3L-2

文脈操作によりコラボレーション記述を簡易化するプログラミング言語

羽二生　典之　　南澤　吉昭　　大場　善次郎
北海道大学情報科学研究科

1　はじめに

プログラム記述の簡易化は EoD (Ease of Development, 開発の容易さ)を向上させる。その例に Java 言語バージョン 3.0[1]が有する Enhanced For Loop がある。

オブジェクト指向言語により記述されたプログラムにはコラボレーション記述が頻出し、その記述を簡易化できる可能性がある。Visual Basic[2]の Using 文はリソース使用の宣言を行う際にコラボレーション記述の簡易化を行えるが、インスタンスの個数が１つである点が限界といえる。また、メソッド呼び出しについても頻出し、記述を簡易化できる可能性がある。

本研究では指定した複数の他のクラスのインスタンスについてのコラボレーション記述の削減、およびメソッド呼び出しについての記述の簡易化のための機能、及びそれを備えた言語「NORL」を提案する。また、記述の容易さについて Java 言語との比較を行う。

2　NORL の概要

2.1　処理系

NORL の処理系が従うべき抽象的な構造を図 1 に示す。一度 Java 言語へ変換する際、複数のソースファイル間でメソッド呼び出しのパラメタ名の名前解決を行う。その後、Java コンパイラを通すことによりクラスファイルが生成される。クラスファイルは Java 仮想マシンで実行する。この方式は API 等の既存資産の利用の面の利点がある。

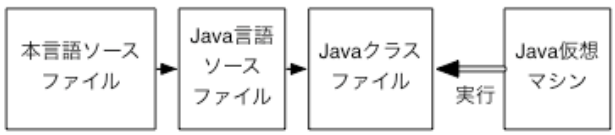

図 1 NORL の抽象的な処理系

2.2　言語仕様

NORL の機能に関連する機能として典型的なオブジェクト指向言語にはカプセル化の助長の目的でクラスのすべてのインスタンスメソッドに対してクラス自身のインスタンス変数やインスタンスメソッドのスコープを与える機能がある。NORL の機能は 1 つのメソッドから指定した複数の他のクラスのインスタンスに含まれるメソッドや変数へのスコープの付与と、キーワード引数により一致した名前の引数に対する自動的な引数渡しからなる。

NORL の文法は Java 言語との差分として BNF で定義される。字句レベルではキーワードとして 'context' を追加する。このキーワードとともに宣言された変数は、メソッドの中の中括弧 '{', '}' からなるブロックのスコープの文脈として登録される。また、初期化文とともに宣言された場合は型を省略することができる。また、値の前に context キーワードを付加することができ、その場合には型がその値の型であり、名前なしの変数が宣言されたものとする。パラメタ宣言の場合は変数名の省略により名前なし変数となる。これらの機能を文脈操作と名付ける。文脈として登録された変数は次の振る舞いを行う。

- インスタンスへの参照の場合のみ、そのインスタンスが含む公開されたすべてのメンバへの単一識別子でのアクセス
- 呼び出すメソッドへの自動的な引数渡しの指示

１番目の機能は名前ありまたはなしのいずれの変数でも用いることができる。もし引数のコンテキストが名前を与えられないパラメタとして宣言されたならば、その引数には名前がないものとする。参照されるメンバの名前は重複することがあり、その場合は Java 言語のインポートと同様に内側のスコープにある文脈を優先し、仮に同じスコープに宣言された２つ以上の同一

名称のメソッドや変数が存在する場合には、参照時にエラーとする。

2 番目については、Python[3]が有するようなキーワード引数機能を用意する。キーワード引数は位置ではなく名前による引数渡しを行うものである。まず、Java 言語にキーワード引数機能を追加する。また、Java 言語にはデフォルト値での引数の初期化機能が存在しないが、これを可能にする。メソッド定義における引数と登録された文脈についてその名前及び型が同じ場合に自動的に引数を渡す。メソッド定義における引数名はクラスファイルからは取り除かれるため、クラスファイルに含まれるメソッド呼び出しのために本機能を用いることはできない。このため、呼び出されるメソッドは NORL から変換された Java 言語ソースファイルとする。

3 NORL と Java 言語の比較

XHTML[4] (XML を用いた HTML)ファイルからタイトルと説明とキーワードを抽出するプログラムを作成し、Java 言語と NORL を比較した。ここでは、XML の API として DOM (Document Object Model)を用いた。DOM の欠点である階層構造を順にたどる処理の記述の煩雑さが一般的なプログラムの性質と異なる可能性があるが、現実にはよく用いられており、NORL の実際的な有用性について確認できるといえる。比較は局所的な簡易化の確認およびプログラム全体でのトークン数により行った。トークンの入力効率については無視した。

Java 言語で記述したプログラムの一部を次に示す。

```
String uri="http://www.w3.org/1999/xhtml";
Element head=(Element)html.
  getElementsByTagNameNS(uri,"head").
  item(0);
System.out.println("Title:"+
  head.getElementsByTagNameNS(uri,"title").
  item(0).getTextContent());
```

NORL で記述したプログラムの一部を次に示す。

```
context uri="http://www.w3.org/1999/xhtml";
context System.out;
context (Element)html.
  getElementsByTagNameNS("head").item(0);
println("Title:"+
  getElementsByTagNameNS("title").item(0).
  getTextContent());
```

この部分から、NORL による記述では文脈操作が追加されている一方で、言語レベルでは次の点で記述を簡易化できていることがわかる。

- '.'の左側のメソッドを呼び出すべきオブジェクトの指定(System.out, head)の省略
- getElementsByTagNameNS メソッドへの uri 引数の省略

また、プログラム全体のホワイトスペース以外のトークン数の比較では、Java 言語では 194 トークン、NORL では 158 トークンとなり、18.6%減少した。このことから、プログラム全体で記述の簡易化がなされたことがわかる。

4 結論と今後の課題

本研究では、変数およびメソッドのスコープを与えるインスタンスの指定と、名前付きメソッドへ指定した引数の自動的な代入を行うプログラミング言語 NORL の開発を行った。また、XHTML からの情報の抜き出しの例により、記述の簡易化を確認した。

スコープにおける名前の競合が与える問題の程度については今後検証すべき点である。

また、今回はスコープをメソッド内部に限定したが、クラスやファイルのスコープについても記述を簡易化できる可能性がある。

本機能は典型的なオブジェクト指向言語のスコープに関する一部の機能を備えていることから、本機能を追加した構造化言語とオブジェクト指向言語の比較も１つの課題である。

参考文献

[1] James Gostling, Bill Joy, Guy Steele, and Gilad Bracha, 2005, The Java Language Specification 3.0, Addison-Wesley.

[2] Paul Vick, 2005, The Microsoft Visual Basic Language Specification Version 8.0, Microsoft Corporation, http://www.microsoft.com/downloads/details.aspx?FamilyId=6D50D709-EAA4-44D7-8AF3-E14280403E6E&displaylang=en

[3] Guido van Rossum, 2006, Python Reference Manual, Python Software Foundation, http://docs.python.org/ref/

[4] World Wide Web Consortium, 2006, Extensible Markup Language (XML) 1.0 (Fourth Edition), http://www.w3.org/TR/xml/

[5] World Wide Web Consortium, 2006, Namespaces in XML 1.0 (Second Edition), http://www.w3.org/TR/xml-names/