3H－5

Java クラスライブラリに対する言語間インタフェース

木俣 功†，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，インターネットを始めとするネットワーク環境の急速な進展にともなって，計算機のさまざまな分野の概念が大きく変化し，拡張されつつある．このような変化の中でプログラミング言語には，ネットワークを越える分散処理の機能が要求されるようになっている．そこで本研究室では Scheme[1, 2, 3] 処理系 TUTScheme[4] をベースに，Scheme による分散処理環境を開発している．

現在，ネットワークに最も大きなインパクトを与えているものの一つとして World Wide Web(WWW) をあげる事ができる．WWW ブラウザの普及によって Java[5] 実行環境が広く存在するようになった．そこで，前述の Scheme による分散処理環境の開発は Java を用いて行なっている．その開発の上で着目したのが Java のもつ豊富なクラスライブラリである．この Java の多種多様なクラスライブラリを Scheme 言語から利用することができれば，Java クラスライブラリの拡充にともなって Scheme の機能も拡充されることになる．

そこで本研究では Java クラスライブラリに対する言語間インタフェースの設計と実現法を提案し，また，本研究室で開発中の Java による Scheme 処理系『ぶぶ』[6] への適用をおこなった．本稿ではその概要を報告する．

2 言語間インタフェース

言語間インタフェースに必要な機能としては Java のクラスの読み込み，インスタンス生成，メソッド呼び出し，フィールド変数の参照及び変更を挙げることができる．このなかで，メソッド呼び出しに関しては Java では引数の静的な型によるメソッドオーバーローディングを採用しているため，注意を要する．Scheme のような静的な型を持たない言語との組合せでは，呼び出すべきメソッドを決定するためにメソッド名の他に引数のシグネチャ(signature) が必要となる場合があるためである．

提案する設計では，メソッド呼び出し時のメソッド決定の問題を解決するために，メソッドのシグネチャを引数の実行時の型から決定する機構を設けるとともに，ユーザがシグネチャを指定するためのインタフェースを設けた．その実装には Java Virtual Machine におけるシグネチャの記述に基づいたシグネチャ文字列を用いた．

3 ぶぶ への組み込み

我々はこのインタフェースの実装を行ない，ぶぶへ組み込んだ．ぶぶ への組み込みは，Scheme 上からのインタフェースである組み込み関数の実装と，JavaClass および JavaObject の 2 つのクラスを Scheme のデータ構造へ取り込むことによって実現した．Scheme のシンプルな言語仕様はこの拡張を非常に容易にするものであった．

ぶぶ 上からの Java クラスライブラリ利用のインタフェースは大別すると以下の 3 種類である．

クラスオブジェクト Scheme において Java のクラスをあらわすデータ．(クラス JavaClass)

オブジェクトオブジェクト Scheme において Java のオブジェクトをあらわすデータ．(クラス JavaObject)

組み込み関数

- クラスオブジェクトの獲得

  `(get-class <クラス名>)`

  指定されたクラスのクラスオブジェクトを返す．<クラス名> はクラスの完全な名前の文字列．

A Language Interface to the Java Class Libraries
KIMATA Isao†, Tsuneyasu Komiya††, Taiichi Yuasa†††
†Department of Infomation Science, Kyoto Univ.

- インスタンスの生成

  (make-instance <クラス>
  <引数$_1$> … <引数$_n$>)

  <引数$_1$>〜<引数$_n$> を初期化のための引数として <クラス> のインスタンスを生成し，それを返す．

- メソッドの呼び出し

  (invoke <オブジェクトまたはクラス>
  <メソッド> <引数$_1$> … <引数$_n$>)

  <メソッド> で指定したメソッドを <引数$_1$>〜<引数$_n$> を引数として呼び出す．<メソッド> は文字列でメソッド名を指定するか，もしくは

  (<メソッド名> <型名$_1$>…<型名$_n$>)

  (<メソッド名> <シグネチャ文字列>)

  の形式のリストで指定する．第1引数がクラスオブジェクトであるときは，クラスメソッドのみ呼び出すことができる．

- フィールド変数の参照

  (java-ref <オブジェクトまたはクラス>
  <フィールド名>)

  <フィールド名> で指定したフィールド変数の値を得る．<フィールド名> は文字列．第1引数がクラスオブジェクトであるときは，クラスフィールドのみ参照可能である．

- フィールド変数への代入

  (java-set! <オブジェクトまたはクラス>
  <フィールド名> <式>)

  <式> を評価し <フィールド名> で指定したフィールドへ代入する．<フィールド名> は文字列．第1引数がクラスオブジェクトであるときは，クラスフィールドへのみ代入可能である．

4 まとめ

このインタフェースによって，ユーザは Scheme の上から Java クラスライブラリの希望する機能を利用することが可能となった．Java の豊富なクラスライブラリを用いることによって，比較的簡単な記述で非常に高い機能を実現することができる．イベント処理やスレッドに関する処理はまだサポートされていない．しかし，今日のソフトウェアに欠かすことができない機能であるネットワーク機能や GUI の機能を Java のクラスライブリを用いることで比較的容易に Scheme の上から利用することが可能となったことから，本研究の言語間インタフェースには一定の意義があると思われる．

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)

[6] 山中政宣 : Java 上の Scheme 処理系「ぶぶ」の実装, 京都大学工学部情報工学科 特別研究報告書 (1997)