FUJITSU
shaping tomorrow with you

OSSライセンスを理解するための Linuxが動く仕組み

2021.2.8

富士通コンピュータテクノロジーズ
浅羽 鉄平

1591ka1

$ whoami

FUJITSU

- 名前：浅羽 鉄平 (あさば てっぺい)
- 所属：富士通コンピュータテクノロジーズ
- 仕事：組込みシステム向けLinuxディストリビューションのメンテナンス。最近はLinuxであれば、「組込み」と付けなくてもよいのではと思う今日この頃。
- 好きなライセンス：GPL-3.0

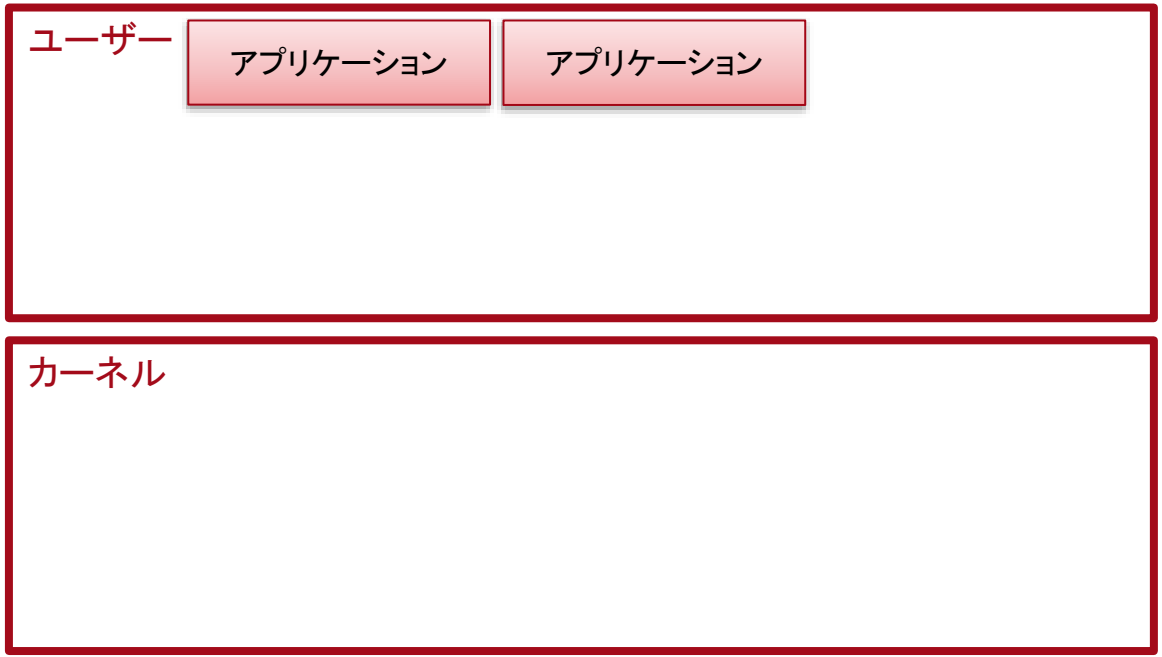
FUJITSU

Linuxのシステム構成

よく見るLinuxのシステム構成

今日お話しするライセンスの伝播を理解するためには、

よく見るLinuxのシステム構成



これだけ意識すればOK

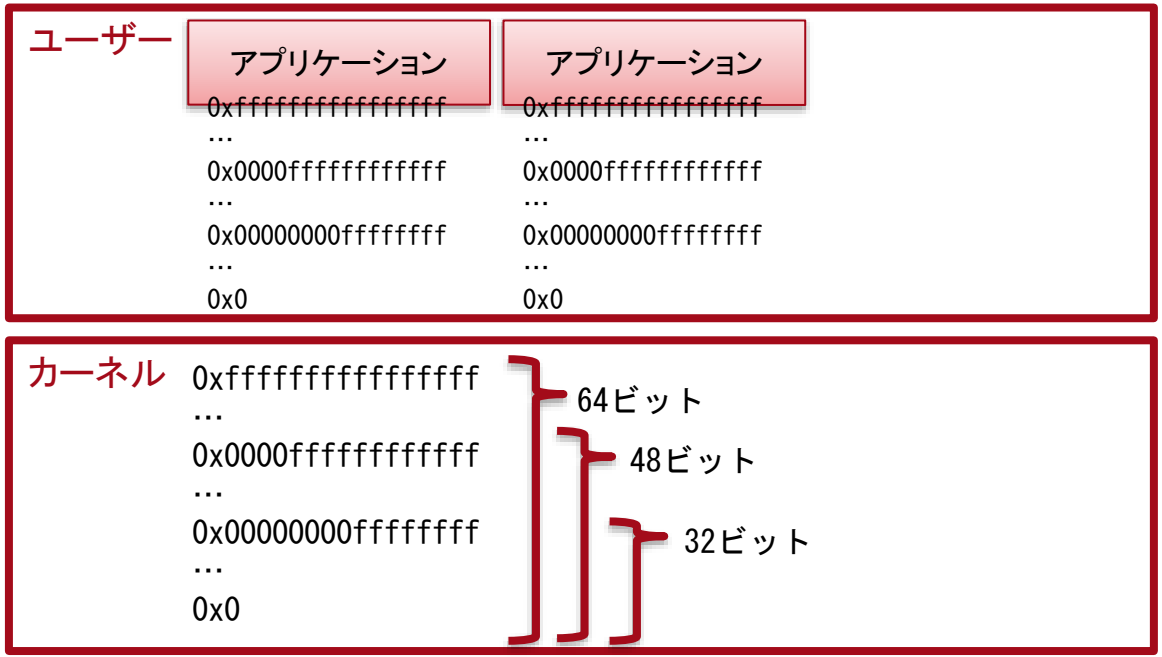
つまり、

FUJITSU

- Linuxシステムは、
  - カーネル
  - ユーザー空間のアプリケーション達

  で構成されています。

どういうことかというと、、



カーネルとアプリケーションの数だけ空間 (アドレス、番地)があるということ

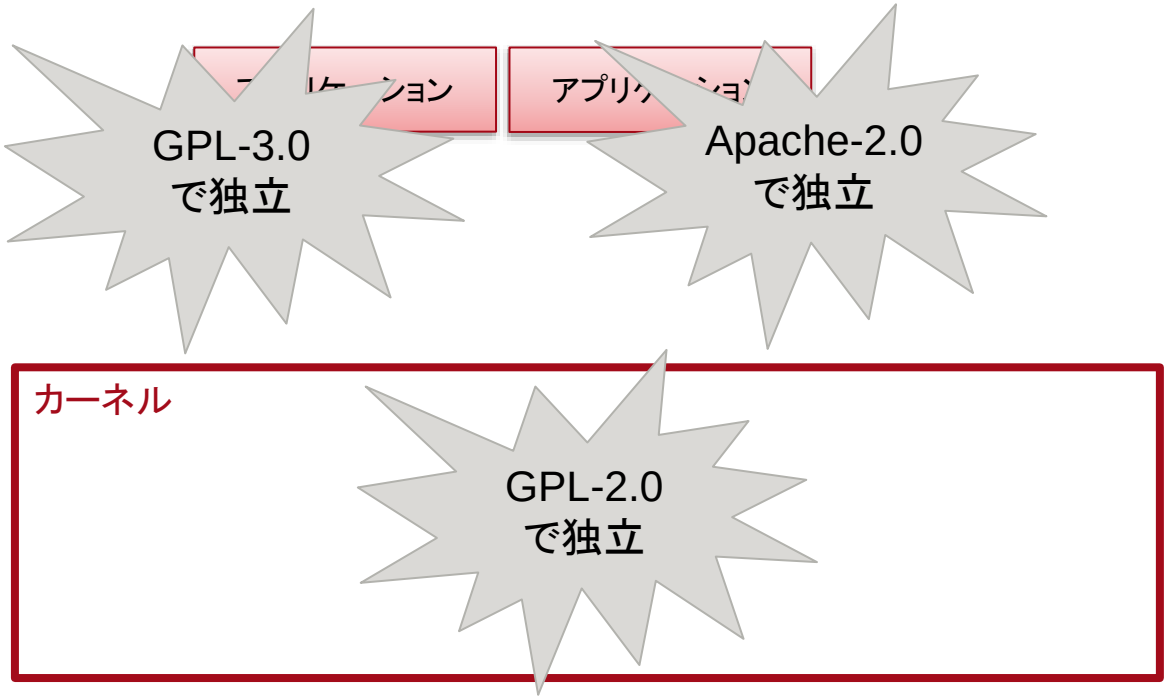
0xfff…は理解できなくてもOK

FUJITSU

- Linuxシステムは、
  - カーネル
  - ユーザー空間のアプリケーション達

  でそれぞれ空間があり、独立していると覚えてください

つまり、



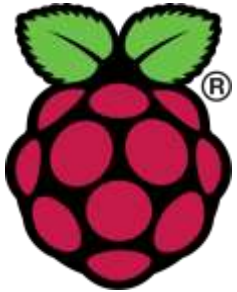
こんなイメージです

FUJITSU

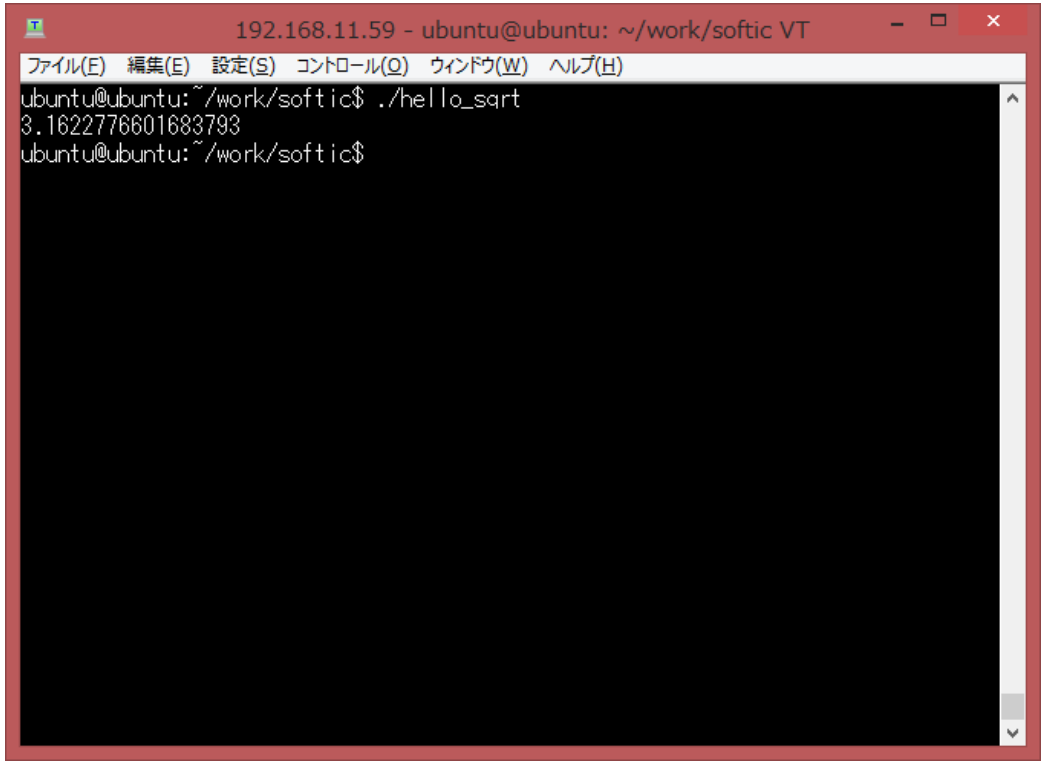
Linux上でアプリケーションを動かしてみよう

デモ

- 実行環境：
  Raspberry Pi 4



Linux (Ubuntu)上で
アプリケーション (√10を計算する)を実行



√10を計算するプログラム

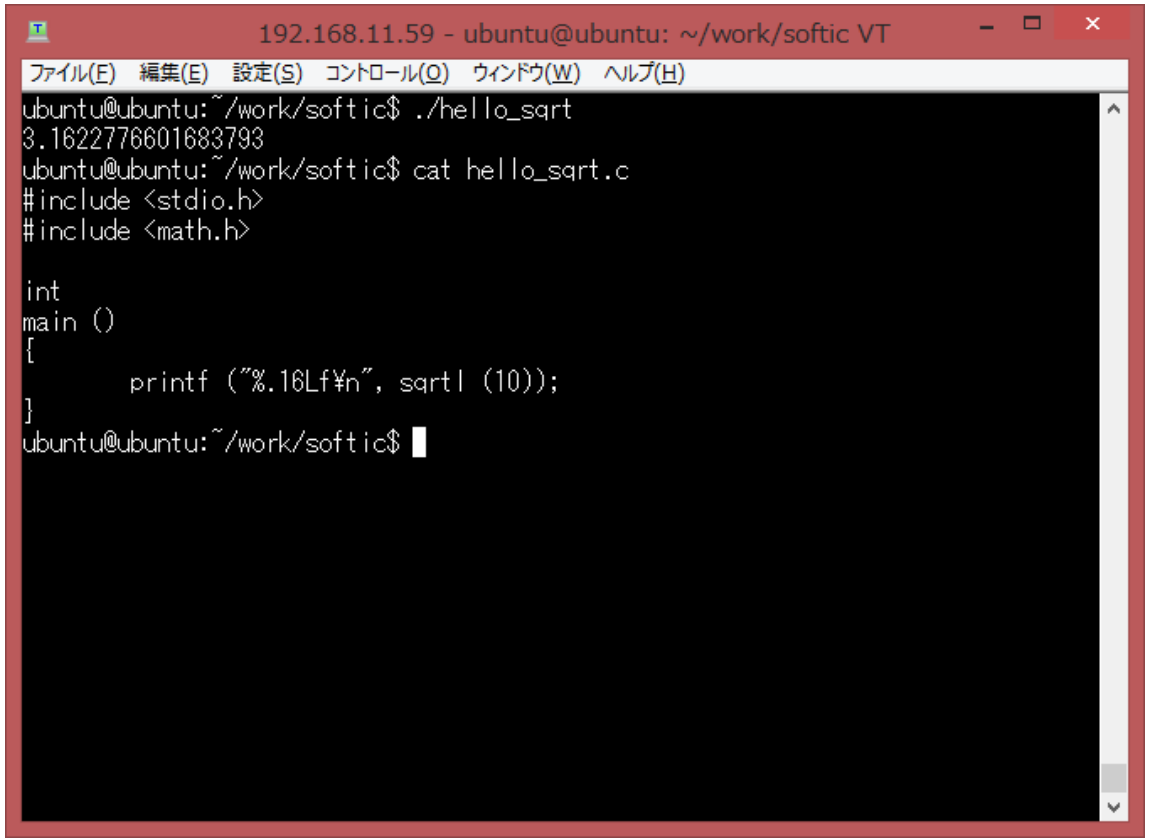
https://github.com/bminor/glibc/blob/master/sysdeps/ieee754/dbl-64/e_sqrt.c

FUJITSU

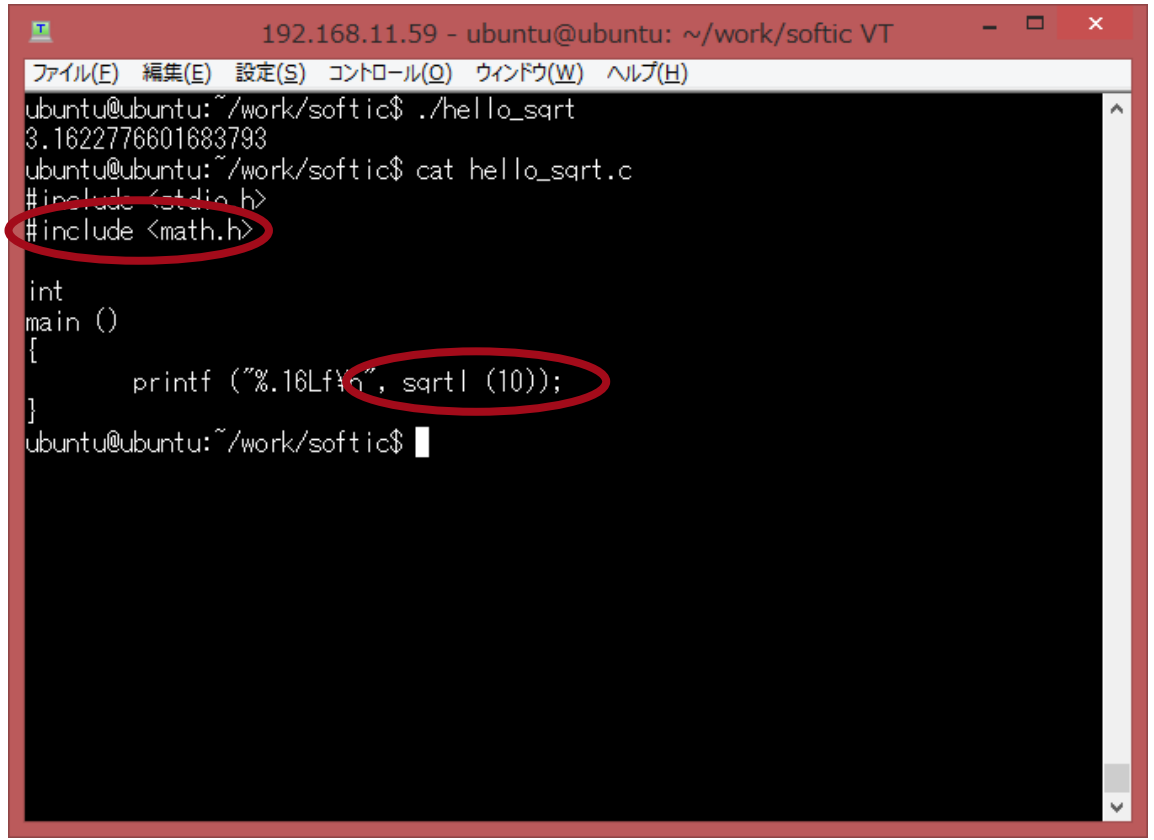
これを√の計算をする度に

書くのは大変、、

でも実際に書いたのは、



ポイントは、



- よく使う機能を棚に並べて必要な時に取り出せるようにしておく。
  これがライブラリ
  - √, X^2, sin, cos, log等の数学関数
  - アルファベットを小文字から大文字に変換する等のテキスト処理
  - Webアクセスする
  - 暗号復号する
  - 画面に●や▲等を描画する

  など

FUJITSU

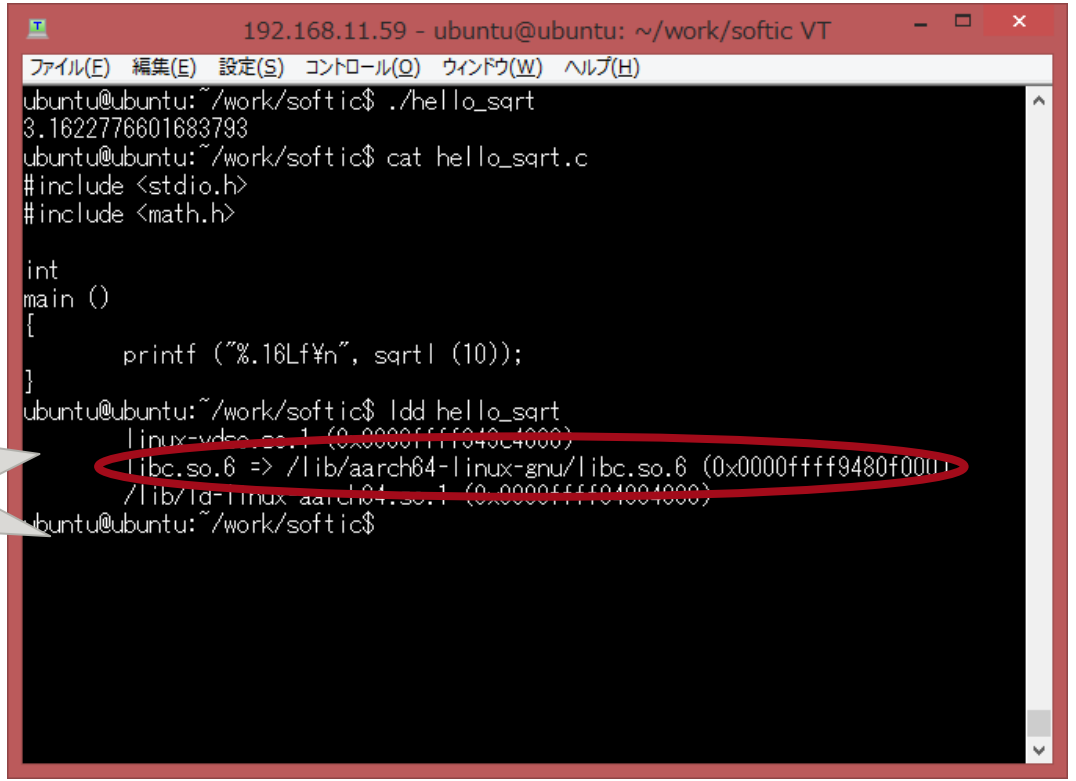
- 例えるなら、
  - 料理で毎回だしを取るのは大変だから、
  - だしを粉末や液体にして溶くだけにしたら、
  - 楽ですね。
  - 一般的なだしのレシピは公知であっても、おいしいだしは商品として売れる。
  - だしの商品がたくさんあるようにデータベースの商品はたくさんある。
  - データベースのライセンスは、、

アプリケーションとライブラリの関係は、、

FUJITSU

アプリケーション
0xffffffffffffffff
…
0x0000ffffffffffff
…
0x0000ffff9480f000 ライブラリ
…
0x00000000ffffffff プログラム
…
0x0

同じ空間で動く



動的リンク vs. 静的リンク

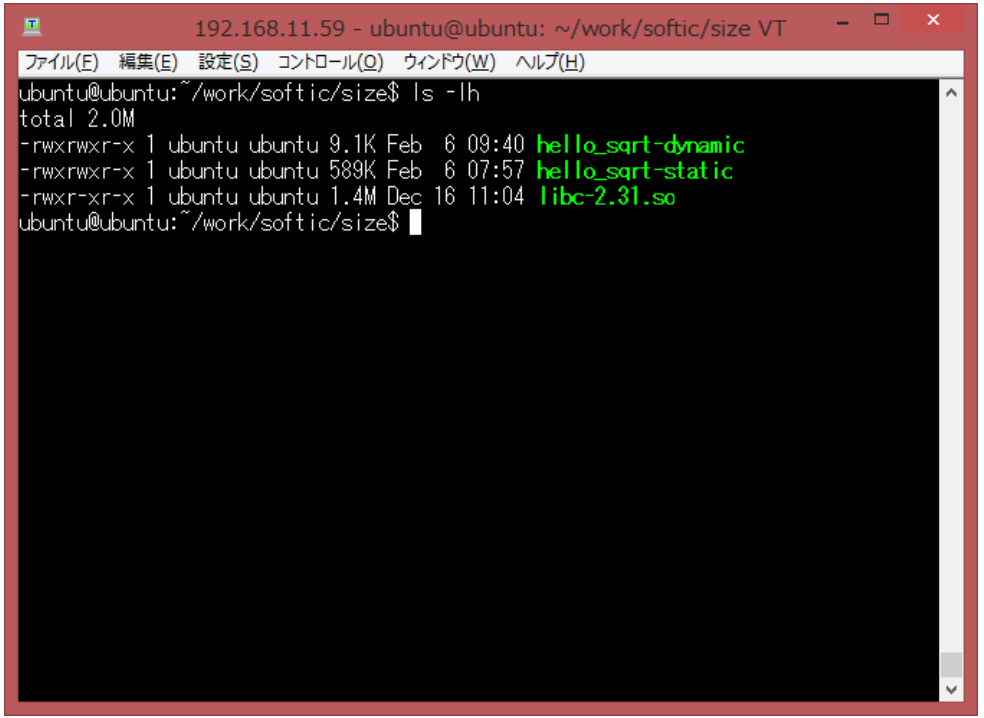
FUJITSU

動的リンク

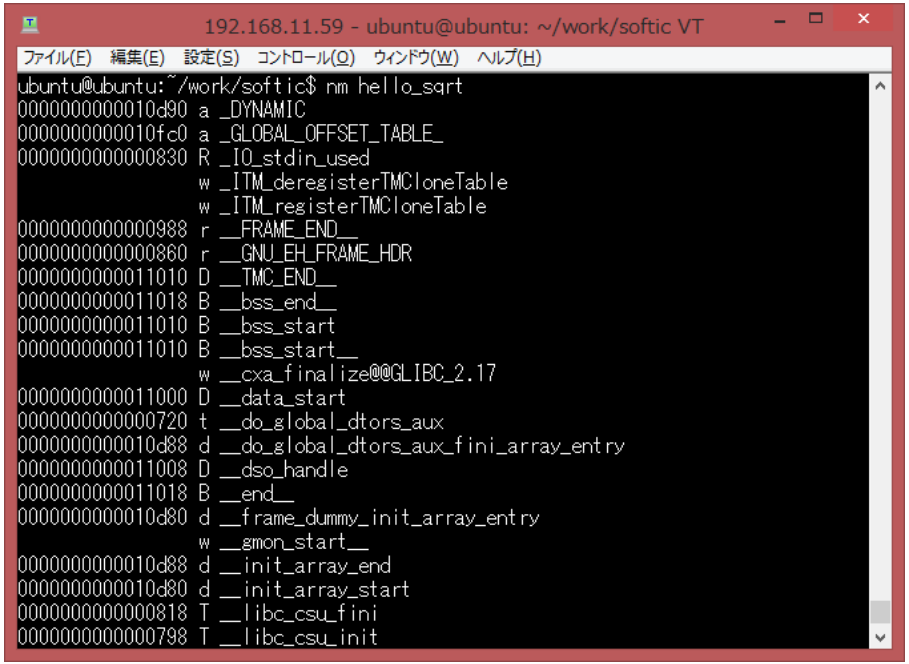
- 実行時にライブラリを呼び出し
- プログラム＋ライブラリの2ファイル
- プログラムのバイナリのサイズが小さい

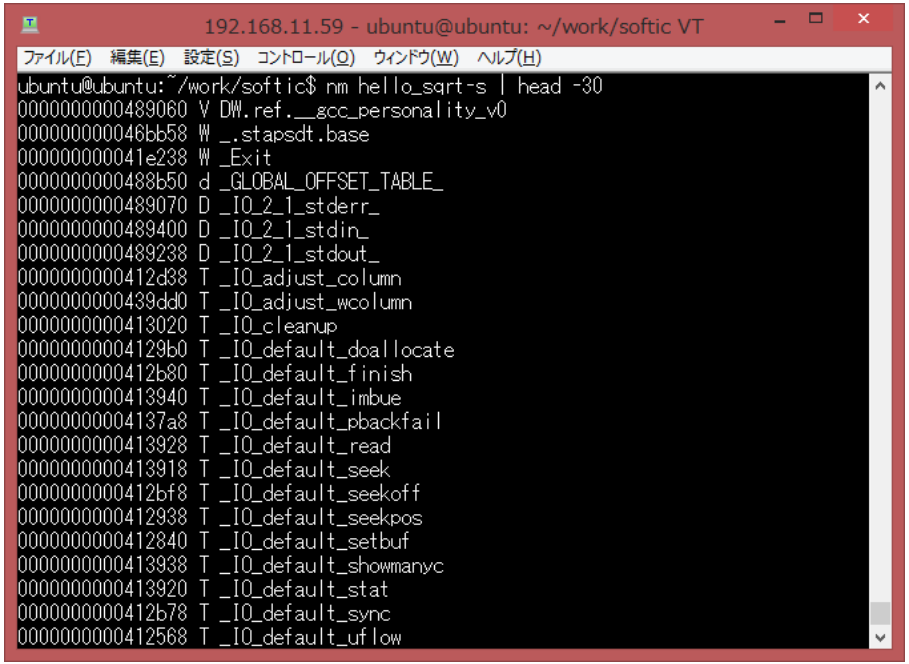
静的リンク

- ビルド時にライブラリを組み込み
- プログラム＋ライブラリで1ファイル
- ライブラリすべての機能を使わないなら、サイズが小さい



動的リンク vs. 静的リンク





動的リンク vs. 静的リンク

動的リンク	静的リンク
0xffffffffffffffff	0xffffffffffffffff
…	…
0x0000ffffffffffff	0x0000ffffffffffff
…	…
0x0000ffff9480f000	0x0000ffff9480f000
…	…
0x00000000ffffffff	0x00000000ffffffff
…	…
0x0	0x0

動的リンクも静的リンクも
同じ空間で動くことに
違いはない

(GPLの)及ぶ作品に対し、静的 vs 動的にリンクされたモジュールについて、GPLには異なる要求がありますか?
(#GPLStaticVsDynamic)

いいえ。GPLの及ぶ作品に静的もしくは動的にほかのモジュールとリンクすることは、GPLの及ぶ作品をもとにして結合著作物を作成することです。ですから、全体としての結合には、GNU一般公衆ライセンスの条項が及びます。GPLソフトウェアにGPLと両立しないライブラリを用いた場合、どのような法的問題が発生するでしょうか?もご覧ください。

https://www.gnu.org/licenses/gpl-faq.ja.html#GPLStaticVsDynamic

「集積物」とそのほかの種類の「改変されたバージョン」の違いは何ですか? (#MereAggregation)

「集積物」はいくつかの別々のプログラムからなり、それらは同じCD-ROMやそのほかのメディアにいっしょに入れられて配布されます。GPLは集積物を作成し配布することを認めています。たとえ、ほかのソフトウェアが不自由あるいはGPLと両立しないものである場合でもです。ただ一つの条件は、あなたは、それぞれのプログラムの個別のライセンスが許す権限をユーザが行使することを妨げるような、あるライセンスのもとで集積物をリリースすることはできないということです。

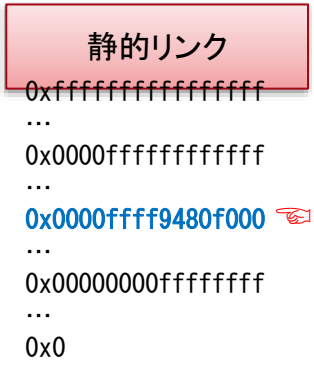
二つの別々のプログラムと二つの部分の一つのプログラムを分ける線はどこにあるでしょうか? これは法的な問題であり、最終的には裁判官が決めることです。わたしたちは、適切な基準はコミュニケーションのメカニズム(exec、パイプ、rpc、共有アドレス空間での関数呼び出し、など)とコミュニケーションのセマンティクス(どのような種の情報が相互交換されるか)の両方によると考えています。

モジュールが同じ実行ファイルに含まれている場合、それらは言うまでもなく一つのプログラムに結合されています。もしモジュールが共有アドレス空間でいっしょにリンクされて実行されるよう設計されているならば、それらが一つのプログラムに結合されているのはほぼ間違いないでしょう。

逆に、パイプ、ソケット、コマンドライン引数は通常二つの分離したプログラムの間で使われるコミュニケーションメカニズムです。ですからそれらがコミュニケーションのために使われるときには、モジュールは通常別々のプログラムです。しかしコミュニケーションのセマンティクスが親密であったり、複雑な内部データ構造を交換したりする場合は、それらも二つの部分がより大規模なプログラムに結合されていると考える基準となりうるでしょう。

https://www.gnu.org/licenses/gpl-faq.ja.html#MereAggregation

なぜLGPLは静的リンクで要求が変わるのか



☜静的リンクされたLGPLのライブラリを差し替える自由として、
「アプリケーションのオブジェクト (リンクされる前の状態)の提供」
が要求されている。
「リバース エンジニアリングの許諾」も同じ理由。

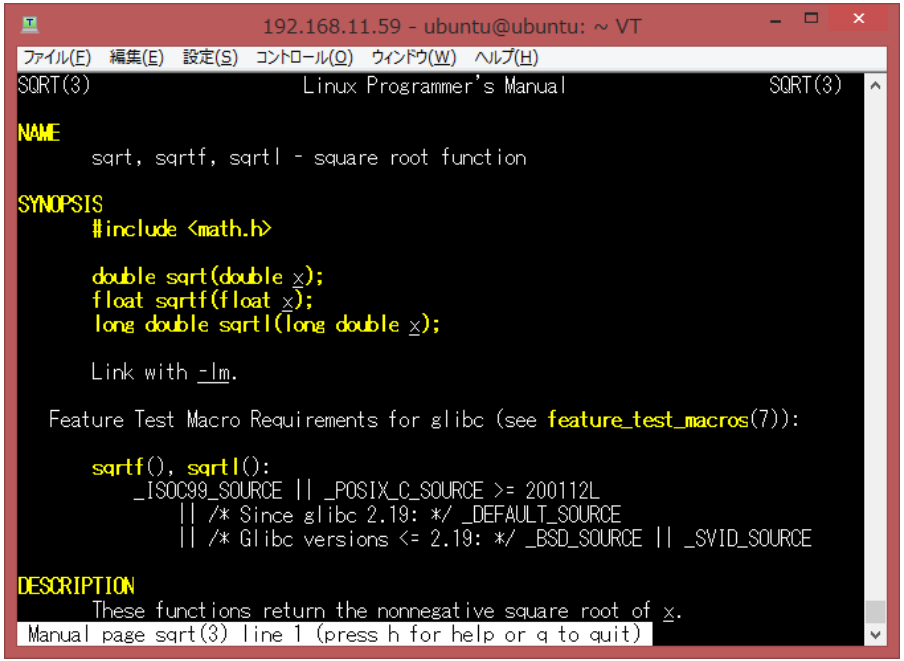
(LGPLの)及ぶ作品に対し、静的 vs 動的にリンクされたモジュールについて、LGPLには異なる要求がありますか?
(#LGPLStaticVsDynamic)

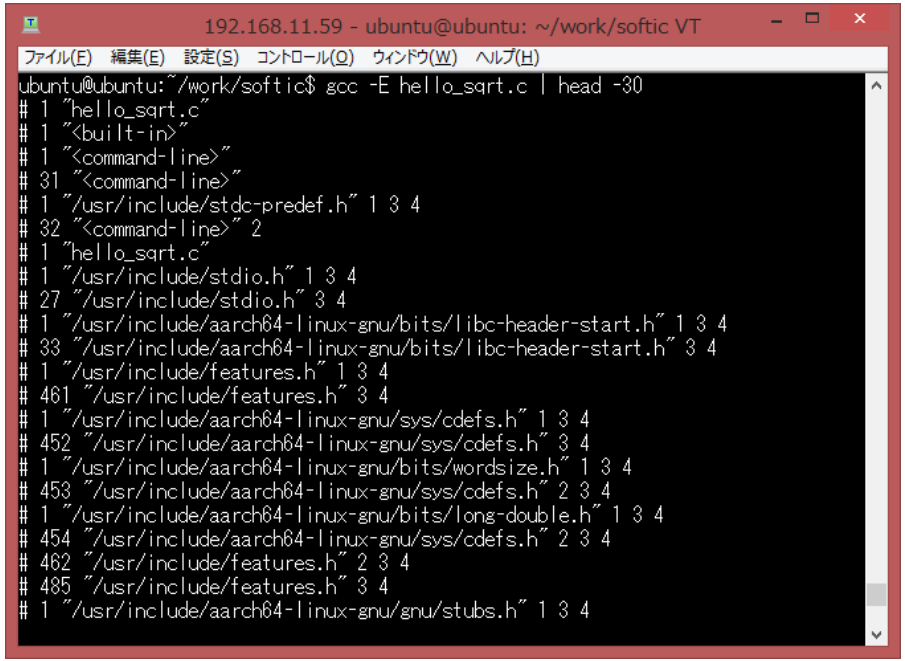
LGPL (現存のどのバージョンでも: v2, v2.1, or v3)に適合する目的では:

(1) LGPLのライブラリに対し静的にリンクする場合、ユーザがライブラリを改変してアプリケーションと再リンクできる機会のために、あなたは、あなたのアプリケーションを、オブジェクト(ソースの必要は必ずしもありません)フォーマットでも提供する必要があります。

https://www.gnu.org/licenses/gpl-faq.ja.html#LGPLStaticVsDynamic

ヘッダとの関係





FUJITSU

アプリケーションとアプリケーションの関係

パイプ

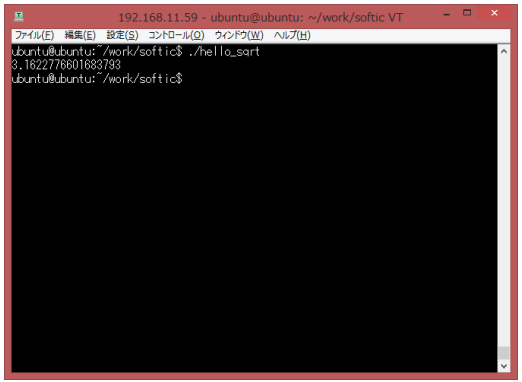
アプリケーション

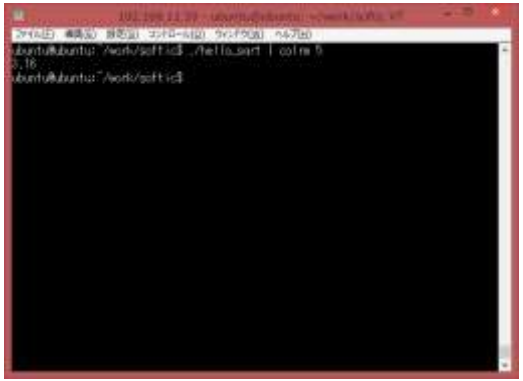
0xffffffffffffffff
…
0x0000ffffffffffff
…
0x00000000ffffffff
…
0x0

出力⇒3.1622776601683793⇒入力

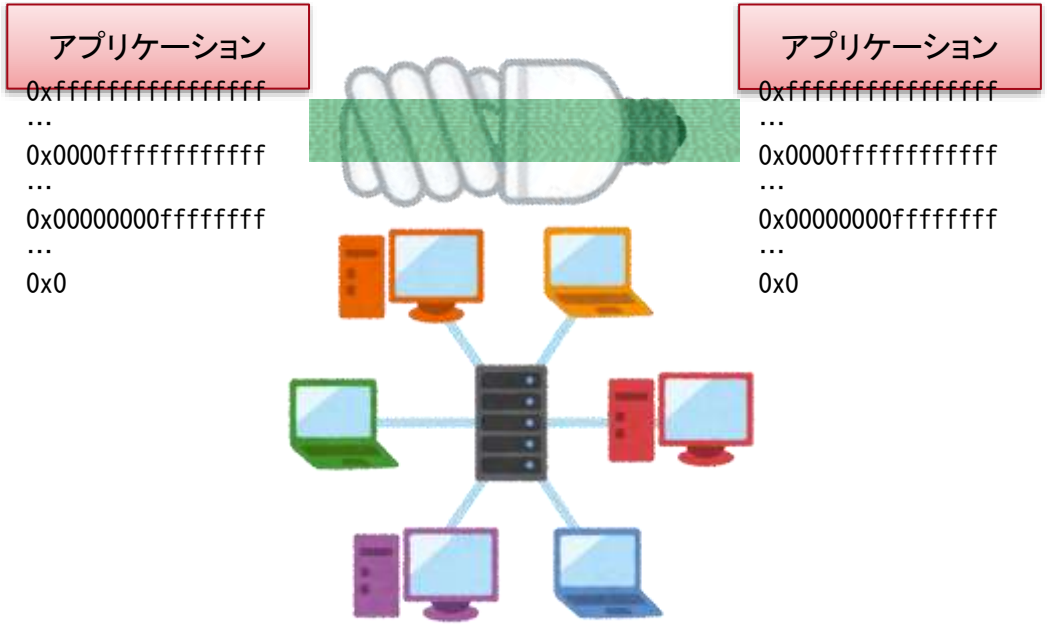
アプリケーション

0xffffffffffffffff
…
0x0000ffffffffffff
…
0x00000000ffffffff
…
0x0





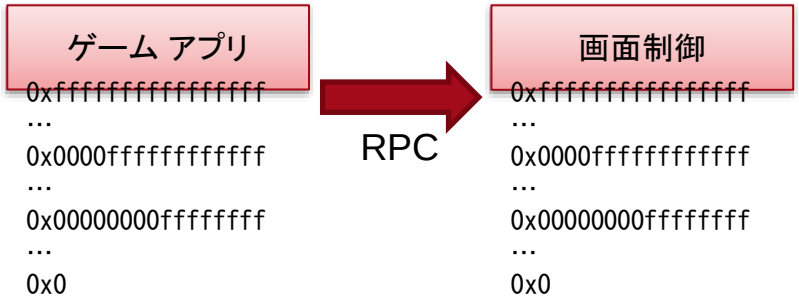
ソケット



プログラムとプログラムでデータをやり取りするネットワークの仕組み。
同じコンピュータ内でも別のコンピュータでも可能。
同じプログラム内でも可能だが、、

FUJITSU

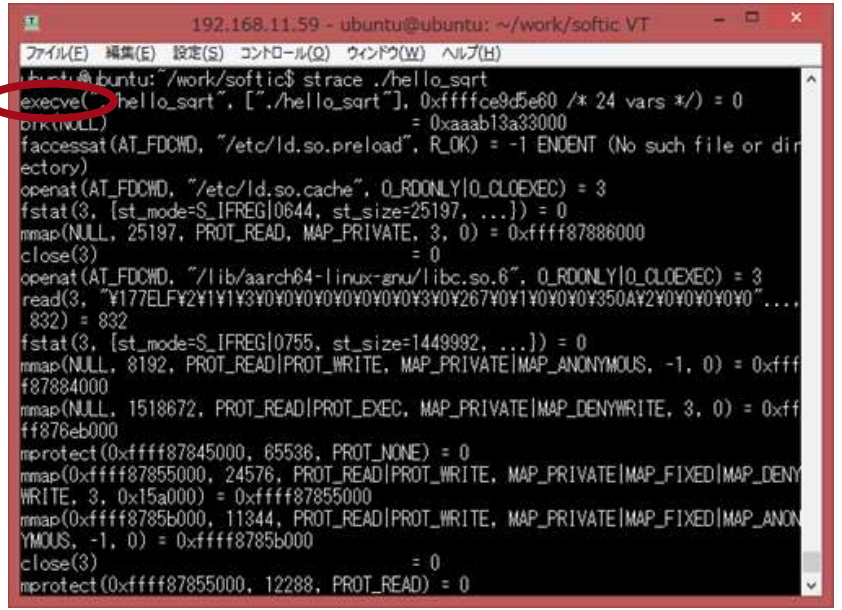
Remote Procedure Call、遠隔操作のこと



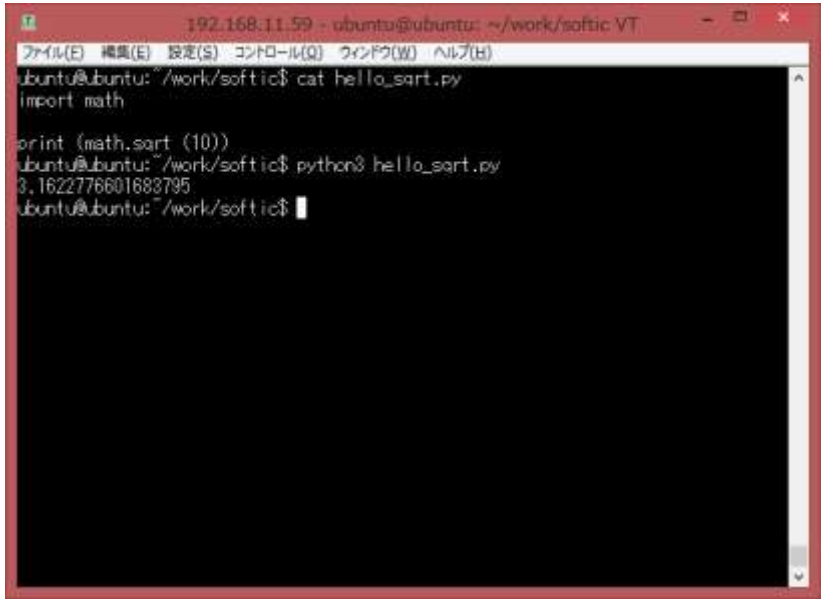
ゲーム アプリがグラフィックを描画したいときに画面全体を制御するミドルウェアに依頼を出すとき等に使われる仕組み。
同じコンピュータでもネットワークの向こう側でもよい。

exec

- execute、実行する
  - コマンドラインから、アプリケーション内から (fork)などトリガーは様々だが、新たなアプリケーションを実行するという意味



コマンドライン引数



プログラミング言語のインタープリタがGPLのもとでリリースされていた場合、そのインタープリタで解釈されるように書かれたプログラムは、GPLと両立するライセンスでなければならないでしょうか? (#IfInterpreterIsGPL)

インタープリタが単に言語を解釈するだけならば、答はノーです。解釈されるプログラムは、インタープリタにとっては単なるデータに過ぎません。GPLのような著作権法にもとづく自由ソフトウェアのライセンスは、あなたがインタープリタ上で利用するデータの種類を限定することはできません。あなたは、好きなデータ(この場合解釈されるプログラム)を使って、好きなようにインタープリタを実行することができますし、そのデータを誰かにライセンシングすることについて必要とされる条件は何もありません。

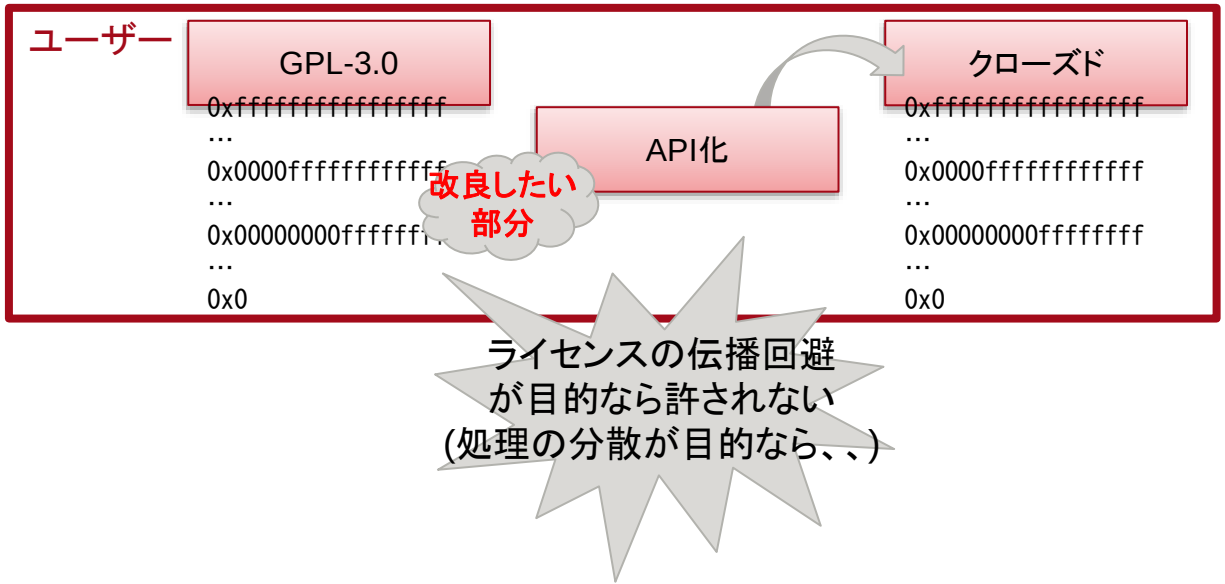
しかし、インタープリタが他の機能(多くの場合ライブラリですが、ライブラリである必要はありません)への「バインディング」を提供するように拡張されている場合、解釈されるプログラムはバインディングを使うことによって事実上それらの機能とリンクされることになります。ですから、もしそういった機能がGPLのもとでリリースされているならば、機能を利用している解釈されるプログラムはGPLと両立する形でリリースされなければなりません。JNI(Javaネイティヴインターフェース)はそのような機能の一例です。JNIによってアクセスされるライブラリは、それらを呼ぶJavaプログラムと動的にリンクされています。これらのライブラリはインタプリタともリンクされています。もしインタプリタがそれらのライブラリと静的にリンクされていれば、あるいは、動的にそれらの特定のライブラリとリンクするように設計されていれば、それもGPLと両立する形でリリースされる必要があります。

https://www.gnu.org/licenses/gpl-faq.ja.html#IfInterpreterIsGPL

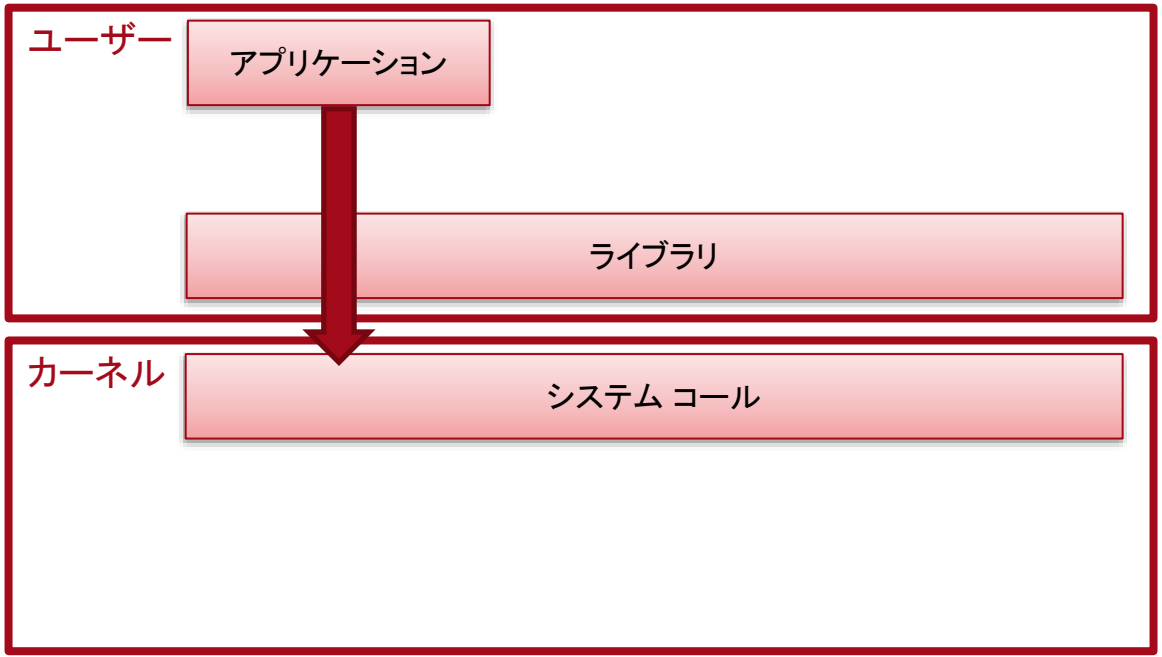
共有アドレス空間での関数呼び出し

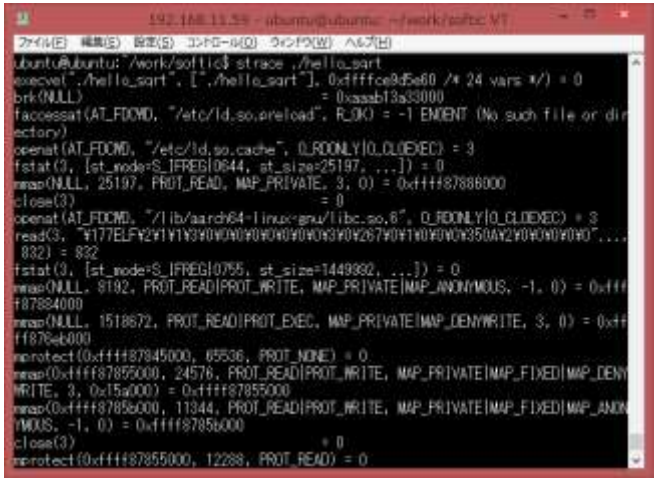
コミュニケーションのセマンティクスが親密

- コミュニケーション＝通信 の
  セマンティクス＝データ が
  親密!?



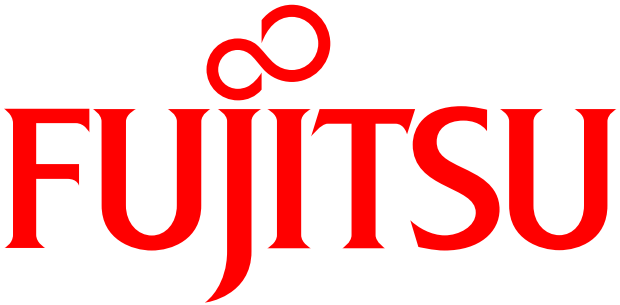
システムコール





FUJITSU

Q&A?



shaping tomorrow with you