Java上のScheme処理系「ぶぶ」の実装

3H－4

山中 政宣，小宮常康，湯淺太一
京都大学大学院工学研究科情報工学専攻

1　はじめに

今日，インターネットを中心とするネットワーク環境は急速な発展を見せている．その中心ともいえるWWWはマルチメディアを利用した情報発信の手段としてまたたく間に広がった．しかし，発信される情報は一方的であり，ユーザはその情報を受け取ることしかできなかった．

この状況はJavaの登場に大きく変わった．Javaはアプレットと呼ばれる対話性を実現できるプログラムをつくることができる．アプレットはネットワークを介してダウンロードされ，Java対応ブラウザによってWebページの中で実行される．このアプレットによりWWWの可能性は大きく広がった．

こういった背景からJavaによるプログラミングはアプレットが中心となっており，従来プログラミング言語により記述されていたアプリケーションの記述例は少ない．従来の言語で記述されてきたプログラムがJavaによってどのように記述できるかを検討することは興味深い．

ところでLispの一方言にSchemeというプログラミング言語がある．Schemeの言語仕様は小さくシンプルであるが，継続と呼ばれる強力な制御機構を備えている．Schemeはプログラミング言語や新しい言語機能などの研究，プログラム開発環境のための土台として使われる．

本稿ではJava上で動作するScheme処理系「ぶぶ」の設計と実装，そしてJavaによる処理系の記述について述べる．

Implementation of BuBu — a Scheme system on Java
Masanori Yamanaka, Tsuneyasu Komiya, Taiichi Yuasa
Department of Information Science, Kyoto Univ.

2　TUTScheme

「ぶぶ」の設計にあたり，既存のScheme処理系としてTUTSchemeを参考にした．TUTSchemeは湯淺研究室で開発されたScheme処理系である．

TUTSchemeはC言語とSchemeで記述されており，バイトコードと呼ばれる中間コードを生成するコンパイラと，それを解釈実行するインタプリタとから構成される．

各種のデータはポインタで表現され，データ型を示すタグを保持する．また，不要となったメモリを回収し再利用するガーベージコレクション機能を備えている．

3　Java

プログラミング言語としてのJavaは，例外処理やマルチスレッド処理などの先進的な機構を備えたオブジェクト指向プログラミング言語である．

Javaの言語仕様はC / C++からポインタや構造体，共用体など複雑な言語要素を排除したシンプルなものである．Javaの特徴にはオブジェクト指向，インタプリタ型，アーキテクチャ非依存，自動ガーベージコレクションなどがある．

JavaプログラムはコンパイラによってJavaバイトコードに変換され，Java VMと呼ばれるインタプリタによって実行される．

4　「ぶぶ」の設計と実装

「ぶぶ」の設計方針は

1. TUTSchemeの実行方式を踏襲し，バイトコードに互換性を持たせる．
2. Javaの特徴を活かす．

であった．

この方針に基づき，バイトコードインタプリタ部を本体，バイトコード命令，組み込み関数に分離し，それぞれを独立したオブジェクトとして定義した．

これにより，バイトコード命令や組み込み関数の拡張が容易に行なえるようになった．

Java ではデータの属するクラスによってデータ型を区別できるため，TUTScheme のようにタグを用いる必要はなくなった．また，データに付随する処理をメソッドとしてデータ側に持たせた．これにより，組み込み関数の記述が簡潔になった．さらにデータ型の追加や変更の影響がそのデータの外に及ばず，拡張が容易に行なえる．

コンパイラやライブラリは TUTScheme のものをそのまま流用した．

実装における要点は次の通りである．

- C 言語における関数へのポインタをクラスとして実現し，関数自体はメソッドとしてそのクラス内に定義した．
- スタックは任意のデータを格納する必要があるため，全クラスのスーパークラスである Object クラスの配列として実現した．
- データ構造における構造体，共用体はクラスとして実現した．構造体，共用体の各要素はインスタンス変数として定義される．また，データに関する操作をメソッドとしてデータ側に持たせた．
- ガーベージコレクションは Java のガーベージコレクションを利用した．

5 まとめ

実装した「ぶぶ」は TUTScheme とプログラム，バイトコードにおいて互換性を持ち，ほぼ同等の機能を備えている．

これをソースコードレベルで TUTScheme と比較すると次の点が異なる．

- オブジェクト指向プログラミング言語の特徴としてプログラム全体が独立したモジュール構成になっている．モジュール単位での変更ができるため，保守性，拡張性に優れている．
- 処理系依存がないため，各プラットフォームに対して異なるプログラムを用意する必要がなく，プログラムには本質的な処理のみが簡潔に記述されている．
- メモリに関する処理を記述する必要がない．特に Scheme 処理系はガーベージコレクションを備えているため，この点で大きな差が生じる．

以上の点より，Java を用いることで機能を犠牲にすることなく，保守性，拡張性に優れた簡潔なプログラムを短期間で作成できることが分かる．

ところでこれらの長所に対し，短所としては実行速度が挙げられる．「ぶぶ」の実行速度は TUTScheme に大きく劣る．しかしこれは Java の発展とともに解消されていくと思われる．

参考文献

[1] IEEE Standard for the Scheme Programming Language, IEEE (1991)

[2] Clinger, W. and Rees, J., eds. : Revised^4 Report on the Algorithmic Language Scheme, MIT AI Memo 848b, MIT (1991)

[3] 湯淺太一: Scheme 入門, 岩波書店 (1991)

[4] 湯淺太一 他: TUTScheme のマニュアル, 豊橋技術科学大学 湯淺研究室 (1994)

[5] J. Gosling, B. Joy, and G. Steele: The Java Language Specification, Addison-Wesley (1997)