2E-3

COMMONLOOPSのUtiLisp上の実現

松木美保子，泉寛幸，吉田裕之，加藤英樹

株式会社　富士通研究所

1. はじめに

COMMONLOOPS は，Common Lisp 上のオブジェクト指向プログラミングシステムとしてD.G.Bobrowらにより提案されたものである［1］［2］．このたび COMMONLOOPSの基本部分を UtiLisp上に実現したので報告する．本稿では本システムをUtiLoopsと呼称する．

我々は知識表現システムの研究をしており，UtiLoopsはその研究用ツールとして作成した．COMMONLOOPS を選んだ理由は，コンパクトなのでインプリメントの効率がよく，拡張性に富んでおり，オブジェクト指向とLISPの考え方が融合しシステム全体の思想が洗練されているので簡潔なプログラムを期待できるからである．

2. COMMONLOOPS

2.1 記法

はじめに COMMONLOOPSの記法を略述する．

・クラスの定義は Common Lispの機能を拡張した DEFSTRUCT を用いて行う．例を示す．

```
(DEFSTRUCT SHAPE
  (X-POSITION 0)(Y-POSITION 0))          ---- (1)
(DEFSTRUCT (COLORED-SHAPE (:INCLUDE SHAPE))
  (COLOR 'GREEN))                        ---- (2)
```

(1)はクラス SHAPEの定義で，SHAPE のインスタンスがX-POSITIONとY-POSITIONという名前の属性を持ち，初期値をそれぞれ 0とすることを示す．(2)は COLORED-SHAPEの定義である．:INCLUDEオプションでは SHAPEを上位クラスとして継承することを宣言する．これにより，COLORED-SHAPE のインスタンスは(2)で指定する属性 COLORと上位クラス SHAPEで指定した属性X-POSITIONとY-POSITIONをもつ．

・インスタンス生成は，クラス定義時に自動的に作られる MAKE-クラス名　のコンストラクタを用いて行う．

```
(SETQ MY-COLORED-SHAPE (MAKE-COLORED-SHAPE))
```

これは COLORED-SHAPEのインスタンス生成例で，MAKE-COLORED-SHAPEがコンストラクタである．

・メソッドは，特定のメッセージが特定のクラスのオブジェクトに送られた時に実行する手続で，DEFMETHOD を用いて定義する．次にメソッドMOVEの例を示す．

```
(DEFMETHOD MOVE ((OBJ SHAPE)
                 (X FIXNUM)(Y FIXNUM))
  ...)                                   ---- (3)
(DEFMETHOD MOVE ((OBJ SHAPE)
                 (X FLONUM)(Y FLONUM))
  ...)                                   ---- (4)
(DEFMETHOD MOVE ((OBJ COLORED-SHAPE)
                 (X FIXNUM)(Y FIXNUM))
  ...)                                   ---- (5)
```

定義するメソッド名MOVEの次にくるリストは次の形式で，引数の属すクラスを指定する．

((仮引数-1　クラス名-1)
(仮引数-2　クラス名-2) ...)

(3)では OBJ,X,Yが仮引数で，クラス SHAPE,FIXNUM,FIXNUMにそれぞれ属すことを指定する．このように引数の型を異なって指定することで，同名のメソッドの複数個定義が可能である．(3)～(5)の ...部分にはメソッドの手続を記述する．

・メソッド呼出は次の形式で行う．

(メソッド名　実引数-1　実引数-2 ...)

例えば，上述のメソッドMOVEは次のように呼出せる．

```
(MOVE MY-COLORED-SHAPE 10 20)
```

2.2 特徴

・COMMONLOOPS ではクラスの多重継承が可能である．クラス定義の際，DEFSTRUCT 文の:INCLUDEオプションに複数個の上位クラスを列挙することができる．

・メソッドはクラスには属さず，独立に定義する．

・メソッドが呼出されると，第一引数だけでなくすべての実引数のクラスを求め，適用可能なメソッドのうち最も優先順位の高いものを評価する．この機能は多重メソッドと呼ばれ，他の多くのオブジェクト指向用システムと異なる．

・メソッド呼出の形式はLISP関数呼出と同じであり，メソッドの機能はLISP関数の機能を実引数のクラスを考慮するように一般化したものととらえられる．この点でオブジェクト指向がLISPの考え方に融合しているといえる．

・メソッド結合が可能である．メソッド評価中に関数 RUN-SUPER を呼出すと，同名の上位メソッドを実行する．

An Implementation of COMMONLOOPS on UtiLisp

Mihoko MATSUKI, Hiroyuki IZUMI, Hiroyuki YOSHIDA, Hideki KATO

FUJITSU LABORATORIES LTD.

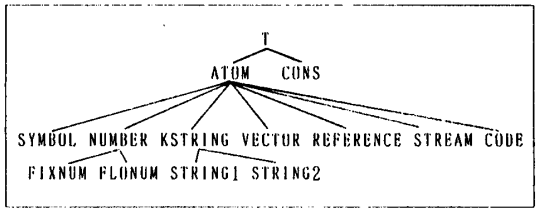
3. UtiLoops

3.1 実現の概要

UtiLisp 上に COMMONLOOPSのカーネルと呼ばれる基本部分を実現した.

UtiLoopsは, 関数EVALを書換えるなどの UtiLispシステムの変更を行わずに実現している. これはメソッドを後述のようなLISP関数として定義することで可能となった.

なお Common Lispと UtiLispの基本的機能の相違によりCOMMONLOOPS と異なる点がある. 例えば, LISPオブジェクトの型とその階層関係はUtiLoops独自に定め, 下図のようにした.



3.2 データ

処理の高速化とメモリ節約をねらい, クラスとメソッドについては関連するデータをその属性リストにベクタで保管して実現した. またメソッド名の関数定義部にはメソッドの実行を処理するための手続を格納する. 前述の例MOVEのシンボルの関数定義は次のとおりである.

```
(LAMBDA (OBJ X Y)
  (APPLY-METHOD 'MOVE (LIST OBJ X Y)))        ..... (6)
```

3.3 メソッドの適用方式

UtiLisp システム自身を変更しないでUtiLoopsを実現するために, メソッド呼出は(6)のようなAPPLY-METHODを呼出すLISP関数とした. APPLY-METHODは, メソッドの実引数のクラスをもとにメソッド優先順位リストを求めて, 最も優先順位の高いメソッドを評価する関数である. メソッド優先順位リストとは, 定義されたメソッドのうち適用可能なものを, 実引数のクラスの継承順位をもとに優先順位の高い順に並べたものである.

メソッド呼出例

```
(MOVE MY-COLORED-SHAPE 10 20)
```

の場合APPLY-METHODは, MY-COLORED-SHAPE, 10, 20のクラス COLORED-SHAPE, FIXNUM, FIXNUM を調べ, それをもとに適用可能なメソッド(3)と(5)を取り出す. COLORED-SHAPE が SHAPE より優先順位が高いことから次のメソッド優先順位リスト

```
(((COLORED-SHAPE FIXNUM FIXNUM)
  (LAMBDA (OBJ X Y) ...))                  } メソッド(5)
 ((SHAPE FIXNUM FIXNUM)
  (LAMBDA (OBJ X Y) ...)))                 } メソッド(3)
```

を得て, 最初のメソッド(5)を評価する. なおこのメソッドの評価中にメソッド結合用関数 RUN-SUPERが呼出されると, 順位が一つ低いメソッド(3)を評価する.

3.4 メソッド適用表の利用

メソッド優先順位リストの作成時間の短縮を図って, メソッド名の属性リストにメソッド適用表を用意する. メソッド適用表は第一引数に適用可能なすべてのクラスをキーとし, 対応する一次優先順位リストをデータとする連想リストである. メソッドの優先順位は一般にはすべての実引数のクラスがわからなければ完全に決らないが, 連想リストのキーのクラスの第一実引数を与えることによって決る範囲で, 優先順位の高い順にメソッドを並べたリストを, 一次優先リストと名付けた. メソッド呼出時には, 適用表から第一実引数のクラスに対応する一次リストを取り出し, 残りの実引数をもとに完全なメソッド優先順位リストを求める.

前述のメソッド呼出例の場合, 適用表から第一実引数 MY-COLORED-SHAPEのクラス COLORED-SHAPEに対応する次の一次リストを取り出す.

```
(((COLORED-SHAPE FIXNUM FIXNUM)
  (LAMBDA (OBJ X Y) ...))                  } メソッド(5)
 (((SHAPE FIXNUM FIXNUM)
  (LAMBDA (OBJ X Y) ...)))                 } メソッド(3)
 ((SHAPE FLONUM FLONUM)
  (LAMBDA (OBJ X Y) ...))))                } メソッド(4)
```

ここで第一引数のクラスが COLORED-SHAPEであることだけからは優先順位の決らないメソッド(3)と(4)は, リストにまとめた. そしてこの一次リストと残りの実引数 10, 20をもとにメソッド優先順位リストを求める方式をとった.

4. おわりに

以上の方式でCOMMONLOOPS カーネルを UtiLisp上に実現した. 現在, 知識表現システムの研究に利用している. その一つとしてUtiLoops上にグラフィックルーチンを作成し利用している.

今後の課題に, 処理の高速化がある.

謝辞

日頃御指導頂く棚橋部長, 林部長代理並びに研究室諸兄に深く感謝致します.

文献

[1] D.G.Bobrow, K.Kahn, G.Kiczales, L.Masinter, M.Stefik and K.Zdybel: COMMONLOOPS, Pre-IJCAI'85 Draft, ISL-85-8, XEROX PARC, 12 AUGUST 1985.

[2] Common Lisp の動向に関する調査報告書, 日本電子工業振興協会, 1986.

[3] UTILISP 手引書, FACOM マニュアル, 1985.