# オブジェクト指向PROLOGの設計

3F−6

碓崎 賢一
九州工業大学情報工学部

## 1．はじめに

知識処理などの高度な情報処理システムの記述言語として，単一化と後戻りに基づく強力で柔軟な処理能力を持つ PROLOG が多用されている．しかしながら，PROLOG には大規模なシステム開発を支援するモジュール機能などが欠如しているために，実用的なシステムを記述する際の問題点となっている．また，知的 CAD，CASE システムなどを構築しようとすると，対象の知識や設計した部品などを柔軟に表現できる必要があり，PROLOG にもこれらの要求を満たすオブジェクト指向機能が必要となっている．本報告では，この様な要求を満たすオブジェクト指向 PROLOG[1]の概要を示す．

## 2．設計方針

### 2．1　PROLOG とオブジェクト指向機構の融合

本報告で示すオブジェクト指向 PROLOG は，従来の DEC-10 準拠の PROLOG に対して上位互換性を持つように考慮し，その使用者が違和感なく利用できるようにする．PROLOG と提案するオブジェクト指向機構との関係は，C に対する C++，Common Lisp に対する CLOS と同様であり，利用者は必要に応じてオブジェクト指向の利点を利用し，特に必要性がなければ，オブジェクト指向機構を意識しなくてもよいようにする．

### 2．2　継承機能

オブジェクト指向の特徴である継承機能としては，多重継承を実現する．クラス定義は，基底クラスが完全に定義されていなくても，定義するクラスだけに着目してクラスの構造やメソッドを定義することができるようにする．基底クラスはそのクラスを使用するプログラムが実行される時点で定義されていればよい．

### 2．3　性能

オブジェクト指向システムには実行時の探索処理などのオーバーヘッドがあるが，探索処理などの実行時の処理を減少[1]させ，オブジェクト指向機構を利用したプログラムの実行速度を向上させる．また，オブジェクト指向機能の追加により，処理内容が増加するために，PROLOG 処理系としての処理速度の低下は避けられないが，性能低下を極力抑えることにより，従来の PROLOG の機能だけを利用する場合にも性能的な問題が生じないようにする．また，コンパイラだけではなく，インタプリタにおいても効率的な処理が行えるような実現手法を導入する．

## 3．言語仕様

### 3．1　基本機能

多重継承されたメソッドは，継承するクラスリストの左から右へ深さ優先で探索し，最初に見つかったものを実効メソッドとして用いる．メソッドは派生クラスで上書きして変更することができる．ESP[2]では，基底クラスに定義されているメソッドと派生クラスで定義されたメソッドを組み合わせて1つのメソッドとして利用するが，本システムでは単純に上書きする方式を採用している．

各オブジェクトの状態を表すスロットとして，クラス全体で共通のクラス変数と，各インスタンスで独立したインスタンス変数を用意する．スロットにはシンボルや数値などのアトミックな項と，リストや複合項などの構造を持つ項を代入することができる．スロットへの値の代入は ::=/2 を用い，左辺にスロット，右辺に値を指定することで代入することができる．構造を持つ項は，そのコピーが作成されてスロットに格納される．スロットが ::=/2 以外の引数として現れた場合には，スロットの値の参照として処理される．

### 3．2　字句要素

基本的には従来の PROLOG と同様であるが，オブジェクト指向機構導入のために，新たな字句要素を導入する．クラスを示す表現として，次に示すようにクラス名に #! を付加した表現を用いる．

```
#!aClass
```

---

Design of an Object-Oriented PROLOG

Ken'ichi KAKIZAKI
Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology

スロットを示す表現として，次に示すようにスロット名に #@ を付加した表現を用いる．

#@aSlot

特殊変数として，自分自身は #@self，基底クラスは #@super で表現する．

### 3．3 メソッド呼び出し

利用できるメソッドは，以下に示す4種類である．

1）クラスメソッド
2）基本メソッド
3）パッケージメソッド
4）プライベートメソッド

クラスメソッドはクラスオブジェクトが認識できるメソッド，基本メソッドはインスタンスオブジェクトが認識できるメソッドである．パッケージメソッドは，クラスオブジェクトとインスタンスオブジェクトが共に認識できるメソッドで，オブジェクト指向機能をパッケージ機能的に利用する際に用いる．プライベートメソッドは，そのメソッドが定義されているオブジェクト内だけで使用できるメソッドである．

メソッド呼び出しには，明示的なメッセージ伝達と暗黙のメッセージ伝達を用いるものの2種類がある．明示的なメッセージ伝達では，次のような構文をとり，レシーバーオブジェクトを指定してメソッドを呼び出す．

オブジェクト <- メッセージ

暗黙のメッセージ伝達では，通常の述語呼び出しと同様に，メソッドとして定義されている述語をゴールとして指定する．この場合，レシーバーオブジェクトは，現在実行されているメソッドが属するオブジェクト自身である．

クラス定義をモジュール定義的にとらえた場合，他のクラスオブジェクトに対する明示的なメッセージ伝達は，Common Lisp における use-package されていないパッケージの手続き呼び出しに，暗黙のメッセージ伝達は，use-package されているパッケージの手続き呼び出しに対比させることができる．

### 3．4 クラス定義

クラスの定義は，クラスの構造を定義する defclass/4 と，メソッドを定義する defmethod/3 の2種類の述語を用いて行う．

```
defclass(クラス名, 基底クラスのリスト,
         クラス変数のリスト,
         インスタンス変数のリスト).
defmethod(クラス名, メソッド型, 節).
```

defmethod/3 のメソッド型は，3.3 節で示した4種類から選択して指定する．defmethod/3 は，クラスとメソッド型を指定する他は，従来の assert/1 と同様の機能を持つ．ソースプログラムを記述する際には，クラス定義述語を利用してプログラムを記述するのは繁雑である．このため，ソースファイルでのクラス定義に適した構文を定義し，その構文をプログラムの読み込み機能が解釈できるようにする．

### 3．5 PROLOG とオブジェクト指向機構の融合

従来の PROLOG の組込み述語は prolog クラスのパッケージメソッドとして定義されている．組込み述語を使用する場合には，prolog クラスのパッケージメソッドをクラスメソッド的に使用する方法と基本メソッド的に使用する方法の2種類で利用することができる．1つめの方法では，次に示すように実行したいゴールを指定して prolog クラスに明示的なメッセージ伝達を行う．

#!prolog <- write(abc).

2つめの方法では，利用者の定義するクラスで prolog クラスを継承する．この方法では，暗黙のメッセージ伝達機能を利用して，次に示すように単純にゴールを指定する．

write(abc).

従来の PROLOG プログラムのように，オブジェクト指向機構を考慮せずに定義されたユーザ定義述語は，user クラスのパッケージメソッドとして読み込まれる．user クラスは prolog クラスの派生クラスとして定義されており，ユーザ定義述語では組込み述語を自由に利用することができる．利用者は prolog クラスに対してメソッドを追加削除することはできないが，prolog クラスの派生クラスで組込み述語の定義を上書きすることが可能である．

シンボルやリストなどの従来の PROLOG のデータ型は，オブジェクトとして利用できる．これらのデータ型をオブジェクトとして用いる場合，改めてクラスの構造を定義することはできないが，メソッドを定義して利用することができる．

### 4．まとめ

本報告では，現在設計しているオブジェクト指向 PROLOG の概要を示した．今後の予定としては，言語仕様をさらに詳細に定義するとともに，効率的な実現方式を用いて処理系を試作し，その評価を行いたいと考えている．

### 参考文献

[1] 砒崎賢一: PROLOG のオブジェクト指向機構，情報処理学会，記号処理研究会資料，62-2, (1991).

[2] Chikayama, T.: "Unique Features of ESP", Proc. of FGCS '84, ICOT, (1984).