

---

# 小型ハードウェアへのmruby アプリケーション適用試行

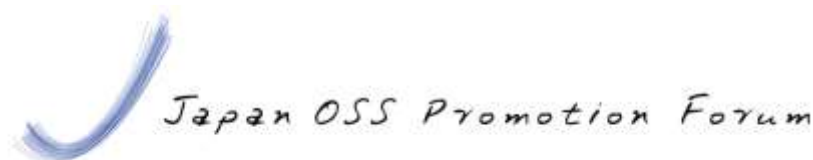
---

2017年2月15日

Japan OSS Promotion Forum

株式会社日立ソリューションズ 技術統括本部 IT技術推進センタ

三好 秀徳



# 目次

---

1. Ruby / mruby の紹介とアプリ部会とのかかわり
2. 実装したmrubyアプリケーションとシステムの紹介
3. ハードウェア・ソフトウェア詳細
4. mrubyアプリケーションの開発方法
5. 発生した問題とその解決
6. まとめ

# 目次

---

1. **Ruby / mruby の紹介とアプリ部会とのかかわり**
2. **実装したmrubyアプリケーションとシステムの紹介**
3. **ハードウェア・ソフトウェア詳細**
4. **mrubyアプリケーションの開発方法**
5. **発生した問題とその解決**
6. **まとめ**

- オープンソースで開発されているプログラミング言語
  - <https://www.ruby-lang.org/ja/documentation/repository-guide/>
  - BSDライセンス、Rubyライセンスのデュアルライセンス
- まつもとゆきひろ氏を中心に多くの日本人が開発
  - **日本語ドキュメントが多数存在**
  - Ruby関連コミュニティが日本各地に存在
- Webフレームワーク『Ruby on Rails』の実装言語
  - 一躍Rubyが人気言語に
  - 世界で使われているプログラミング言語 **20位以内**  
<http://www.tiobe.com/tiobe-index/>

- さまざまな言語でRuby処理系が開発されている
  - CRuby
    - ◆ C言語で開発されているRuby処理系
    - ◆ 一番使われているRuby処理系
  - JRuby
    - ◆ Java言語で開発されているRuby処理系
    - ◆ Javaアプリとの親和性が高い
  - Rubinius
    - ◆ Ruby言語で開発されているRuby処理系

- Rubyと同じ文法を持つ組み込み向けプログラミング言語
- オープンソースとして開発
  - <https://github.com/mruby/mruby>
- 処理系(CPUやOS)に非依存
- 既存C言語資産との互換性あり
  - mrubyからCアプリの呼び出し、C言語からmrubyアプリの呼び出しが可能
- 言語機能の追加だけでなく削除も自由に可能
  - 実行バイナリサイズや使用メモリの低減が可能

## ■ 適用事例多数

- IIJ社のインターネットルーター『SA-W1』
  - ◆文字列処理が得意な点を利用し、設定管理などの機能を容易に実装
- QPS研究所による小型人工衛星システム
  - ◆処理系(CPU、OS)に非依存な点を利用し、高汎用性を実現
- Webサーバの機能拡張『mod\_mruby』『ngx\_mruby』
  - ◆Rubyがもつ高い可読性を利用し、複雑な設定ファイルの記述を読みやすい形で表現
  - ◆C言語への組み込みが可能である点を利用し、高速な動作を実現
- <http://monoist.atmarkit.co.jp/mn/articles/1601/12/news016.html>

## ■ 『mrubyとCRubyとの性能比較』

- Japan OSS Promotion Forum 2015 富田昌宏氏(株式会社富士通)

<http://ossforum.jp/jossfiles/JOSSPF-2015-7-3.pdf>

## ■ 『IoTへのRuby/mruby適用試行』

- Japan OSS Promotion Forum 2016 廣田哲也氏(株式会社シーイーシー)

[http://ossforum.jp/jossfiles/05\\_01\\_application.pdf](http://ossforum.jp/jossfiles/05_01_application.pdf)

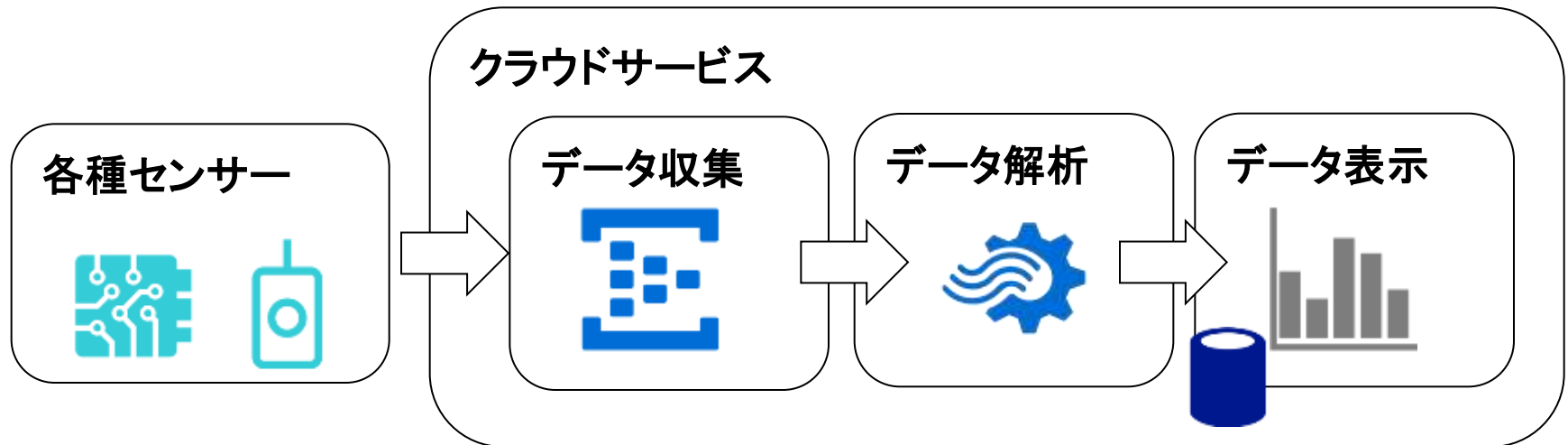


- 小型ハードウェア上で動くmrubyアプリを開発
- mrubyアプリ開発における知見を紹介
  - どのように開発を始めればよいのか
  - 実装したい機能の実現可能性
  - トラブルとその対処方法

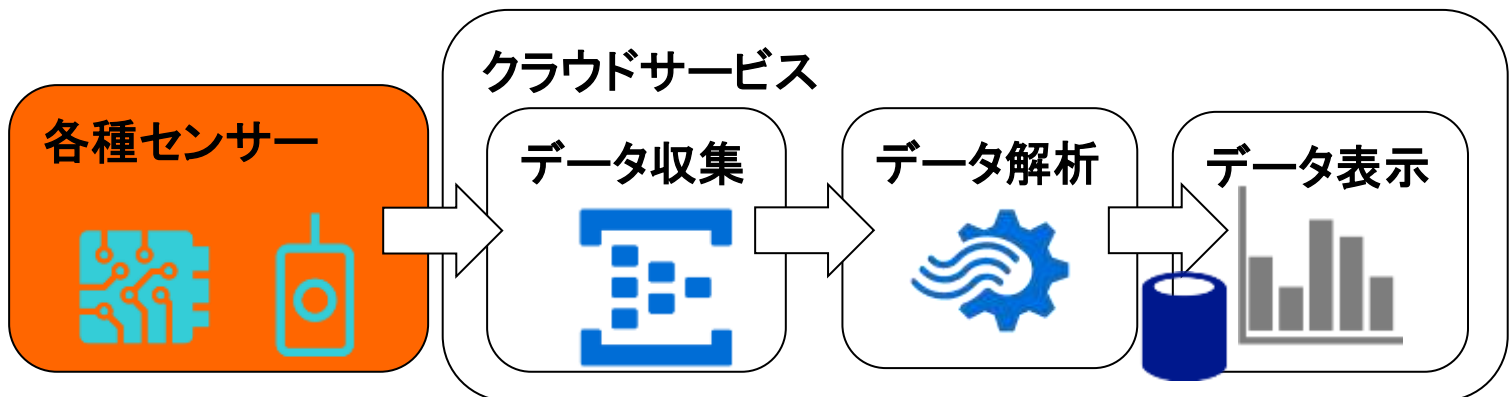
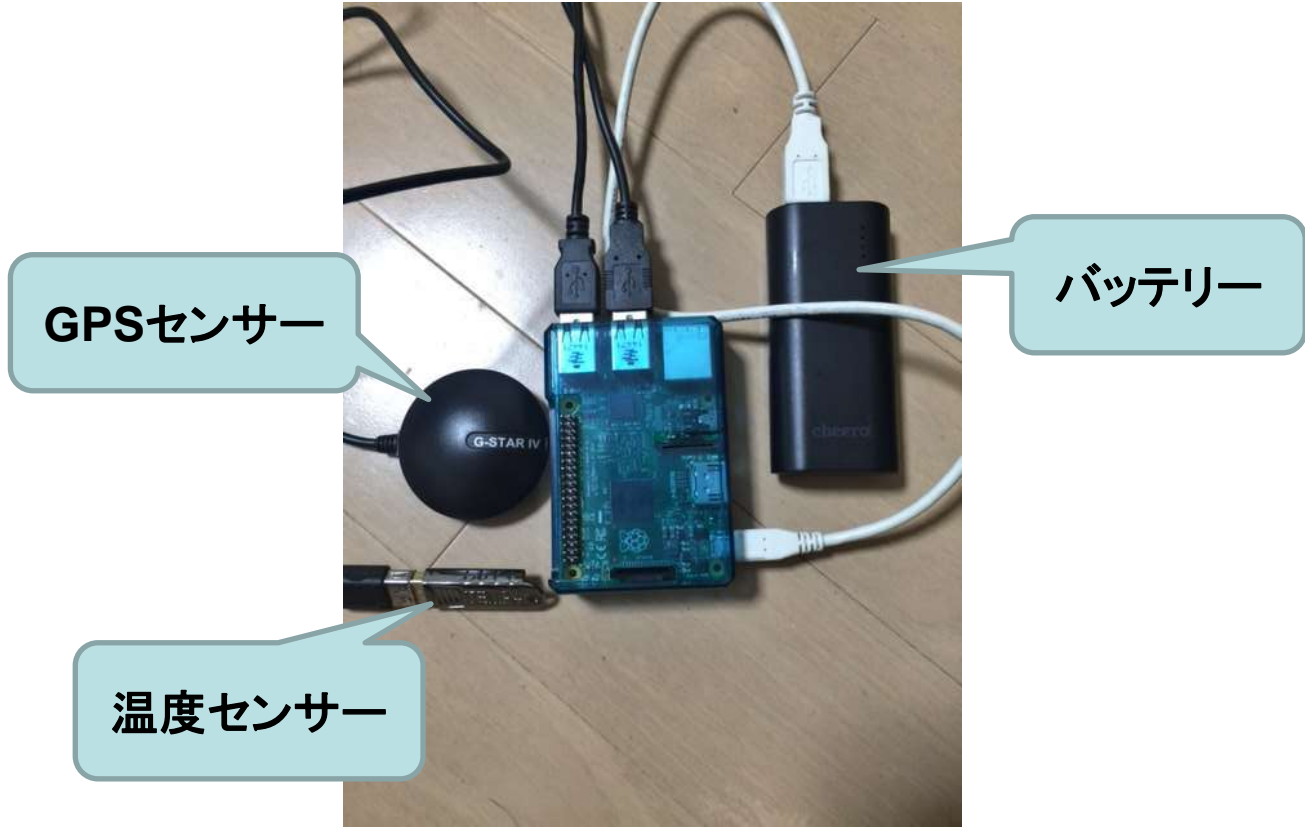
# 目次

1. Ruby / mruby の紹介とアプリ部会とのかかわり
2. **実装したmrubyアプリケーションとシステムの紹介**
3. ハードウェア・ソフトウェア詳細
4. mrubyアプリケーションの開発方法
5. 発生した問題とその解決
6. まとめ

- ランニング用リアルタイム位置情報発信アプリの開発
  - 位置情報(GPS)と気温を随時クラウドサービスに送信
  - 最低限4時間は動かすことを目標



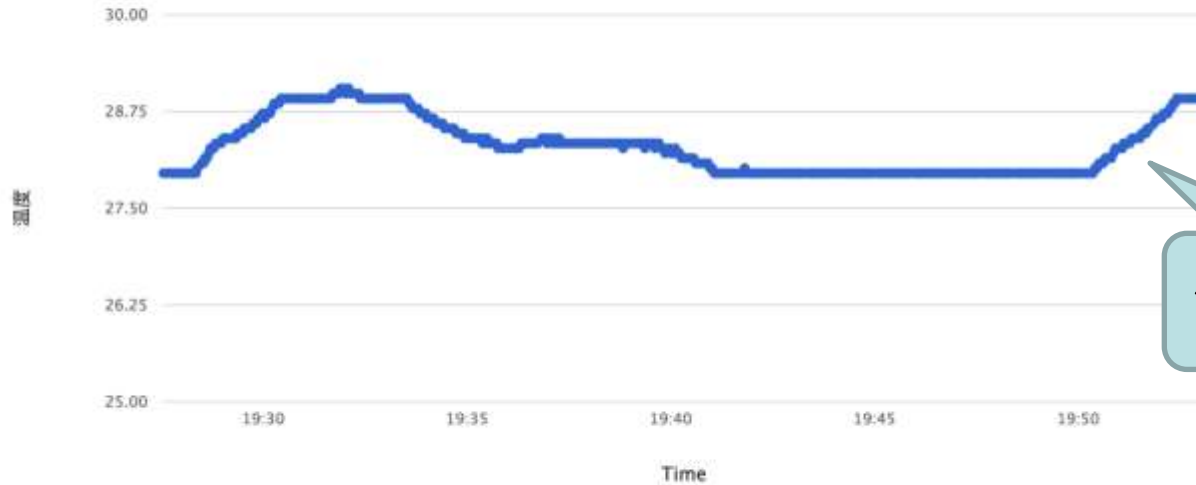
# 2-2.各種センサーの紹介



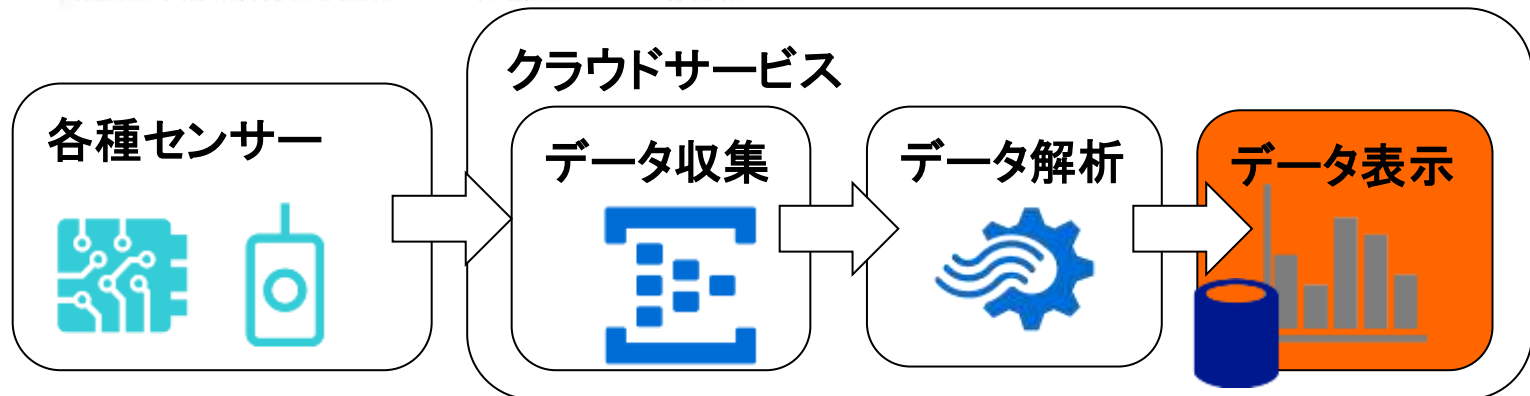
# 2-3. データの表示 (温度)

## Runlog RaspberryPi2 Download GPX file

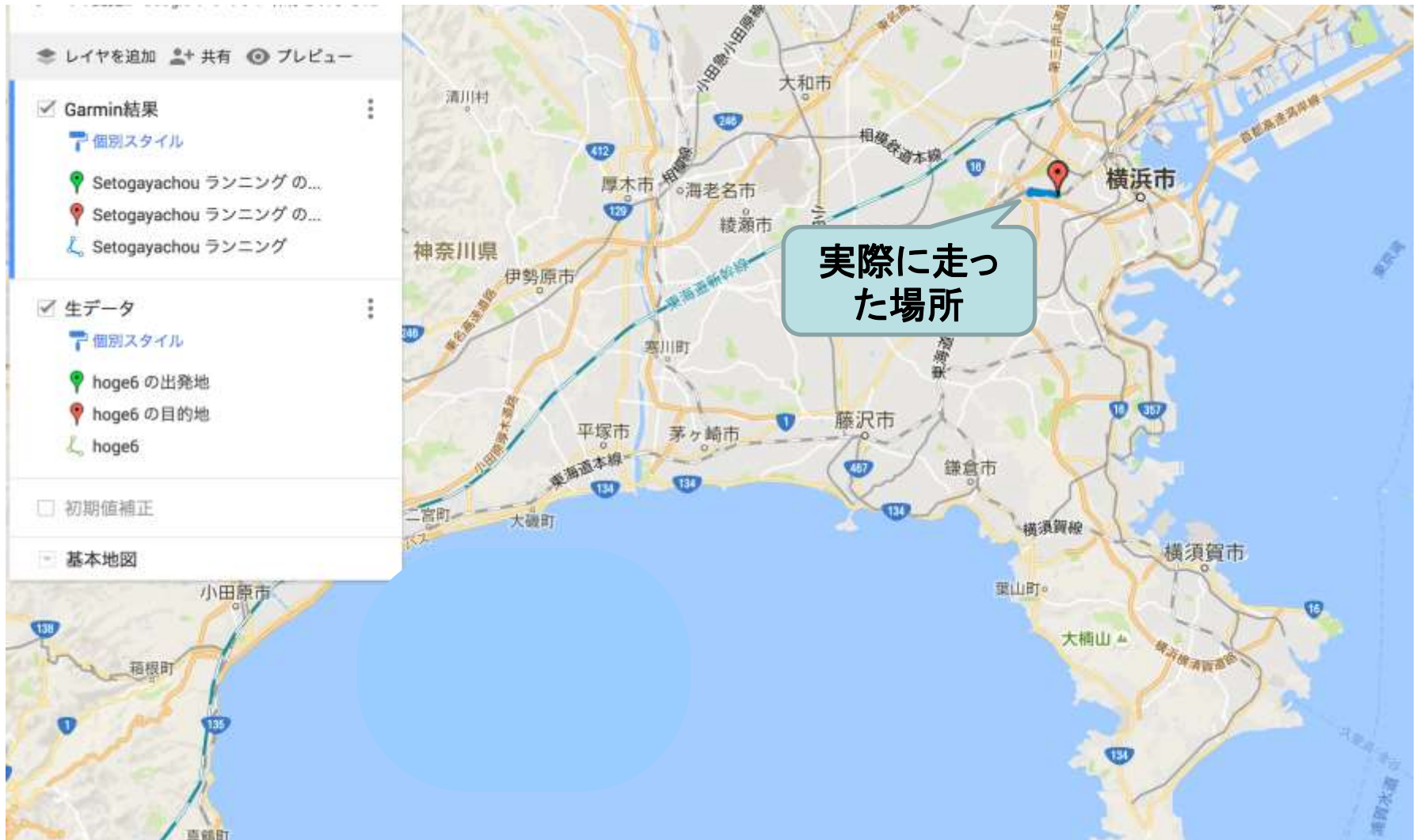
GPX download



温度の時系列  
変化



# 2-4.GPSデータを地図上にマッピング



# 2-5.GPSデータ(初期値を補正)



# 目次

---

1. Ruby / mruby の紹介とアプリ部会とのかかわり
2. 実装したmrubyアプリケーションとシステムの紹介
3. **ハードウェア・ソフトウェア詳細**
4. mrubyアプリケーションの開発方法
5. 発生した問題とその解決
6. まとめ



## ■ mrubyアプリの開発に集中できるようにする

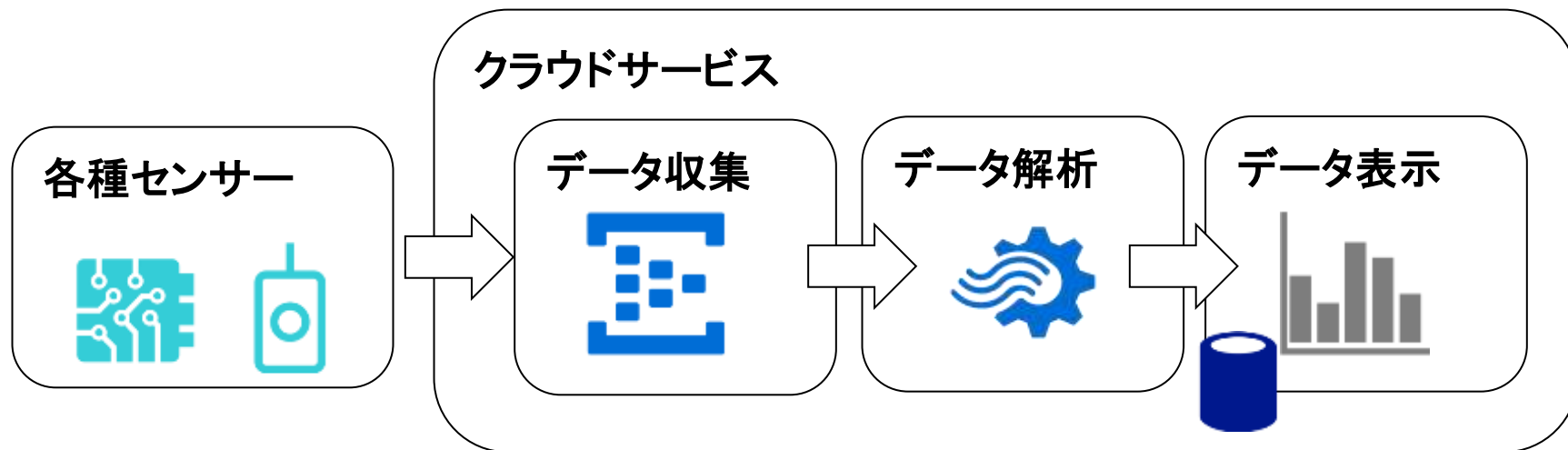
- 取り回しが容易なハードウェア

  - ◆ハンダ付け作業を行わずともセンサー類の取り付けが可能

  - ◆mrubyの動作実績あり

- 既存の基盤技術を活用する

  - ◆クライアントから送信されるデータの処理基盤はクラウドサービスを活用



## 3-2. 準備したハードウェア

### ■ Raspberry Pi 2

- サイズ : 85mm × 56mm × 17mm
- センサー類の接続がUSBやGPIOで実施できる
- 使い慣れたLinux OS (Raspbian) が使える



### ■ 温度センサー

- PCsensor.com TEMPper gold
- USBシリアル通信でデータの取得が可能



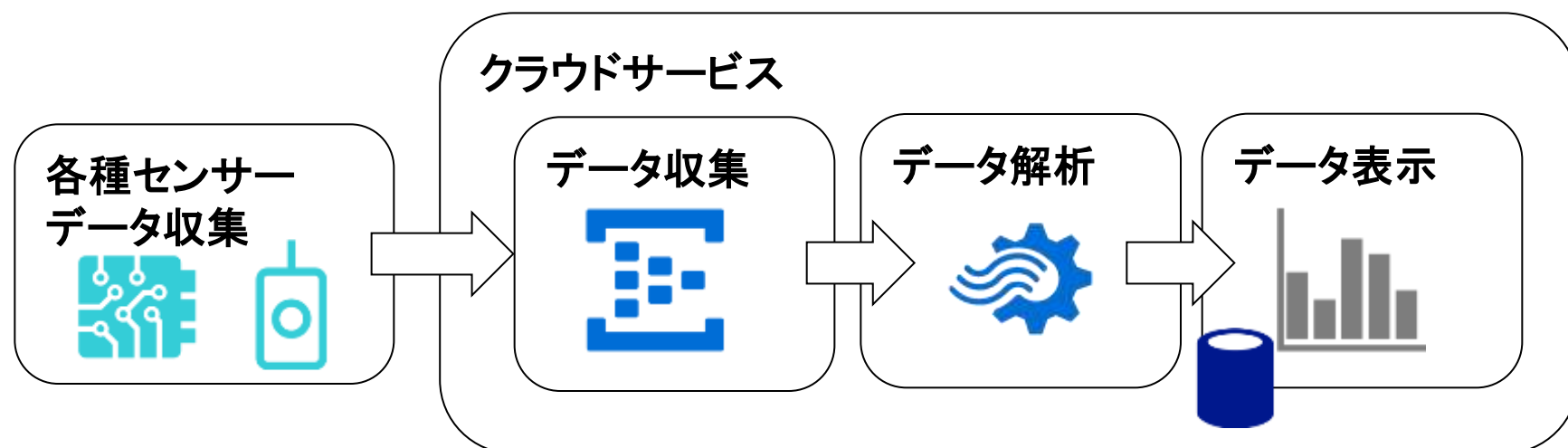
### ■ GPSレシーバー

- GLOBALSTAT BU-353 S4 GPSレシーバー
- USBシリアル通信でデータの取得が可能



# 3-3.ソフトウェア詳細

大分類	小分類	使用したもの
各種センサーデータ収集	OS	Raspbian
	言語	mruby 1.2.0
クラウドサービス	データ収集	Microsoft Azure IoT Hub
	データ分析	Microsoft Azure Stream Analytics
	データ表示	Microsoft Azure App Service
	データ表示 アプリ	JRuby 9.1.5.0 + Ruby on Rails 4.2.

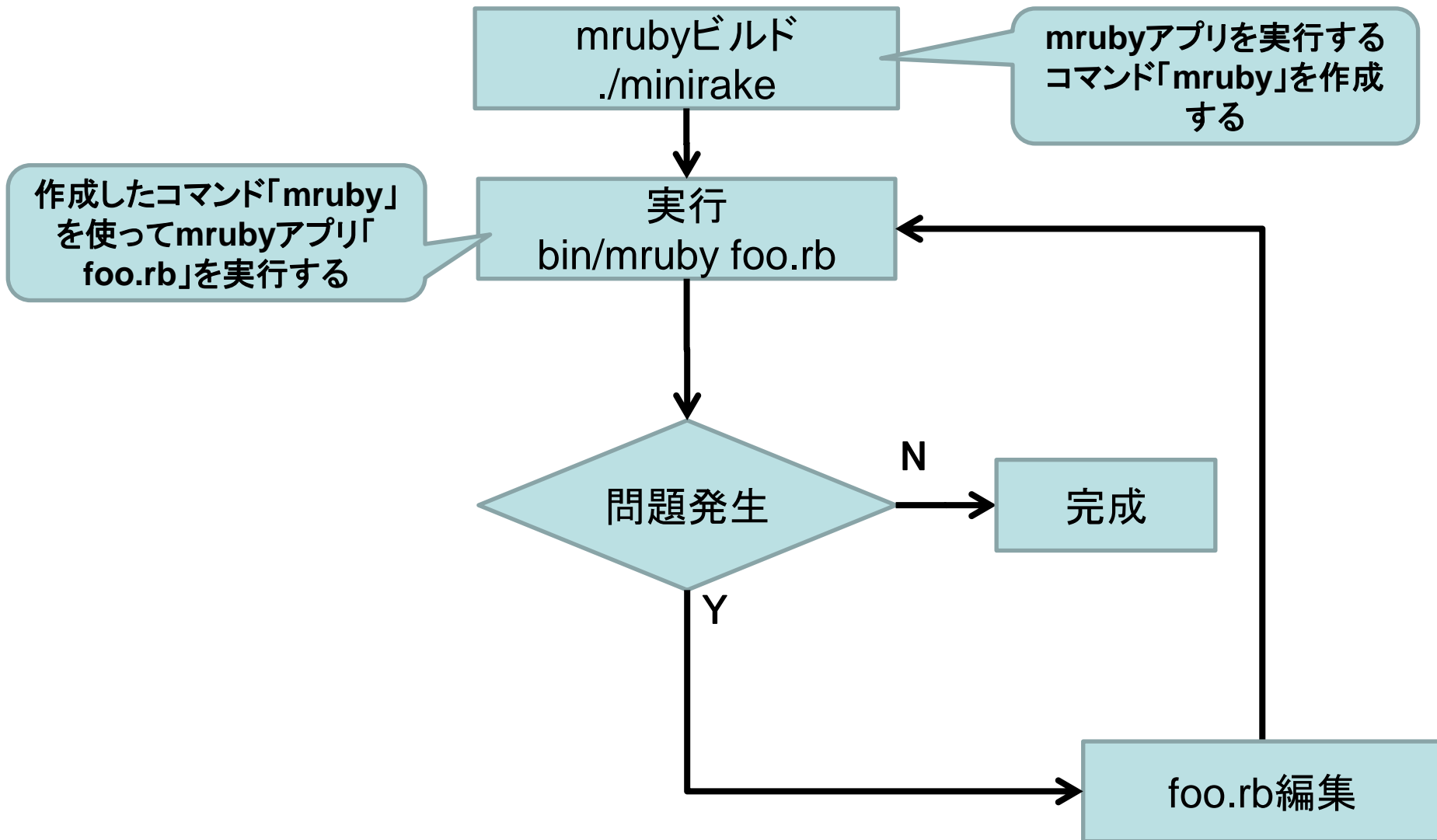


# 目次

---

1. Ruby / mruby の紹介とアプリ部会とのかかわり
2. 実装したmrubyアプリケーションとシステムの紹介
3. ハードウェア・ソフトウェア詳細
4. **mrubyアプリケーションの開発方法**
5. 発生した問題とその解決
6. まとめ

# 4-1.一般的なmrubyアプリ 開発の流れ



- 最初にCRubyで実装。その後、mrubyで実行する
  - 文法はCRuby/mrubyで差異なし
    - ◆ 基本的な機能はCRuby/mrubyともに有しているだろうという思い込み
  - システムの構築に必要な機能を洗い出す
    - ◆ HTTPS方式でのデータ送信
    - ◆ BASE64エンコード化
    - ◆ シリアルポートの読み込み
- 極力CRuby用の外部ライブラリ(gem)は使わない
  - 今回はシリアルポートの読み込みのみgemを使用
  - その他の部分はCRubyに付属のライブラリで実装できた

## 4-3.いざ実行！

```
[~/mruby-1.2.0]$ head -3 runinfo_sender.rb
DEVICE_NAME = ENV["DEVICE_NAME"]
SHARED_ACCESS_KEY = ENV["SHARED_ACCESS_KEY"]
HOST_NAME = ENV["HOST_NAME"]
[~/mruby-1.2.0]$ bin/mruby runinfo_sender.rb
```

## 4-3.いざ実行！

```
[~/mruby-1.2.0]$ head -3 runinfo_sender.rb
DEVICE_NAME = ENV["DEVICE_NAME"]
SHARED_ACCESS_KEY = ENV["SHARED_ACCESS_KEY"]
HOST_NAME = ENV["HOST_NAME"]
[~/mruby-1.2.0]$ bin/mruby runinfo_sender.rb
runinfo_sender.rb:1: uninitialized constant ENV
(NameError)
[~/work/mruby-1.2.0]$
```

gemを使わないCRubyスクリプトがmrubyでは動かない



- mrubyで実行したときに動かなかった箇所
  - 環境変数の取得
  - 日時データのフォーマット化
  - HTTPS方式でのデータ送信
  - BASE64エンコード化
  - etc...
- システムの構築に必要な機能として洗い出した機能全部

- CRuby添付の各種ライブラリがmrubyでは添付されていない
  - mrubyは本体を小さくし、さまざまな機能は外部ライブラリで実現する方針
  - mruby用の**外部ライブラリである“mrbgems”**を探す
  - 探してきたmrbgemsを設定ファイル(build\_config.rb)に記述してmrubyを再度作成

```
MRuby::Build.new do |conf|  
  # Use mrbgems  
  # include the default GEMs  
  conf.gembox 'default'  
  # 以下省略  
end
```

build\_config.rb

# 4-6.mrbgemsの探し方

## ■ 情報元は主に以下URLから

- <https://github.com/mruby/mruby/wiki/Related-Projects>

- ◆ mruby GitHubリポジトリのWikiページ

- <http://forum.mruby.org/download/index4.html>

- ◆ 軽量Rubyフォーラム mruby 1.2.0 リリースアナウンスページ

- ◆ Windows/Mac/Linuxの動作検証結果が掲載

特定非営利活動法人  
mruby 軽量Rubyフォーラム specified non-profit corporation mruby Forum

Home | about Forum | Join Forum | Seminar/Event | Members | Download | Docs

**Download Stable v1.2.0**

Satbel v 1.0 download (Jan. 2014) | Satbel v 1.1 download (Nov. 2014)

**mruby v1.2.0 リリース**

\* mruby Stable版V1.2.0 (2015.11.16) はこちらからダウンロード出来ます。  
リリースノートは[こちらから](#)

Linux/Ma/Windows安定版で評価済外部mrbgemsリスト

mgem name	Description	Mac	Ubuntu	Windows		
				Visual Basic	msys	cygwin
mruby-bcrypt	OpenBSD-style Blowfish-based password hashing.	○	○			
mruby-pjson	pure mruby JSON parser	○	○	○	○	○
mruby-md5	MD5 digest function	○	○	○	○	○
mruby-io		○	○			
mruby-kmp	IO and File class for mruby	○	○	○	○	○
mruby-updategems	update all git based mrbgems	○	○	○	○	○
mruby-ipaddr	IPAddr class for mruby	○	○			
mruby-bin-mirb-hostbased	Hostbased mirb for serial communication.	○	○			

- クロスコンパイルを行うときは互換性に注意
  - 開発環境と実行環境が異なるときに問題が顕在化
    - ◆ 開発環境がMac、実行環境がRaspberry Pi
- ライブラリの使い方がCRubyのものとは違う
  - メソッド名、引数、戻り値、定数
  - ドキュメントが充実しているもの、テストコードが存在するものを選ぶ

## ■ 探してきたmrbgemsを取り込む

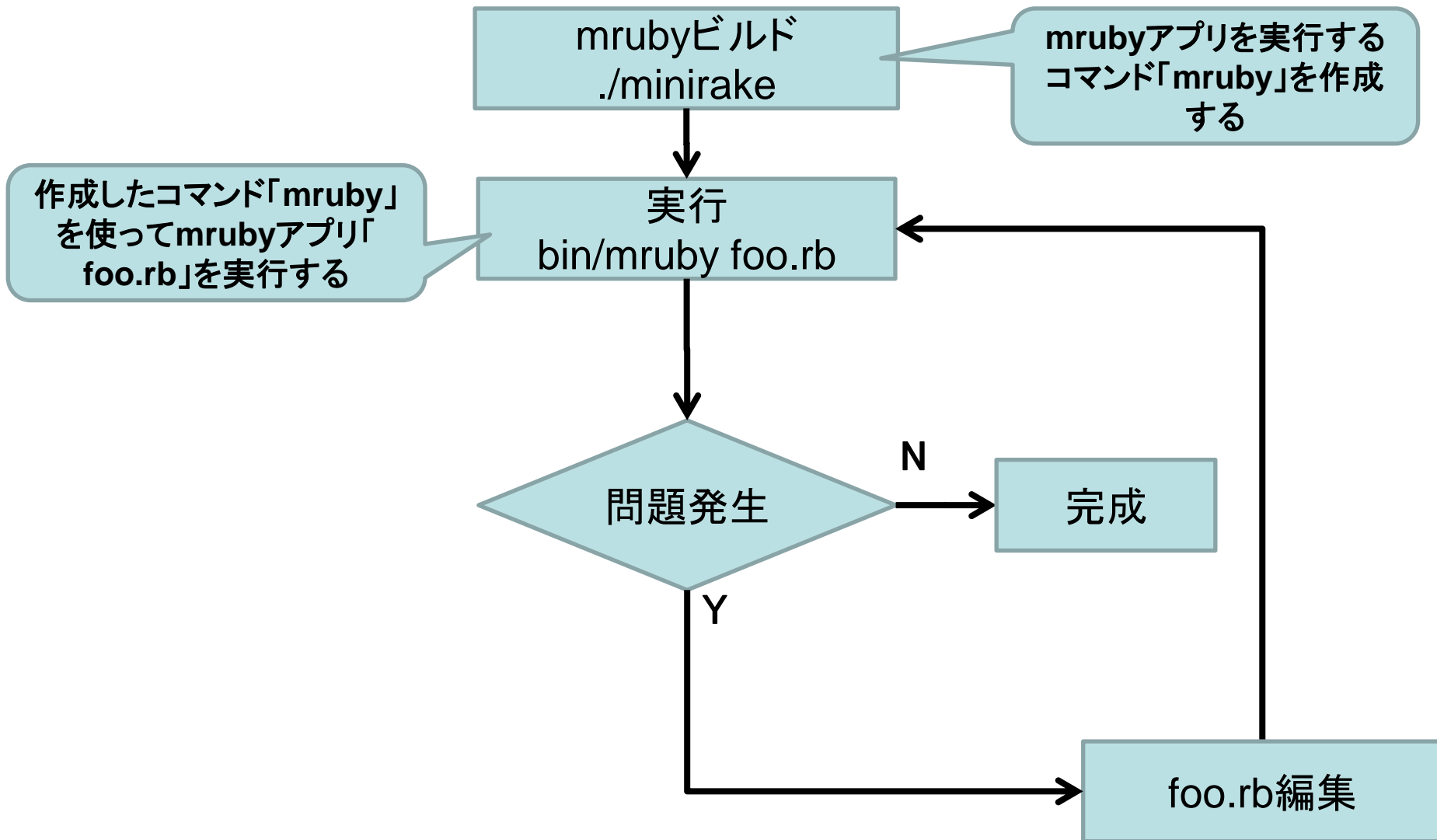
- mrubyはmrbgemsを追加・変更・削除した場合はmruby自身の再構築が必要
- 探してきたmrbgemsを設定ファイル(build\_config.rb)に記述する
- minirakeコマンドでmrubyをリビルドする

```
MRuby::Build.new do |conf|  
  # Use mrbgems  
  # include the default GEMS  
  conf.gembox 'default'  
  conf.gem github: 'mattn/mruby-json'  
  conf.gem github: 'mattn/mruby-base64'  
  conf.gem github: 'mattn/mruby-http'  
  conf.gem github: 'iij/mruby-digest'  
  # 以下省略  
end
```

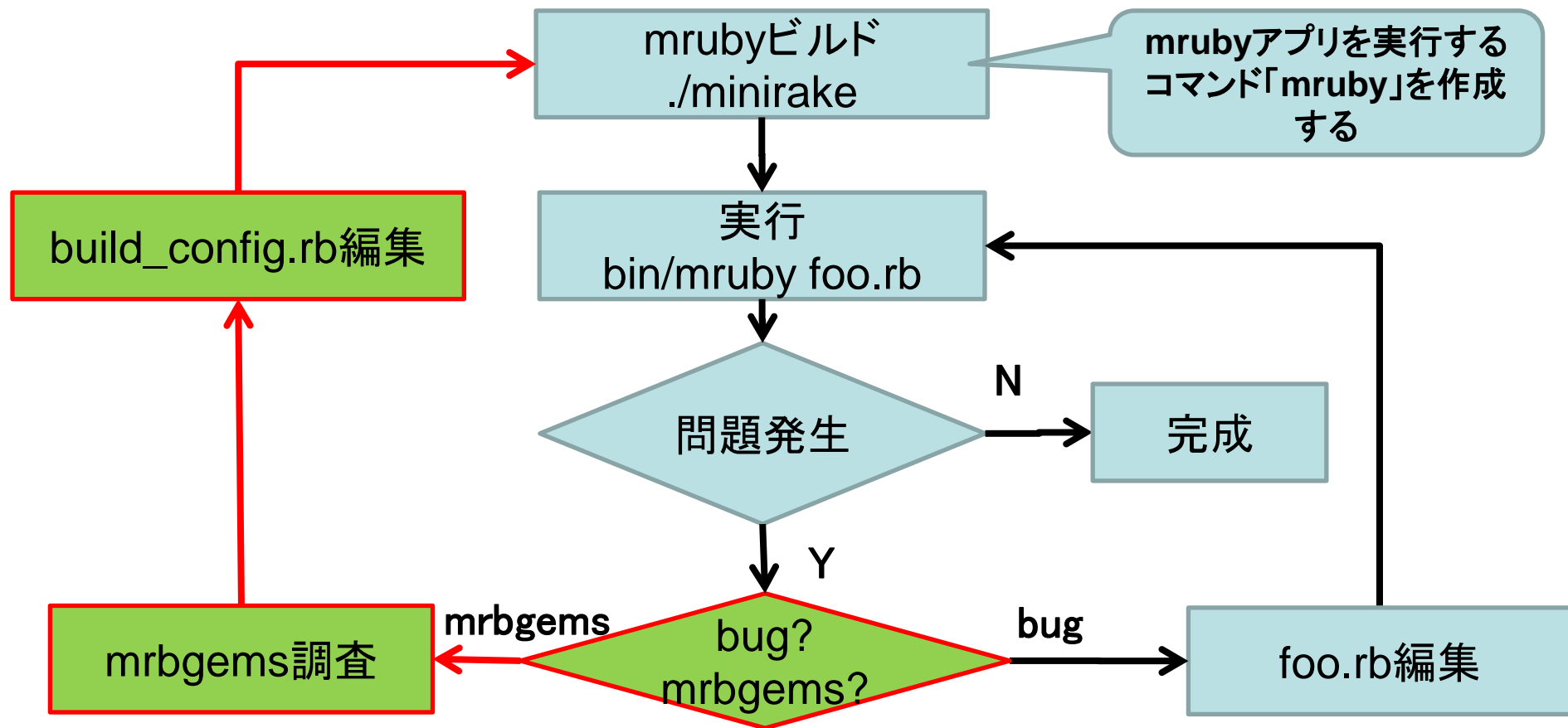
build\_config.rb

必要なmrbgemsを  
追記

# 4-9. 一般的なmrubyアプリ 開発の流れ(復習)



# 4-10.mrbgemsを使った場合の開発の流れ



# 目次

---

1. Ruby / mruby の紹介とアプリ部会とのかかわり
2. 実装したmrubyアプリケーションとシステムの紹介
3. ハードウェア・ソフトウェア詳細
4. mrubyアプリケーションの開発方法
5. 発生した問題とその解決
6. まとめ



- 1時間ほど経過すると、mrubyアプリが異常終了する
  - 目標稼働時間(4時間)もたない
- さまざまなログを取って検証を実施

```
{"Time": "2016/9/15 23:44:24", "Mem": 236464, ... }  
{"Time": "2016/9/15 23:44:25", "Mem": 236496, ... }  
{"Time": "2016/9/15 23:44:26", "Mem": 236528, ... }
```

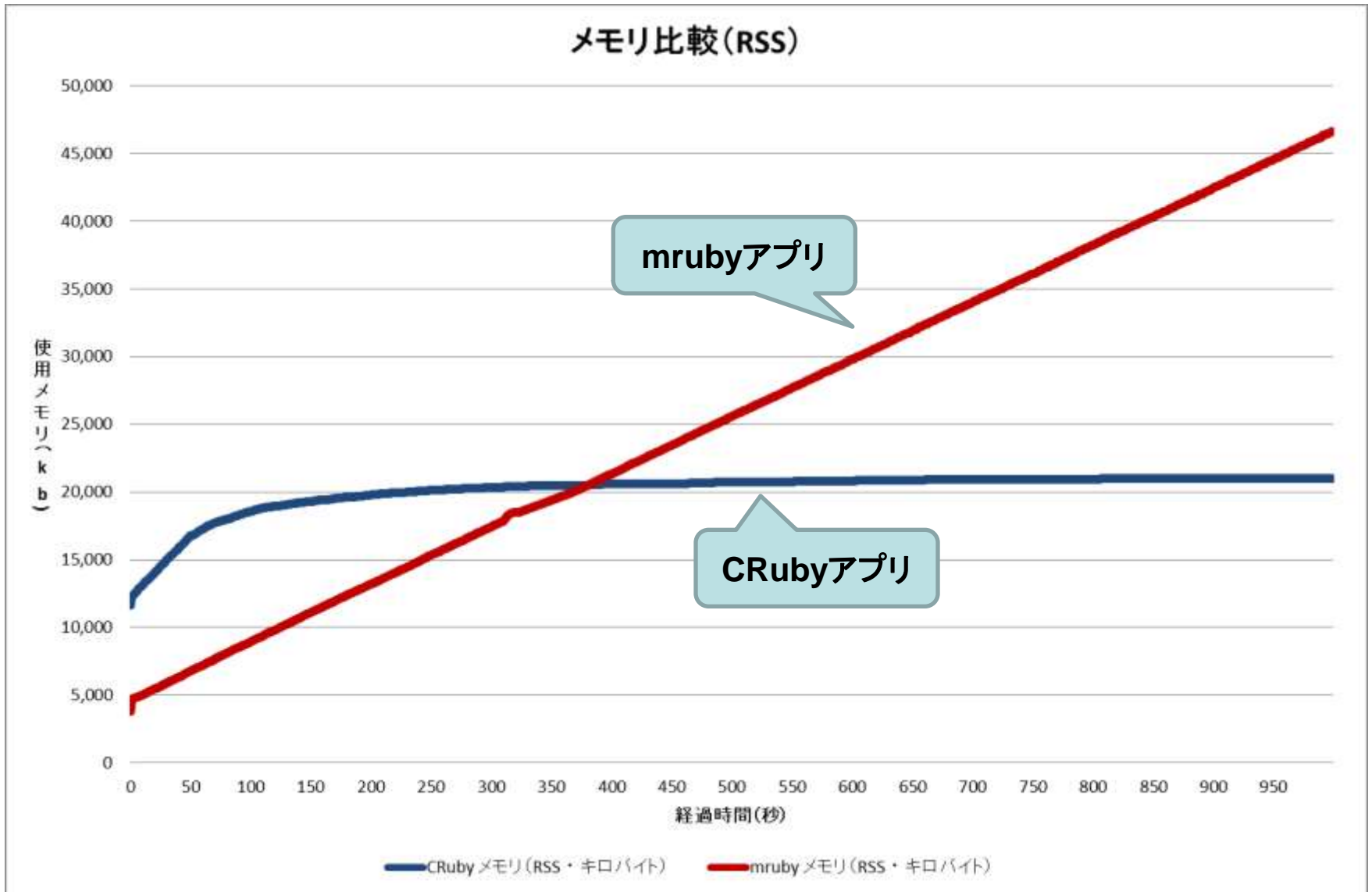
trace:

```
[5] /home/dev1/work/mruby-1.2.0/build/mrbgems/mruby-  
io/mrblib/io.rb:33:in IO#open  
(中略)
```

```
[0] runinfo.rb:37  
/home/dev1/work/mruby-1.2.0/build/mrbgems/mruby-  
io/mrblib/io.rb:33: pipe_open failed. (RuntimeError)
```

使用メモリが単調増加している？

# 5-2. 使用メモリ量の変化



- 無限ループでHTTP送信処理を実施している箇所が怪しい
  - 多くのライブラリが依存し、根本解決は困難

```
##(前略)
```

```
http = HttpRequest.new
```

```
loop do ## 無限ループで処理させる
```

```
# (中略) 送信するデータ(payload)の作成処理など
```

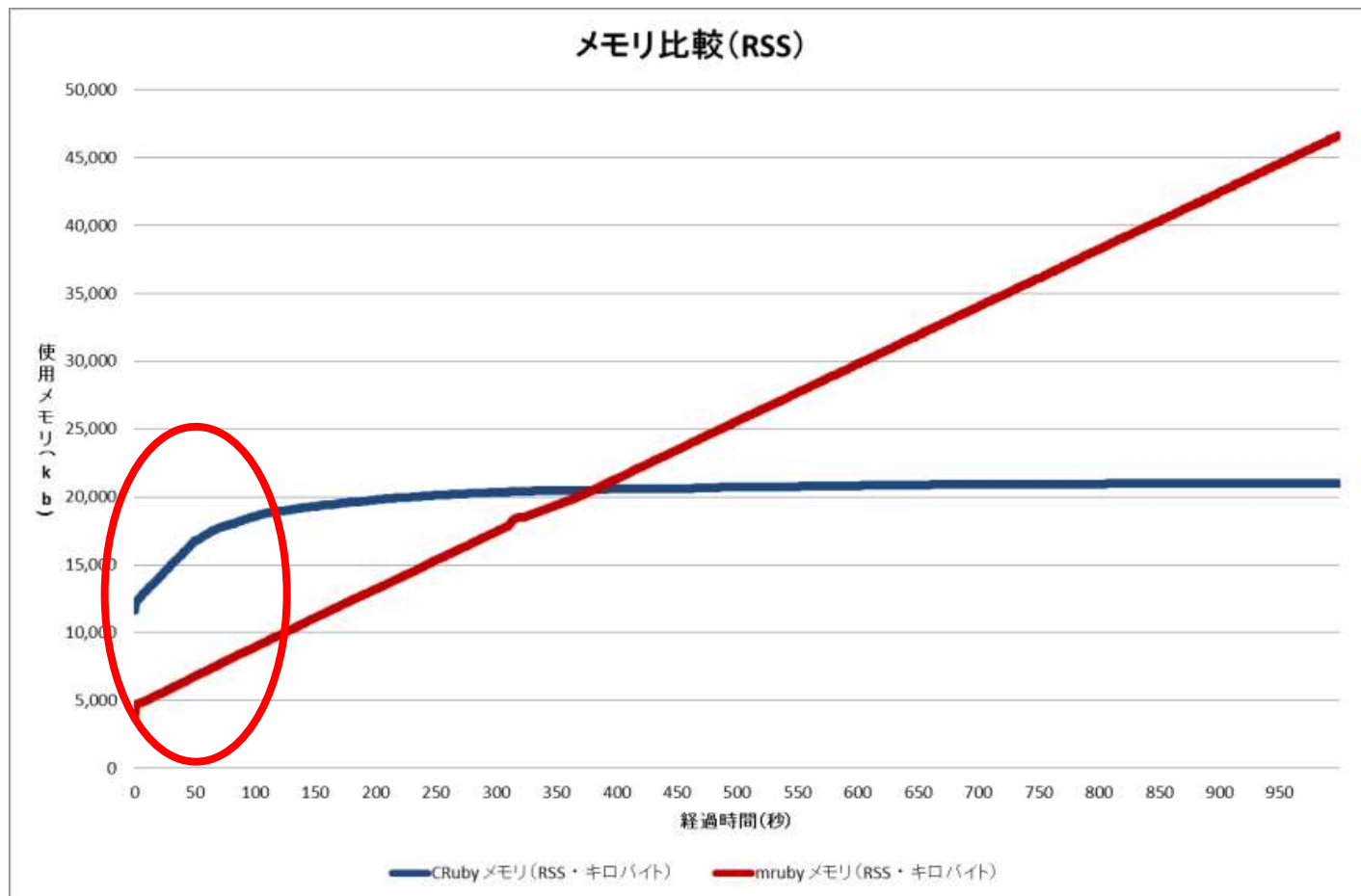
```
## Azure IoT Hubにデータ送信
```

```
http.post(uri_with_port, payload, {  
  "Content-Type" => "application/json",  
  "Content-Length" => payload.length.to_s,  
  "Authorization" => sas_header  
})
```

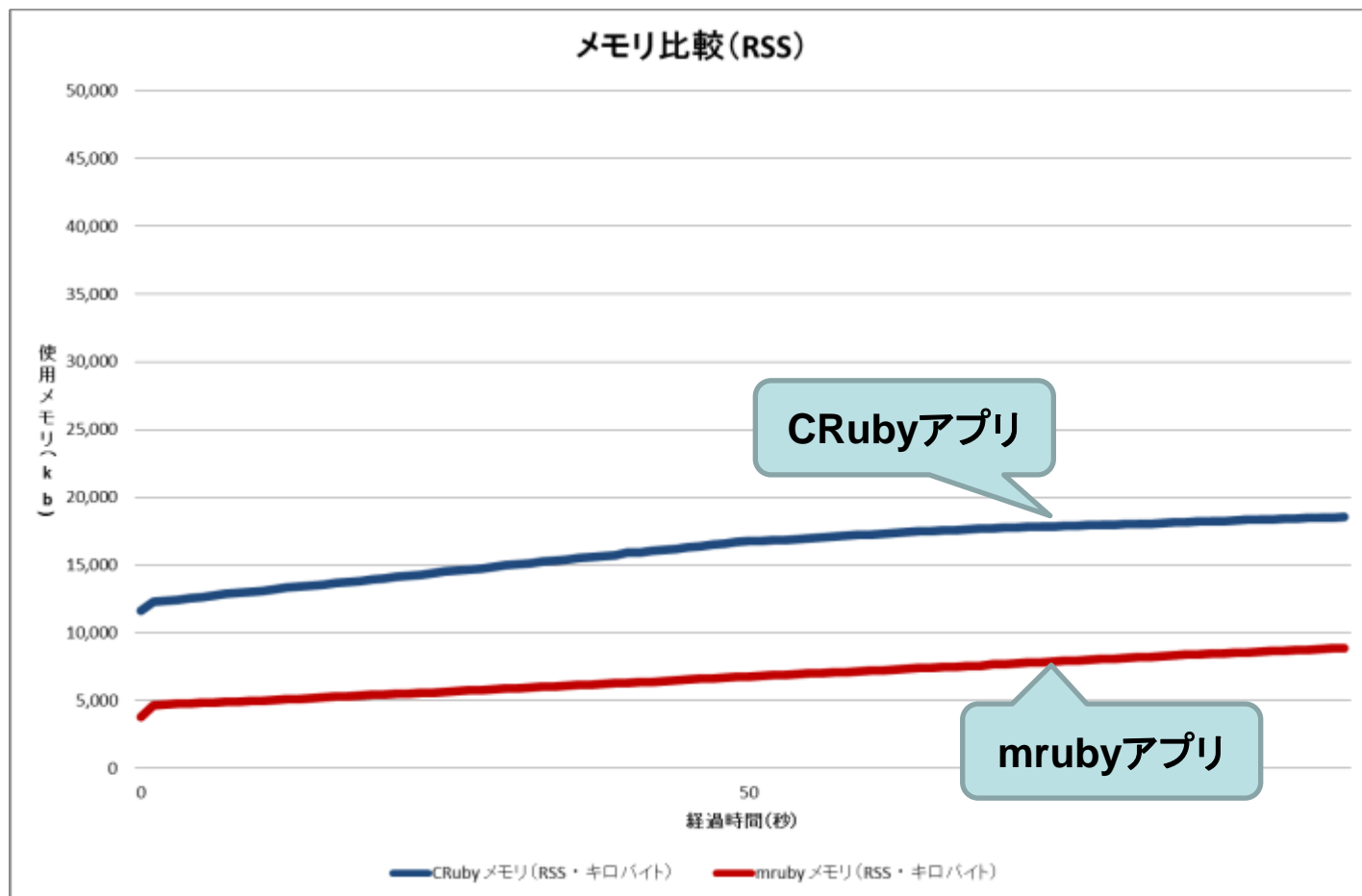
```
Sleep::sleep(1) ## 間隔をあける
```

```
end
```

- mrubyアプリで無限ループすることをやめる
  - 根拠1: ループ回数が少ないときは、使用メモリは少ない

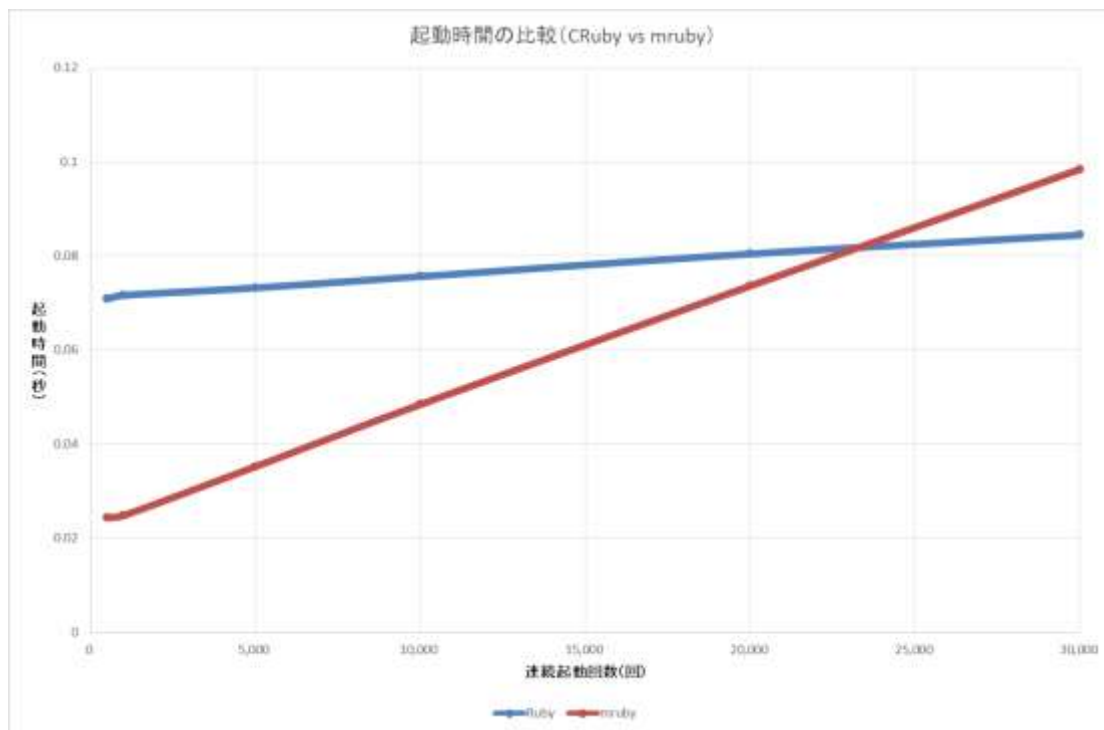


- mrubyアプリで無限ループすることをやめる
  - 根拠1: ループ回数が少ないときは、使用メモリは少ない



## ■ mrubyアプリで無限ループすることをやめる

- 根拠 1: ループ回数が少ないときは、使用メモリは非常に少ない
  - 根拠 2: mrubyアプリの起動はCRubyアプリよりも早く起動する
- ◆ cf. mruby/c in TokyoRubyKaigi#11 (田中和明氏)



## 5-5. 実装例

```
#!/bin/sh
```

実行するシェル

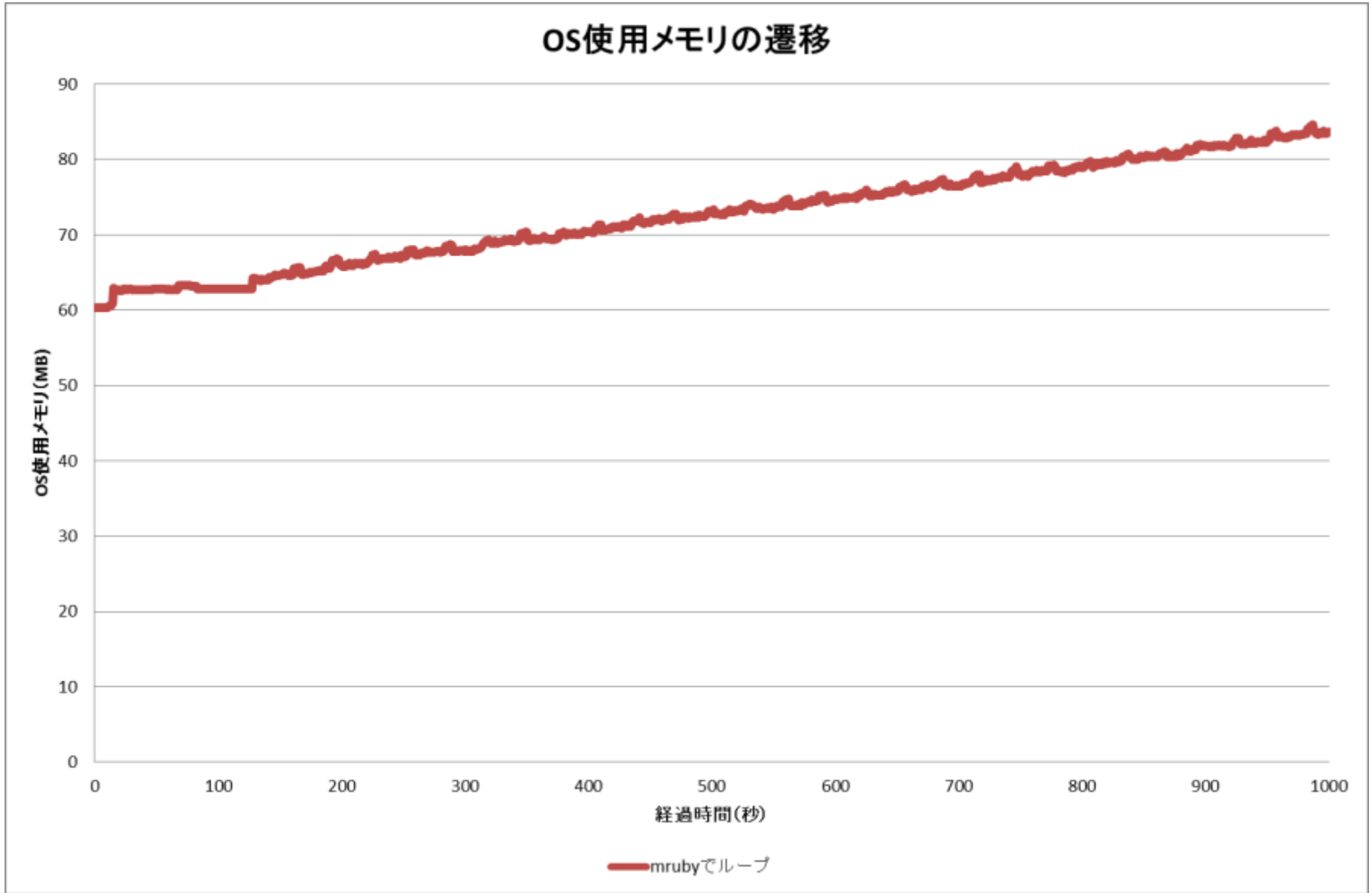
```
while :  
do  
  ./bin/mruby run_log.rb  
  sleep 1  
done
```

```
##(前略)
```

run\_log.rb

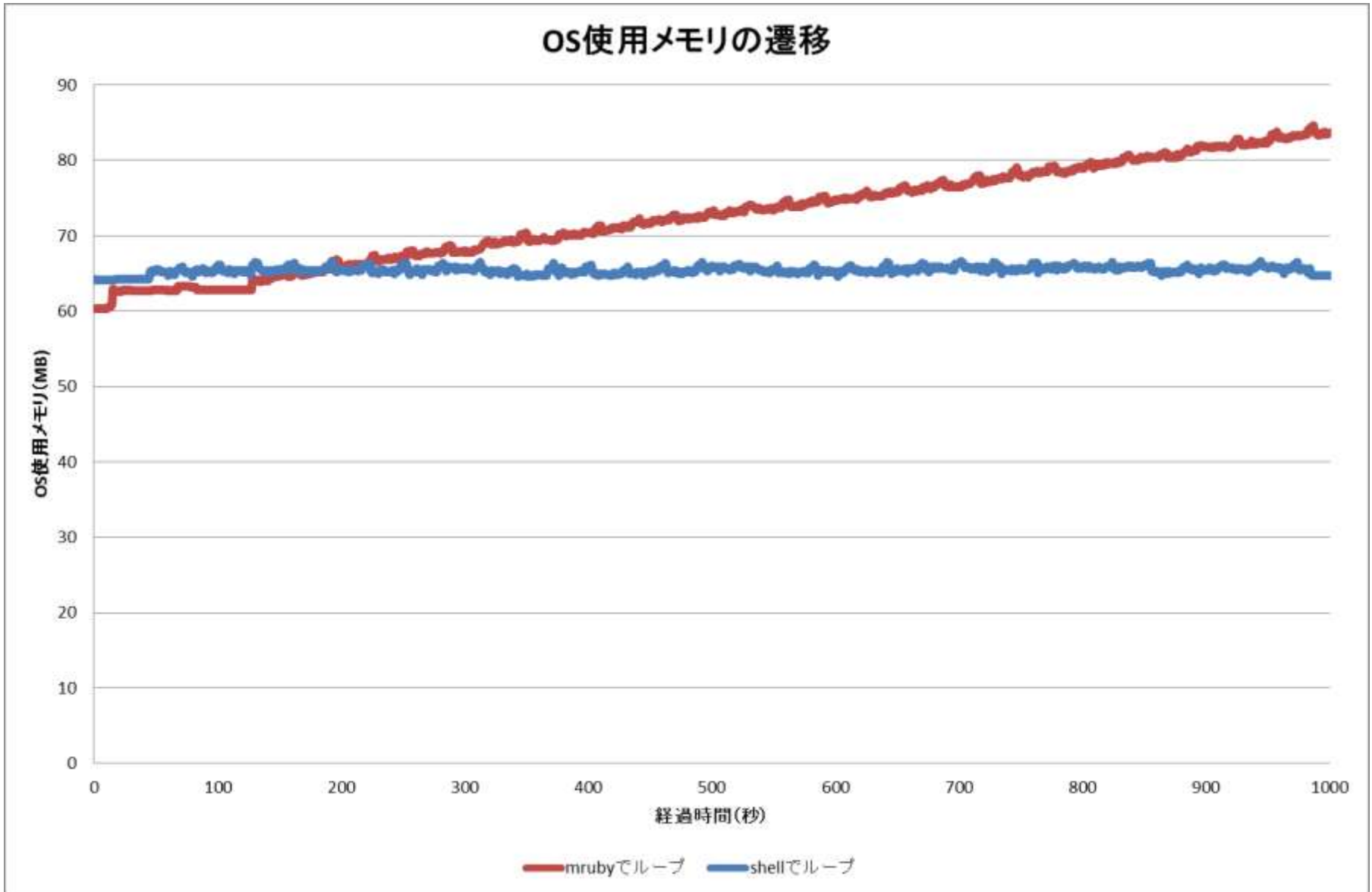
```
http = HttpRequest.new  
# loop do ## シェルで実装  
# (中略) 送信するデータ(payload)の作成処理など  
## Azure IoT Hubにデータ送信  
http.post(uri_with_port, payload, {  
  "Content-Type" => "application/json",  
  "Content-Length" => payload.length.to_s,  
  "Authorization" => sas_header  
})  
# Sleep::sleep(1) ## シェルで実装  
end
```

# 5-6.実行結果(対策前)





# 5-7.実行結果(対策後)



# 目次

---

1. Ruby / mruby の紹介とアプリ部会とのかかわり
2. 実装したmrubyアプリケーションとシステムの紹介
3. ハードウェア・ソフトウェア詳細
4. mrubyアプリケーションの開発方法
5. 発生した問題とその解決
6. まとめ

- 日本OSS推進フォーラムとしてmrubyアプリを作成
  - 我々が行ったmrubyアプリの作り方を紹介
  - mrbgemsの探し方・選び方を紹介
- mrbgemsの充実により、できることが広がっている
  - mrbgemsのドキュメントは不足気味
- mrubyアプリに使用メモリが単調増加する問題が発生
  - mrbgemsを含めた根本原因の解決には至らず
  - mrubyの特徴を生かして、問題を(とりあえず)解決

### ■ 機器が大きい

- 測定機器が大きくて走りにくい。バッテリーもぎりぎり。
- 小型化に向けた対策が必要
  - ◆ Raspberry Piからの脱却(例えばESP-WROOM-02とか)
  - ◆ よりコンパクトなmrubyの実装、mruby/cの利用  
<http://www.s-itoc.jp/activity/research/mruby/>
  - ◆ クロスコンパイルを用いた開発効率向上



小型マイコン付きWiFiモジュール  
ESP-WROOM-02



Ruby on Railsは、David Heinemeier Hansson氏の米国およびその他の国における商標または登録商標です。  
Windows、Microsoft Azureは米国Microsoft Corporationの米国およびその他の国における商標または登録商標です。  
Macは、Apple Inc.の米国およびその他の国における商標または登録商標です。  
LinuxはLinus Torvalds氏の米国、日本およびその他の国における登録商標または商標です。  
Raspberry Piは、英国Raspberry Pi財団の米国およびその他の国における商標または登録商標です。  
Cortexは、ARM Limitedの米国およびその他の国における商標または登録商標です。  
その他、記載されている会社名、商品名は、各社の登録商標または商標です。