研究開発におけるソフトウェアライセンスの選択指針

2020/03/30

塚本　明
産業技術総合研究所 (AIST)

目次

- ソフトウェアライセンスとその重要性
- ソフトウェアライセンスが生まれた背景
- オープンソースライセンスの主旨
- オープンソースソフトウェアによくみられる誤解
- 代表的なライセンスとその特色
- ソースコード公開義務のメリットとデメリット
- インターネット時代到来でオープンソース開発の活発化
- ソースコードへのソフトウェアライセンス記述方法
- ソフトウェアライセンスの一つであるGPLの伝搬性について
  - Bootloader, OS, ライブラリー、アプリ、TEEなどの状況でどこまで伝搬するか
- GPL 以外のライセンスを選択するときに考慮すべき GPL 互換性
- GPLv2 と GPLv3 の違い
- オープンソースプロジェクトが向いている用途・向いていない用途
- 著作者人格権について
- まとめ

ソフトウェアライセンスとその重要性

- ソフトウェアは申告なしで著作権により知財として守られる
  - 著作物を創造した時点で著作権が発生し知財として権利化が容易
    - 特許は出願し審査機関に認められて初めて知財となる
    - ソースコードを記述した時点で審査なしですべてが著作権がある知財となる
  - 保護期間が長い
    - 特許権の保護期間は、特許を出願した日から20年
    - 著作者の死後50年

ソフトウェアライセンスが生まれた背景

- ソフトウェア開発の工数と品質の特性
  - ソフトウェアエンジニアリングの教科書では、ハードウェア開発よりソフトウェア開発は一般的に3倍効率が悪いとされる
    - 例、Intel が毎年新しい CPU を発売可能であるが、世界トップ3に入る体力も持つMicrosoft をもってしても新しいWindows のリリースに3年以上かかっている。ソースコードの量が多すぎることでバグ修正が永遠に終了しない
      - 2001年　Windows XP　(内部バージョン NT 5.1)
      - 2007年　Windows Vista（内部バージョン NT 6.0 ）
      - 2009年　Windows 7　(内部バージョン NT 6.1)
      - 2012年　Windows 8　(内部バージョン NT 6.2)
      - 2015年　Windows 10　(内部バージョン NT 10.0)
- ソフトウェア開発の工数負担の削減と品質維持のため、会社をまたがって協力（協創）が必要になる
  - 60年代にメインフレーム開発者はイベントで集まりオープンリールでソースコードの交換を始めた
  - 会社をまたがってソフトウェアを融通しあうには取り決めが必要になる
- どういう条件を満たせば他人が著作権を持つソフトウェアを使えるかをソフトウェアライセンスで規定することになった

オープンソースライセンスの主旨　（1/2）

- もともとは組織の壁を越えて協力する目的であったことから、ソフトウェアを提供する側も、そのソフトウェアを使用する側も、協力関係が継続できるようにいくつか取り決めごとを規定する必要があった

- ソフトウェアライセンスにある代表的な記述事項 (1/2)
  - 免責事項条項：本ソフトウェア利用による事象に対して著作者はいかなる場合も一切の責任を負わない
    - ソフトウェアを使用する側は自己責任で使用すること。そもそもソフトウェアを提供する側は金銭的取り決めなしで提供していることから、特許訴訟も含めて供給責任を負わされていてはソフトウェアを提供できない。メンテナンス義務も問い合わせ対応義務もない。
  - ソースコード開示義務の有無
    - いくつかのソフトウェアライセンスは改変した場合にソースコードの開示義務の条項がある（代表的にはGPL）(参照 4)。
      - 理由1：わざわざソースコードを提供するからには提供側にもメリットが欲しい。提供したソースコードを使用する側がバグを修正したり、機能拡張されたときは、提供側に教えてほしい
      - 理由2：オープンソースソフトを使用した製品を購入した人が、製品のバグ修正や機能拡張したい時は自分でソースコードを変更して直せるようにしたい。（GPLは本理由が主である）いわゆるソースコードを修正できる権利の主張に近い
    - ソースコード開示義務がないライセンスとしては BSD ライセンスが代表的
      - 理由1：条件を付けないことで使用する側からフィードバックをもらえないことで開発速度では不利になるが、製品化が容易になる

オープンソースライセンスの主旨　（2/2）

- ソフトウェアライセンスにある代表的な記述事項 (2/2)
  - 商用利用に対する制限の有無
    - ライセンスによっては、商用利用を制限する条項があるソフトウェアライセンスもある。このようなラインセンスは、大学などの学術機関が策定したライセンスに存在が確認されている
  - ライセンスの種類によっては、付随して追加項目がある
    - コピーするときはもともとの著作者を表示してほしい。使用者がいつか金持ちになったら旅行先からポストカードを送ってほしいなど

オープンソースライセンスにみられるよくある誤解

- オープンソースライセンスに基づいているソフトはフリーソフトである
  - 「フリー」は無料という意味にも解釈できるが、ほとんどのライセンスには製品化時に無料提供しなければならないと規定する条項はない
  - 一番条件が厳しいGPLでも、ビジネス利用を促進する条文があり、実費の徴収を行ってもよいと記述がある(参照２，３)
  - したがって,多くの家電製品、携帯電話、事務機器、クラウドサービスなどオープンソースソフトを活用してビジネス展開が行われている

- オープンソースソフトを製品に利用する場合、製品で使われている他のソフトウェアのソースコードも開示なければならない
  - 一番条件が厳しいGPLであっても、GPLの制限に従って、ソースを開示したくないプロプライエタリのソフトや、別のライセンスのソフトと同居できる。ただし幾つかの条件に対応する必要がある。詳細については別の章にて後述

- オープンソースプロジェクトには学生や新卒など技術レベルが幼稚な人が参加者している
  - 世界のトップレベルのエンジニアからのレビューを受けるため、技術レベルが低い場合は相手にされないことから、各企業ともトップレベルの人材を投入している

オープンソースライセンスにみられるよくある誤解

- オープンソースソフトには製品レベルの品質がない
  - オープンソースソフトであるから品質が低い、プロプライエタリのクローズドソースのソフトであるから品質が高い、とい関連性は技術的にはない。唯一関係があるとすれば、品質問題があったときにプロプライエタリのソフトである場合は購入先にサポートをお願いできる場合がある程度。ただし、例としてMicrosoft 社のソフト製品の利用条項には免責事項があり、本当に技術的に意味があるサポートを受けることができるか状況によると思われる。したがってRedHat社のようにオープンソースソフトのサポートを受け持つ企業が現れている
  - 多くの開発者が参加して活発なオープンソースプロジェクトは、多くのエンジニアの目でソースコードのレビューを受ける結果となり、クローズドソースのソフトウェアより一般的に品質が良くなる傾向がある
  - 実際にはオープンであるかクローズであるかに関係せずに、使用する側の企業のソフトウェアに対する技術力の有無が、よいサポートを受けることができるかに影響する

- オープンソースソフト開発はボランティアである、または標準化である
  - プロジェクトはボランティアではない。iPhoneと MacOSX に使われている BSD kernel 開発を行う Darwin プロジェクトや、同様に Intel 社、IBM社、ARM社のように多くの場合自社のプラットフォーム製品の普及のために純粋にビジネス目的でオープンソースプロジェクトに貢献している
  - Linux kernel や gcc のようにオープンソースプロジェクトが成功して結果的にデファクトスタンダードになり標準化に似た効果が得られた場合もありますが、JIS, ISO, UL のように認証機関が最初に規格をドキュメントで規定する標準化とは意味合いが大きく違う

代表的なライセンスとその特色

- ソースコード公開義務ありのライセンス
  - GPL
    - ソースコード公開義務があるライセンスの代表格。またほかのソースコードへ GPL 適応を強要する条項あり(ライセンス伝搬)
  - LGPL
    - ライブラリーのようにリンクする境界でライセンス伝搬しない。ほかは GPL と似たライセンス
- ソースコード公開義務なしのライセンス
  - 3条項/2条項BSDライセンス(以降BSD)、MIT
    - ソースコード公開義務がないライセンスの代表格
  - Apache
    - 特許に関する記述あり。使用したソフトに他社特許が含まれることを気にする場合に使われる参照11
- 営利目的、非営利目的を選択できるライセンス
  - Creative Commons
    - もともとはハードウェアの回路図を公開するときのライセンスとして始まった。営利目的などそれぞれの条件を許可・不許可を選択できるライセンス。ソフトウェア自体に使われることは少ないが、ソフトのドキュメントに使われることが多い

ソースコード公開義務有ライセンスのメリットとデメリット

- メリット
  - ソフトウェアを利用する側がbug修正や機能拡張のソースコードを開発元に提供することで、開発速度が上がりオープンソースプロジェクトとして活発化しやすい
  - プロジェクトが成功すると結果的にデファクトスタンダードとなりやすい

- デメリット
  - ソフトウェア技術とソフトライセンスの知識の両方を持った人がプロジェクトに参加していない場合 GPL のライセンス伝搬を考慮した設計による開発がほぼ不可能
    - 不必要に多くのソースコードを公開することなることが多い
  - 国により傾向は違うと思われるが、誤解、勘違いから社内機密が漏れると誤解を受けることが多い
  - 特許訴訟で訴えられると誤解を受けることが多い
    - オープンソースソフトウェアが特許訴訟で負けたケースはなく、実際にはクローズドソースの製品のほうが危険確率は高い
  - 社内稟議の負荷が大きくなることが多い
    - 知財部と法務部にソフトウェア開発経験者がいないことが多いことも影響

ソースコード公開義務無しライセンスのメリットとデメリット

- メリット
  - 暗号鍵相当の情報など外部に漏洩させることができないコードを追加することが可能
  - GPL 伝搬を考慮した設計をせずに済む
    - ソフトウェアの設計自由度が上がる
  - 社内稟議の負荷が少なくなることが多い
    - ソースコードを公開しやすい

- デメリット
  - 利用者からbug修正や機能拡張のフィードバックが行われないことが多く、オープンソースのメリットであるソフトウェアの開発速度向上や効率向上につながりにくい
    - 公開したソースコードをコピーして利用するだけの利用者が占める割合が増えることで、ギブアンドテイクによる発展が起きにくい
    - コピー元から同様な機能のオープンソースプロジェクトが乱立する場合がある
  - 必ずしもデファクトスタンダードを目指していなくとも、自社のソフトウェアのソースコードを外部企業に強要するような意図には向いていない

インターネット時代到来に伴い、オープンソース開発が活発化

- もともとは物理メディアを持ち寄り、ソースコードの交換を行っていたため、オープンソースソフトウェアという概念が一部の人にしか知られていなかった。

- インターネットが普及することで web サイトまたは ftp サイトでソースコードを公開できるようになり、メーリングリストで開発を進め、その議論を反映したコードをweb/ftp サイトで公開という方式がとられるようになった。これにより、従来一つのプロジェクトに30人ぐらいが限度であった開発が、インターネット上でアクセスできる人が誰でも参加できることから、kernel や Apache など常時数百人が開発に従事できるようになり、オープンソースプロジェクトの開発速度がプロプライエタリ開発を超えるようになった。（いわゆるバザール方式参照１）

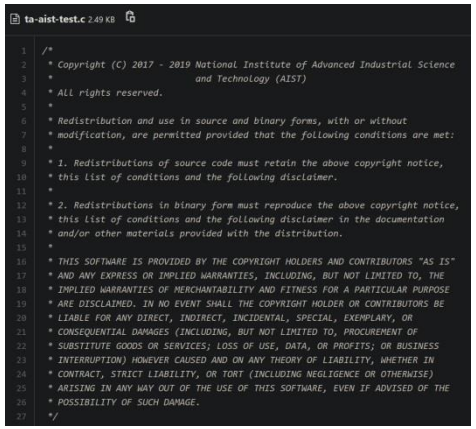
- その後、ソースコード管理サーバーの進化がオープンソース開発を促進した。従来の cvs のように一つのサーバー上でのみ開発可能なソースコード管理ツールから、git のように常時インターネット接続を要求しない分散型の開発ができるソースコード管理ツールが実現し、地域的に分散した多くのエンジニアが一つのプロジェクトへの同時開発がより容易になった。

ソースコードへのソフトウェアライセンス記述方法 (1/2)

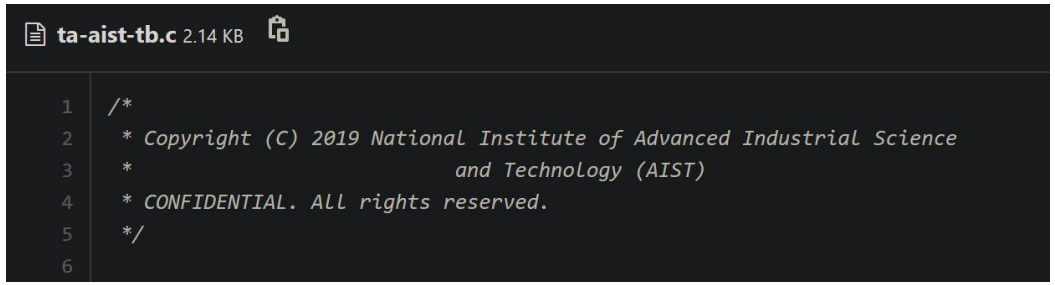
- ライセンス条項の内容を準拠しない場合はライセンス違反となり訴訟問題となる（busybox 訴訟など）。ソースコードにライセンスを判別しやすいように明記する方法のルールがある。

- 従来の方式
  - ソースコードのすべてのファイルの冒頭に著作権表示と一緒にライセンス条項全文を記述

BSD 2条項ライセンス



プロプライエタリ

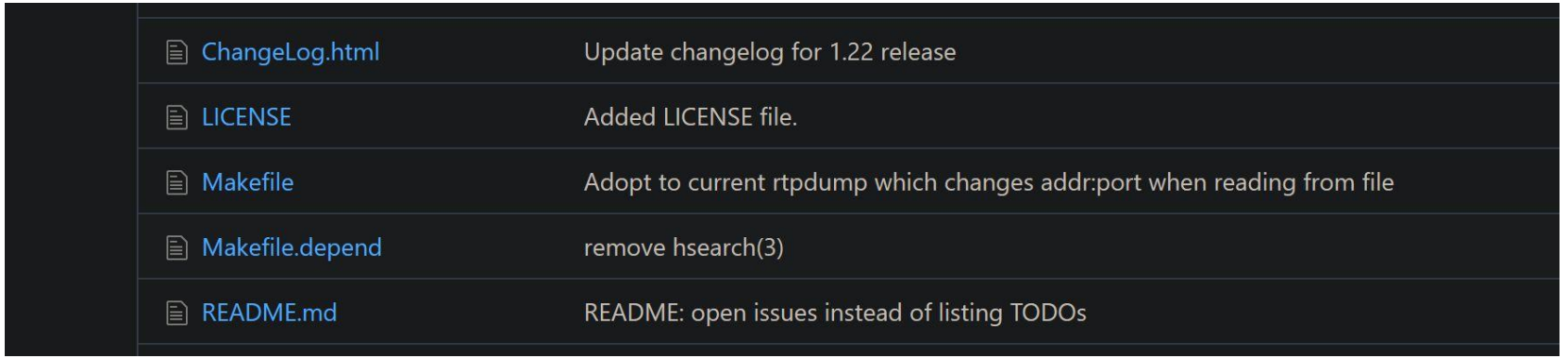


  - 課題：ソースコードの開発者が全ソースのファイルに同じ内容を記述する必要があることから、誤植のミスを起こしやすい。また利用するときはライセンス違反を行わないように、どのライセンス使われているか調べる必要があるが、全ファイルの記述を調査する膨大な工数がかかる。

ソースコードへのソフトウェアライセンス記述方法 (2/2)

- github 時代の方式
  - ソースコードのリポジトリーのプロジェクト毎にライセンス条項が記述された LICENSE ファイルを置く
    - LICENSE ファイルを置くことは 90年代から行われていたが、github にて大きく普及した
  - 各ソースファイルの先頭に SPDX-License-Identifier でライセンスの識別子のみを記述する

LICENSE ファイル



SPDX-License 識別子

```
1  /*
2   * (c) 1998-2018 by Columbia University; all rights reserved
3   *
4   * SPDX-License-Identifier: BSD-3-Clause
5   *
```

GPLの伝搬性の重要性

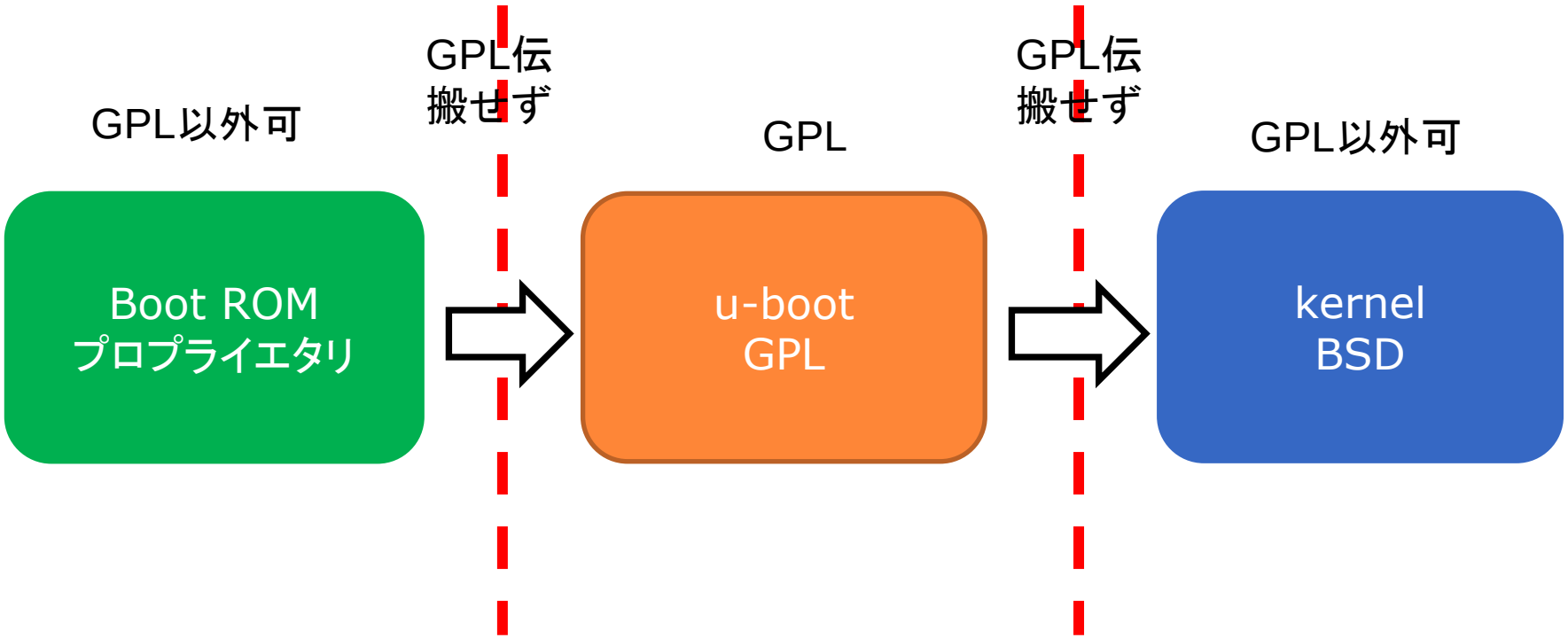
- 実際の企業や大学での研究開発では、複数のソフトウェアライセンスをつかったソフトウェアを同居する形にならざるを得ないことが多い。

- GPL には一緒にしたすべてのソースコードに GPL を適応しないといけない条項がある。
  - 公開したくないソースコードを一緒にできない（GPLはソースコード公開義務があるため）
  - クローズドソース、オープンソースに関わらず、すでにほかのソフトライセンスが使われているソースコードは一緒にできない（著作者以外は一般的にライセンスを変更できない）

- そこで、一番条件が厳しい GPL/LGPL のライセンスがどこまでほかのソースコードに波及するかの見極めが研究開発プロジェクトの知財を守るうえで重要になる
  - 現在の基本的な GPL の解釈ではコンテキストスイッチが起こる境界を越えて GPL は伝搬しない
  - LGPL はリンクする相手には LGPL が伝搬しない

GPLの伝搬性　ケース１

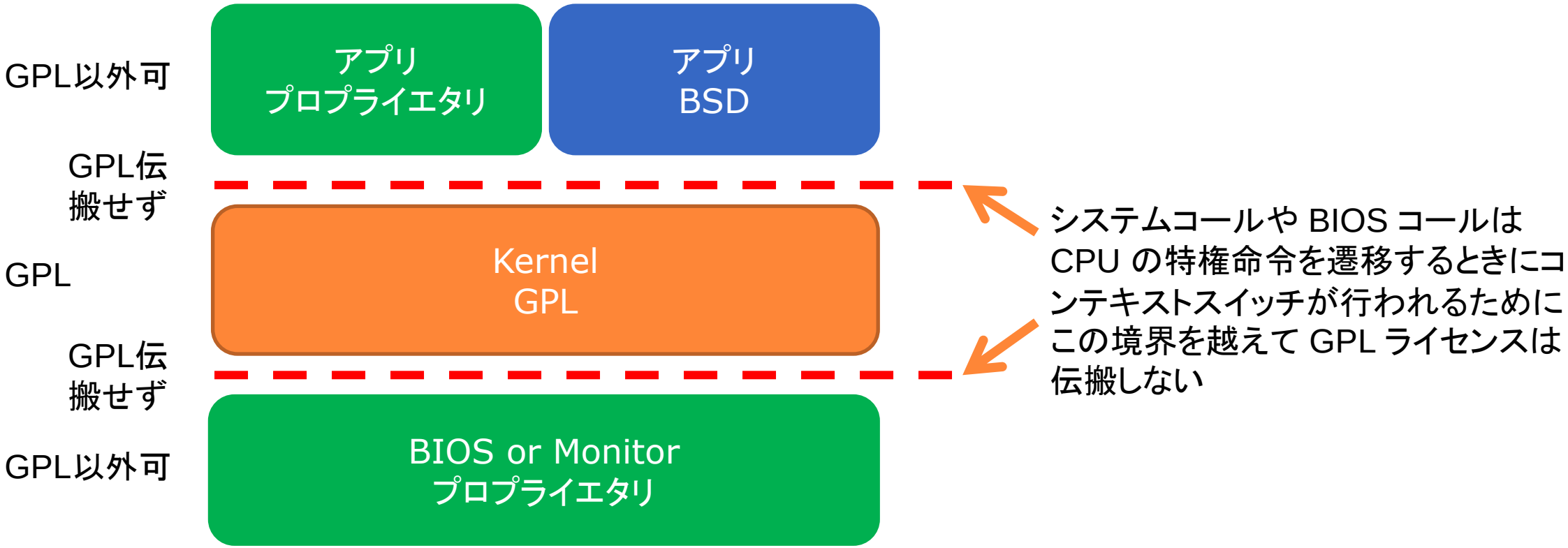
- Bootloader のように同時にソフトが実行されないケース



Boot ROMの実行が終了してから u-bootが実行され、
u-boot の実行が終了してから、Kernel が実行される

GPLの伝搬性　ケース２

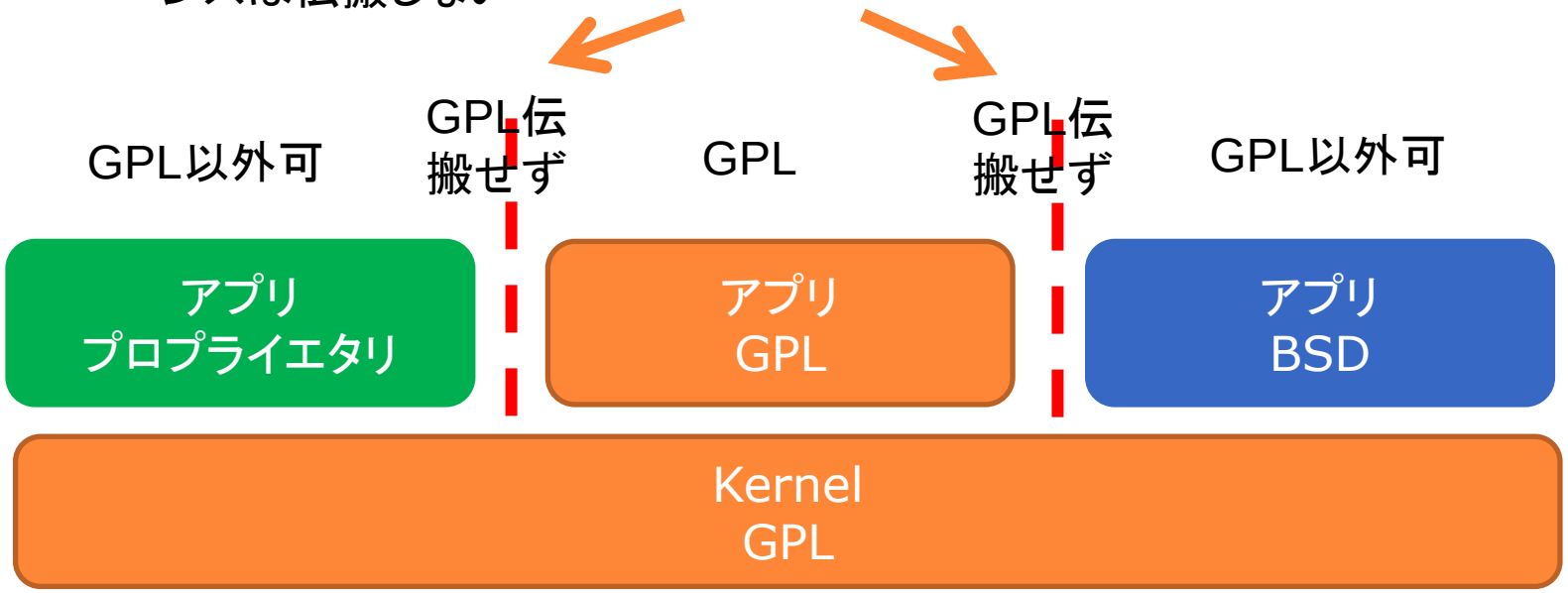
- 特権命令間のコンテキストスイッチがある場合



GPLの伝搬性　ケース３

- プロセス間のコンテキストスイッチがある場合

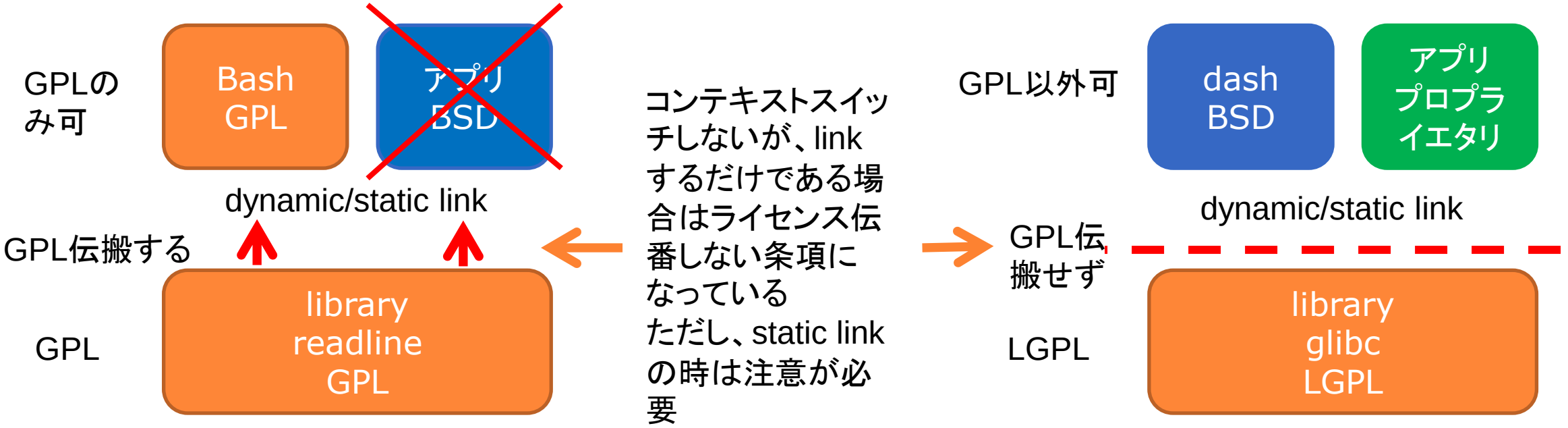
別々のプロセスで実行しているアプリの間は必ずOSを通してコンテキストスイッチが行われるために、この境界を越えて GPL ライセンスは伝搬しない



- 注意事項：mmu がない CPU では、GPL を使うことはほぼ不可能
  - アプリ間にプロセスの境界がなく一つのアプリが GPL であるとほかのアプリも GPL にする義務が発生

GPLの伝搬性　ケース４

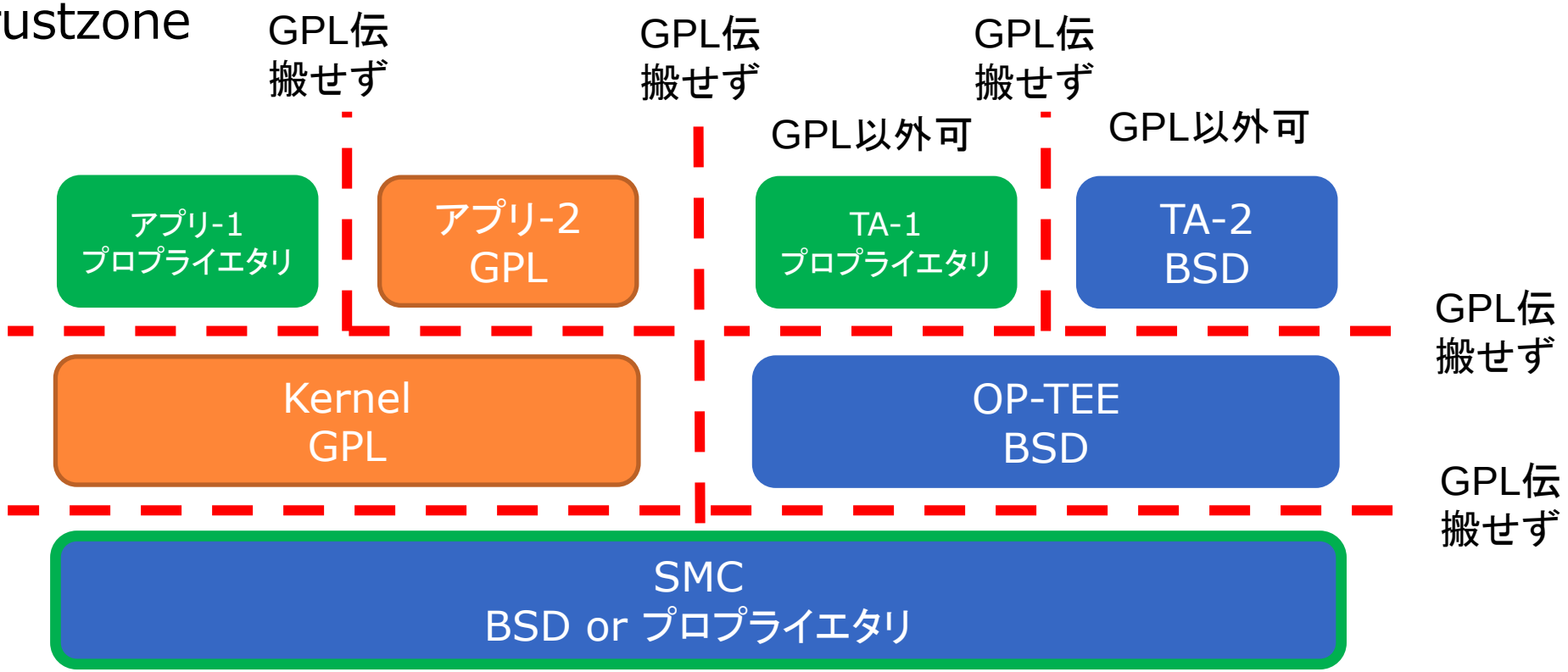
- アプリとライブラリーにおける GPL と LGPL のライセンス伝搬の違い



- 注意事項：
  - static link の時は、クローズドソース部分のバイナリーとビルド方法をエンドユーザーに提供する必要がある。
    エンドユーザーがLGPL部分のみソースのバグ修正や機能拡張を自分できるようにするため

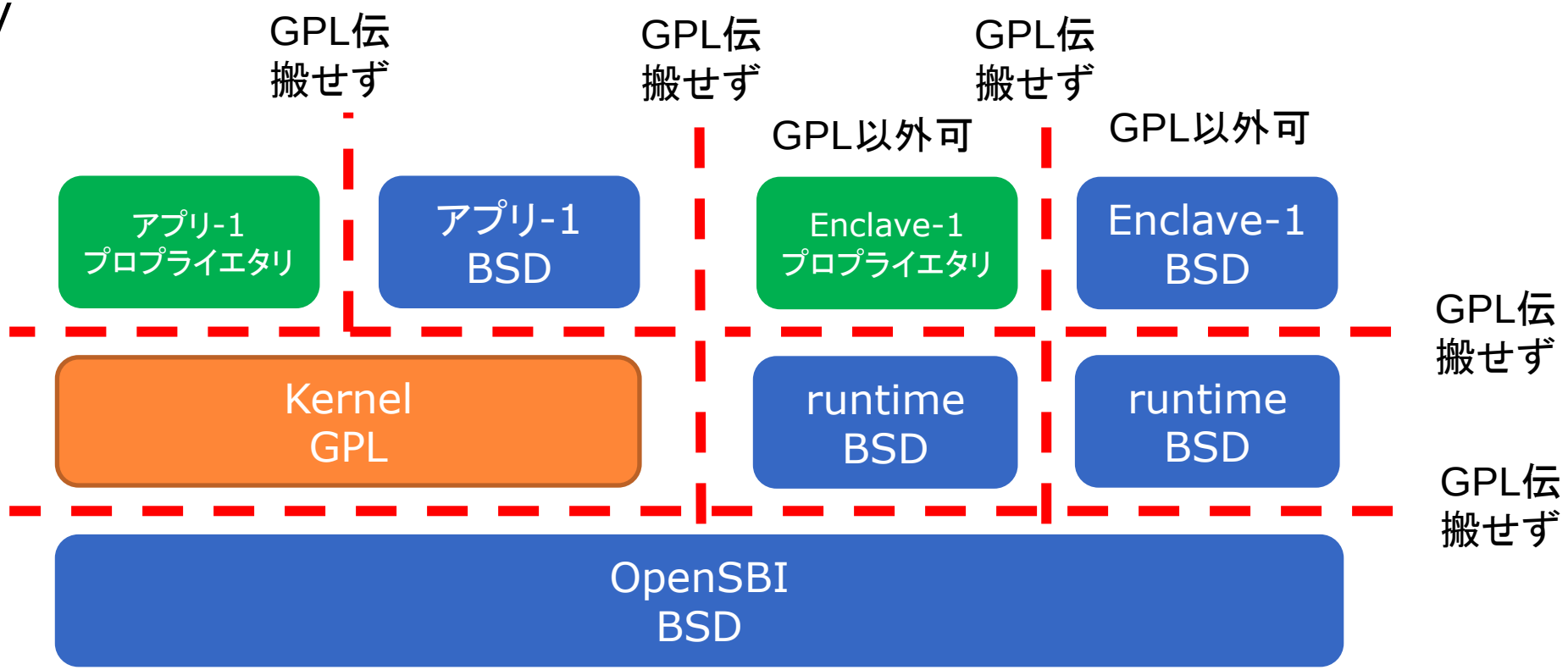
GPLの伝搬性　ケース５

- TEEによるコンテキストスイッチがある場合
  - ARM trustzone



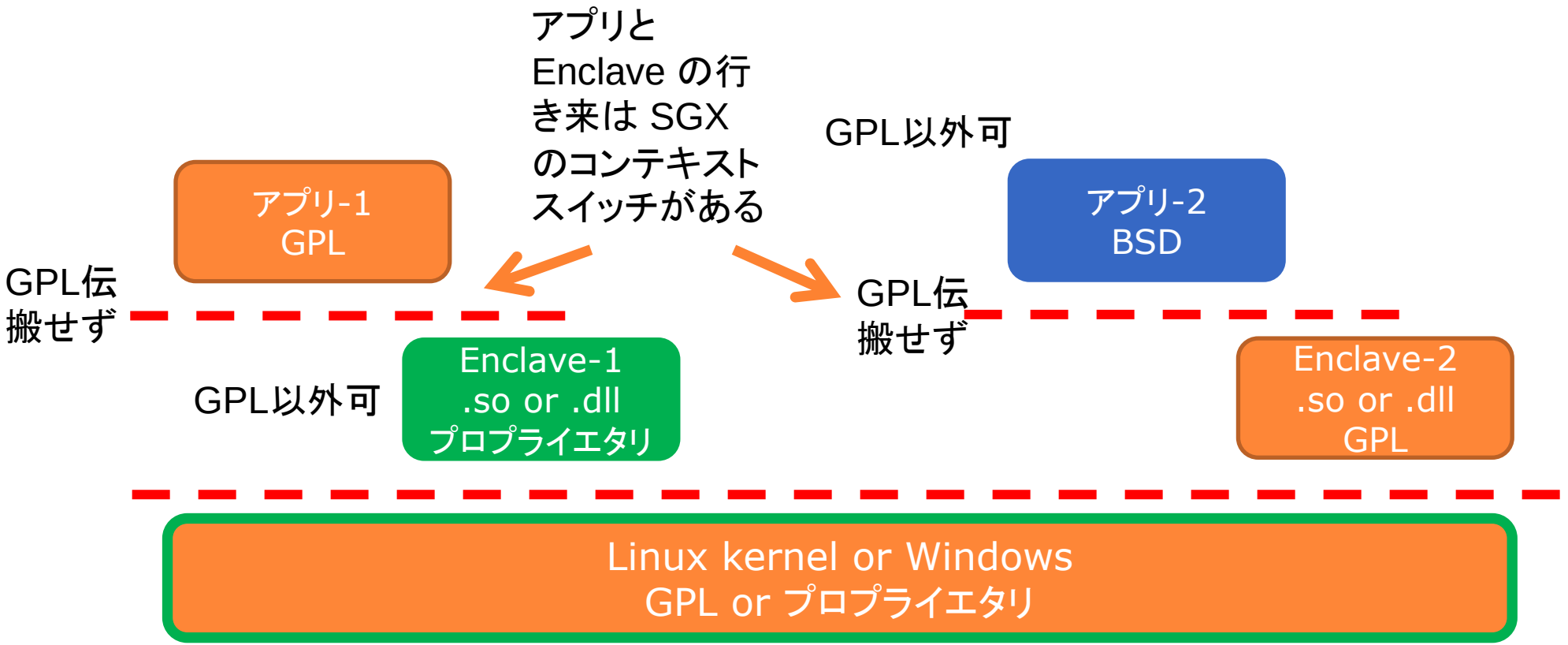
GPLの伝搬性　ケース6

- TEEによるコンテキストスイッチがある場合
  - RISC-V



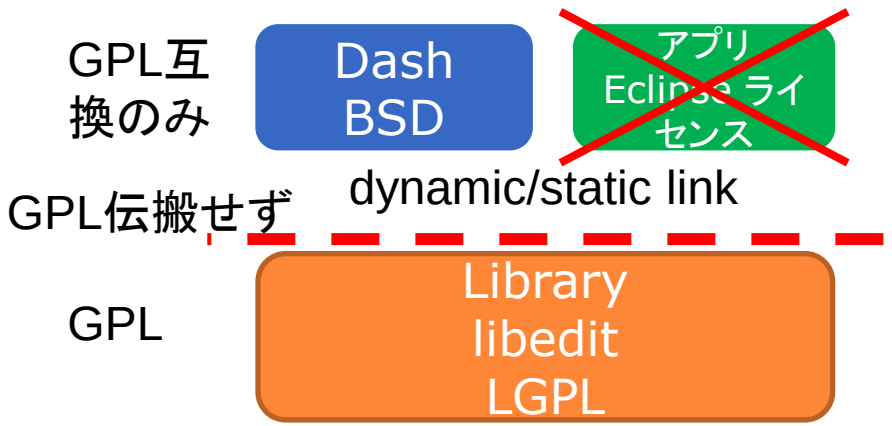
GPLの伝搬性　ケース7

- TEEによるコンテキストスイッチがある場合
  - Intel SGX



ライセンスを選択するときに考慮すべき GPL 互換性

- 実際の企業や大学での研究開発では、システム全体として内部に最低一つは GPL であるソフトウェアが使われることになることが多い
  - GPL 互換性の考慮が必要になる
- GPL/LGPL はライセンス伝搬がない状態で、ほかのオープンソースライセンスのソフトと一緒に製品として配布する場合、GPL でないほかのソフトのライセンスが GPL と互換性があることを要求する(参照 6,7)
  - GPLの条件が厳しかったとことから、GPLにある一部の条文を否定して条件を緩くするオープンソースライセンスが複数生まれたことで問題化
  - 解決方法は、GPL 互換性があるオープンソースライセンスの採用、または配布を別々にする



- GPL互換のオープンソースライセンスの代表例
  - 3条項/2条項BSDライセンス
  - Apacheライセンス
- GPL非互換のオープンソースライセンスの代表例
  - 4条項BSDライセンス参照12
  - Eclipse ライセンス

GPLv2 と GPLv3 の違い

- GPLv3 が生まれた背景
  - デジタル著作権管理技術によりソースコードを実質改変できなくなる事例が生まれた
    - もともとの GPL の主旨は「自分でソースコードを変更して製品のバグ修正や機能拡張したい」権利の主張であった。Linux など多数のオープンソースソフトウェアを使う DVD プレーヤーなどは、公開されていることでソースコードを入手できるが、ユーザーがファームウェアをアップデートできず、オープンソースソフト部分を自分で修正しても製品を改良することが不可能
    - ソースコードをビルドした実行形式(object code)をインストールする方法を明記する義務が追加された(参照 8)
  - 特許について言及
    - オープンソースに含まれる特許に対してロイヤリティー請求する事案がおきた。ロイヤリティー請求できない条項が追加された(参照 10)
- GPLv3 の評判
  - GPLv2 と比較して製品に採用しにくくなり、敬遠する企業が増えた
    - GPLv2 を維持し、GPLv3 を採用しない事例：Linux kernel
- 例外事項
  - 製造メーカーまたは第3者がファームウェアのアップデートを実施できる場合は、実行形式(object code)をインストールする方法を明記しなくてもよい記述がある（抜け穴に近いが、企業側の心理的負担は高い）（参照 9）

オープンソースプロジェクトが向いている用途・向いていない用途

- 向いている用途
  - 様々なオープンソースソフトウェアや自社製ソフトウェアを組み合わせた、商用レベルのサーバー運営ビジネス
    - 典型的な例は、Amazon AWS, Microsoft Azure, Google Cloud
    - 証明書運営ビジネスや動画配信サービスなど
  - オープンソースソフトウェアを活用したハードウェア製品
    - ソフトウェア部分だけでの課金が難しい。ハードウェアにバンドルして販売
    - 例：Mac, MacBook製品 (Apple社)、Android 携帯（各社）
  - 海外展開の意図がある用途
    - インターネット上で公開することで、国内外の顧客開拓や利用者と相互の直接交流が可能
      - 初期の Yahoo や Google の検索エンジン
      - インターネット技術自体など、リファレンス実装の公開

- 向いていない用途
  - 会社など組織にとってビジネス上の差別化部分すべてをオープンソースにすること。
    - ソースコードを公開していることから、企業組織を維持するほどの利益を上げることは実施的に不可能
    - MacOS はオープンソースを多く活用しているが、Apple社の差別化ポイント(使い勝手が売り)である GUI 機能を実現する Cocoa レイヤー以上はすべてクローズドソース。
  - オープンソースソフトウェアだけの使用によるソフトウェア販売
    - 課金をユーザーが受け入れることは非常に難しい

著作者人格権について

- 日本の著作権法では、ソースコードを記述した時点で自動的に著作者に著作権が発生し、著作権を著作者は放棄できない。
  - 本理由で米国のようなフリーソフトとも呼ばれるパブリックドメインソフトが日本では許されていない
- 日本での研究開発では、ソフトウェアを外注することが多いという背景がある
  - 外注先が著作者となる

- したがって発注元のみが著作権を執行できるように、外注先が著作者人格権を行使しない契約を行うことが必要

まとめ

- ソフトウェアは著作権にて守られ、特許権より知財として権利化が容易
- 開発効率と品質向上を目的とした技術者によりオープンソースプロジェクトがうまれ、世界中から開発に参加を可能にしたインターネット時代到来により、活動が活性化。
- 必要とされるソースコードの量が年々大きくなり、オープンソースとクローズドソースの合わせた研究開発と製品化が主体となる。
- ただ単にすべてをオープンソースにしては組織は研究開発のメリットを受けにくい。オープンソースとクローズソース混在には設計段階からソフトウェアライセンスの知識が必要
- オープンソースとクローズドソースには向き・不向きがあり、上手に活用することが現代の研究開発では求められている。

参照先

1. 伽藍とバザール
   - https://ja.wikipedia.org/wiki/%E4%BC%BD%E8%97%8D%E3%81%A8%E3%83%90%E3%82%B6%E3%83%BC%E3%83%AB
   - 企業内チームでの開発と、インターネット上で多数による集合知を集める開発の効率との比較

2. GPLv2: ソースコードを受け取れる権利の条項、商用利用は許可
   - https://www.gnu.org/licenses/old-licenses/gpl-2.0.html
   - Preamble: Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

3. GPLv2: ソースコードの配布に金銭の要求を許可
   - https://www.gnu.org/licenses/old-licenses/gpl-2.0.html
   - Section1: You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee

4. GPLv2: 全ソースコードの公開義務
   - https://www.gnu.org/licenses/old-licenses/gpl-2.0.html
   - Section3: Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange

5. SPDX 識別子
   - https://spdx.org/licenses/

6. 他のオープンソースライセンスにGPL互換性を要求する条項
   - https://www.gnu.org/licenses/old-licenses/gpl-2.0.html
   - Section10: If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission.

7. GPL互換と非互換ライセンスの一覧
   - https://www.gnu.org/licenses/license-list.en.html#GPLCompatibleLicenses

8. GPLv3：実行形式(object code)をインストールする方法を明記する義務

- https://www.gnu.org/licenses/gpl-3.0.en.html
- Section 6:
- If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information.

9. GPLv3：インストールする方法を明記に対する例外条項

- https://www.gnu.org/licenses/gpl-3.0.en.html
- Section 6:
- But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

10. GPLv3：特許のロイヤリティに対する記述

- https://www.gnu.org/licenses/gpl-3.0.en.html
- Section 11. Patents.
- Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

11. Apache ライセンス：特許のロイヤリティに対する記述

- https://www.apache.org/licenses/LICENSE-2.0
- Section 3. Grant of Patent License.
- Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted.

12. 4条項 BSD と3 or 2 条項BSDの違い

- https://spdx.org/licenses/BSD-4-Clause.html
- https://opensource.org/licenses/BSD-3-Clause
- 4条項 BSDの一つの条項で次の記述がある。
  - 3. All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by the organization
- 上記は利用者の製品に元の開発者の表示を義務付けている。ただし、GPLでは開発者の表示を義務づけておらず、これは GPL が定義する以上の制約を許可しない次の条項に反している。
  - GPLv3:
    - Section 7: If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term.
  - GPLv2:
    - Section 6: You may not impose any further restrictions on the recipients' exercise of the rights granted herein.
- したがって、GPLと互換性を持たせるために4条項 BSDから Section 3を除いた、 3 or 2 条項BSDが作られた。