CASEツールによるJava言語プログラムの理解支援

5AE－3

岡山 敬　福田 薫　金子 博　松本 憲幸*
株式会社　東芝**

1. はじめに

プログラムを理解することは、ソフトウェア開発において非常に重要な事柄である。効率的にプログラム理解を支援する方法として、プログラムの各記述の内容を表現するコメントを自動的に付加する方法を実現した[1]。

プログラムの各記述を理解する際には、変数定義や参照部分など理解しようとする記述に関連する他の記述についても理解している必要がある。また、Java 言語の場合、生成したオブジェクトは、インスタンス名で扱うため、メソッド実行文においてクラスに関する情報が含まれていない。

そこで、各記述に対する翻訳を行なう前に、プログラム全体を解析することによって各記述の理解に役立つ情報を抽出しておいて、各記述の翻訳の際に利用する方法を考える。

本稿では、プログラムの意味情報を自動的に付加する方式を実装し、さらにプログラムのマクロ的な理解に利用可能な設計書の生成機能をも装備したプログラム理解支援システムについて、その構成、意味情報の付加方式及びその効果について述べる。

2. プログラム理解支援システムの構成

プログラム理解支援システムの構成を図1に示す。概略アルゴリズムは、以下のとおりである。

Step 1 プログラム全体を読み込んで、クラス、メソッド、パッケージに関する継承・包含・呼び出し関係や概要コメントなどの意味情報を抽出して意味情報辞書に格納する。

Step 2 プログラムを1記述ずつ読み込んで、変数などの説明を持つシンボル辞書とパターンに関する規則を持つパターン翻訳辞書を利用して翻訳パターンを取り出す。

Step 3 翻訳結果と意味情報をコメントとしてプログラムに付加することにより、コメント付きプログラムを生成する。

Step 4 コメント付きプログラムと抽出した構造情報から各種の設計書を生成する。

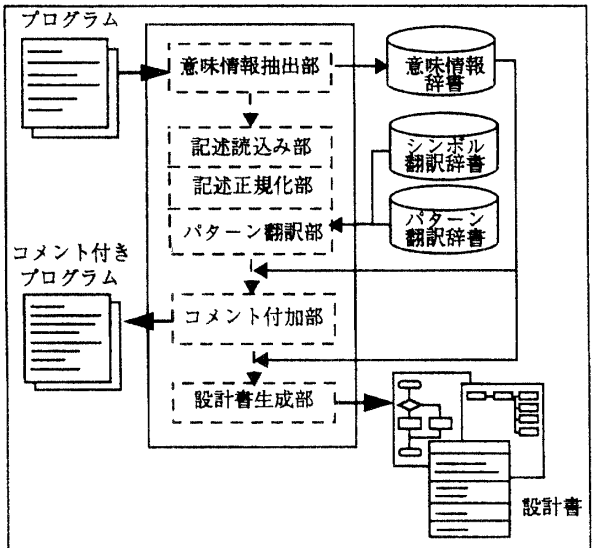

図1　プログラム理解支援システム

3. 意味情報の付加方式

プログラムを解析することにより抽出可能な情報の中で、コメントとして付加することにより、その記述の理解に役立つ情報として次のものを考える。

3.1 メソッド実行文

メソッド実行文に出現する変数名は、インスタンス名であるため、宣言文及び生成文を調べないとどのクラスから生成したオブジェクトなのか知ることができない。また、メソッド名だけではメソッドの

Understanding Support by CASE tool for Java Language Program

* *Takashi Okayama, Kaori Fukuda, Hiroshi Kaneko, Noriyoshi Matsumoto*
** TOSHIBA Corporation

処理内容を知ることはできない。よって、メソッド実行文を理解するためには、”メソッド実行文→オブジェクト生成文→変数宣言文→クラス定義→メソッド定義”と辿っていく必要がある。

そこで、これらの意味情報を結合してメソッド実行文のコメントとして付加することにする。クラスの説明として、クラス概要だけでなく実際のクラス名を付加しておくことにより、ここからクラス定義部分への参照操作が容易に実行できる。

（例１）定義情報を付加した場合

```
// 円を定義するクラス Circle の線作成メソッドを実行する
f.drawLine();
```

3.2 クラス定義、メソッド定義

プログラム作成者は、クラスやメソッドの定義部分に概要や定義内容に関する説明をコメントとして記述することはあるが、そのクラスがどのクラスで実際に利用されているか、このメソッドがどのクラスで呼び出されているかなどの関係情報は記述しない。しかし、これらの情報は、定義内容を理解するには必要なものであり、定義されているクラスやメソッドが実際に利用されているかどうかの確認にも使用できる。よって、解析により関係情報を抽出しておいて、定義部分にコメントとして付加する。

（例２）クラスの関係情報を付加した場合

```
// 多角形を定義するクラス Poligon の中心を求めるメソッド getCenter で利用されている
// 円を定義するクラス Circle に包含されている
class Point {
    ...
}
```

3.3 オーバライドメソッド

オーバライドされたメソッドを理解する場合、オーバライドする前のメソッドの意味を理解することが必要である。このことから、オーバライドされたメソッドに対して、オーバライドする前のメソッドの概要コメントを抽出して付加する。

（例３）オーバライド情報を付加した場合

```
// 面積を求めるメソッドをオーバライドしている
int getArea() {
    /* 円の面積を求める処理 */
    ...
}
```

4. 生成する設計書

本システムにより自動生成可能な設計書には、プログラム全体の構造の理解に役立つクラス継承図やメソッド呼び出し図、各定義内容の詳細の理解に役立つパッケージ・クラス・メソッドの各仕様書がある。翻訳コメントを付加したプログラムから設計書を生成することにより、仕様書の各説明欄に詳細な意味情報を反映させることができる。

5. 効果

抽出した意味情報をコメントとして各記述に付加することにより、各記述の内容を他の部分を参照することなく、より効率的に理解することができる。また、生成された設計書を利用することによりプログラムの全体構造を容易に把握することができる。

6. おわりに

Java 言語プログラムを解析することにより抽出する情報を記述理解に役立つ部分にコメントとして付加し、さらに設計書を自動生成するシステムの構成、意味情報付加方式及び効果について述べた。

今後、より効率的な情報抽出方式や他の抽出情報の検討や他の言語への対応などを研究していく予定である。

参考文献

[1] 岡山，福田，金子，松本："Java 言語、C 言語プログラムパターンの翻訳"，情報処理学会第 54 回全国大会講演論文集 (1) pp.219-220, 1996.

[2] Tim Ritchey: "Java 環境入門"，プレンティスホール，1997.