の改善
- ・ 製品の市場参入の迅速化
- ・ コミュニティへの関与によるエンジニアリング能力の向上
- ・ 自社のオープンソース活動に対する開発コミュニティのコミットメントの強化

オープンソース ソフトウェアを使用しコントリビュートする企業の顧客による共通のメリットは、以下のとおりです。

- ・ 製品とアプリケーションの費用を低減
- ・ 高品質で信頼性のある製品とアプリケーション
- ・ 新製品、機能、アプリケーションの迅速な市場参入

オープンソース戦略は、全体的な企業目標の実現にどのように役立つか？

あなたが答える最初の質問は、オープンソース ソフトウェアが全体的な企業目標のどこに適合するかを特定することです。企業は多くの分野において、オープンソースの使用とコントリビューションから恩恵を得ることができ、戦略は特定のニーズに適応するように調整することができます。この定義により、企業は、オープンソースによって最も利益を得られる分野に集中することができます。設定した目標は、以下の事項のいずれかを含みます。

- ・ 製品のエコシステムにおけるリーダーシップの地位を獲得するための長期的で優れたロードマップ
- ・ 製品とサービスを開発し維持するための費用と複雑性を軽減
- ・ さらに高い利益率を目標とした差別化されたポジションの構築
- ・ オープンソースによる代替品を開発することによる競争製品もしくはサービスのコモディティ化
- ・ 製品とサービスの全体的な品質の改善
- ・ オープンソースに関与することによる社外の認知度とブランド力の向上

オープンソース戦略は IP 戦略の実現にどのように役立つか？

オープンソース ライセンスは、プロプライエタリ ライセンスとは異なります。企業は、ライセンスの違いが、オープンソース ソフトウェアの

使用と開発から恩恵を受ける能力に、どのような影響を与えるかを考慮しなければなりません。設定する目標には、次のいずれかを含みます。

- ・ より良いプロプライエタリ製品を実現する一方で、企業が社外活動に関与することで利益を得るために最適なライセンス戦略を決定する。
- ・ 製品とサービスで使用するオープンソース ソフトウェアのコンプライアンスを確実にすることにより、知的財産のリスクを軽減する。
- ・ コアとなるオープンソース コンポーネントを改善することで、自社が持つ知的財産の差別化価値をさらに高める。

オープンソース戦略は、達成不可能とも思える機会を掴むのにどのように役立つか？

オープンソースは、オープンソース戦略を通じてのみ得ることができる特有の機会を提供します。

このカテゴリに分類される一般的な目標は、以下のとおりです。

- ・ 製品の差別化機能を補完する重要なオープンソース技術を強化するための研究開発への投資に集中することにより、市場のリーダーシップを発揮する。
- ・ 主要なオープンソース イニシアチブやコンソーシアムをサポートし、競合他社または競合する市場を混乱させるオープンソースとして独自の機能を選択的にリリースしたり、技術競争条件を公平化するためにオープンソースを使用することによって既存の市場ポジションを守る。
- ・ すぐに使用可能なオープンソースのコモディティ化機能や市場拡大力を製品に組み込み、売上原価の改善を促進する。結果的に、製品出荷コストは段階的に低減される。

オープンソースの戦略ラダーにおける現在位置と目標位置を明確にする

オープンソースの戦略ラダーには、使用、参加、コントリビューション、リーダーシップという4つの主要な戦略があります。それぞれの戦略が、その前の戦略の成功を必要としており、企業がこの戦略のラダーをどこまで上がるかは、すべてあなた次第です。

図6は、これら4つの基本的なオープンソース戦略と位置を表しています。1つの位置から他の位置へ移行する際、それぞれが重なり

合っているのが見られます。通常、初期段階である使用と参加は、開発を加速する技術的メリットに基づいてエンジニアがさまざまなオープンソース コンポーネントを使用し始め、特定プロジェクトで会話に参加したり、小規模なコントリビュートをするなどの限られた参加活動を行うことで進められます。この方法は、時間とともに企業の上級レベルに知られるようになり、勢いが増した時、このような関与は、確固たる戦略に基づいて、ビジネスへと結実します。

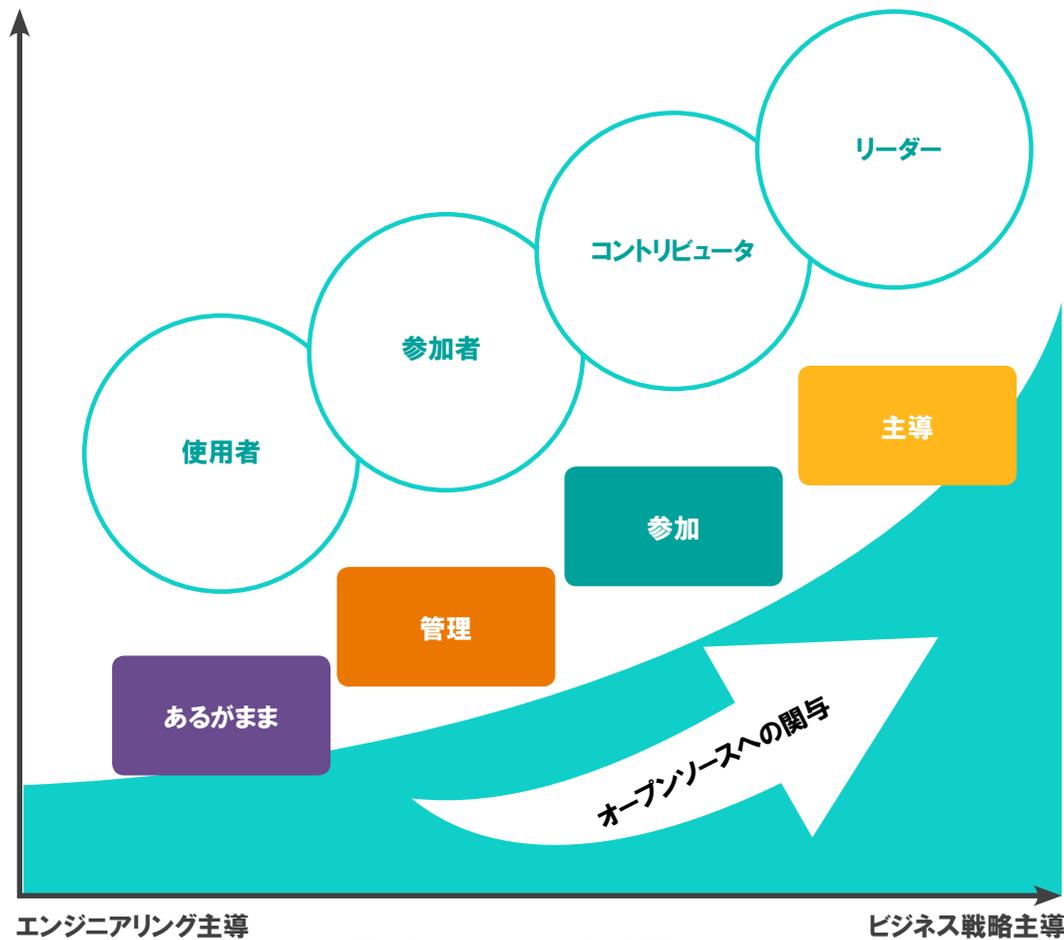


図6: オープンソースへの段階的関与

単に使用者として目標を達成し、そのレベルに留まることに甘んじる企業もありますが、リーダーとしての地位を手に入れて、さらに上位のランクにのし上がる野望を抱く企業もあります。すでにリーダーのどこかのレベルにいる可能性が高く、現在位置と目標位置の両方を確認し、現在位置から目標位置へ移行する道順を図に表すことは重要です。

戦略：オープンソースの使用

組織の一般的なスタートポイントは、自社の商用製品がオープンソースを活用することです。積極的にオープンソース コンポーネントを活用することで、差別化の能力を高め、また製品を供給するのに費やす時間と費用を削減することができます。オープンソース使用戦略に必要な要素は以下のとおりです。

- ・ どのようなオープンソース ソフトウェアを活用するかを決定する戦略的な分類スキームを使用します。
- ・ 企業が確実にオープンソース ソフトウェア使用の全責任を負うようにします。
- ・ オープンソースの使用を評価・承認するための自動ワークフロー ソフトウェアを導入します。
- ・ すべてのオープンソース アクティビティの情報センターとして機能を果たすため、速やかにオープンソース レビュー ボード (OSRB) を立ち上げます。
- ・ エンジニアリング、製品管理、および法務関連の人員やインフラへの投資を拡大し、クローズド ソースとオープンソースが複雑に結び付いたソフトウェアを管理できるようにします。

戦略：オープンソースへの参加

企業が製品やサービスでオープンソース ソフトウェアの使用に成功すれば、オープンソース コミュニティへの参加に向けて戦略を拡大することができます。コミュニティから経験豊かな開発者をまだ雇用していない場合、まず初めに、コミュニティに接近し、認知度を高め、有能な人材を引き付ける必要があります。オープンソースへの参加戦略に必要な要素は以下のとおりです。

- ・ チャットサーバー、メーリングリスト、フォーラム、ウェブサイトなどのコミュニティ コミュニケーション プラットフォームを観察し、プロジェクトの開発に関する情報を常に収集し続けます。
- ・ 関連する会議や会合に参加し、コミュニティとの関係を築きます。
- ・ コミュニティ内で認知度を高めるため、イベントやファウンデーションのスポンサーになります。
- ・ オープンソース プロジェクトへの参加やコントリビューションの方法を開発者に教えます。

戦略：オープンソースへのコントリビューション

オープンソース プロジェクトへの参加やコード コントリビューションを開始する準備ができたなら、企業のニーズを推進するために、目標とするプロジェクトやコミュニティを選別して関与する必要があります。

戦略的なオープンソース プロジェクトへのコントリビューションは、企業のニーズに合った将来機能の形成をするとともに企業価値を高めることにつながります。

オープンソース コントリビューション戦略に必要な要素は以下のとおりです。

- ・ オープンソース戦略をリードし、OSRB を管理する責任者を雇用する。
- ・ 製品開発に重要なオープンソース コミュニティに対するコントリビュータとコミッタを雇用する。

- ・ オープンソースの使用とコントリビューションをサポートするオープンソース コラボレーション ツールを導入する。
- ・ オープンソース開発者のリソースを追加する。
- ・ 既存の社外コミュニティと関与するために、エンジニアリング、製品管理、法律問題へ出資を段階的に増やす。

戦略：オープンソースのリーダーシップ

オープンソースの戦略ラダーの最後のステップは、リーダーシップです。このシナリオは、これまでのすべてのシナリオを基に最新の技術トレンドを利用し、リーダーシップの地位を確立します。

既存のオープンソース コミュニティでのリーダーシップの役割は、プロジェクト メンバとの信頼の確立とプロジェクトへの継続的な高いレベルでのコントリビューションを維持することにより、掴むことができます。

このシナリオでは、リーダーシップを確立するために、ターゲットにするオープンソース コミュニティやコンソーシアムへの大幅な投資が必要です。社外コミュニティや産業コンソーシアムにおけるリーダーシップを確立するために、エンジニアリング、製品管理、法務関連への段階的な投資も必要です。

以下は、オープンソースのリーダーシップ戦略に必要な要素です。

- ・ 対象のオープンソース コミュニティとの関係を増やす。
- ・ 企業のニーズを推進するために、選別してオープン スタンダードに関与する。
- ・ オープンソースのファウンデーションに関与する。
- ・ オープンソースのプロジェクト、組織、財団などを設立する。
- ・ 既存の社外コミュニティに関与するために、エンジニアリング、製品管理、法務関連への投資を段階的に増やす。

移行

技術企業であれば、すでにオープンソース ソフトウェアを評価し、使用し、展開しているでしょう。プロジェクトへの参加やコントリビューションもしているでしょう。そしてできれば、オープンソース プログラムがそれらの取り組みを促進し、リスクを減らし、参加活動を企業戦略への

恩恵に還元するのが理想です。図7は、一組織としての参加を段階的に表しています。初期段階は、ガイダンスのあるなしに関わらず到達することができますが、リーダーシップ段階での成功は、オープンソース プログラムが導入されていなければ達成できません。

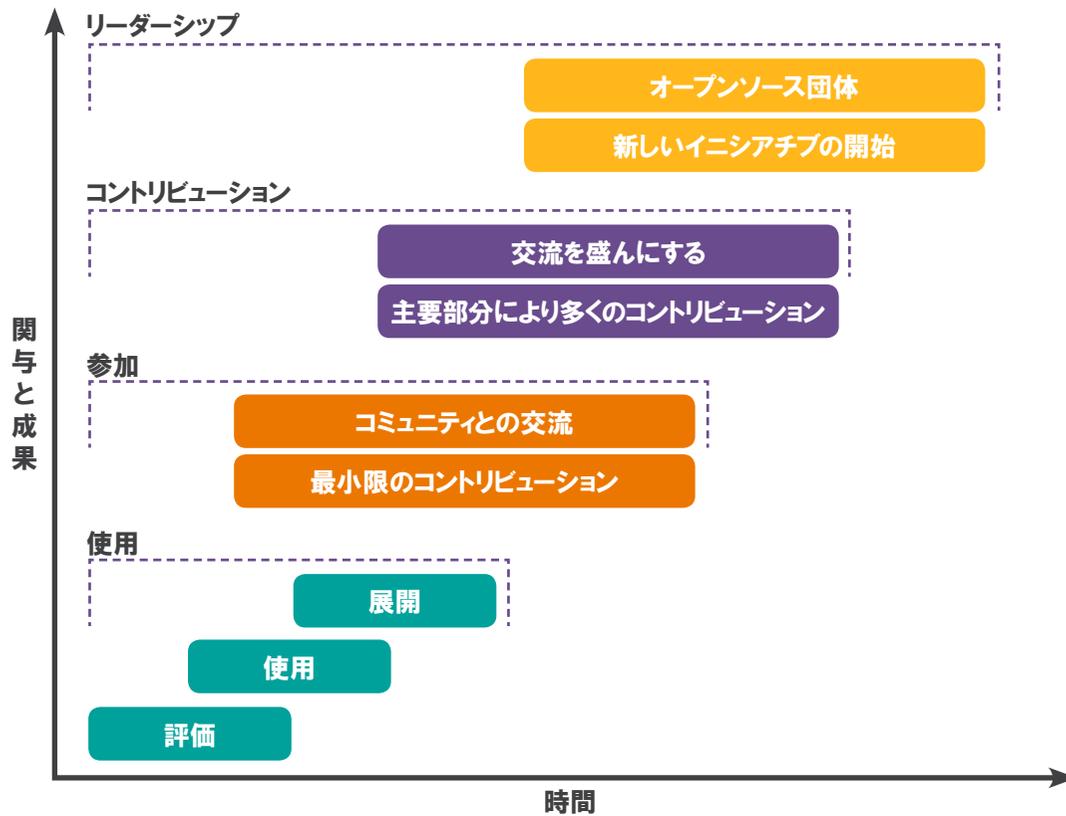


図7は、4つの重要な戦略と、各戦略の主な活動を表しています。この図の目的は、オープンソースの緩やかな増加と、企業がその導入の促進やコントリビューション作業の習得を促進するために取り得る行動を明確にすることです。

図 7: 成果と活動を増大することにより、1つのオープンソース戦略から他のオープンソース戦略へどのように進展するか？

オープンソース インフラの実装



図8: オープンソース インフラの中心的要素

企業のオープンソース戦略を確認したら、オープンソース エンジニアリングの成果をサポートするインフラを構築する必要があります。図8は、インフラがサポートする必要がある4本の柱、つまりコミュニティへの関与、オープンソース コントリビューション、オープンソース コンプライアンス、およびオープンソースの使用を示しています。

コミュニティは、これらの柱の中でユニークな存在です。なぜなら、企業が消費・コンプライアンス・貢献の側面から関与しているオープンソースプロジェクトと企業との間のすべてのやりとりに関わっているからです。

オープンソースの使用とコンプライアンスのインフラ

使用とコンプライアンスは、商用環境において、特に、使用が製品やサービスでのオープンソースの使用と同じ意味である場合、密接に関係しています。したがって、これらの中心的要素は、少なくともインフラストラクチャの観点からは、まとめて表わされることがよくあります。オープンソースソフトウェアを商業的に使用するには、対応するライセンスに準拠する必要があるからです。

図9は、オープンソースの使用とコンプライアンスをサポートするインフラの構築に必要なさまざまな要素を示しています。

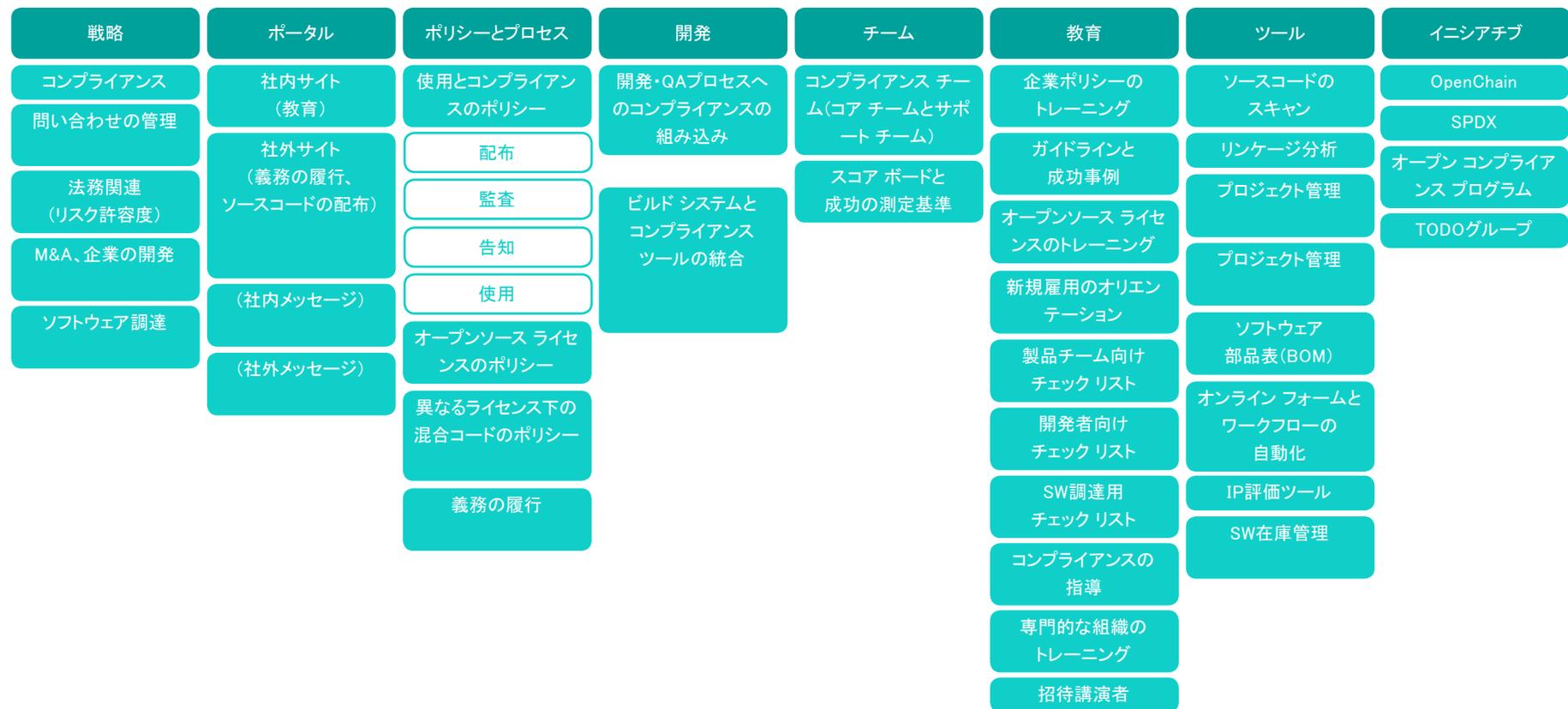


図9: オープンソースの使用とコンプライアンスのインフラ

戦略を立案する

オープンソース コンプライアンスについては、以下に関する戦略のフレームワークを含む必要があります。

- ・ 法的コンプライアンスとリスク許容度
- ・ M&A と企業開発
- ・ ソフトウェア調達
- ・ コンプライアンス調査の管理

プロセスを定義する

オープンソース プログラム オフィスは、企業がどのようにコード配布、監査、告知、使用を取り扱うかについてのプロセスとポリシーを定義します。さらに、プログラム オフィスは内部ポリシーを公表し、どのライセンスがどの種類の用途に許容されるか、ライセンスをどのように組み合わせられるか、企業としてどのようにコンプライアンスを強化するかなどに焦点を当てたトレーニングを提供します。

チームを結成する

コンプライアンスを確実にするための専門チームを結成し、チームの成功指標に同意します。

ツールを入手する

監査の自動化は、コンプライアンスを確実にするのに役立つため、ソースコード スキャン ツールに投資することを検討します。

プロセスに組み込む

オープンソース コンプライアンスを開発・QA プロセスに直接組み込みます。可能であれば、コンプライアンス ツールをビルド システムに直接組み込みます。

計画を伝える

オープンソース プログラムをサポートするには 2 つのウェブサイトが必要になるでしょう。イントラネットのサイトで、社内のオープンソースポリシーとプロセスを公表し、オープンソース コンプライアンスについて社員に説明し、ライセンス、義務、許可を確認するための社内サービスに導きます。また、社外向けのウェブサイトで、オープンソース プロジェクトを宣伝し、ニュースをリリースします。コンプライアンス レポート、告知、ソースコードについての情報公開も必要になる可能性があるかもしれません。

その他の関係者を支援

ソフトウェアの調達、開発、配布、雇用に関わる全員が、オープンソース コンプライアンスを確実に認識していることを確認します。質疑応答がしやすい環境を作りましょう。

コミュニティに参加する

オープンソースの調達、配布、コンプライアンスに特化するファウンデーションがありますが、このようなファウンデーションと関与し、企業がファウンデーションとの関与から何を得るかについての戦略を持つことは価値があります。例えば、Linux Foundation は使用とコンプライアンスの活動を支援・実現できるプロジェクトをホストしています。これらのプロジェクトについては、「参照サイト」を参照してください。

オープンソースのコントリビューション インフラ

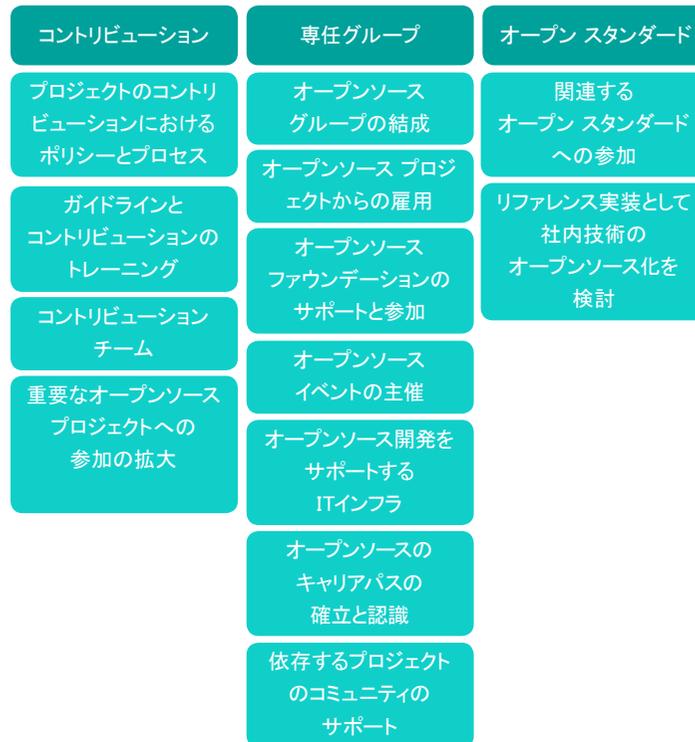


図10: オープンソースのコントリビューション インフラ

オープンソース プロジェクトへのコントリビューションに進む準備ができたなら、それらのコントリビューションをサポートし、特定のプロジェクトに何がコントリビュートでき何ができないか、という企業ポリシーを強化するのに役立つフレームワークを設定するためのインフラが必要になるでしょう。こうすることで、無駄なコントリビュートを回避することができます。

オープンソース コントリビューションのインフラには、3つの中心的要素があります（図 10）。

- ・ コントリビューションの基本要素：ポリシー & プロセス、オープンソース プロジェクトへの貢献を審査する責任を負うチーム、および一般的なガイドライン。
- ・ オープンソース グループ：オープンソースへの参加や、関連プロジェクトへの関与に専念するグループ。
- ・ オープン スタンダードへの参加：関わっている開発活動やオープンソース プロジェクトに関連するオープン スタンダードへの参加。

コントリビューションを可能にする要素

以下のコントリビューション サポートは、オープンソースのコントリビューション インフラの最も重要なコンポーネントです。

- ・ コントリビューションのポリシーとプロセス：これらは、企業のポリシーと、それをコントリビューション プロセスによって実行するかを表す。
- ・ ガイドライン：オープンソース プロジェクトに貢献する際には、これに従う。
- ・ トレーニング：コントリビューションに関する企業のポリシーやプロセスについて、開発チームが確実に理解するため。
- ・ コントリビューション アクセプトチーム：すべてのコントリビューションを審査する責任を持つ。
- ・ 優先順位：製品開発を推進する上でコントリビューションが最も必要な分野を取り上げる。

オープンソース専任チーム

アップストリーム コントリビューションに専念するチームは非常に重要になるため、コミュニティの人材を登用して、オープンソース ファウン

デーションをサポートし、オープンソース コミュニティをモデルにした IT インフラを構築し、開発者の人事キャリアパスにオープンソース メトリクスを加えます。

オープン スタandardとの関わり

関連する標準化団体や業界団体への参加は、通常オープンソース コントリビューションの要素として不可欠であり、関連する標準の最

新の変更情報を会社の中で常に把握する責任者が必要になります。

Linux Foundation のコンプライアンス イニシアチブへの参加

Linux Foundation は、フリー & オープンソース ソフトウェアのライセンスに伴うコンプライアンスを改善することを目的としたコンプライアンス イニシアチブを主催しています。主なプロジェクトは、OpenChain と Software Package Data eXchange® (SPDX®) の 2 つです。オープンソース コンプライアンスの実践をサポートするこれらのイニシアチブへの参加を強く推奨します。

OpenChain

OpenChain プロジェクトは、高品質なフリー&オープンソース ソフトウェア (FOSS) のコンプライアンス プログラムの中核となるコンポーネントを確認し、共有するのに役立ちます。OpenChain は、物事をよりシンプル・効率的・一貫性のあるものにするので、オープンソースへの信頼を築きます。サプライ チェーン全体でオープンソースのコンプライアンスを管理するための業界標準です。

OpenChain プロジェクトは、オープンソースのライセンス コンプライアンスを簡素化し一定化することにより、オープンソースにおける信頼を築きます。OpenChain は、3 つの中心要素で構成されています。

- **OpenChain の仕様**は、それぞれの品質コンプライアンス プログラムが満たさなければならない要件の中心部分を定義しています。

- **OpenChain のカリキュラム**は、OpenChain の仕様の主な要件を満たすと同時に、オープンソースのプロセスと解決のための教育的な基盤を提供します。
- **OpenChain 適合**により、組織はこれらの要件に遵守していることをアピールできます。結果として、オープンソース ライセンスのコンプライアンスは、ソフトウェア サプライチェーンの参加者にとって、より予測可能で理解しやすく、かつ効率的になります。

Software Package Data eXchange®

Software Package Data Exchange® の仕様は、ソフトウェア パッケージに関連するコンポーネント、ライセンス、著作権をやりとりする標準フォーマットです。

SPDX 標準は、ライセンス情報がソフトウェア サプライ チェーン全体で共有される方法を標準化することにより、フリー & オープンソース ソフトウェア ライセンスへのコンプライアンスを促進します。

SPDX は、ソフトウェアのライセンスと著作権の重要なデータを共有するために、企業やコミュニティに共有フォーマットを提供することに

より、不必要な業務を削減し、結果としてコンプライアンスを効率化し、改善します。

オープンソース チームのリーダーの雇用と昇進

オープンソース エンジニアリングの取り組みにおいて強いリーダーシップを確実に発揮するには、オープンソース エンジニアリングの方法論についての深い理解を持つ人材を雇用することが重要です。この人材は、以下のような特性を持つ必要があります。

- ・ エンジニアリングの豊かな経歴
- ・ 一般的なオープンソースのライセンスと義務についての深い理解
- ・ 業界慣行の知識
- ・ 企業全体のポリシーとプロセスを確立する知識と経験

- ・ 企業の製品に関する技術的な知識
- ・ オープンソースの歴史的な見解
- ・ コミュニティのコンセンサスと実践の知識
- ・ 重要なオープンソース プロジェクトのコミュニティとの接点
- ・ オープンソース団体との接点

TODO グループは、特定のニーズにカスタマイズできる職歴のテンプレートを <http://todogroup.org/blog/sample-job-req/> で公開しています。

オープンソース キャリアパスを制度化する

オープンソース開発者として雇用されたことにより、クローズドソースの開発者よりも、キャリアが昇進したことを印象付けるために、人事 (HR) システムにオープンソース開発者のキャリアパスを整備しましょう。さらに、オープンソース開発業務の目標を含む業績連動ボーナス制度を適用すべきです。プロプライエタリ、またはクローズドソースの開発者の実績を査定する測定基準は、オープンソース開発者の測定基準とは異なることが多々あります。

オープンソース開発者とクローズドソース開発者の間に明確な差がある企業も存在します。ただし、企業によっては、組織構造やオープンソース戦略によって、境界線が非常にあいまいです。しかし、現代の開発者のすべてがオープンソースを使用して開発しており、クローズドソースのみの開発者は存在しない、というのが現実です。さらに、ある場合はコードが社内に留まっていることもあれば公開されることもあります (第三者へのコントリビューション、または新規プロジェクトとして公開)。人事 (HR) システムのキャリアパスとインセンティブは、組織固有の構造とオープンソースへのアプローチを反映しなければなりません。

最終的に、一般的な企業の勤務ポリシーとは別に、オープンソース開発者には、在宅勤務を許可します。最近も多くの企業で、これとは逆の在宅勤務を禁止したり厳しく制限したりする状況が見られます。オープンソースの世界においては、オープンソースのエキスパートは世界中に存在するので、在宅勤務ポリシーはほぼ必須であり、

在宅勤務でのみ雇用が実現するケースもよくあります。また、柔軟な作業ポリシーは、運営上のメリットをもたらします。

オープンソース トレーニングの開発またはアウトソーシング

教育は、オープンソース プログラム オフィスにおいて重要な基本要素であり、2つのカテゴリに分かれます。オープンソースの技術的知識を増やすための技術トレーニングと、オープンソース ソフトウェアの使用管理ポリシーを従業員に確実に理解させるためのコンプライアンス トレーニングです。

これらのトレーニングを提供する目的は、オープンソースのポリシーと戦略の認識を高めることと、製品やソフトウェアのポートフォリオにオープンソース ソフトウェアを組み込む際のビジネス リスクと法的リスク、およびオープンソースのライセンスの問題と事実に関する一般的理解を構築することです。また、トレーニングは、コンプライアンス ポリシーと組織内プロセスの公開とプロモーションの役割を果たし、またコンプライアンスの文化を育てる場にもなります。

企業が、すべての上級開発者やエキスパートを雇用することは、不可能です。これは、Linux カーネルなどの著名なオープンソース プロジェクトに当てはまる考え方です。したがって、特定の技術領域における開発者の能力を伸ばすための方法を確立しなくてはなりません。技術的なトレーニングに加えて、オープンソースの開発モデルやオープンソースの法的コンプライアンスの基本的概念を指導するトレーニングが必要です。

トレーニング コースの例は、以下のとおりです。

- ・ オープンソース ソフトウェアの特定分野を対象とする技術トレーニング：これは通常、専門知識を増やすために、管理者もしくは上級開発者によって行われます。オープンソースのコミュニティから熟練した開発者を雇用することはかなり難しいため、専門知識を伝えることは非常に重要です。
- ・ オープンソース開発の方法論のコース：オープンソースに初めて関わるスタッフに、オープンソース開発の進め方、最善の取り組み方を指導します。
- ・ オープンソース コンプライアンス コース：コンプライアンス指針とオープンソースのライセンスの基本事項をスタッフに指導します。また、企業のポリシーとプロセスについても指導します。

OpenChain のカリキュラムは、オープンソースのコンプライアンスについてソフトウェア調達のプロセスに関係する個人を教育するために使用することができるリソースです。さらに、Linux Foundation は、オープンソースのコンプライアンスが業務とどのように関係するかを開発者に指導する **"Compliance Basics for Developers (開発者のためのコンプライアンス・ベーシック)"** トレーニング コースを提供しています。

進捗状況の追跡に役立つ測定基準の制定

オープンソース施策の履行を開始したら、要求される開発行動を促進する適正な測定基準が必要となるでしょう。しかし、生産組織で使用されている従来の測定基準は、オープンソース開発の場合は適用できません。

例えば、変更点やコードラインの数を追跡することは、オープンソース開発活動の影響度合いを知る良い測定基準となりますが、自社のオープンソース開発者がコミュニティにサポートの依頼を働き掛けたことによって、自社の必要機能が複数実装されたかもしれません。この場合、変更点の数やコード行数よりも、チームメンバがアップストリームに含め、それによって企業のダウンストリームのメンテナンスにおける労力を削減することになった、という技術リーダーシップの方が、意味があります。したがって、追跡測定基準は、両方の活動を考慮に入れなければなりません。

まず、初めに、開発者のコントリビューションと効果を追跡する社内システムを開発します。測定基準は、アップストリーム開発、製品開発チームのサポート、知識の伝達（指導、トレーニング）、知名度（出版物、講演）、新規オープンソースプロジェクトの立ち上げ、および他のチームやグループとの社内コラボレーションプロジェクトの開設を含めることができます。

サブミットまたはコミットされるパッチの数

オープンソースのリポジトリにサブミットまたはコミットされるパッチの数は、追跡のための最も基本的な測定基準であり、プロジェクトの活動の概要を把握するのに使用されます。誰がコードの作者で、どのプロジェクトにサブミットされたか、いつコミットされたかなどの情報

ソースコードのコントリビューションの追跡に役立つツールキットがいくつかあります。Linux Foundation は、gitdm と呼ばれるツールを使用しています。[Linux カーネル開発年次レポート](#)で報告されているデータは、このツールで作成されています。このツールキットは、個々の開発者とチーム全体のパフォーマンスを追跡するために使用できます。個々の開発者は、サブミットするパッチの数、パッチのアクセプト率（サブミットされたパッチ数 ÷ アクセプトされたパッチ数）、パッチのタイプ（新機能、既存機能の強化、バグ修正、ドキュメンテーションなど）で追跡されます。その他、Grimoire Lab のようなツールも、追跡したい測定基準の図示や視覚化するために使用できます。

本節の残りの部分では、追跡を考察するためのいくつかの測定基準について説明します。

を含みます。単独で使用されるべきではなく、下記の他の測定基準とともに使用するか、さらなる質的分析の開始点を特定するために使用されるべきです。

パッチのタイプ

コードがオープンソース プロジェクトにサブミットされる理由はさまざまであり、それぞれに異なるタイプの価値があります。プロジェクトで解決すべき問題の種類に応じて、次のようなコード タイプをインセンティブにすることができます。

- ・ バグ修正
- ・ 既存機能の改善
- ・ 新しい小規模機能の実装
- ・ 新しい大規模機能の実装
- ・ テスト ケースとテスト コードへのコントリビューション
- ・ ドキュメンテーションへのコントリビューション

パッチのアクセプト率

オープンソース プロジェクトのためにコードを書いたからといって、ピア レビュー プロセスを通してそれが全面的にアクセプトされることを保証される訳ではありません。時には、長期にわたって複数の修正が要求されることもあります。プロジェクトにコミットしたパッチの数対

コード ベースにアクセプトされたパッチの数の比率は、最初に自社のエンジニアがコードをアクセプトされることにもどの程度成功しているかを追跡するのに便利な測定基準です。

コラボレーション プロジェクトの一部としてコミットされたパッチ

オープンソース エンジニアが他のチームに代わって直接コードをサブミットしている場合、その影響を追跡することも有用でしょう。この測定基準によって、他のチームとの社内コラボレーション、または大

学のような社外の組織とのコラボレーションから直接生じるアップストリーム作業を追跡することができます。

認知度の測定基準

オープンソース エンジニアリングのメリットの1つに、コミュニティ内のリーダーになることで得られる柔らかな影響があります。それは、従業員がそれぞれのオープンソースのコミュニティ内で認知度を上げることによって成り立ちます。出版物、ブログ掲載、会議でのプレゼン、

メディア露出などの数を追跡する測定基準を使用することで、オープンソースのエンジニアが時間をかけてオープンソースのリーダーシップを強化するのを奨励できます。

新規プロジェクトとイニシアチブ

満たされていないニーズに応える新規のオープンソース ソフトウェアを開発することが目標の1つである場合、チームが新プロジェクト立ち上げを動機付けるために新規プロジェクトの数を追跡することができます。

オープン スタANDARDへのコントリビューション

いくつかのケースでは、標準成果と標準にリファレンス実装を提供するオープンソース実装の間に直接リンクが存在する場合があります。

この場合、従業員の標準へのコントリビューションを追跡することは有益でしょう。

オープンソース ファウンデーションとの関係を築く

オープンソース ファウンデーションは、オープンソース コミュニティにおける影響力を高めるための大きなリソースです。まずは、自社の製品や利益に関連する取り組みを主催するファウンデーションと関係を築くのがよいでしょう。多くの企業は、The Linux Foundation、Mozilla Foundation、Apache Software Foundation のような著名で確立されたファウンデーションへの参加に価値があると考えています。

法的な問題に関心がある場合は、[Software Freedom Law Center](#) や [Open Invention Network](#) などの団体に参加できます。企業が依存するエコシステムをサポートするオープンソースの組織と関与することから生じる機会を認識することが主な目的です。

プロプライエタリ ソースコードをオープンソース ライセンスでリリースする プランを立てる

多くの企業はオープンソース ライセンスの下で、コードを公開するのにテンプレートやチェックリストが役に立つと理解しています。これらは、実施すべき事柄を明確にする時間を最小限にしてくれます。当然、オープンソースとして何を公開するか、公開する目的は何なのかは、それぞれのケースで異なりますが、その仕組みは同じです。さらに、これらのツールを使用する時はいつも、将来、プロセスを改善する

のに役立つように使い方やその他のメモを残しておくことは、良いことです。

Linux Foundation の「[オープンソース プロジェクトを立ち上げる](#)」は既にオープンソースに明るい企業が自身のオープンソース プロジェクトを開始するために知るべきことを学べるように作成されました。

このガイドは、何をオープンソースにするかを決定し、予算や法律を考慮するなどのプロセスについて説明しています。また、概念的な側面から、プロジェクトの開始とメンテナンスまで、すべてを完結するために必要な主なタスクをすべて網羅した**チェックリスト**のサンプルを提

供します。チェックリストは、企業が使用できる多くのガイド例を提供しており、さまざまなチームが、オープンソース化のプロセスに何が含まれるかを、全体的に把握するための知識を得ることができます。

社内コラボレーションを奨励する

自社製品で同じオープンソース プロジェクトを使用する他の部署との社内コラボレーションは、オープンソース エンジニアリング チームの影響を拡大する最善の方法です。これらのコラボレーションでは、以下のフォームの1つまたは複数を採用することができます。

- ・ 製品開発者へのトレーニングを実施
- ・ 特定のトピックまたは問題についてのワーク ショップの開催
- ・ 新機能の開発
- ・ 問題やバグのトラブル シューティングと解決
- ・ 自社にリソースが無く活動できない既存コードのアップストリーム化
- ・ 古いフォーク版の使用を止め、アップストリーム版を使用するための支援

社内コラボレーションは多くの目的を担いますが、特に以下の2点が重要です。

- ・ 企業内の他の組織やチームに対するオープンソース チームの認知度を驚異的に向上する機会を提供します。
- ・ 研究開発や製品チームのアップストリーム パートナーになるなどオープンソースの社内エキスパートになる機会を得ます。

社内のコラボレーションは、特に大企業においてはチャレンジです。オープンソース ソフトウェアのオープン性やアップストリームとの連携が大きな利益をもたらすことは、困難さを多少軽減します。社内の他のチームは、通常これを実現するためにコラボレーションを強く望んでいます。

柔軟な IT インフラを提供する

オープンソースの開発者が、少ないチャレンジでオープンソース プロジェクトと関わり、業務にあたることができるように柔軟な IT インフラを提供します。さらに、社外で使用されているツールを社内の IT イン

フラで使用することは、社内のチームとオープンソース コミュニティの間のギャップを埋めるのに役立ちます。インフラの多くは、企業のオープンソース文化とともに自然に発展しますが、実装の必要性和計画を

認識することが重要です。オープンソース開発に使用される IT サービスには、3つの主要な領域があります。それは、知識の共有（wiki、コラボレーション編集プラットフォーム、公開ウェブサイト）、コミュニケーションと問題解決（メーリングリスト、フォーラム、リアルタイム チャット）、コードの開発と配布（コード リポジトリとバグ追跡プラットフォーム）です。これらのツールのすべて、またはいくつかは、オープンソース開発を適正にサポートするために、社内で作成される必要があります。

これらのオープンソース プラクティスは、通常、制限された多くの標準的な IT ポリシーから解放された IT インフラを必要とするため、既存の企業規模の IT ポリシーと衝突するかもしれません。その場合は、この衝突を解決し、オープンソース開発者が熟知しているツールを使用できるようにすることが必須です。

オープンソースのイベントを主催する

ローカル コミュニティの会合、ハッカソン、サミットなどのオープンソースの会議やイベントに、開発者が出席し参加するようにサポートしましょう。このような場面では、開発者が個人レベルでピア（コミュニティでともに活動しているメンバ）と接点を持ち、関係を築き、対面で社会的交流を持ち、プロジェクトの方向を指南する技術的な討論会に参加する機会を持つことができます。

他の開発者興味があるような作業を自社の開発者が行っている場合、彼らの発表コンテンツの準備を支援しましょう。なお、大小イベントのスポンサーになり、プロジェクトのコミュニティ内で社外での認

知度を高めることもできます。また、このようなイベントは、有能な人材を見つけることができる格好の場でもあります。

Linux Foundation のイベントに参加することを検討し、業務にあたっている技術に関連するイベントを探すことは、手始めとしての良い機会でしょう。知識や影響力を増やしたいと考えている開発者をイベントへ参加させることができます。また、管理責任者が重要な個人を確認し、新しい人材を雇用するために、イベントに参加することもできます。

オープンソース研究開発プロジェクトで大学とコラボレーション

多くの大学は、ソフトウェア開発の実践経験が得られるため、学生が学べる機会を提供する企業と連携することを強く望んでいます。こ

のような関係は、既存のオープンソース コミュニティで新たに有能な人材を発掘すよい機会となるため、企業にとっても有益な関わりとな

ります。特に、経験豊かな開発者が不足しているプロジェクトでは有益な方法です。世界中の才能のある人材をすべて雇用するのは無理

なので、社外プロジェクトで新しい知識を取り入れ、好ましい結果に導く方法を見つける必要があるでしょう。

社内プロジェクトのインナーソース プラクティスを調査する

インナーソースとは、オープンソース開発の方法論から学んだ教訓を企業内で使用されている開発方法論に適用するプロセスを表す用語です。コラボレーティブな開発モデルとその方法論から得られる価値と同じものを社内ですべて育てることが目標です。インナーソースの方法論は、

職務上の枠を超えた、複数の製品分野に関わる従業員同士の新しい関係を可能にします。同様の関係が、自社の従業員とオープンソースコミュニティの間にも存在します。なお、オープンソース コミュニティにも自社の他の従業員も含まれていることも多々あります。

なぜオープンソースの原則を取り入れるのか？

なぜ、オープンソースの原則を社内組織に取り入れるのでしょうか？まず、初めに、オープンソースは、非常に大きな規模で稼働します。このプロセスは、世界で最も大きなソフトウェア プロジェクトのいくつかを成功へと導き、多くの Fortune 500 企業によって採用されている証明済みのメソッドなのです。

インナーソースの方法論を採用することで、通常以下のような多くのメリットが生まれます。

- ・ 公開の頻度を早める
- ・ ソースコードの品質向上
- ・ イノベーションの拡大
- ・ 社内情報の共有拡大
- ・ 開発コストの削減

- ・ 社内コラボレーションの拡大
- ・ モラル保持の向上
- ・ 社内コミュニケーションの向上

このようにすることによって、従業員がコンテキストの切り替えなしで他の従業員や社外コミュニティのメンバーと交流することができるので、企業は、社外のオープンソース コミュニティと効果的に連携することができますようになります。新規採用の従業員は、この開発モデルを既に熟知しているので、より迅速に適応することができるでしょう。最後に、パートナーは、おそらくすでにこれらの開発プラクティスの多くを導入しているので、自社の経験は自身が属している商品エコシステムとの統合を強化することにつながります。

実際にはどのようなことか？

これを現実にするためには、まず社内のすべての従業員に対して、ベースとなるソースコードを公開する必要があります。そして、誰もが使用したり、コントリビュートしたり、必要に応じてフォークしたりできるようにします。コードの世話をした責任を持ったチームを用意し、社員からのコードの提供をレビューし、コードを適応させるようにすることが理想です。インナーソース メソッドを使用する企業は、通常、

社内の誰もが自由に変更点についての議論や質問したりできる、完全にオープンなメーリング リスト、チャット ルーム、チケット発行システムなどを使用しています。

インナーソース プラクティスの実行方法については、巻末で案内している参考サイトで確認してください。

企業のための重要なオープンソースのワークフロー プラクティス

適正なインナーソース プラクティスを実行するための基本的なステップがいくつかあります。

- ・ 可視化
- ・ フォーク
- ・ プル リクエスト／マージ リクエスト
- ・ ピア レビュー
- ・ 早期のリリース、頻繁なリリース
- ・ テスト
- ・ 持続的統合
- ・ ドキュメンテーション
- ・ 問題追跡

本節の残りの部分では、これらの概念をそれぞれ説明します。

可視化

社内のすべてのソフト プロジェクトは、デフォルトで企業の全従業員が見られるようにすべきであり、コミュニケーション手段もできるだけ

オープンにすべきです。これは、チーム間の相互作用を可能にし、従業員が責任感を共有する機会を提供します。また、組織内の従業員間のアドホックなコミュニケーションや関係構築を促進します。

時には、プロジェクトは開発に関連する機密情報のために特別な保護を必要とし、選ばれたグループだけの非公開のものになります。しかし、これらは規則というより、例外措置とすべきでしょう。

フォーク

社内のコードを閲覧できるすべての人が、自由に変更できるコピー（フォーク）を作成することが許可されるべきです。フォークは、すべての人が同様の操作ができるよう、可視化されるべきです。これは、組織内のさまざまなチームにおけるソフトウェア スタックに関する深い理解とより良い統合を促進し、結果として有益な相互作用を生み出します。

プル リクエスト／マージ リクエスト

プロジェクト外の人々が、プロジェクトの変更を提案できるようにすべきです。従業員が社外チームからの参加者を歓迎し、コントリビューションを検討するよう奨励すべきです。

ピア レビュー

ピア レビューは、オープンソース開発において、スタイルのばらつきを減らし、コードがプロジェクトの品質基準に沿うようにするための不可欠な要素です。コードは、コミットされる前に、コード ベースに深い理解を持つ人々に常にレビューされるべきです。レビューアーは、サブmitterがスタイルを改良し、アクセプトの可能性を増やせるよう助言を与えるべきです。

早期で頻繁なリリース

少ない機能でより頻繁なリリースと定期的な更新のリリースは、リリースのための統合とテストを容易にするので、コード開発の効果を向上するために重要です。さらに、バグとリグレッションの確認および修正を容易にします。毎夜、毎週、マイルストーン、リリース、長期などのリリース階層の計画を立てるべきでしょう。

テスト

単体テストと統合テストは、ソフトウェアの開発プロセスに直接組み込み、新たな問題が生まれることをあまり心配せず変更できるようにすべきです。潜在する問題は、技術的負債の蓄積を回避するために早期に特定して対処しなければなりません。

継続的インテグレーション

ソフトウェア開発のプロセスは、継続的インテグレーションを実行しなければなりません。すべての修正提案が自動的にテストされ、その結果が修正そのものに表れるようにすべきです。

ドキュメンテーション

すべてのソフトウェアのプロジェクトは、何のソフトウェアか、なぜ重要か、どのように稼働するか、どのように開発するかを記載する README を含む必要があります。優れたドキュメンテーションは、プロジェクトの明暗を分けます。

イシュー トラッカー

すべての社内プロジェクトのイシュー トラッカーは、バグのサブミット、機能の要求、質問のために、組織のすべての人にオープンでなければなりません。フィードバック数を増やすことによって、プロジェクト自体に利益を与えます。他のチームが使用できる専門知識を拡大することにより、他のチームにも利益を与えます。

TODO グループへの参加

TODO group は、オープンソース プログラム オフィスを運営するためのポリシー、実践、プラクティクスについて連携する技術系企業の集まりです。このコラボレーションは、Linux Foundation の下でコミュニティ プロジェクトとして管理され、オープンソース プログラムを確立しようとしている企業に情報を提供します。TODO グループは、オープンソース プログラム オフィスのためのガイドを公開しており、メ

ンバは、ベスト プラクティクスについてオープンソースのカンファレンスで頻繁に発表しています。

詳細については todogroup.org でご覧いただき、参加方法については、info@todogroup.org にお問い合わせください。

M&A 実践の更新

あなたの会社が合併を検討している、または買収の対象である場合、必要なレベルの情報を開示し、表明を提供するようコンプライアンス プログラムを構築する必要があります。

合併、買収の取引に関する企業ポリシーは、オープンソース ソフトウェアを考慮して更新する必要があります。

経営企画部門は、合併や買収の前にソースコードをコンプライアンスの観点から評価し、議論の妨げや企業の評価に影響を与えたりする可能性を回避しなければなりません。

買収企業にとって、包括的なコード評価は、ソフトウェア資産の適正な評価を保証し、予期しないライセンス問題が将来の価値を損なうリスクを軽減します。さらに、買収側の会社は、買収契約の中にオープンソースの開示を要求する条項を含めることができます。オープンソースの開示を要求するように勤勉慣行を更新し、開示されたオープンソースとライセンスのレビューに関するガイダンスを含めるべきです。

アウトソース開発契約の更新

ソフトウェアのアウトソース開発に関する契約は、オープンソース コンプライアンスの手順を反映するように、またこれらの契約の他の条件（表明と保証など）がオープンソースによってもたらされるリスクをカバーするのに十分広範囲であるように、更新されなければなりません。企業の経営企画部門は、使用されているすべてのオープンソー

スを発見し、ライセンス義務を満たす適正な行動が取られていることを確認するために、アウトソーシングした開発センターから受けるすべてのソースコードがコンプライアンス プロセスを経るよう義務付ける必要があります。

直面する課題

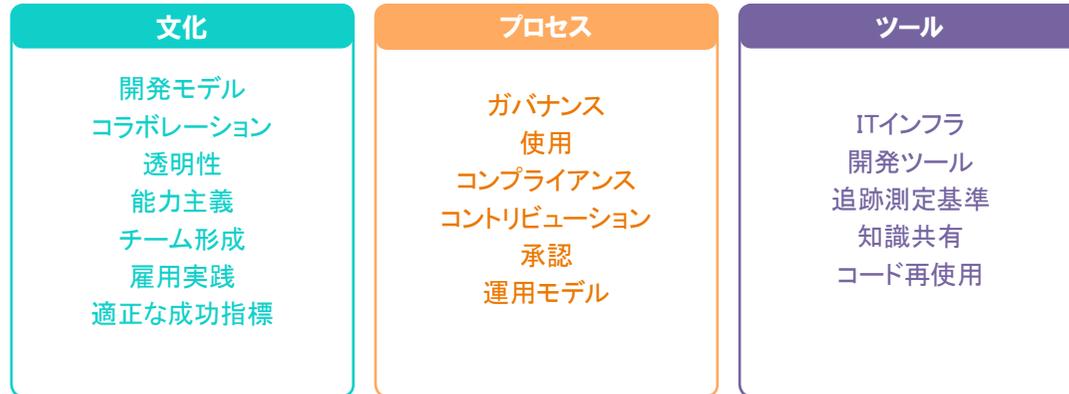


図11: オープンソース エンジニアリングの3つの主要な課題分野

オープンソース プログラムが通常重要な課題に直面する一般的分野が3つあります。それは、文化、プロセス、ツールです（図11）。それぞれのカテゴリには、オープンソース モデルに合わせて対処する必要があるいくつかの要素があります。

文化

文化的課題は、従来のソフトウェア開発経験とオープンソース開発の要件との間に差があることから起こります。オープンソースのエキスパートを雇用し、オープンソース開発モデルに熟知していない他のグループを訓練させることにより、差を埋めることができます。これらのエキスパートは、次のような助言を与えます。

- ・ 早期で頻繁なリリースとピア レビューのオープンソース開発手法に従う社内プロセスを開発する。
- ・ 部署間の透明性を高め、部門の枠を超えたコラボレーションを促進する。

- ・ 能力主義の理念でエンジニアリング チームを形成する。
- ・ オープンソースと部署を越えたコントリビューションを促進する適正な成功指標を定める。

プロセス

オープンソース開発はダイナミックで、迅速に進み、コンプライアンスの特別な要件を持ち合わせます。社内プロセスをこの種の開発に合うよう適応させない企業は、すぐに取り残される可能性があります。開発者は、コードをアップストリームに迅速にコントリビュートできる必要があります。社内のコードのポリシーは、これに対応するように修正される必要があります。

第一に、法的な問題を回避するために、適正なオープンソース コンプライアンスの維持を担当するチームを持つことが必須です。また、オープンソースの使用とコントリビューションのための簡単な社内の承認モデルを持つ必要があります。非常に複雑で面倒なポリシーから、ソースコード コントリビューションを受け取り、レビューし、承認するシンプルなアプローチへ移行する必要があるでしょう。それは、関連するすべての関係者間（法務関連、エンジニアリング、オープンソース）のバランスをとる機能です。このような歩み寄りが、献身的なオープンソース チームの助けになります。彼らは全面的な承認を得てさまざまなオープンソース プロジェクトにコントリビュートします。すべてのチームに当てはめる必要はありません。また、コントリビュートされるコードのタイプによって、異なる容認レベルにすることも可能です。（例えば、簡単なバグ調整、既存の機能を改善するコード、新機能を提供するコード、新規プロジェクトの開始）。

ツール

最後の重要なチャレンジは、初めからオープンソース開発モデルと互換性のあるツールを構築することです。オープンソース プログラム オフィスのニーズを満たし、企業の IT ガイドラインを満たすよう設定を作成する必要があります。

ここでの最大のチャレンジは、オープンソース エンジニアが主に一般に公開されているコラボレーション ツールやコミュニケーション ツールを扱うことによって生じます。これらのコミュニケーション プラットフォームは、社内の IT に直接関連がないため、サブミットするコードや関連するコミュニケーションについては、一定の考慮をする必要があります。オープンソースのエンジニアは、社外の参加者と、メール、チャット、コード開発のプラットフォームでコミュニケーションが取れるような柔軟性が必要です。また、IT ツールがこれを手助けする必要があります。例えば、オープンソース プロジェクトへのメールには、送信元の企業の知的財産権を主張するようなファイルを添付してはいけません。また、公開されているメーリングリストに企業アカウントを使ってコミュニケーションを取ることが妨害なく許可されるべきです。

さらに、広範囲なオープンソース開発の大部分は、Linux 上で行われています。開発者には、コミュニティ活動におけるやり方と同様の条件で参加できるようにするため、これを意識し IT 環境を用意しなければなりません。そのため、エンジニアは、自分が使用するディ

ストリビューションをサポートしているデバイスを必要とするでしょう。Linux を使用するすべてのエンジニアが、Linux 上で、または別の互換デバイスを介して、すべての重要な社内ツールやリソースにアクセスできるようにしてください。

最後に、オープンソース開発は世界中で行われているため、ツールは、遠隔オフィスか自宅で業務を行う完全に分散されたチームをサポートする必要があります。企業の場所が 1ヶ所の場合は、遠隔地で業務を行う人が、VPN や類似の技術を通して社内のビジネス リソースに接続することを可能にするツールが必要です。ヘルプデスク サポートなど、リモートで開発する従業員の IT の問題を解決するセキュリティを考慮したアクセス方法が適切かといった、IT のポリシーを再評価する必要もあります。例えば、遠隔地で業務を行う人は通常、社内のリソースに問題がある場合、個人的に IT ヘルプデスクを訪れることはできないので、他の選択肢を用意する必要があるかもしれません。

結論

"Open Source is eating the World" になることは明白です。あなたはそのショーを見る側になることもその一員になることも可能です。オープンソースをマスターするためには強固な戦略が必要です。消費、参加、コントリビューション、そして、リーダーシップです。さらに、それぞれには、個別の追加の努力とオープンソース エンジニアリングに対する投資が必要です。

- **使用** - 適切なオープンソース プラクティスを可能にし、オープンソース ポリシー、プロセス、チェックリスト、トレーニングを含んだ社内のインフラを構築します。
- **参加** - コミュニケーション プラットフォームやイベントで、オープンソース コミュニティとの交流を始めましょう。製品が依存するオープンソース ソフトウェアにとって重要なプロジェクトや組織のスポンサーになりましょう。
- **コントリビューション** - 特にオープンソース コントリビューションに集中する開発者の雇用とトレーニングを行い、社内のオープンソースのエンジニアリングをサポートするために必要なツールを開発しましょう。
- **リーダーシップ** - オープンソース コミュニティ、オープンスタンダードの団体、ファウンデーションとの交流を増やしましょう。新規オープンソース イニシアチブを設立し、オープンソース コミュニティでの認知度を高めましょう。

オープンソースをマスターすることには、秘密はありません。本書に掲載されている手順と原則にしたがえば、自身に合った最良の方法を見つけることができます。エンタープライズ オープンソースによるこそ。

参考サイト

OpenChain

<https://www.openchainproject.org/ja/>

SPDX

<https://spdx.org/>

Open Compliance Program

<https://compliance.linuxfoundation.org/>

TODO

<http://todogroup.org/>

Open Source Compliance in the Enterprise

<https://www.linuxfoundation.jp/compliance-and-security/2018/12/open-source-compliance-in-the-enterprise/>

Linux Foundation Enterprise Guides

<https://www.linuxfoundation.org/resources/open-source-guides/>

企業のためのオープンソース ガイド

<https://www.linuxfoundation.jp/resources/open-source-guides/>

Practical GPL Compliance

<https://www.linuxfoundation.jp/open-source-management/2017/05/practical-gpl-compliance/>

Open Source Audits in Merger and Acquisition Transactions

M&A取引におけるオープンソース監査 必須となるその基礎知識

<https://www.linuxfoundation.jp/resources/open-source-audits-merger-acquisition-transactions/>

オープンソース デベロッパーの採用
The Linux Foundation公開の[ガイド](#)

OpenChain Curriculum (OpenChainカリキュラム)
[こちらからダウンロード](#)できます。ソフトウェア調達プロセスに関わる人々がオープンソース コンプライアンスについて学べる素晴らしい
リソースです。

Linux Foundation Free Compliance Training For Developers
[Linux Foundation Compliance Basics for Developers](#)は無料のトレーニング リソースです。開発者がオープンソース コンプライアンス
と開発作業の関係を学べる素晴らしいリソースです。

Software Package Data eXchange®
<https://spdx.org/about>

Innersource: A Guide to the What, Why, and How
[インナーソースとその応用](#)に関する実践的ガイド (by Jono Bacon)



The Linux Foundation は、オープンソースがクローズドなプラットフォームと競い合う上で必要なリソースやサービスを一元的に提供し、Linuxの普及促進、保護、および標準化を推進しています。

The Linux Foundationやその他のプロジェクトの詳細については、 www.linuxfoundation.org をご覧ください。