1J-3

mruby言語採用による組込みソフトウェア開発の生産性向上

池田 智之† 三井 浩康‡
東京電機大学† 東京電機大学‡

1 はじめに

近年，組込みシステム開発はシステム機能の複雑化に伴い高機能化，大規模化しており，優先度の高い課題として開発コストの削減，開発期間の短縮が挙げられている[1]．一方で高性能なMPUやSoC，大容量メモリの普及によって演算処理能力が高まり，組込みソフトウェアの開発により高級な言語を用いる事例が増えている．

しかし，多くの組込みソフトウェアの開発は従来通りC言語で開発されており[2]，C言語のソフトウェア資産が大半を占める中で他言語へ移行することは難しいという問題がある．そこで，いかにC言語の資産を生かしたまま開発の大規模化に対応していくかが一つの課題である．

本研究では，近年新たに登場した組込み向け言語であるmrubyに注目し，C言語との併用による開発の効率化の可能性を検討する．同一のシステムをC言語のみを用いて開発したものと，mrubyとC言語の両言語を用いて開発したものの2種類を用意する．そして両者を比較することにより，組込みソフトウェア開発において，mrubyとC言語の2種言語利用による開発に移行した場合の，mrubyの有用性について評価する．

2 本研究で使用する技術

2.1 mruby[3]

mrubyは松本行弘氏が開発したRubyの組込み向けエディションである．mrubyはRiteVMという仮想マシン(VM)上で動作し，VMの実行に必要な情報は「mrb_state」という構造体に格納されC言語中から組込みAPIとして利用できる．また，mrubyは機器への依存性が低いため様々な環境に導入でき，様々なコンピュータでメモリ管理はC言語で書き，高度な処理はmrubyで書くというような運用が可能である． mrubyはガベージコレクション機能，シンプルな記述の文法，オブジェクト指向言語の組込み向け言語であることを特徴としている．Rubyユーザーは日本人が多く，可読性が高い，コード量が少ないと定評があるため，mrubyは日本における複雑化する組込みソフトウェア開発の生産性向上に寄与する可能性がある．

本研究では，C言語と比較したmrubyの可読性，コード量，計算機資源の増加量について着目する．

Improving productivity in embedded software development using mruby language

† Tomoyuki Ikeda, Tokyo Denki University
‡ Hiroyasu Mitsui, Tokyo Denki University

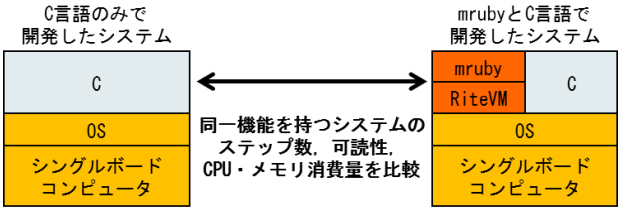

図 1 評価対象概要図

3 研究概要

本研究の目標は，C言語のソフトウェア資産がある上でのmrubyの有用性について評価することである．そこで，対象システムをC言語で構築したものと，一部処理をmrubyで開発してC言語とmrubyを協調利用したものを用意し，比較・評価する．評価対象項目は，ステップ数，可読性，実行時のCPU・メモリ消費量の3項目とする．

評価対象システムの構成図を図1に示す．コンピュータには高性能なSoCやGPIOポート，USB，イーサネットなどを備えた高機能なシングルボードコンピュータをLinux OSをインストールして用いる．評価対象となるプログラムはこのシングルボードコンピュータのOS上で実行する．mrubyはC言語プログラム中でmrb_state構造体を用意し，C言語プログラムから呼び出して実行する．

4 実装

評価対象であるシングルボードコンピュータの評価環境として，センサデータをサーバへ収集するM2Mシステムを構築する．M2Mデバイス（センサノード）にマイクロコンピュータ，ゲートウェイにシングルボードコンピュータ，M2Mサービス（サーバ）にPCを用いる．評価環境の概要図を図2に示す．

4.1 センサノード

センサノードではマイクロコンピュータを用い，温度センサから読取ったデータを温度に変換し，UART通信でシングルボードコンピュータに温度情報を送信する．

マイクロコンピュータにNXP mbed1768（NXP セミコ

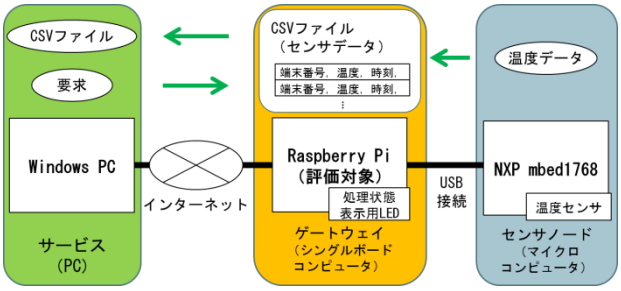

図 2 評価環境概要図

ンダクターズ社）（以下mbed）を用い，センサに温度センサを用いた．mbedとシングルボードコンピュータをUSBケーブルで接続し，温度データを送信するようにした．

4.2 ゲートウェイ

ゲートウェイではセンサノードから取得した温度情報をシングルボードコンピュータでcsvファイルにまとめ，サーバから要求があったときにcsvファイルを送信する．また，処理状態表示LEDをGPIOに接続し，制御を行う．

シングルボードコンピュータにRaspberry Pi Model B（ラズベリーパイ財団）（以下RasPi）を用い，OSにはRaspbian OSを用いた．データ処理は大きく分けて2つあり，①USBポートからデータを読取り，[mbedの識別番号，取得した温度，温度情報を取得した時刻，]を1行とするcsvファイルを生成してデータを追加する処理，②PCからファイル要求を示す文字列を受け取った際にcsvファイルをTCP/IPで送信する処理がある．さらに，それぞれの作業の処理状態表示LEDをRasPiのGPIOに2つ接続し，各作業中の場合はLEDを点灯するようにした．これら3つの処理を，C言語のみのプログラムと，mrubyとC言語による協調プログラムの2種類用意した．

4.3 M2Mサービス（PC）

PCではRasPiへ要求を送信し，ファイルを受け取る．

PCのOSにWinodws7 OSを用いる． RasPiとTCP/IP通信を行いファイル要求を表す文字列を送信し，csvファイルを受信するプログラムを用意した．

5 評価と考察

5.1 評価

評価対象であるゲートウェイ（RasPi）上で動作するC言語プログラムと，C言語とmrubyの協調プログラムの評価を行った．評価方法として，それぞれのプログラムのステップ数，可読性，CPU・メモリの使用量を比較した．センサノードから受信した文字列buf ="T:24.00¥n"から温度情報を取り出し，csvファイルに書き込むための文字列を用意する処理の各プログラムの評価を以下に示す．

各プログラムのステップ数を表1に示す．各プログラムの可読性の例として，C言語プログラムをリスト1，mrubyプログラムをリスト2に示す．ただし，アルゴリズムに直接かかわらない部分は省略した．可読性の例に挙げた文字列処理プログラム実行時のCPU・メモリ使用量を測定した．測定方法にはLinuxの実行中プロセスの情報を表示するps␣u コマンド（'␣'はスペース文字の意）をsystem関数でプログラム中から実行する手法を用いた．

5.2 考察

表1の結果から，C言語のみと比べてmrubyの導入でステップ数は減らずに増えた．これはmrubyには標準入出力などのライブラリが用意されていないため記述量が増えたことに加え，mruby化をうまく実現できなかったためだと

表 1 文字列処理の各プログラムのステップ数

	C言語のみ	mrubyとC言語の協調
ステップ数	25	28

リスト 1 C言語による文字列処理プログラム

```
// ※buf="T:24.00¥n¥0", time="Mon Jan 6 19:14:23 2014¥n¥0"
// ※serial="mbed001¥0", int i,j; , char csvBuf[64];
for(i=0; buf[i]!=':'; i++);
for(i=i++,j=0; buf[i]!='¥n'; i++,j++) buf [j] = buf [i];
buf[j] = '¥0';
sprintf(csvBuf, "%s,%s,%s,¥n",serial,time,buf);
```

リスト 2 mrubyによる文字列処理プログラム

```
# ※buf="T:24.00¥n", time="Mon Jan 6 19:14:23 2014"
# ※serial="mbed001"
csvBuf = serial+','+time+','+buf[2,5]+",¥n"
```

表 2 文字列処理時のCPU・メモリ使用率

	C言語使用時	mrubyとC言語使用時
CPU使用率(%)	0.0	0.0
仮想メモリ(KB)	2980	3692
物理メモリ(KB)	752	1320

考える．次に，リスト1とリスト2の結果から，mrubyの方が簡潔にまとまっている．アルゴリズムに直接関係のない記述が少なく関係性を捉えやすいため，mrubyは可読性が高いと言える．最後に表2の結果から，CPU使用率は0.0(%)とmrubyの導入による資源増加量を示す有効な数値を示すことができなかった．これは本研究に用いたコンピュータが実行プログラムに対して非常に高性能だったためだと考える．また，メモリの使用量では，mrubyの導入によって仮想メモリは124%増加，物理メモリは176%増加した．これはVMを使用していることによると考える．

以上から，mrubyの導入によって可読性が高まり，デバッグ時やソースコード資産の引継ぎ時に貢献すると考えられる．しかし，ライブラリの不在，言語習得のむずかしさ，開発環境や仕様がまだ不安定である点からみて現時点では実用性に劣り，これから一層の研究が必要である．

6 まとめと今後の予定

C言語によるシステムにmrubyを導入することができた．また，簡単なシステムにmrubyを導入することによる効果を評価し，mrubyの課題を示すことができた．mrubyは登場して間もなく情報が少ないため，mrubyの習得が難しく，また，自身で拡張しなければならないことが多いためRubyの仕組みについて詳しい必要があり，mrubyの習得や実装に時間をかけてしまう問題がある．実績を作り，mruby開発の環境を整えることが今後の課題である．

参考文献

[1] ガートナージャパン株式会社, “組込みシステム産業の施策立案に向けた実態把握のための調査研究,” 経済産業省, 2011.

[2] 経済産業省 商務情報政策局 情報処理振興課, “2010年版組込みソフトウェア産業実態調査報告書－事業責任者向け調査－,” 経済産業省, 2010.

[3] まつもとゆきひろ, まつもとゆきひろ直伝 組込向けRuby mrubyのすべて 総集編, 日経BP社, 2013.