OpenCOBOLとPerlを使った汎用機ダウンサイジング

有限会社ランカードコム

峰松　浩樹

概略

- 自己紹介
- OpenCOBOLについて
- 今までの経緯と現在の状況
- OpenCOBOLの実際
- 長崎県版OpenCOBOL
- JCLの移植例
- 長崎県での状況
- 移植作業例
- 将来

自己紹介

- 所属
  - 有限会社ランカードコム
- 所在地
  - 東京都新宿区四ッ谷1-10-2-408
  - 長崎県長崎市扇町33-40
- 業務内容
  - 長崎県にてOSSを活用した業務提案
- 主要顧客
  - 長崎県庁、他自治体など

私

- 名前
  - 峰松　浩樹
- 所属
  - 有限会社ランカードコム
  - OSSコンソーシアム
    オープンソースCOBOLソリューション部会
- Twitter: @minemaz
- Mail: mine@lancard.com

OpenCOBOL

- COBOLからC言語への変換とコンパイル
- 元開発者は日本人（西田　圭介 氏）
- 現開発者はRoger While氏
- 現バージョンは OpenCOBOL 1.0
- OpenCOBOL 1.1ではncursesを使用した画面制御機能の拡充とデバッグ機能の強化等
- OpenCOBOL 2.0ではGPL3へ移行

OpenCOBOL（2012年の状況）

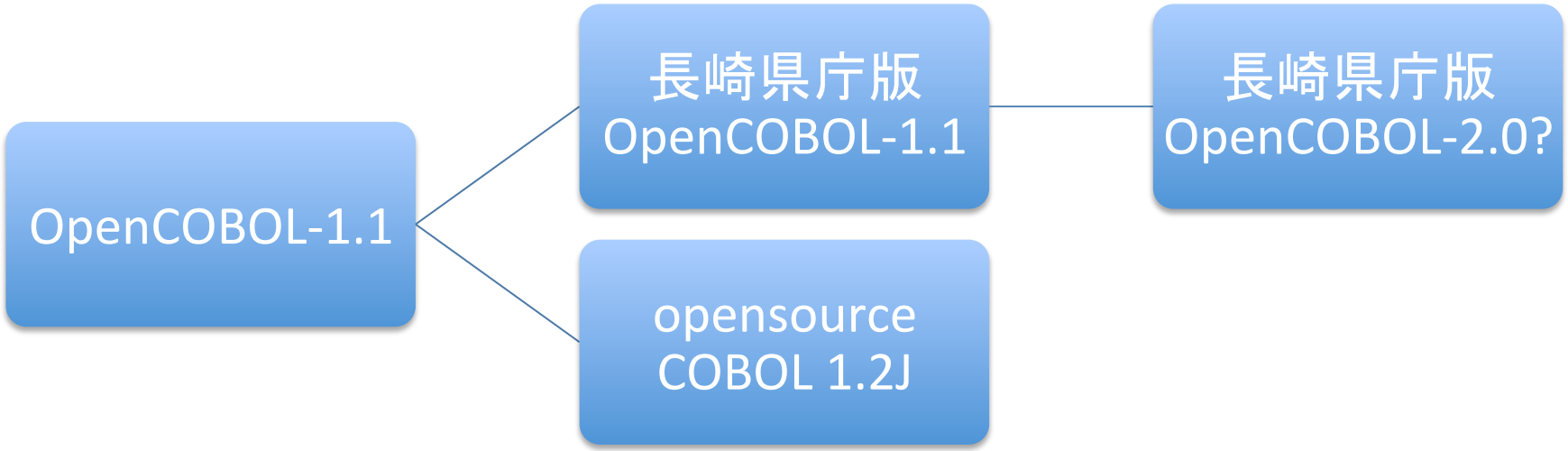
- OSSコンソーシアム
  - オープンソースCOBOLソリューション部会 立ち上げ(2012/07/05)
  - Opensource COBOL 1.2J 2012/07/31 リリースされました！

- OpenCOBOLコミュニティエディション
  - 現メンテナのアクションを待てない開発者がプロジェクトをフォーク (OpenCOBOL-CE)

opensource COBOL 1.2J

- OSSコンソーシアム
  - オープンソースCOBOLソリューション部会
  - 日立ソリューションズ様、弊社にてリーダーをつとめさせて頂いてます(2012/08/03 現在)
  - 「osscons cobol 1.2J」で検索
  - 汎用的な日本語対応
  - OpenCOBOL-1.1から 302箇所もの修正、変更

大まかな流れ

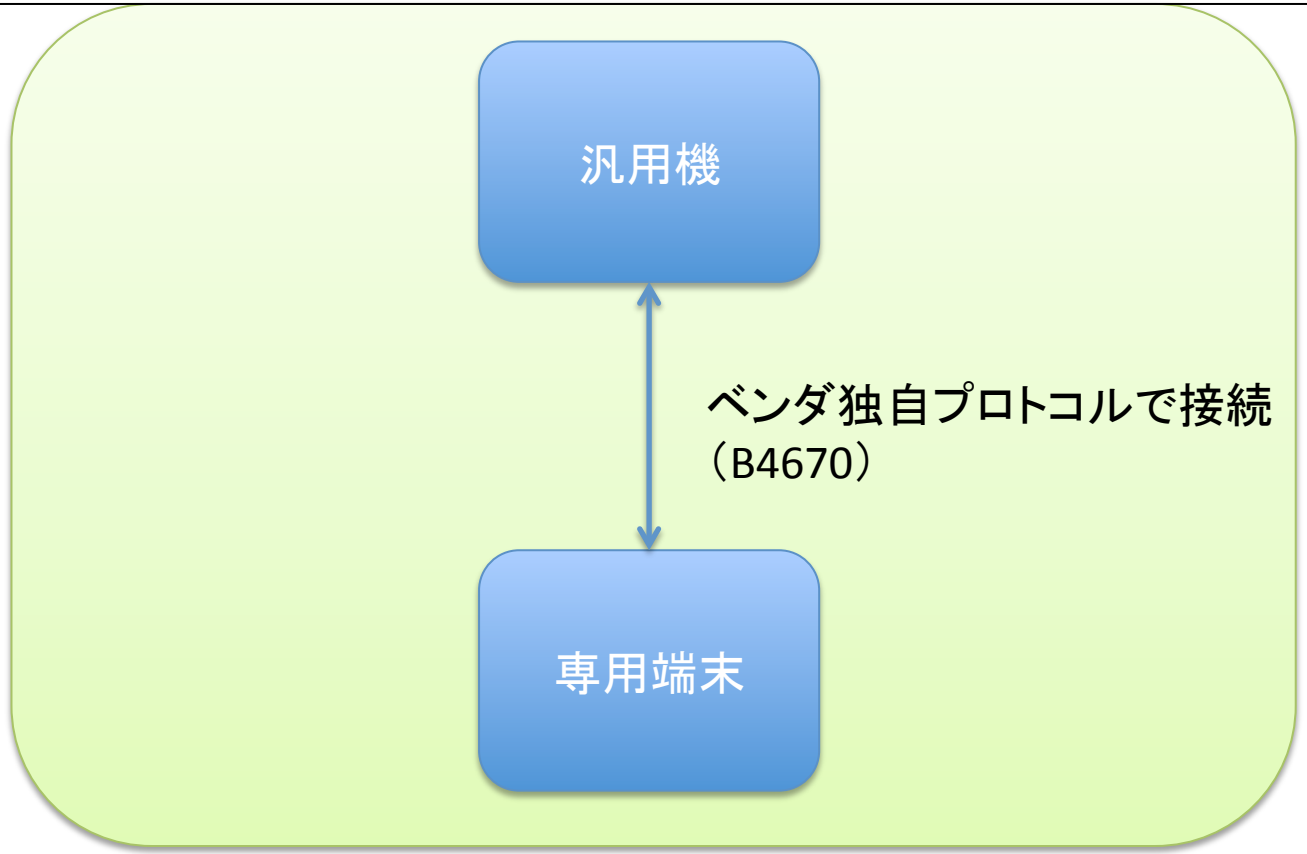

OpenCOBOL-1.1
長崎県庁版
OpenCOBOL-1.1
opensource
COBOL 1.2J
長崎県庁版
OpenCOBOL-2.0?

長崎県：今までの経緯

- 2008年：長崎県様よりOpenCOBOLを使った移植可能性について調査依頼
- 2009年：Webアプリとして既存のCOBOLソースを流用できないか調査依頼
- 2010年：JCL等バッチ処理について本格的な移植ツールの実装
- 2011年：バッチ編集、管理画面の実装。既存データの移行等
- 2012年：実機と並行運用開始

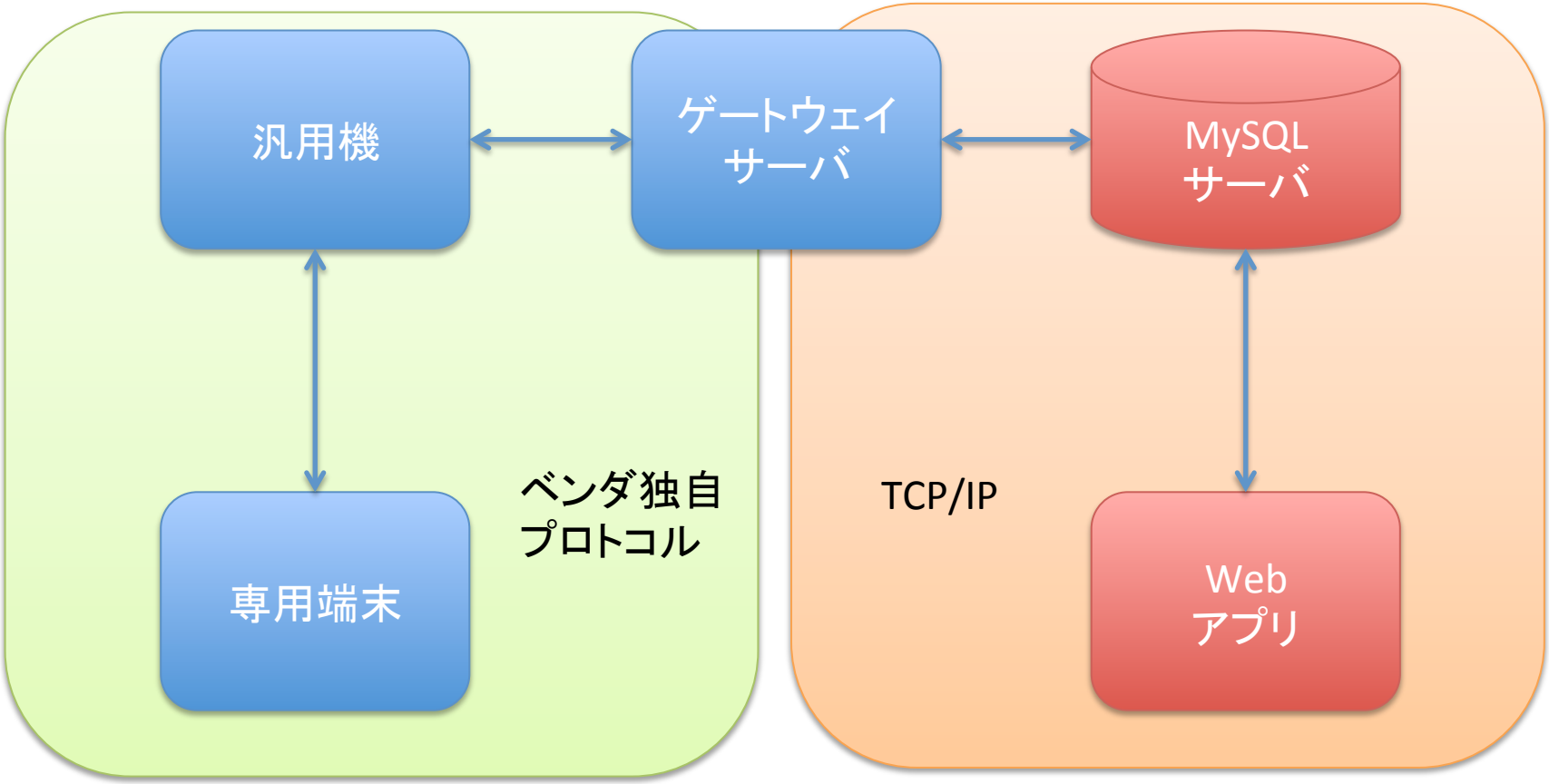
汎用機からの移行

移行前環境。汎用機と専用端末（エミュレーション使用）での接続



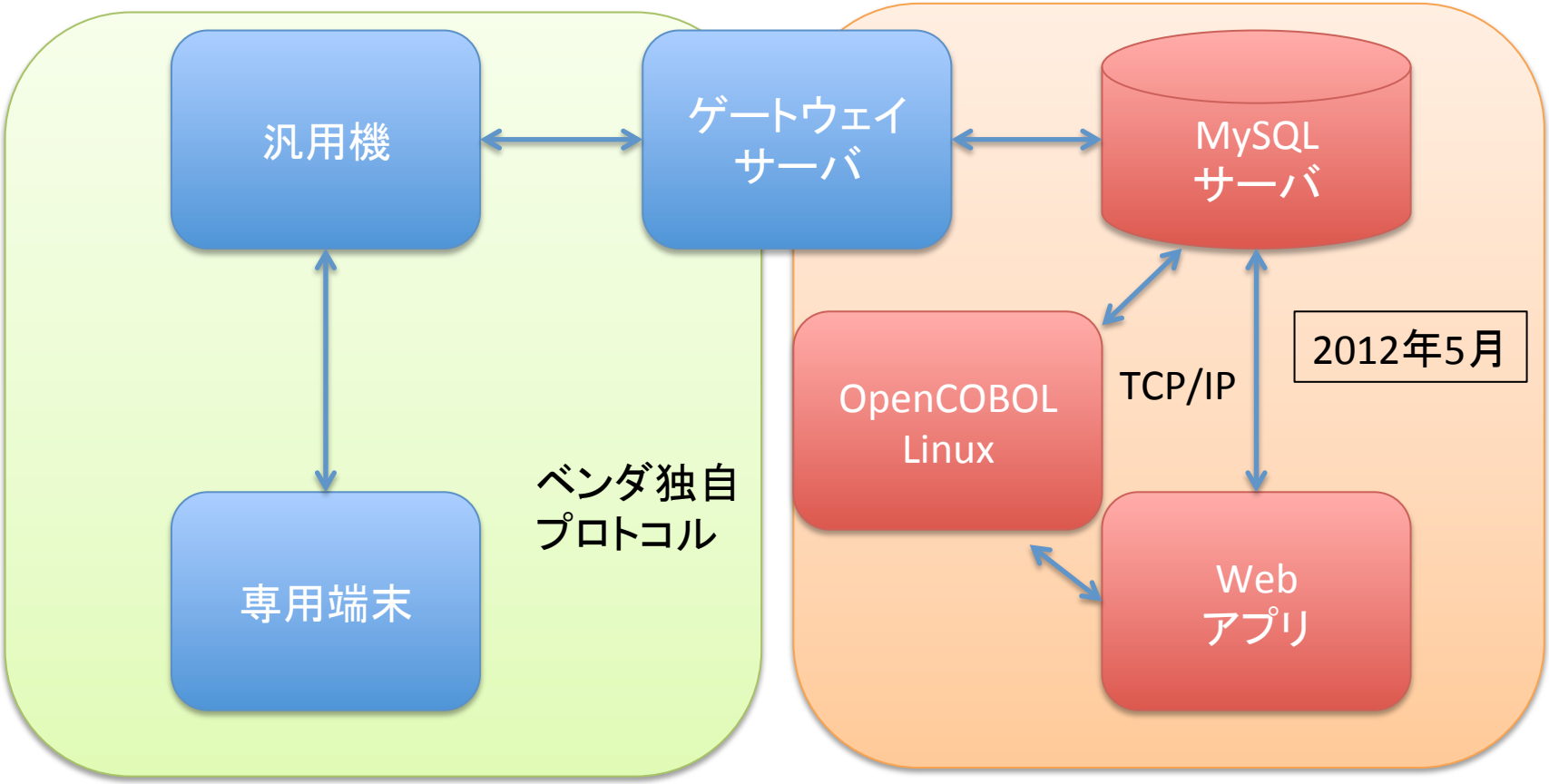
汎用機からの移行(入力端末移行)

専用端末以外の端末からもデータエントリを可能とし将来の置き換えに備える



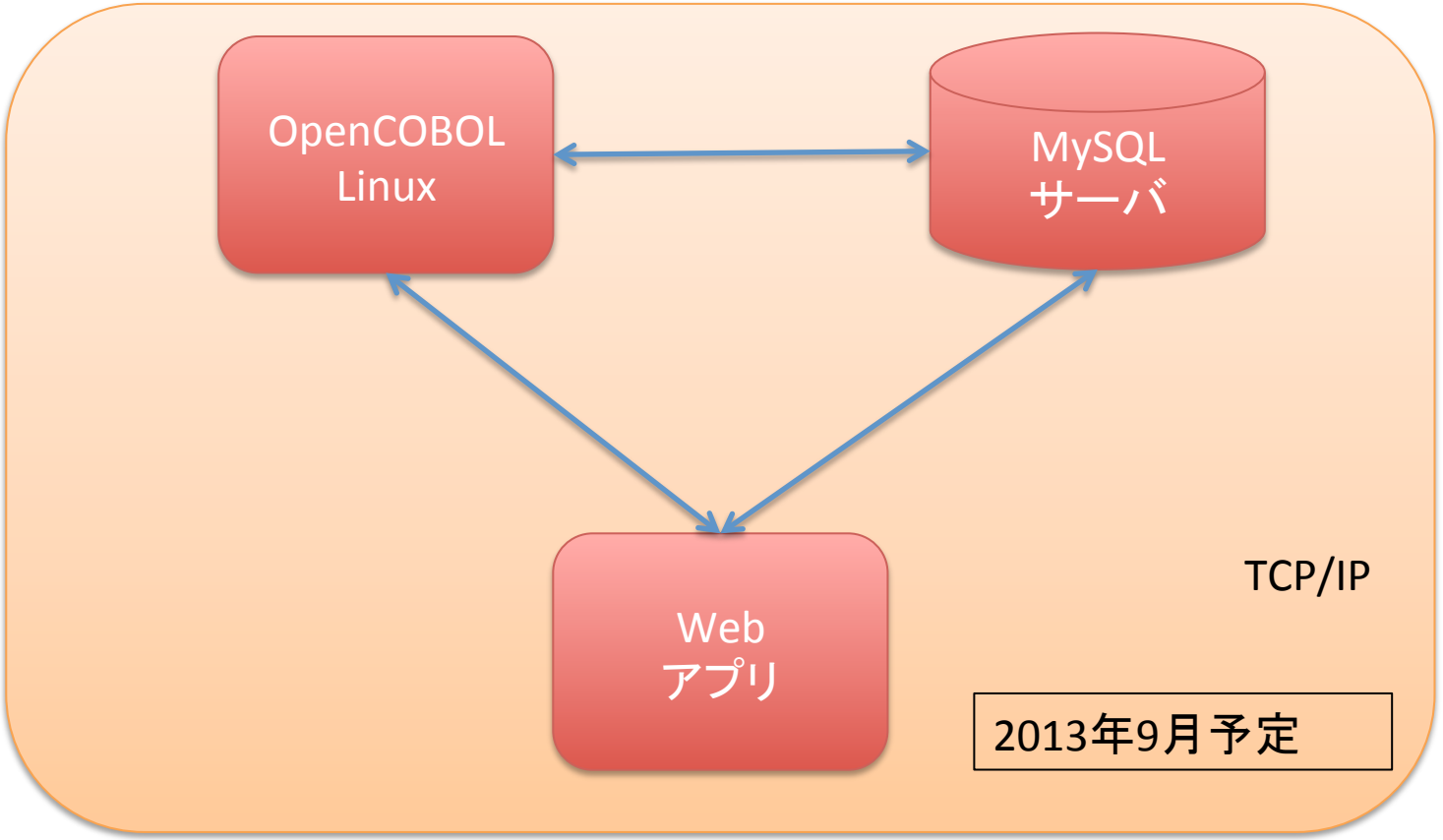
汎用機からの移行(並行運用)

汎用機とlinuxの並行運用。データの突き合わせを行いながら
動作相違箇所のチェック、運用上のノウハウの蓄積を行う



汎用機からの移行(移行完了)

汎用機撤去後の環境
OpenCOBOL/Linuxにて処理を行い、データエントリもWebアプリから実施する



実装環境

- 既存環境：
  - 国産メーカ汎用機
  - 専用ページプリンタ
  - 専用端末エミュレータ

- 移行先：
  - Linuxサーバ数台（業務種類毎に1台ずつ）
  - 印刷出力はPDFファイルにて生成
  - Webブラウザで（JCL、COBOLプログラムの編集など）端末操作

作業対象

- 職員総合、財務会計、予算編成の3システム
- COBOLプログラム
  - 合計で約500万行、約1万5千本
- JCL（ジョブコントロール言語）
  - 合計で約100万行、約6500本
- 帳票レイアウト
  - 職員総合システムで約400本
  - 他システムは未知数（同数程度、計1200本？）

作業方針

- 既存のCOBOLソースはなるべく手修正しない
  - COBOLプログラマの工数が不足している
- 帳票レイアウトは流用する
  - 帳票デザインツールの開発や再デザインが大変
- データレイアウトはそのまま使用する
  - COBOLはプログラム中にデータレイアウトがハードコードされている（COPY句)
- JCLもなるべく変えない
  - システム運用はお客様が行うため。

対応

- COBOLプログラムを大規模に書き換える必要があるなら、コンパイラ側を変更するか
  ツールで自動書換する
- 既存プリンタのエミュレーションを行い
  PDFファイルに印刷出力する
- MySQL,可変長データなどの操作用にCOBOL側を拡張しライブラリにて機能吸収する
- JCLは習得と導入が簡単な他言語へ変換する

現在の状況(2012/08/03)

- COBOLプログラム
  - 汎用機の特殊な言語方言をOpenCOBOLへ導入
- JCL
  - ソート、マージ、集計、フィルタ処理用ツールは
    ほぼ実装完了。
  - 並列動作、負荷モニタなど~~調整中~~試験中
- 帳票
  - ほぼ完全に再現
  - 280ページ約12秒
  - 帳票用にOpenCOBOLを拡張

OpenCOBOLの実際

- 移植するにはOpenCOBOLの機能はかなり不足していた
  - PIC N項目(NATIONAL項目)が PIC X項目の2倍のbyte数確保してくれる程度のL10N（メーカーに依りますが…）
  - 半角カナ変数名、漢字変数名には未対応
  - 印刷出力も単純なラインプリンタのみ対応
  - 通貨記号に「¥」が使えませんでした
  - 海外では業務利用例を見かけるが、国内ではORCAでの利用を除けば学習用の位置づけ

これでは業務に使うのは無理

・・・　ということで改造

長崎県版OpenCOBOL

- CHARACTER TYPE、COLUMNでの印刷文字種、位置指定機能
- DATE IS NATIVE指定時に日付を和暦にて返す機能
- COB_DATE環境変数設定時にACCEPT～FROM DATE等で設定値を返す機能
- SPECIAL-NAMESで定義した外部値の取り込みと吐き出し
- RETURN-CODEとは独立したSYSTEM-STATUS値
- DUMMYファイル機能(/dev/nullのような機能、ISAMでも利用可)
- SEQ形式（固定長）ファイルのレコード長、キー長チェック機能
- PIC XからPIC NへのMOVE時に半角→全角の自動変換
- PIC NへのMOVE時に全角空白での補完
- 固定長ではなく改行コードをレコード区切りとする可変長入力データ形式対応
- SETでのファイル状態のコピー
- PIC NにてALL SPACEを指定した場合に全角空白でのフィル
- COPY～REPLACINGでの部分一致置換
- 半角カナ変数名の使用
- データハンドラの外部化によるmysql他データベースとの連携
- 印刷出力時の行について絶対位置指定・印刷出力時に行復帰コードの自動挿入
- ファイル名の代わりに URI渡しが可能
- 2次キーの取り扱いについて柔軟な対応
- STRING ～ DELIMITED BY SPACE 等 STRING文の日本語対応
- 発生した例外をファイルへロギング
- 印刷時の出力先ファイル名の指定等、印刷出力時に独自パラメータ出力可能
- デバッグ実行中の行番号、モジュール名の取り出し
- ステップ実行機能
- etc,

結構手を入れてます。

非互換性の例

- 変数Aに0、変数Bが0の場合の A / B
  (DEVIDE A BY B)
- 通常は ZERO除算例外が発生する
- 変数Aが0の場合だけ結果が0になる処理系
- 都度0チェックを追加すると
  大変な移植コストになる
- 演算ライブラリ側を調整して対応

JCLの移植

- 特定汎用機メーカーのJCLのOSS実装は無い
  →動作環境を新規開発するしかない

  でも新たに処理系作りたくない

- お客様の要望により大幅な改造は無理
  - 文がセミコロンで終端していて、
  - LABEL:　で定義されたラベルにGOTOで飛べて
  - 強力な文字列置換機能をもち
  - マルチスレッドで動作して
  - 大文字のキーワードが使えて
  - ヒアドキュメント機能があって
  - プログラマが割と多くて
  - 動作が安定しており実績があって仕様も枯れた言語

  ということでPerlを使うことに決定

JCLの移植例(JCL)

JCLでのバッチ処理の記述例

```
¥JOR  OUTID=SPR OUTDEV=PR/M050/JSTD CLASS=A       ;

¥COMM  月次パラメタＳＴＯＲＥ;
¥INPUT    ACCEPT1,TYPE=DATA,LIST=YES;
AIDF0270 1 4210918            例月（１）＋△＋支給日
AIDF0271 4210822 4210918      償還完了者リスト抽出期間（自～至）
********                        ＊前回支給日の翌日～今回支給日
¥ENDINPUT;

¥CREATESEQ
       INFILE=(ACCEPT1 FILESTAT=SYSIN )
    OUTFILE=(AID.F089K RECSIZE=80 BLOCKSZ=11440 RECFORM=FB BPB=4);

¥COMM  引去金ファイル作成;
AIDM2510:¥STEP   AIDM2510 FILE=AIA.IL1 DUMP=DATA;
          ¥ASSIGN AIDF0010 AID.F101    SHARE=ALL HOLDMODE=NO;
```

JCLの移植例（perl）

```
#!/usr/bin/perl
use strict;
use warnings;
use JCL;

JOR  "OUTID=SPR OUTDEV=PR/M050/JSTD CLASS=A      ";

COMM  "月次パラメタＳＴＯＲＥ";
INPUT   "ACCEPT1,TYPE=DATA,LIST=YES",<<_EOT;
AIDF0270 1 4210918           例月（１）＋△＋支給日
AIDF0271 4210822 4210918     償還完了者リスト抽出期間（自～至）
********                       ＊前回支給日の翌日～今回支給日
_EOT
ENDINPUT;
CREATESEQ
      "INFILE=(ACCEPT1 FILESTAT=SYSIN )",
   "OUTFILE=(AID.F089K RECSIZE=80 BLOCKSZ=11440 RECFORM=FB BPB=4)";

COMM  "引去金ファイル作成";
AIDM2510: STEP   "AIDM2510 FILE=AIA.IL1 DUMP=DATA";
          ASSIGN "AIDF0010 AID.F101    SHARE=ALL HOLDMODE=NO";
```

perlでのバッチ処理の記述例

移植作業

- COBOLソースの変換とコンパイル

```
$ acos2oc.pl COBFILE.txt
```

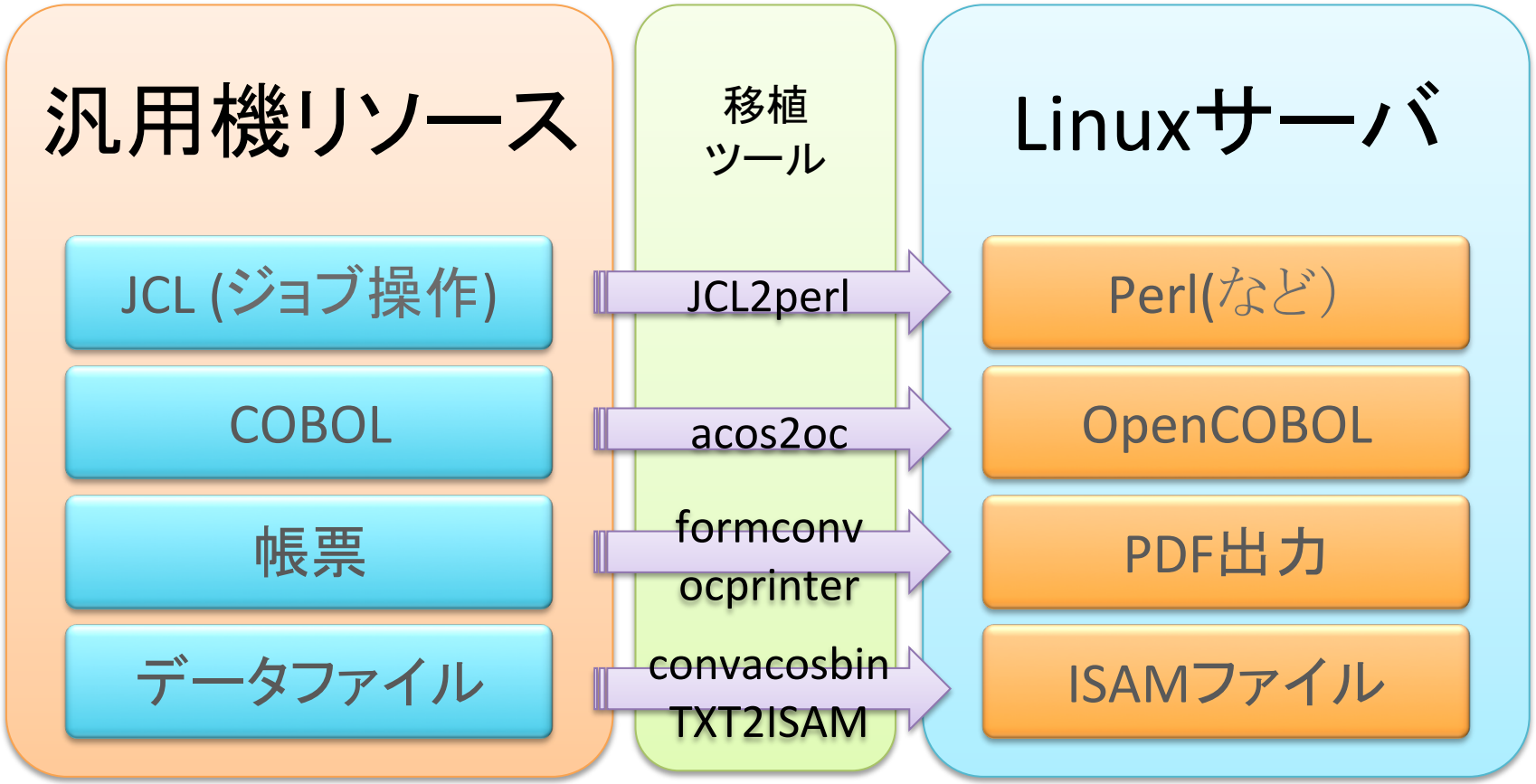
* JCLの変換

```
$ JCL2perl.pl JCLFILE.txt > JCLFILE.jcl
```

* 帳票レイアウトファイルの変換

```
$ cp FORMFILE.txt form/FORMFILE.form
```

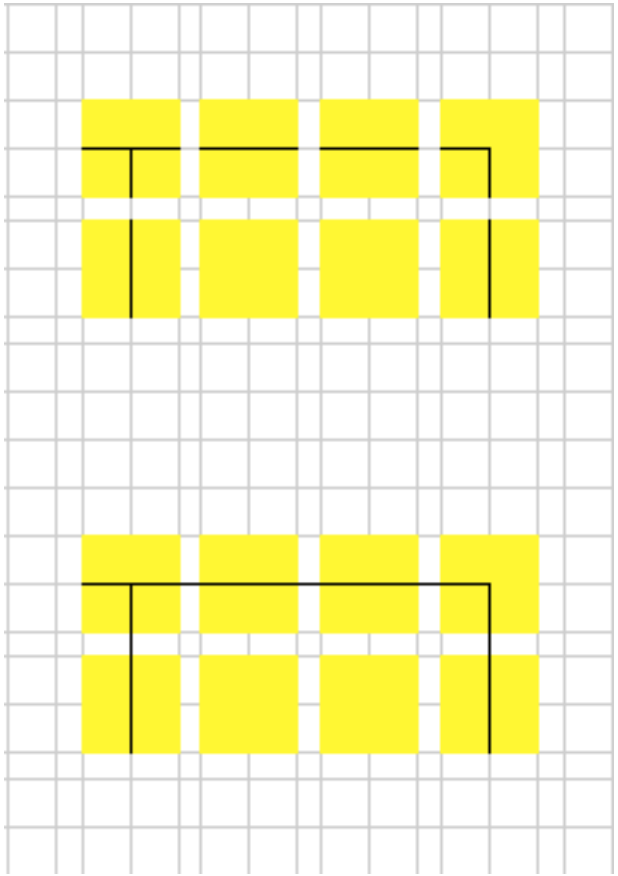
移行イメージ


汎用機リソース
JCL (ジョブ操作)
COBOL
帳票
データファイル
移植
ツール
JCL2perl
acos2oc
formconv
ocprinter
convacosbin
TXT2ISAM
Linuxサーバ
Perl(など)
OpenCOBOL
PDF出力
ISAMファイル

実演

-

文字罫線対応

- 

- 罫線文字そのまま出力すると文字間が開く

- 仮装プリンタドライバで文字間の罫線を補正

長崎県での状況

- プロジェクトはredmineで管理
- プロジェクトのソース、バイナリはsvnで配布
- 長崎県職員と複数の業者メンバーが参加
- CentOS5+perl+OpenCOBOL+mysqlなので各社内でコンパイル、実行、帳票デザインが可能になった
- OSSで全体的な開発コストが低下

長崎県での状況

- 平成24年度当初予算等の公表で使用
  - http://www.pref.nagasaki.jp/zaisei/yosan/gian/24gian_tousho.html

将来

- ファイル名をURI渡し出来るように改造
  - misam://directory/file (実装済)
  - mseq://directory/file (実装済)
  - perlfs://class_name/param (実装済)
  - ~~mysql://servername/db?table= (実装済)~~
    - perlfsに統合
  - hdfs://resource (実装中)
  - 等 各種データ形式を直接操作
  - C言語以外のソースへの変換(Perl, Ruby, Scala, Java 等)
  - 等々

php+OpenCOBOL+Perl+MySQL

- COBOLは固定長のバイト列を扱う
- Perlで固定長のバイト列作ってやればいい？
- MySQLとのやり取りをperlに任せればOK?

php+OpenCOBOL+Perl+MySQL

- PHP側記述例

```
if(!extension_loaded('opencobol')) {
    dl('opencobol.so' . PHP_SHLIB_SUFFIX);
}
$err    = opencobol_init();
$handle = opencobol_load("/tmp/TEST0001.so");
$err    = opencobol_call(
              $handle,
              "TEST0001",
              array(
                "WK-KETA" => "9876543",
                "WK-A" => "ABCDEFGHIJKLMNOPQRSTUVWXYZ" ));
```

php+OpenCOBOL+Perl+MySQL

- OpenCOBOL側記述例

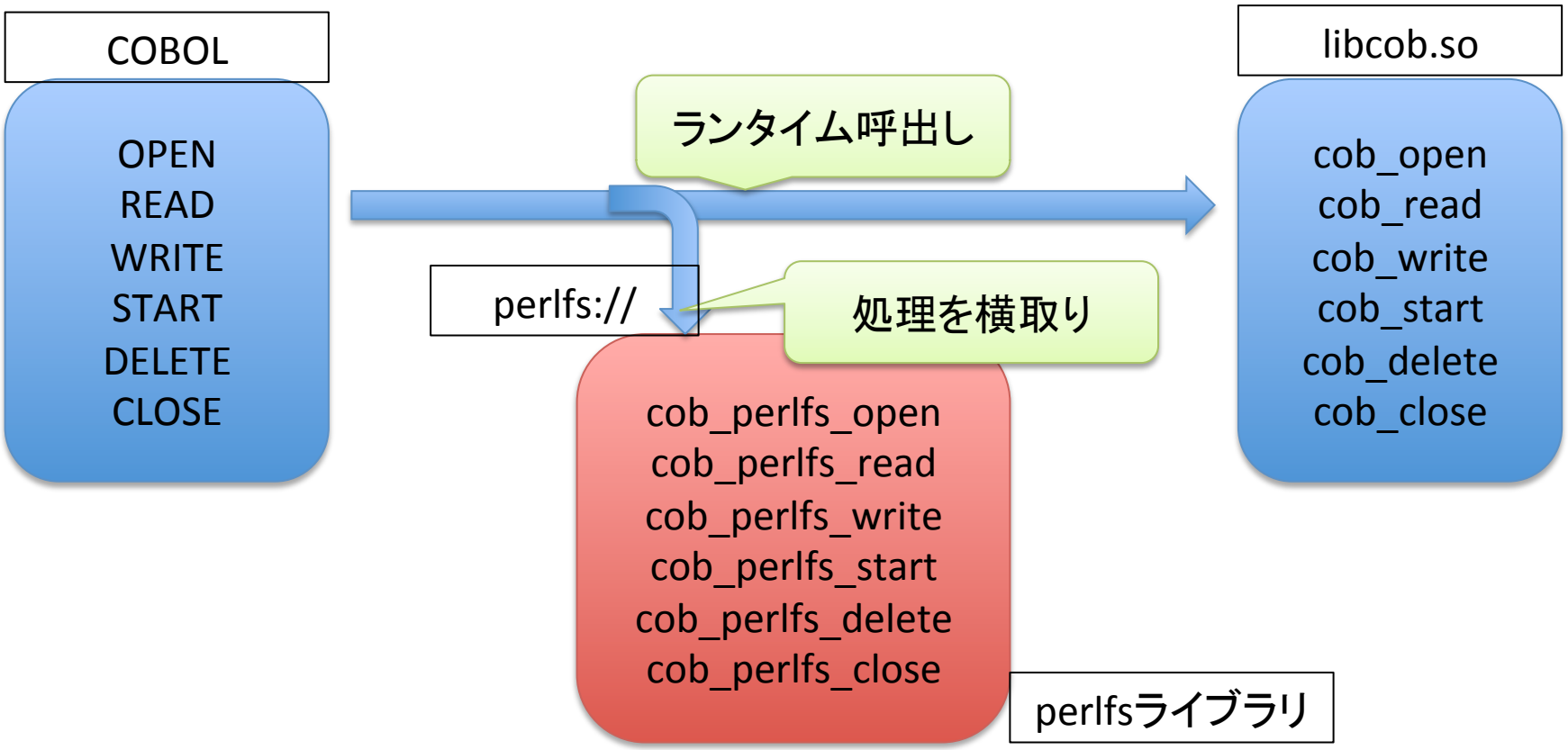
```
*     OPEN        INPUT                    IN4-F.
      CALL "cob_perl_require" USING "COB_PERL.pm"
      CALL "cob_perl_call" USING "mydb_open"
           "IN4-F" "AIACF050".
      CALL "cob_perl_results" USING
            COBPERL-STATUS LN010-FSTAT1.

～略～

      CALL "cob_perl_close".
```

php+OpenCOBOL+Perl+MySQL

- Perl側記述例

```
sub mydb_open {
    my ($fh, $dbname) = shift;
    eval {
        require $dbname.".pm";
    };
    my $db;
    eval "¥$db = new $dbname();";
    $COBPERL_DB::dbnames{$fh} = $db;
    $db->{'dbname'} = $dbname;
    $db->{'status'} = "00";
    $db->{'fstatus'} = "00";
    $db->Open;
    ($db->{'fstatus'}, $db->{'status'});
}
```

ファイル操作をフック


COBOL
OPEN
READ
WRITE
START
DELETE
CLOSE
ランタイム呼出し
perlfs://
処理を横取り
cob_perlfs_open
cob_perlfs_read
cob_perlfs_write
cob_perlfs_start
cob_perlfs_delete
cob_perlfs_close
perlfsライブラリ
libcob.so
cob_open
cob_read
cob_write
cob_start
cob_delete
cob_close