デジタル化社会におけるオープンソースプラットフォーム

なぜ世界の一流企業はオープンソース活用に熱心なのだろうか

宗像尚郎

ルネサスエレクトロニクス株式会社

オートモーティブソリューション事業本部シニアダイレクタ

2021-1-19

Linux kernel 開発の総本山＝ kernel.org (as of 2021-1-5)

Protocol	Location
HTTP	https://www.kernel.org/pub/
GIT	https://git.kernel.org/
RSYNC	rsync://rsync.kernel.org/pub/

Latest Release
5.10.4

mainline:	**5.11-rc2**	2021-01-03	[tarball]		[patch]	[inc. patch]	[view diff]	[browse]	
stable:	**5.10.4**	2020-12-30	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
stable:	**5.9.16 [EOL]**	2020-12-21	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**5.4.86**	2020-12-30	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**4.19.164**	2020-12-30	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**4.14.213**	2020-12-29	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**4.9.249**	2020-12-29	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
longterm:	**4.4.249**	2020-12-29	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse]	[changelog]
linux-next:	**next-20210104**	2021-01-04						[browse]	

https://www.kernel.org/

kernel 5.10 の経済価値 （sloccount というツールによる算出)

Total Physical Source Lines of Code (SLOC) = 20,365,466
Development Effort Estimate, Person-Years (Person-Months) = 6,689.12 (80,269.45)
(Basic COCOMO model, Person-Months = 2.4 * (KSLOC**1.05))
Schedule Estimate, Years (Months) = 15.22 (182.67)
(Basic COCOMO model, Months = 2.5 * (person-months**0.38))
Estimated Average Number of Developers (Effort/Schedule) = 439.42
Total Estimated Cost to Develop = $ 903,609,303
(average salary = $56,286/year, overhead = 2.40).

SLOCCount, Copyright (C) 2001-2004 David A. Wheeler
SLOCCount is Open Source Software/Free Software, licensed under the GNU GPL.
SLOCCount comes with ABSOLUTELY NO WARRANTY, and you are welcome to
redistribute it under certain conditions as specified by the GNU GPL license;see the documentation
for details. Please credit this data as "generated using David A. Wheeler's 'SLOCCount'."

Linux kernel 5.10 のプロファイル

項目	値
ソースコード行数(コメント等を除く)	20,365,466 (ver5.9 は 20,365,469)
ファイル数(ディレクトリを除く)	63,265個
sloccount によるソフトウエア評価額試算	$903,609,303 (約 932億円)
ver. 5.10 の開発期間 (5.9 → 5.10)	63日 (9週間)
ver. 5.10 の開発に参加した開発者数	1,971人
その内、初めてパッチを投稿した開発者数	252人
加えられた変更数 (=チェンジセット)	16,174 個
ver. 5.10 で追加された価値 (sloccount 差分)	$139（新規追加コード = 削除コードの影響）

企業に所属するプロフェッショナル が Linux kernel を開発している

Most active 5.10 employers

	By changesets			By lines changed	
	Huawei Technologies	1434	8.9%	Intel	96976	12.6%
*	Intel	1297	8.0%	Huawei Technologies	41049	5.3%
	(Unknown)	1075	6.6%	(Unknown)	40948	5.3%
	(None)	954	5.9%	Google	39160	5.1%
	Red Hat	915	5.7%	NXP Semiconductors	35898	4.7%
	Google	848	5.2%	(None)	30998	4.0%
*	AMD	698	4.3%	Red Hat	30467	3.9%
*	Linaro	670	4.1%	Code Aurora Forum	29615	3.8%
	Samsung	570	3.5%	Linaro	29384	3.8%
	IBM	521	3.2%	Facebook	27479	3.6%
*	NXP Semiconductors	439	2.7%	BayLibre	24159	3.1%
	Facebook	422	2.6%	AMD	23343	3.0%
	Oracle	414	2.6%	(Consultant)	19905	2.6%
	SUSE	410	2.5%	IBM	18312	2.4%
	(Consultant)	404	2.5%	MediaTek	15893	2.1%
	Code Aurora Forum	313	1.9%	Arm	13390	1.7%
*	Arm	307	1.9%	Texas Instruments	11814	1.5%
	Renesas Electronics	283	1.7%	SUSE	11063	1.4%
*	NVIDIA	262	1.6%	Oracle	10542	1.4%
*	Texas Instruments	218	1.3%	NVIDIA	10481	1.4%

https://lwn.net/Articles/839772/

＊：組込み系半導体ベンダー
＊：基幹系半導体ベンダー

現代の Linux kernel 開発は
多くの半導体ベンダーにより
支えられている点に注目！

以前はサーバー系ベンダー
（IBM、RedHat、SUSE 等）
だったのから様変わりしている

Linux kernel 開発貢献企業 (2007 年〜2019 年の累計)

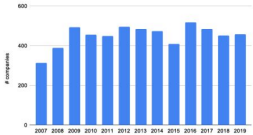
Highly Diversified Corporate Contributors

The Linux Kernel has attracted a wide range of organizations contributing to it over the years. From looking at the commit data from `Git`, we see that we have a very long tail of organizations contributing to the kernel. From 2007 to 2019, there were **780,048** commits accepted into the Linux kernel from **1730** organizations. The top 20 organizations listed below accounted for **68%** of the commits, and there is a long tail of companies that only made a single commit. The top 20 committers are at left.

In these tables and graphs, the label of "(Unknown)" are those patches for which a supporting employer's existence could not be determined. When "None" is used, it indicates the patches from developers known to be working on their own time. This convention is used in the `gitdm` tool[34] that was used to generate these tables and was initially documented in the statistics about the 2.6.20 release[35].

Organization	# of Commits 2007-2019	%
None	93,225	11.95%
Intel	78,068	10.01%
Red Hat	69,443	8.90%
(Unknown)	31,919	4.09%
IBM	29,538	3.79%
SUSE	27,239	3.49%
Linaro	24,740	3.17%
(Consultant)	23,081	2.96%
Google	21,779	2.79%
Samsung	20,160	2.58%
AMD	17,781	2.28%
Renesas Electronics	15,542	1.99%
Texas Instruments	13,855	1.78%
Oracle	13,295	1.70%
Broadcom	9,572	1.23%
Huawei Technologies	9,379	1.20%
Mellanox	9,267	1.19%
NXP Semiconductors	9,223	1.18%
Arm	7,646	0.98%
Linux Foundation	6,109	0.78%

Table 3: Top 20 Committers



Number of Companies Committing to Linux per Year

https://www.linuxfoundation.jp/blog/2020/08/download-the-2020-linux-kernel-history-repor

自己紹介 (Who am I)

OS（Linux kernel）と OSS（= オープンソース） に取り組んできました

- ルネサスエレクトロニクス株式会社
  - 日立、三菱、NEC の半導体事業が統合、グローバルな半導体専業メーカーに発展
  - 自動車向け大規模ソフトウエア基盤（OS、プラットフォーム）開発に従事
  - 社内のオープンソース開発活動（Linux kernel 開発など）を管掌
- オープンソース開発コミュニティ（会社公認の社外活動）
  - The Linux Foundation 理事（Board of Director）
  - AGL (Automotive Grade Linux) プロジェクト、yocto プロジェクトなど
  - LTSI (Long-Trm Sable kernel Initiative) プロジェクト

OS/OSS エンジニアが企業内で評価を得るのは容易ではない（のかもしれない）

半導体と OS/OSS は本来はとても相性が悪いのですが...

では何故いま半導体ベンダーが Linux kernel 開発をリーディングしているの？

- OS = デバイスの特徴（ユニークさ）を隠蔽してしまう仕組み
  - 半導体ベンダーは デバイスのユニークな機能や性能で勝負したいのに...
  - OS の原点は DOS（ドライブメーカーを問わず共通の SW を利用できる）
  - SW のサステナビリティ が重視されるようになった（SW ファーストの誕生）
- OSS = ソフトウエアを実質無料で流通させる仕組み
  - 企業にとってソフトウエアのソースコードは極めて重要な知的財産（= 売り物）
  - OSS ライセンスの順守 は企業にとって大きな負担となる
  - オープンな技術から企業の競争優位性を生み出すことは容易ではない

OSS 活用では大部分の日本企業は世界の最前線からは相当ビハインドだと感じます

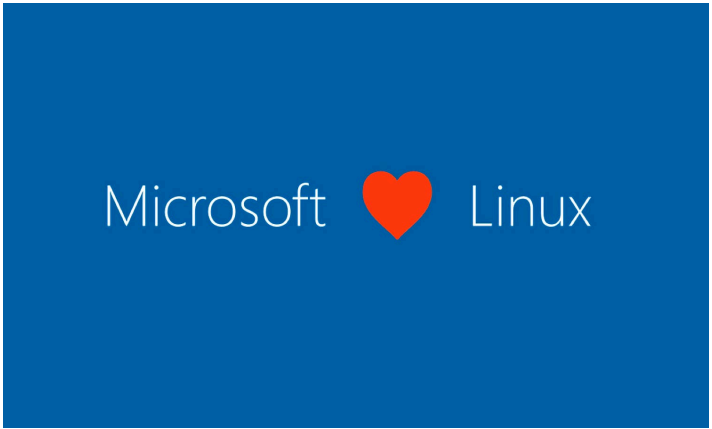
本日の講義でお伝えしたい 3 つのテーマ

営利企業の目的は利潤追求なのにオープンソース開発に投資して儲かるの？

- オープンソースが誕生した経緯 の理解
  - プロプラエタリに対するアンチテーゼ として誕生
  - 繰り返し利用可能な公共財としてのオープンソースソフトウエア
- オープンソース 開発コミュニティの実像 の理解
  - コラボレーション開発はどのように進められるのか、どうすればそこに参加できる？
  - 「オープンソースライセンスの本質」 を理解しよう
- 何故 オープンソースを活用できなければ一流企業とは言えないのか
  - 何故いま 「オープン性」 や 「サステナビリティ」 が重要なのか
  - 企業の壁を越えた連携の舞台となっている コンソーシアム活動

オープンソースの誕生と発展の歴史

マイクロソフトは Linux が大好きだ (by CEO Satya Nadella, 2015)

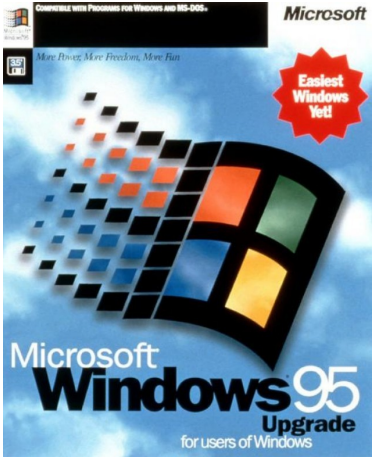


https://www.microsoft.com/ja-jp/cloud-platform/Windows-Server-blog-Loves-Linux.aspx

実はつい最近まで オープンソースの仮想敵は Microsoft だったのです

https://gigazine.net/news/20070520_anti_linux_propaganda/

Netscape Navigator vs. Windows95 inc. Internet Explorer



Microsoft の IE バンドルに対抗？してソースコード公開に踏み切った

Netscape sets source code free

After three months of anticipation, Netscape Communications releases the source code for its Communicator suite.

Janet Kornblum March 31, 1998 12:10 p.m. PT

After three months of anticipation, Netscape Communications today finally released the source code for its Communicator suite.

Netscape CEO Jim Barksdale took this photo in France a month after the company released source code for its Communicator suite.
Alain BUU

Netscape this morning unveiled the much-anticipated release with a teleconference featuring breathy executive statements touting the significance of the move. The company actually posted the approximately 8 megabytes of compressed Communicator 5.0 code at 10 a.m. PT to Mozilla.org, the site Netscape has set up to be the central clearinghouse for source code-related information.

https://www.cnet.com/news/netscape-sets-source-code-free/

Netscape のコード公開が (anti MS としての) OSS 活性化を加速

OSI が発表したOSD (Open Source Definition) は現在でも OSS を名乗る条件である

- 1. 再頒布の自由
- 2. ソースコード (を入手できること)
- 3. 派生ソフトウェア（を作れて、同じライセンスを付与すること）
- 4. 作者のソースコードの完全性 (integrity)（が担保されること）
- 5. 個人やグループに対する差別の禁止
- 6. 利用する分野 (fields of endeavor) に対する差別の禁止
- 7. ライセンスの分配 (distribution)（に際し追加ライセンスが不要）
- 8. 特定製品でのみ有効なライセンスの禁止
- 9. 他のソフトウエアを制限するライセンスの禁止
- 10. ライセンスは技術中立的でなければならない
- http://www.opensource.jp/osd/osd-japanese.html

Richard Stallman と FSF (自由ソフトウエア運動)

1983 年の Stallman に何がおこったのか

- 当時 Stallman は MIT のロボット研究所に所属
- FBI がハッカー集団 414 を逮捕、社会の脅威とされた
- http://enterprisezine.jp/iti/detail/4917

- ACM (Association for Computing Machinery) が アンチハッカー宣言、 一連の記事を掲載
- Stallman はハッカーの本質は違うと主張 するも 著作権自体を否定していると誤解を受ける

- ACM 学会誌上で議論となり、MIT ラボを辞任
- 1985 年 3 月に GNU 宣言を発表

http://ergoemacs.org/emacs/i/Richard_Stallman_at_MIT_dancing_1970s.jpg

Stallman は 社会貢献を企図してフリーソフトウエア を提起した

フリーソフトウエア（= 自由ソフトウエア）とは以下の 4 つの自由 が担保されたもの

- どんな目的に対しても、プログラムを望むままに実行する自由 (第零の自由)。
- プログラムがどのように動作しているか研究し、必要に応じて改造する自由 (第一の自由)。ソースコードへのアクセスは、この前提条件となります。
- ほかの人を助けられるよう、コピーを再配布する自由 (第二の自由)。
- 改変した版を他に配布する自由 (第三の自由)。これにより、変更がコミュニティ全体にとって利益となる機会を提供 できます。ソースコードへのアクセスは、この前提条件となります。
- https://www.gnu.org/philosophy/free-sw.ja.html

GNU Project Origin (GNU Manufesto)

GNU宣言

下記のGNU宣言はリチャード・ストールマンによって、1985年にGNUオペレーティング・システムの開発に支持を求めて、書かれたものです。1987年までは、開発を説明するのに少々更新されましたが、それからは、変更なしのままで置いておくのがよいでしょう。

その時から、よくある誤解について、そしてそれが異なる言い回しで避けられることを、わたしたちは学びました。1993年から脚注を付け足し、この点を明確にしました。

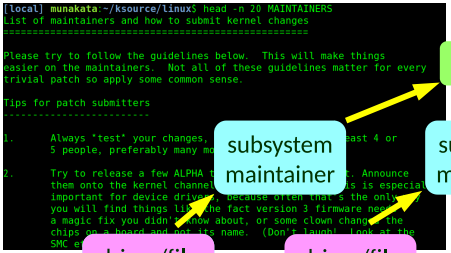
GNU/Linuxシステムをインストールしたい方には、わたしたちは100%自由ソフトウェアのGNU/Linuxディストリビューションのひとつを使うことをを推奨します。どのように貢献するかについては、http://www.gnu.org/helpをご覧ください。

GNUプロジェクトはソフトウェアのユーザの自由のキャンペーンである自由ソフトウェア運動の一部です。GNUを「オープンソース」の用語と関連付けるのは間違いです。この用語は1998に自由ソフトウェア運動の倫理的な価値に同意しない人々によって思いつかれたものです。かれらは同じフィールドで倫理とは無関係のアプローチを推進するのにこの用語を使っています。

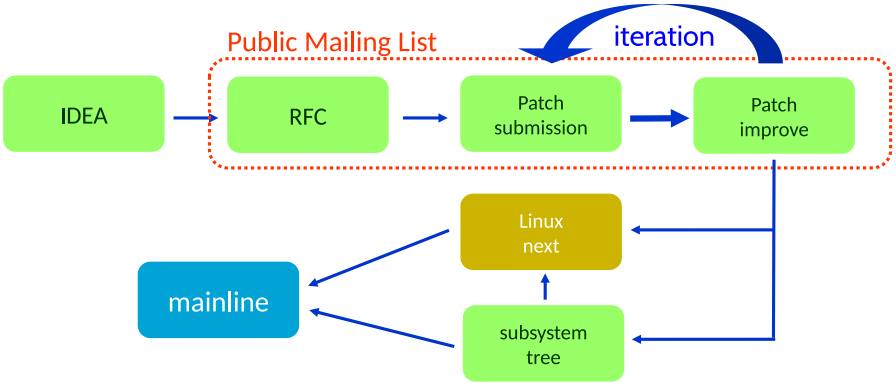
https://www.gnu.org/gnu/manifesto.ja.html

オープンソース開発コミュニティの実像

OSS 開発プロセス（1）：階層化された開発コミュニティの構造

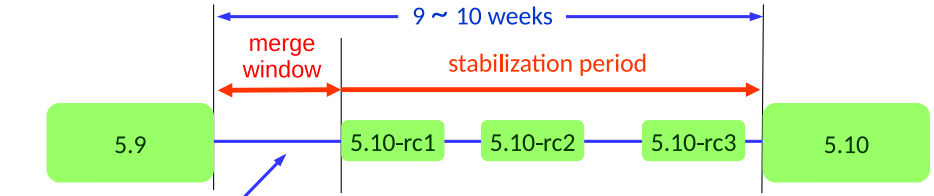


OSS 開発プロセス（2）：公開 ML を使ったオープンな議論



公開のML 上で新提案をオープンに 検討→改善 を重ねる（イタレーション）、
どんな経験者でも数回の書き直しが要求される（顔パスは無い、皆で推敲して良いコードにする）

OSS 開発プロセス（3）：パッチのレビューとマージ



- Linux-next の段階で毎日 kernel build を実施してマージで問題が出ないことを確認しておく
- 一つ前のカーネルがリリースされると、Linux-next のコードを取り込む2週間のマージウィンドウが開く
- RC1 リリース後は新規マージを停止。安定化のためのシステム検証を行う。 通常は RC8 ~ 9 までいく

パッチレビューに合格して Linux-next に登録されてからマージが完了するまで 最大で19週 かかる

OSS のクオリティの源泉は 徹底したコードのピアレビュー（査読）

周期的なバージョン更新を確約することによって 締め切り間際での無茶を徹底排除

- オープンソースプロジェクトではコミュニティ開発者から投稿されたパッチを集約してプログラムを構築するので 低品質なコードが混入しないよう 入り口での制御 (コードレビュー) が重要な砦となる。
  - 組織的なコードレビューの仕組み (Gerrit など) があるか
  - メンテナーはタイムリーにパッチをレビューしているか、
  - レビュー内容に”透明性”や”中立性”があるか
  - メンテナーの技術バックうラウンド、スキルセットは適切か
  - 幅広く関連する別プロジェクトの開発者もレビューに参加しているか
- パッチレビューでは ベテラン開発者でも最低 3 回程度は書き直す のが普通

各界の識者による徹底したピアレビューによってサスティナブルなコードを生み出す

SW メンテナンス (バグレポート、対応状況管理、パッチリリース)

完全無欠なソフトウエアは無い、永遠にメンテナンスを続ける覚悟が必要

- プロジェクトにプログラムの 品質保全体制 (バグ報告 → 対策 → 来歴管理) が適切に運用できているかを確認する必要がある。実際の運用状況はまちまちである。
  - 公開バグ管理システム (=BTS) にバグ報告が集められているか
  - プロジェクト内でバグの原因究明と対策パッチ作成が行われているか
  - バグ修正パッチがタイムリーにリリースされているか
  - バージョンアップ時に過去バグの 累積的なリグレッションテスト の実施
- ローカルにバグ修正をおこなった場合、その内容をマスターコードにも反映しないと品質があがらない (= 将来同じ問題が再発する可能性がある)
- 組織的なバグ管理 (報告、履歴管理、リグレッションテスト等) 体制が必須

(会社にはとても嫌われるだろうが) SW 開発はエンドレスゲームと考える必要がある

ソフトウエアの開発は、いわばガーデニング（by James Kuffner）

トヨタは無駄のないシステムをデザインするのが本当に得意で、トヨタの核となる原則の一つ。これもまた重要なことです。しかしこれにのっとれば、必要最低限 の メモリーと計算能力を載せるのが良い、ということになります。TRI-AD の初期の頃の会話は、まさにそうでした。ソフトウェアを開発している我々が「こういうスペックのものを作りたい」と提案をするとハードウェア側のトヨタは「いや、これはトヨタの原理に完全に反している。 なぜメモリーや計算能力を余分に載っけるのか？ それはムダだ」と。でも本当はこれは「ムダ」 ではなく「バッファー」なんです。それこそが車の期待値になるし、将来アップグレードするために必要な 「余裕」 とも言えます。

ソフトウェアの開発は、いわばガーデニングのようなものです。 雑草を抜き続けたり、水や日光を与えたりと常にメンテナンスが必要です。　これは、トヨタがこれまでに培ってきた常により良い方法を模索するという「カイゼン」に通じるものです。私は、トヨタのハードウェアがそうであるように、ソフトウェアでも強いトヨタ生産方式を確立したいと思っています。

https://newspicks.com/news/5114410/body/

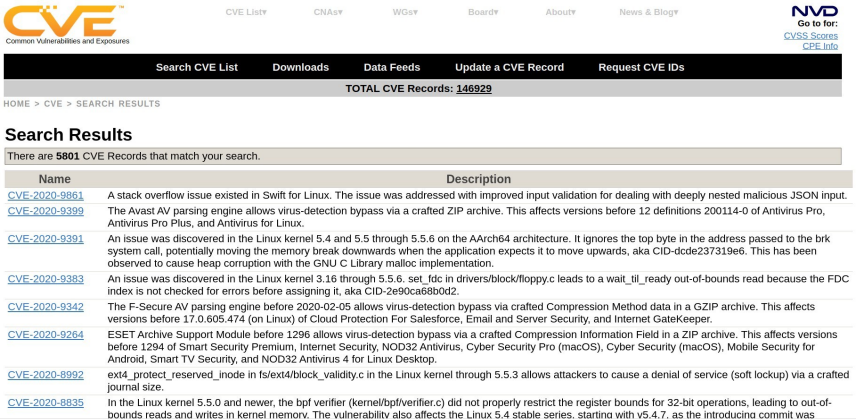
サイバーセキュリティ等の脆弱性への対応

サイバーセキュリティ対応は SW 提供者の社会的責任 となってきている

- 新たに発見された ソフトウエア脆弱性 (＝セキュリティホール) に対して、タイムリーに対策パッチを提供するための組織的な取り組みが必要
- ソフトウエア脆弱性開示 (CVE) とのリンクした活動が行われているか
- リスクが公開されてから対策パッチが出るまでの時間が重要 (ゼロディ攻撃)
- 広く利用されているソフトウエアであってもコミュニティでの開発が既に終了していては脆弱性に対処できないケースもあった（OpenSSL など）
- サイバーセキュリティリスクなど重大なソフトウエア脆弱性が公開された時には 速やかに対策パッチを提供 することが求められる

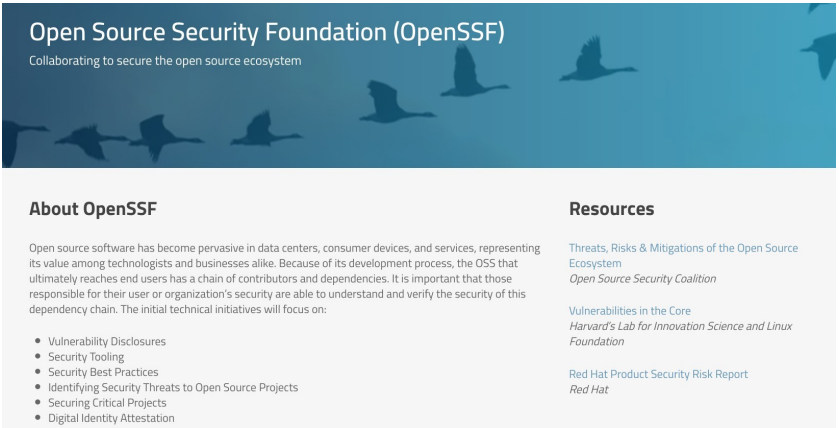
多くの開発コミュニティが脆弱性対策に真摯に取り組んでいるが、支援も必要である

Common Vulnerabilities Exposures (CVE)＝脆弱性情報公開



https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=Linux

OSS のサーバーセキュリティ対応を推進するコンソーシアム活動



https://openssf.org

オープンソースには動作保証はない (GPL v2 Section 11)

- 『プログラム』は代価無しに利用が許可されるので、適切な法が認める限りにおいて、『プログラム』に関するいかなる保証も存在しない。書面で別に述べる場合を除いて、著作権者、またはその他の団体は、『プログラム』を 表明されたか言外にかは問わず、商業的適性を保証するほのめかしやある特定の目的への適合性 (に限られない) を含む一切の保証無しに「あるがまま」で提供する。『プログラム』の質と性能に関するリスクのすべてはあなたに帰属する。『プログラム』に欠陥があると判明した場合、あなたは必要な保守点検や補修、修正に要するコストのすべてを引き受けることになる。
- http://www.opensource.jp/gpl/gpl.ja.html

オープンソースを使っていても製品としての品質を担保するのは機器メーカの責任

プロフェッショナルとして OSS を活用していくには

OSS ベースだから... という言い訳は一切通用しないので

- 決められた期限 (納期) は絶対に守らなければならない
  - ひとくちにオープンソースといってもコード品質やメンテナンス状況はさまざま
  - 外から持ってきた OSS 故に 想定外の問題が発生 して開発がつまずくリスクもある
  - オープンソースでは アジャイル型開発プロセス を採用しているケースが多い
  - アジャイル型開発では機能単位で検証を積み上げていくアプローチが重要
- 開発規模 (= コードサイズ) が桁違いに大きい
  - チームで分担して並行分散開発するのが普通
  - 商品として品質保証があることが当然期待される
  - ユーザー毎の色々な使われ方 (= ユースケース) を開発段階から考慮する必要がある
  - が、なにひとつバグが残っていない完全無欠なソフトウエア を目指すのは非現実的
  - むしろ ソフトウエアの更新が提供されることを保証する 方が建設的だろう

製品出荷後のメンテナンス（ソフトウエアアップデート）

4G/5G の普及によって あらゆる機器の SW のアップデート が求められる時代に

- ユーザーは 不具合対策や機能アップのためのアップデート を期待している
- 重要なサイバーセキュリティ対策 のためのソフトウエア更新は必須
  - 将来の ソフトウエアバージョンアップを想定しておく 必要がある
  - 重大なセキュリティホール が見つかったケースには対応が必要
- オリジナルの開発者でなくてもソフトを変更できるように
  - 誰でも容易に修正できるような コードの可読性 の確保
  - 操作マニュアル以外に 開発者向けのドキュメント が必要
  - 履歴 (変更理由、変更思想、検証内容など) を残すこと
- ソフトウエアを配布し、更新する仕組みも考えておく必要がある
  - 通信機能があるものは オンライン更新 (OTA=Over The Air) の仕組み
  - そうでない場合には SD カードや USB 接続を使ったソフトウエア更新 が必要

企業が OSS を製品開発などに利用 する時には心配もある (1)

企業が OSS を利用する時の 技術面の懸念

- OSS には 技術ロードマップ が提示されていない
  - 何時頃どんな機能がサポートされていくのか予測できない
- ソースコードに全て書かれていると言われても....
  - 簡単に読める敷居の低い ドキュメントがほとんど無い
  - 実際には英語ならとても参考になる技術紹介記事 (無料で読める) が沢山ある
  - https://kernelnewbies.org/
  - https://lwn.net/Kernel/
  - 英語の勉強と最先端技術の習得の一石二鳥に是非チャレンジ してほしい
- 頻繁なバージョンアップについていけない
  - 新しいソフトは品質的に枯れていないのではないか？
  - リリース時に 誰が何を検証したのか トレースできない

企業が OSS を製品開発などに利用 する時には心配もある (2)

企業が OSS を利用する時の 管理面の懸念

- 商品開発に使った場合、有償開発サポートの仕組み があるか？
  - 開発コミュニティは 無保証、ASIS 利用 を前提にコードを配布している
  - 開発コミュニティは 原則 Best Effort ベースのサポート 対応しかできない
- 品質責任の所在が不明確
  - 誰が品質を担保しているの？　問題が発生したら誰に相談すれば良いの？
- 商品出荷後に セキュリティ対策パッチ が必要になったらどうする？
  - 対策パッチは何処から入手すればよい?
  - そもそも 脆弱性の情報は何処で見つける ことができる?
- いろいろな団体から OSS ディストリビューションが出ているけど...
  - Ubuntu, Fedora, Debian,.. どれを選択すればよいのか？
- 結果的に OSS を使った方が むしろ費用が高く付く のでは？

企業エンジニアを OSS 開発コミュニティに参加 させる時の障害は？

以前とくらべると 経営者のオープンソースへの理解は大幅に改善 しているが

- 開発中の 自社技術 (知的財産）の社外流出 になのではないか?
  - 社内で競争領域と共創領域の明確な線引き をしてもらう必要がある
  - 共創領域の技術開発では、開発成果を活用できるような方向付けが重要
- ビジネス上の コンペチターと共同開発 するってどういう事？
  - コンプライアンス遵守 は重要だが (不正競争防止法 = アンチトラスト法)
  - 実際には 同じ悩みを抱える開発者の集まりなので” 非常に居心地が良い場” である
  - コミュニティで開発した OSS を商品に利用することに懸念 を持たれる可能性も
- 会社の 技術発表審査にとても長い時間 がかかる
  - それでは ML に来たフィードバックに返事が書けない
  - 契約によって情報の開示先が限定されてしまうケースもある

White Paper 『技術的負債とオープンソース開発』

アップストリーム開発の役割

最終目標がアップストリームのブランチに貢献することであるかぎり、ブランチアウトまたはフォークして開発を進めることになるでしょう。図 2 は、このプロセスを示しています。

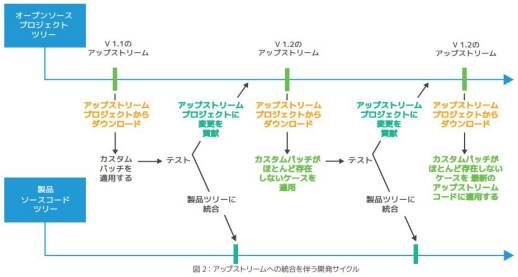


図 2：アップストリームへの統合を伴う開発サイクル

- **継続的インテグレーションと継続的デリバリー / デプロイメント:** 開発者は機能をより迅速に利用できます。新しいコードが開発者ツリーに迅速に取り込まれます。各コード コミットには、より高いレベルの品質でテストが必要で、リグレッションやバグの発見がより容易になります。

https://www.linuxfoundation.jp/blog/2020/07/solving-technical-debt-with-open-source/

開発者会議などに参加して他の企業の同志とつながろう（1）

https://events.linuxfoundation.org/open-source-summit-japan/

開発者会議などに参加して他の企業の同志とつながろう（2）

The Linux Foundation CE Workgroup
Japan Technical Jamboree

Date: June 26th / 日付: 6月26日（金）

- ONLINE (WebEx)
- Please join "celinux-dev" mailinglist to get the latest announcement.
- "celinux-dev" メーリングリストにこのイベント関係のアナウンスが流れます。ぜひ参加してください。
- How to join the mailing list.

We are calling for session proposal now! / セッション提案募集中！

Session proposal how-to. / 提案の方法

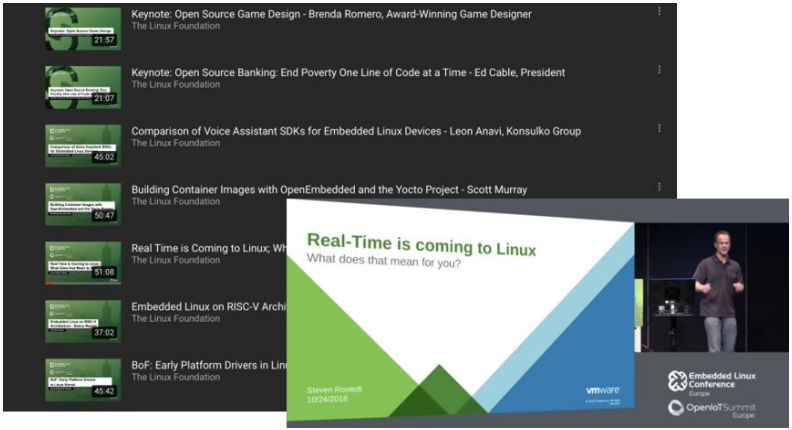
Contents [show]

Agenda / WebEx Remote Presentations

Time (Japan)	Title and presenter	Notes Presentation Materials
9:00am..	**Status of Embedded Linux** • **Tim Bird** • Linux Kernel, technology, conference and industry Update (in slides)	• PDF_English • In English / 英語のセッションです • 🇯🇵 🇺🇸 (<put youtube link here> Video)
10:00am..	**Talk about Maintain In-house Embedded Linux Distro on 17 years** • **Takuma Ueba, Fujitsu**	• PDF_Japanese
11:15am..	**some topics** • **Teppei Asaba, Fujitsu**	
11:30am..	**インナーソースチェックリストの和訳 / Japanese translation of the Inner Source Checklist** • **Hirotaka Motai, Mitsubishi Electric**	• PDF-slide_Japanese • PDF-doc_Japanese • In Japanese / 日本語のセッションです
11:50am..	**<<DEAD LIMIT>> Venue booked until 9:00 pm.**	

https://elinux.org/Japan_Technical_Jamboree_73

開発者会議などに参加して他の企業の同志とつながろう（3）



https://www.youtube.com/playlist?list=PLbzoR-pLrL6qThA7SAbhVfuMbjZsJX1CY

コンピュータプログラムは著作物 であり複製や改造には制限がある

著作権 (著作財産権、著作者人格権) の適用対象となるもの

- 言語 (論文、小説、脚本、詩歌、俳句、講演など)
- 音楽 (楽曲及び楽曲を伴う歌詞)
- 美術 (絵画、版画、彫刻、漫画、書、舞台装置など)
- 建築（芸術的な建造物（設計図は図形の著作物）
- 地図や図形 (地図と学術的な図面、図表、模型など)
- 映画 (劇場用／テレビ、ビデオソフト、ゲームソフト)
- 写真、グラビアなど
- コンピュータ・プログラム

著作権（コピーライト）は明らかにオープンソース活動とは相いれない権利を主張する

著作財産権（= 狭義の著作権）: 対価の請求権 で第三者への譲渡も可能

著作権（財産権）

複製権	著作物を印刷、写真、複写、録音、録画などの方法によって有形的に再製する権利
上演権・演奏権	著作物を公に上演したり、演奏したりする(上演、演奏の録音物を再生することを含む)権利
上映権	著作物を公に上映する権利
公衆送信権・公の伝達権	著作物を自動公衆送信したり、放送したり、有線放送したり、また、それらの公衆送信された著作物を受信装置を使って公に伝達する権利 ＊自動公衆送信とは、サーバーなどに蓄積された情報を公衆からのアクセスに応じ自動的に送信することをいう。また、そのサーバーに蓄積された段階を送信可能化という。
口述権	言語の著作物を朗読などの方法により口頭で公に伝える(口述の録音物を再生することを含む)権利
展示権	美術の著作物と未発行の写真の著作物の原作品を公に展示する権利
頒布権	映画の著作物の複製物を頒布（販売・貸与など）する権利
譲渡権	映画以外の著作物の原作品又は複製物を公衆へ譲渡する権利
貸与権	映画以外の著作物の複製物を公衆へ貸与する権利
翻訳権・翻案権など	著作物を翻訳、編曲、変形、翻案等する権利（二次的著作物を創作する権利）
二次的著作物の利用権	自分の著作物を原作品とする二次的著作物を利用（上記の各権利に係る行為）することについて、二次的著作物の著作権者が持つものと同じ権利

参考条文…著作権法第21条〜第28条

https://www.cric.or.jp/qa/hajime/hajime2.html

著作者人格権 = 著作者の心情（モラール）を保護する権利

著作者人格権は 第三者によるコードの改変や再配布を制約 する

- 公表権（著作権法 18 条 1 項）
  - 無断で公表されない権利、すなわち未だ公表されていない自分の著作物について、公表するかどうか、いつ、どういう方法及び条件で公表するかを決定する権利
- 氏名表示権（著作権法 19 条 1 項）
  - 自分の著作物を公表する際に、著作者名を表示するかどうか、どのように表示するか（実名で表示するのか、ペンネームなどの変名で表示するのか）を決定できる権利
- 同一性保持権（著作権法 20 条 1 項）
  - 自分の 著作物の内容、題号を著作者の意に反して無断で改変させない権利 です。

著作者人格権は財産権と異なり、第三者に譲渡したり相続したりはできない

著作物の 私的利用では著作権が免除 されているケースも多いが....

個人利用ではインストール時に 内容をよく確かめずに Yes を押していませんか？

- ネットで使えそうな OSS が無いか探してみる
- 使えそうな OSS のインストーラが見つかった。早速ダウンロード
- インストール中にライセンス同意確認ダイアログが出たが...
- ライセンスの中身は読まず”Yes”を押してインストールを進める
- インストールや使い方はネットで収集、サポートは必要なし
- もし途中でエラーに当たったらエラーメッセージをネットで検索
- それでもダメだったら別のソフトを選べばよい

製品開発で同じことをしたら取り返しがつかない大損害（訴訟、販売取りやめ等）も

OSS 利用時には 使用許諾条件（ライセンス）の順守 が求められる

コピーレフト型ライセンス

- 代表は GNU GPL/LGPL
- （共通）改変、再配布、利用を許諾
- ソースコードの開示義務あり
- Linux kernel など
- OSS コミュニティの活性化に寄与
- GPL は他のライセンスと結合できない

パーミッシブ型ライセンス

- 代表は MIT、Apache2
- （共通）改変、再配布、利用を許諾
- 改変したコードの開示を義務づけない
- Android、AI 関連ツールなど
- OSS の商用製品への適用を促進
- 一方で分岐（fork）が発生しやすい

Open Source Initiative (=OSI) が各種のオープンソースライセンスを認定している

オープンソースライセンス （OSI が認定した代表的なライセンス）

Open Source Licenses by Category

License Index

- License Approval Process
- License Information
- Origins and definitions of categories from the License Proliferation Committee report

In the lists below, a parenthesized expression following a license name is its SPDX short identifier, if one exists, except for two items in the first list (GNU General Public License and GNU Lesser General Public License). For these, the parenthesized expressions ("GPL" and "LGPL" respectively) are the common non-version-specific names of these licenses today (note also that the full name of the first version (2.0) of the LGPL is the GNU Library General Public License). There is no non-version-specific SPDX short identifier for the GPL and LGPL.

open source initiative
Approved License

Licenses that are "popular and widely-used or with strong communities"

The below list is based on publicly available statistics obtained at the time of the Report of License Proliferation Committee.

- Apache License 2.0 (Apache-2.0)
- 3-clause BSD license (BSD-3-Clause)
- 2-clause BSD license (BSD-2-Clause)
- GNU General Public License (GPL)
- GNU Lesser General Public License (LGPL)
- MIT license (MIT)
- Mozilla Public License 2.0 (MPL-2.0)
- Common Development and Distribution License 1.0 (CDDL-1.0)
- Eclipse Public License 2.0 (EPL-2.0)

https://opensource.org/licenses/category

企業の 製品開発に OSS を利用 するときのライセンスの確認

開発対象の 利用シーンを想定 して適切なライセンスを付与するべきである

- (大原則) 既存の OSS を改変した場合はオリジナルのライセンスを変更できない
- オープンソースのライセンス条件が自分の 製品要件と整合性がとれるか 確認
- Apache2.0 などの パーミッシブ型ライセンス が採用される例も増えている
- 以下は注意が必要なライセンス
  - GPLv3 ライセンス (特に インストール条項)
  - Apache 2.0 ライセンス (特に 特許許諾条項)
  - デュアルライセンス採用時のライセンス移行性
  - 商用アプリケーションとの組合せ (ffmpeg の取り扱いなど)
- オープンソースライセンスに規定されたソースコードの取り扱いを理解する必要がある (例えば アップデートバイナリー配布時のソースコードの扱い 等)

世界のトップ企業とオープンソースの関わり

Instagram は 最先端のデジタル技術の組合せ で実現されている

Instagram が採用している最先端のデジタル技術群

- オープンソースとして開発されているテクノロジ
  - Python (プログラミング言語), React (JavaScript フレームワーク)
  - Nginx (Web サーバー), django (Web アプリケーションフレームワーク)
  - gunicorn (Python WSGI HTTP サーバー), Solr (全文検索システム)
  - PostgresSQL (データーベース), redis (インメモリ型キーバリュー型データストア)
  - Gearman (ジョブキューサーバ), Ubuntu (Linux)
- パブリックなインフラストラクチャ
  - Amazon AWS
  - Amazon Route53、Amazon cloudfront

13 人のデジタル技術の目利きが短期間で画像共有のシステムを構築

2012 年 4 月に Facebook が Instagram を 1B ドル（=10 億円）で買収

April 9, 2012

Facebook to Acquire Instagram

MENLO PARK, CALIF.—April 9, 2012—Facebook announced today that it has reached an agreement to acquire Instagram, a fun, popular photo-sharing app for mobile devices.

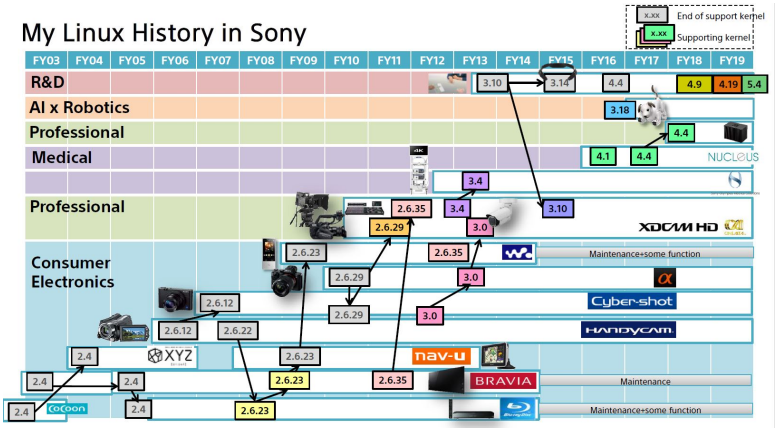
The total consideration for San Francisco-based Instagram is approximately $1 billion in a combination of cash and shares of Facebook. The transaction, which is subject to customary closing conditions, is expected to close later this quarter.

Mark Zuckerberg, founder and CEO of Facebook, posted about the transaction on his Timeline:

> I'm excited to share the news that we've agreed to acquire Instagram and that their talented team will be joining Facebook.
>
> For years, we've focused on building the best experience for sharing photos with your friends and family. Now, we'll be able to work even more closely with the Instagram team to also offer the best experiences for sharing beautiful mobile photos with people based on your interests.

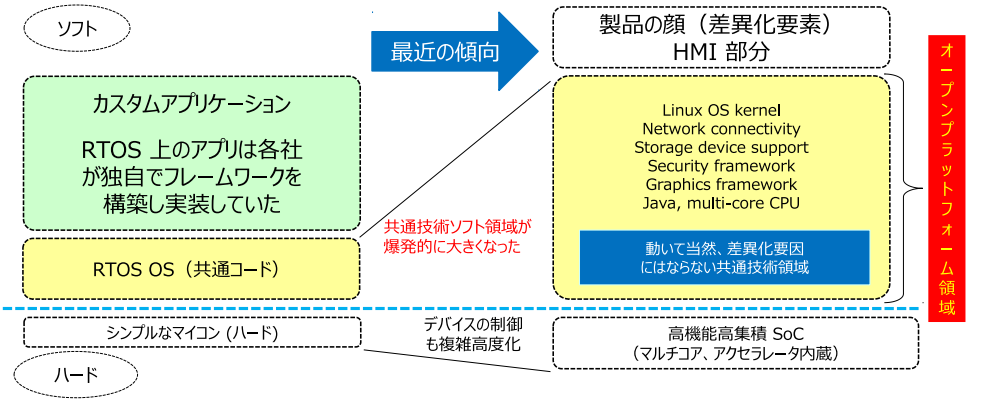
https://newsroom.fb.com/news/2012/04/facebook-to-acquire-instagram/

OSS は 商品開発の現場で幅広く採用 されている（SONY の例）



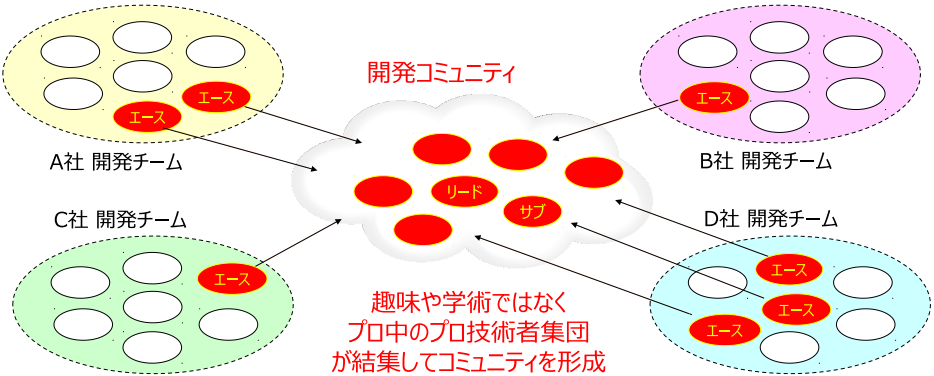
https://elinux.org/images/3/3e/%E3%82%BD%E3%83%8B%E3%83%BC%E3%81%A7%E3%81%AELinux%E3%81%A

商品差異化につながらない 非競争領域のソフトウエア比率 が急拡大



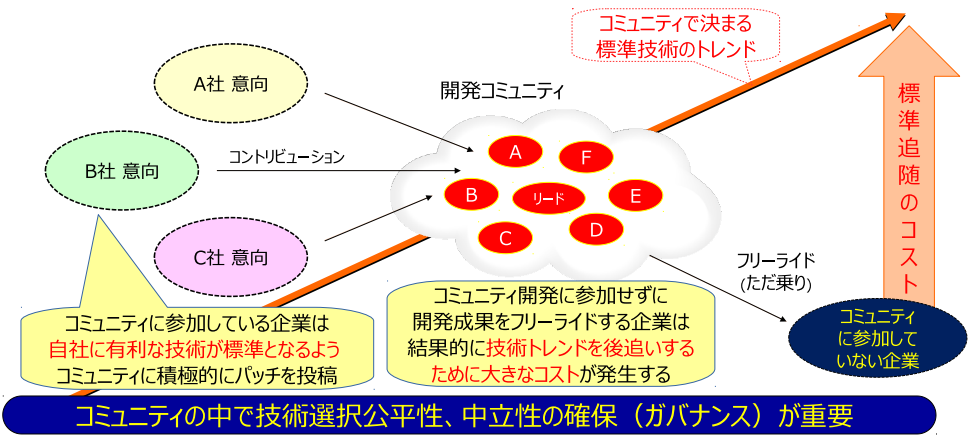
ソフトウエア規模が巨大化、もはや単独企業ですべて作るのは現実的でなくなった

先行企業は開発コミュニティに自社のエースを送りこんでいる

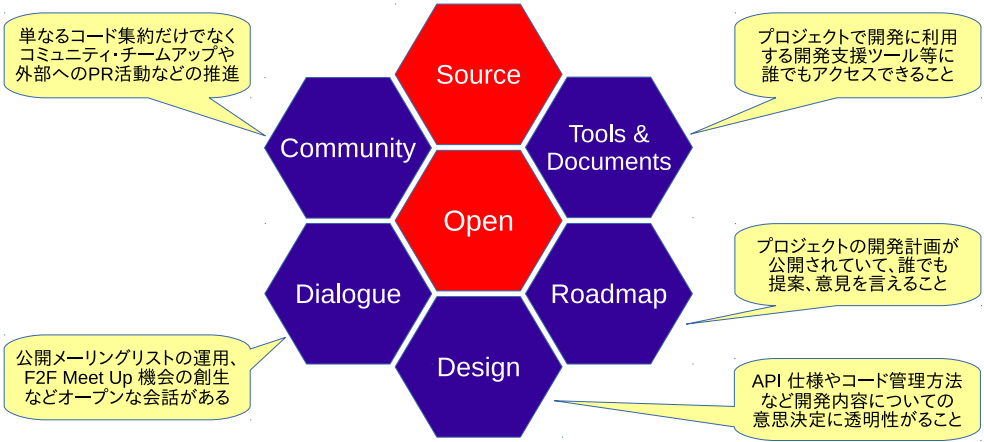


各社がエース級開発者をコミュニティに送り込んで共同で共通技術の開発を推進

実は Contribution の本質 = 陣取り合戦 でもある



ソースコードの公開だけではコミュニティに影響力を行使できない



まとめ (今日 お伝え/お願い したかったこと)

- 本来著作権の保護対象であるコンピュータープログラムについて ソースコードを入手し自由に改変し再配布するオープンソース の確立には、これまで多くの努力があったことを理解しよう。　これは当たり前に出来たことではない。
- オープンソースは 企業や国境といった境界を越えたグローバルに開かれた民主的なコミュニティ で開発されている。　オープンソースの利用だけでなくみなさん自身が開発に参加する扉は常に開かれているのである。
- デジタル社会を支える電子機器やインフラはオープンソース無しには成立しない。このため、世界の先進企業の多くがオープンソースに対し極めて積極的な取り組みを行っており 皆さんも将来そのような企業人として活躍できる可能性がある。