軽量 Ruby の開発と評価

田　中　和　明[†1]

近年，組込み機器の複雑化と大規模化に伴い，システム開発のコストが増加している．しかも，開発コストに占めるソフトウェア開発費の割合は 60%を超えると言われる．スクリプト言語である Ruby は，開発のしやすさから Web アプリケーション開発で広く採用されている．我々は Ruby を組込みシステムの開発にも適用するプロジェクトを進めている．Ruby を組込システムのソフトウェア開発に利用しようとすると，使用するメモリが大きく，ガーベージコレクションによりリアルタイム処理が困難であることが課題となる．我々のプロジェクトで軽量 Ruby を開発し，これらの問題の解決を図った．

Implementation and Evaluation of Lightweight Ruby

Kazuaki Tanaka[†1]

Recently, due to the complexity and large-scale embedded devices, the cost of system development has been increasing. Moreover, the ratio of software development costs in the total development cost is greater than 60

Ruby is a scripting language, which is widely used in Web application development. We are promoting the project to develop the Ruby suitable for embedded systems.

There are two challenges of embedded system software development in Ruby. The first is the problem of large memory to be used by Ruby. Next, real-time processing is difficult because of the garbage collection. Our project has solved the problem by developing a lightweight Ruby. Lightweight Ruby is composed of a Ruby compiler and Ruby VM.

1. はじめに

組込みシステムのソフトウェア開発をスムーズに行うことは，システムのコスト削減だけでなく，品質と高機能化にも関連する重要な事項である．組込みソフトウェア開発では，90%以上が C/C++による開発となっているが，C/C++は開発コストが高いと言われており，特に大規模なソフトウェアにおいてはそのコスト増加が顕著となる．

スクリプト言語である Ruby は，開発のしやすさから Web アプリケーションで幅広く利用されており，開発期間が短いことから，スタートアップ企業では標準的な開発言語となっている．

Ruby の持つ開発しやすさという特徴を，組込みソフトウェアの開発で利用する．Ruby はインタープリタ型言語であり，その実行には多くのメモリを必要とする．また，実行時にガーベージコレクションによりメモリを管理するため，リアルタイム処理においてリアルタイム性の確保が困難である．

軽量 Ruby（Ruby コンパイラと Ruby VM）を開発することで，これらの問題の解決を図った．

2. 軽量 Ruby

Ruby はインタープリタ型のプログラミング言語である．我々が開発した軽量 Ruby は，少ないメモリでプログラムを実行できるように実装した Ruby 処理系である．

実行時に必要となるメモリを減らすため，Ruby コンパイラにより中間コードを生成する．中間コードは Ruby VM により逐次実行される．Ruby VM はコンパクトな構成であり，多くの組込みデバイス上で動作するよう，デバイスに依存する機能について柔軟に構成を変更できる．

2.1 軽量 Ruby の仕様

軽量 Ruby の**軽量**という表現は，軽量 Ruby の言語仕様にも反映している．現在 Web アプリケーション開発等で使用されている Ruby は多くのライブラリで構成されている．この豊富なライブラリが開発者の負担を軽減し，結果として開発期間の短縮に寄与している．

†1 九州工業大学
Kyushu Institute of Technology

しかし，組込みソフトウェアでは必要のないライブラリも多く，デバイスによってはハードウェアの仕様上，機能が意味を持たないこともある．そこで，プログラム言語 Ruby を満たす最低限の仕様を対象とした．具体的には，JIS X 3017 および ISO/IEC 30170 で策定されている仕様を採用した．

2.2 軽量 Ruby の特徴

軽量 Ruby は，Ruby コンパイラと Ruby VM に分かれ，実行時には Ruby VM のみが必要である．Ruby VM は 2MB 以下のメモリで十分に動作することを検証しており，多くの組込みシステムにおいて利用できる[⋆1]

コンパイル後のオブジェクトプログラム（または中間コード）は，静的型付けによる実装が一般的である．軽量 Ruby は，コンパイル後の中間コードがデバイス非依存であり，しかも，クラス名とメソッド名による名前解決によりメソッド呼び出しを実現している．コンパイルにより得られた中間コード内のメソッド呼び出しは，動的バインディングを採用しており，実行時に呼び出されるメソッドが決定する．このことは，Ruby VM が実行する中間コードの開発には，クロス開発環境が不要であることを意味しており，あらかじめターゲットデバイスの Ruby VM を用意しておくだけで，組込みソフトウェアを開発できることになる（図 1）．

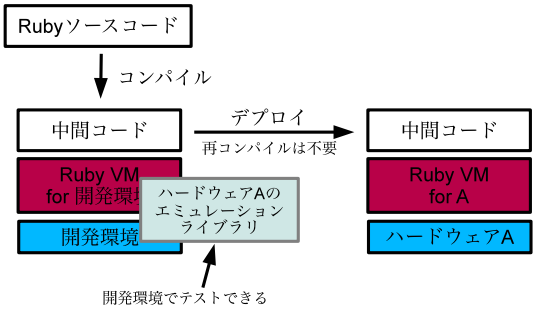

図 1　中間コードと Ruby VM

3. 軽量 Ruby 中間コードと Ruby VM

Ruby VM は，中間コードの逐次実行を行う．中間コードがサポートするのは，一般的な仮想計算機が提供する RTL（Register Transfer Language）の機能と，オブジェクト指向のメッセージパッシング機能を持っている．中間コードが扱うレジスタは，すべてオブジェクトを格納しており，レジスタに対するメッセージパッシングによりメソッドを呼び出す．

Ruby VM が呼び出すメソッドは，中間コードで記述されたメソッドと，Ruby VM がネイティブに持っているメソッドのいずれかである．ネイティブメソッドには，デバイス依存の実装を含めることができる．ネイティブメソッドであっても，メッセージパッシングにより呼び出されるため，中間コードとメソッドは動的なバインディングである．これにより，図 2 に示すハードウェアに依存した処理を，Ruby VM で抽象化できる．

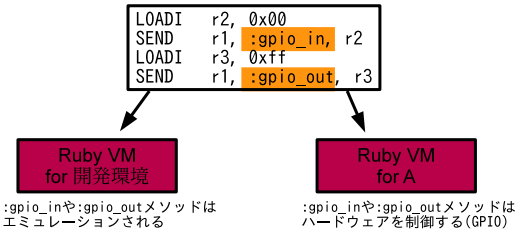

図 2　メソッド呼び出しとテスト

4. おわりに

軽量 Ruby は，2012 年 4 月にオープンソースとして公開され[⋆2]，現在もアップデートされ続けている．組込みシステムで Ruby を利用とすればリアルタイム性の問題，ハードウェア制御の問題など残っている課題も多い．一方，クロス開発が不要で，プログラムの実装にかかる時間が短い，コードの見通しが良いなど，開発者にとってのメリットは大きい．

今後，組込みシステム開発での軽量 Ruby の適用範囲を拡大すべくプロジェクトを進めていく．軽量 Ruby コミュニティの設立も予定されており，多くの開発者からのフィードバックに期待している[⋆3]．

謝辞　本研究の一部は，経済産業省 平成 22 年度 地域イノベーション創出研究開発事業「軽量 Ruby を用いた組込みプラットフォームの研究・開発」の助成を受けている．

⋆1 現在，1MB 以上のメモリを持つ組込みシステムは，全体のおよそ 40%であり，年ごとに増加している．2011 年度「組込みシステムにおけるリアルタイム OS の利用動向に関するアンケート調査報告書」（T-Engine フォーラム）

⋆2 https://github.com/organizations/mruby

⋆3 http://forum.mruby.org/