実行可能な形式仕様言語 CafeOBJ(2)

6V－5

Cafe システムの核アーキテクチャ

澤田 寿実[†*]　中川 中[†‡]　本間 毅寛[‡]　谷津 弘一[‡]　二木 厚吉[*]

†(株)SRA ソフトウエア工学研究所, ‡情報処理振興事業協会, * 北陸先端科学技術大学院大学

概 要

Cafe システムは実行可能形式仕様言語 CafeOBJ による仕様記述を支援するための基本ツールを統合したシステムであり，その一部が実装されている．システムの中心的な部分の一つである CafeOBJ の解釈系は順序ソート書き換え論理を順序ソート項書換え系によって実現したものである．本稿では，システム核の実装のあらましを述べる．

1 はじめに

Cafe は大規模な形式仕様の開発を支援することを目的としたシステムである．Cafe では単独の仕様記述言語の支援にとどまらず，同じ論理体系に基づく言語の一族を支援することを目的としている．CafeOBJ はシステムで提供される標準の汎用仕様記述言語であるが，その他に特定のモデル化方法(ペトリネット，状態遷移，等)や記述の枠組(オブジェクト指向等)に特化した言語を提供することが可能であり，これらの言語を同時に使用することが出来る．システムは拡張可能な核言語を持っており，この核言語を拡張することにより複数の言語が実現される．

我々は基本となる論理体系として順序ソート書き換え論理 [1] を採用した．この論理は通常の等号論理をその部分として包含する他，動的な状態変化を非常に高い抽象レベルで宣言的に記述する能力を持っている．オブジェクト指向設計の枠組等も自然な形で取り込むことが可能である．また，多くの他の論理体系をシミュレートする能力を持っており，拡張可能な核言語とあいまって，広範囲の問題領域のモデルを自然に記述することが可能となった．

Cafe で記述される仕様は実行可能なサブセットを持つが，これは順序ソート書き換え論理を項書換え系によって実行することによって実現される．本稿では核言語の拡張の枠組と項書換え系の実現について現状を述べる．

2 核言語の拡張機構

核言語は自身の言語構成要素を記述する能力を持っており，他の言語はこの核言語によって定義される．言語の定義はその構文定義と，定義された各構文要素を核言語へ変換する変換規則を与えることによって行なわれるが，これは実行可能仕様として記述される．

Cafe における仕様記述は代数的仕様記述と同様のものであり，問題を表現するための構文をシグニチャによって記述し，公理を与えることによって意味記述を行なう．言語の定義もこれに沿って行なわれ，言語の構文をシグニチャによって記述し，変換規則を公理によって記述する．これらの公理は各構文要素の意味関数の意味を定義する物と考えることが出来る．

Cafe では OBJ 言語 [2] 風の強力なモジュール化機構があり，すべての仕様はモジュール化され，これが問題の仕様を記述する際の基本構築部品となる．核言語の言語構成要素やそれらに対する操作もシステム組み込みのモジュールとして提供されており，これを輸入/拡張したモジュールを定義することによって，新たな言語を定義する．定義された言語(モジュール)はさらに他のモジュールから参照することが可能であるため，段階的な拡張が可能で

ある．また，共通の言語構成要素を定義しておき，複数の言語で共有することが出来る．

非常に簡単な例を下に示す．問題記述の構文的な側面のみを定義することを目的としたsigという新たな言語要素をCafeOBJv(CafeOBJ言語のvariant)に対して追加したものである(参照のため行番号をふった)．

```
1  %language(CafeOBJv)
2  module SIG {
3    import CafeOBJv
4    [ sig; sig-construct > sort-declaration
              operator-declaration ]
5    sig_{_} : module-id sig-construct*
                  -> sig
6    sig M:module-id { sc:sig-construct* }
     = module M { sc }
  }
```

第1行目は核言語の項であり，言語CafeOBJvを用いることを宣言したものである．第2行目以降がsig構文の定義であり，SIGという名のモジュールとして定義されている．モジュールの本体ではCafeOBJv言語の定義が輸入され(第3行)ている．第4行目は新たなソートの宣言とソートの順序関係の宣言である．sort-declarationとoperator-declarationは，CafeOBJvで宣言されているソート記号であり，それぞれソート宣言，オペレータ宣言という構文要素に対応する物である．これらはソートsig-constructのサブソートであると宣言されており，sig-constructは新たな構文sigの構成部品であるので(5行目)，sigではソート宣言と，オペレータ宣言のみが可能となる．6行目は等式によって“sig M ... ”は“module M ... ”に変換されることを記述したものである．この，“module M ... ”はさらに変換され，最終的に核言語のモジュール宣言へと変換される．

3 順序ソート書き換え論理の項書換え系による実装

項が型(ソート)を持ちまたソート間に半順序関係が定義出来ることから，非常に大きな表現力が得られるが，反面項書換え系はソート無し，あるいは多ソートの場合に比べて複雑な処理が必要となる．順序ソート項書換え特有の実装上の問題点や，その解決法については[5]を参照されたい．以下では，書き換え論理を項書換えによって実装する際の問題点に付いて指摘する．

紙幅の関係より，ごく直観的な説明をする．書き換え論理は等号(=)と矢印(⇒)の二つの述語記号を持つ一階の論理体系である．等号を用いて等式($t1 = t2$)が記述される．等式の集合Eは項の間の合同関係($=_E$)を定義する．これに対し，矢印は規則($t1 \Rightarrow t2$)を記述するもので，規則は状態の変化を表現するものである．この場合矢印の両辺の項は状態と解釈される．書き換え論理では等式の集合Eによって定義される同値類(これを$[t]$のように表記する)に対して規則の集合Rから得られる関係$[t] \rightarrow_R [t']$が定義される。この関係は反射律および推移律を満たすが、通常の等号関係とは異なり、対称律を満たす必要はない。

もし，$=_E$に対する完備な書き換え関係$\rightarrow_E$が得られた場合，Eで割った書き換え$\rightarrow_R$を$\rightarrow_E$による書換えとRによる書換えの組み合わせで実現したい気持ちになるが，これは必ずしも正しくない．これが正しいものとなるためには，$\rightarrow_E$と$\rightarrow_R$との間にcoherencyと呼ばれる関係が成立することが必要十分である[4]．これは同値類の代表元の選び方が問題とならないことを保証するものである．

利用者がcoherencyを満足するように仕様を記述することが，特殊な場合を除いてさほど困難なことではないこともあり，現在のCafeの実装ではこれを検査する機構は未だ実装されていない．しかし，良く使用されるが，完備な項書換え系を得ることが出来ない特殊なクラスの同値関係(associative, commutative, これらの組み合わせ等)で割った書き換えが実装されており，多くの興味深い問題がこれを利用することによって記述・実行が可能となっている．

参考文献

[1] José Meseguer, “Conditional rewriting logic as a unified model of concurrency”, *TCS*, Vol. 96, 1992, p.73-155

[2] J.Goguen, T.Winkler, J.Meseguer, K.Futatsugi, J.-P Jouannaud *Introducing OBJ*, SRI International, SRI-CSL-92-03, 1992

[3] J.Goguen, J.Meseguer, “Order-sorted algebra I”, *TCS* Vol. 105, 1992, p.217-273

[4] J.-P Jouanaud, H. Kirchner, “Completion of a set of rule modulo a set of equations, *SIAM Journal of Computing*, Vol. 15, No.4, pp.1155-1194, 1986

[5] 澤田他,“順序ソート項書換え系の実現法”, 第8回ソフトウェア科学会全国大会予稿集