EWS上のPrologプログラミング環境 Prologtool

4M-3

近藤 真己*　古関 義幸**
*日本電気技術情報システム開発㈱
**日本電気㈱ C&Cシステム研究所

1 はじめに

近年、エキスパートシステムに対する関心が高まってきている。それに伴い、エキスパートシステムには、優れたマンマシンインターフェースが必要になってきた。今まで、エキスパートシステムの多くは、Lisp、Prologといった知識処理向きの高級言語を搭載したLispマシン等の専用マシンで実現されていたが、高価で、既存システムや外部機器との結合性が悪いという欠点があった。しかし、マイクロプロセッサの性能の向上と共に、汎用ＯＳ、マルチウィンドウシステムを搭載した、安価なＥＷＳ(Engineering Work Station)が出現し、専用マシン並の性能を持ちながら、既存システムとの結合性が良い知識プログラミング環境を構築出来るようになった。

このような背景から、我々は、ＵＮＩＸ[*]ベースの汎用ＥＷＳ上で、メニュー入力、グラフィック出力等の高度なマンマシンインタフェースを持ったPrologプログラミング環境を開発した。本稿では、このプログラミング環境Prologtoolと、この環境の実現の為にPrologに付加した機能について述べる。

2 Ｐｒｏｌｏｇｔｏｏｌ

Prologtoolは、ウィンドウ環境とＵＮＩＸ環境をPrologから利用出来るようにしたプログラミング環境である。このPrologtoolには、次の特徴がある。

(1) メニュー入力が可能
(2) ウィンドウ内の選択文字列の読み込みが可能
(3) 簡易グラフィックが可能
(4) ユーザーや他システムとのインターフェースの構築が容易
(5) 機能分散ができるので、Prolog処理系の負担が少なく、機能変更が容易

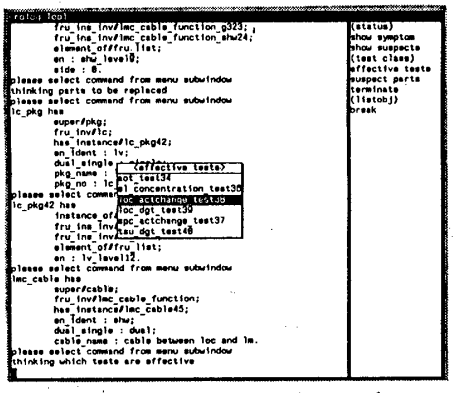

図1 Ｐｒｏｌｏｇｔｏｏｌ

3 Ｐｒｏｌｏｇの付加機能

Prologtoolの実現のために、核となるPrologにはマンマシンインターフェースや、既存システムや外部機器との結合を記述する為の、高度なシステム記述性が必要である。しかし、Prologはシステム記述性の高い言語ではない。従って我々は、エジンバラ大学で開発されたC-Prolog[1]を拡張し、Prolog言語レベルの割り込み処理機能と他プロセスとの通信機能を付加し、システム記述性を高めた。

このPrologを以下では、Prolog+ と呼ぶ。

4 プロセスのコントロールと通信機能

4－1 プロセスコントロール

Prologtoolは、ＵＮＩＸ環境下で既存システムと並列動作を行うことを前提としている。この為に、Prolog+ は次のプロセスのコントロール機能を持つ。

(1) プロセスの生成機能: Prologの子プロセスとしてＵＮＩＸ上のプログラムを実行する。
(2) シグナルの送出機能: ＵＮＩＸ上の走行中のプロセスに対してシグナルを送出する。

4－2 通信機能

Prologtoolは、既存システムや外部機器との通信を行う。この為、Prolog+ は他プロセスとの通信機能を持つ。従来は、Prologと他プロセスとの通信を、ファイル経由か、又は、通信部分を異種言語で記述して、それをリンクして使うかのどちらかで実現していた。しかし、この両方ともPrologの記述は、単純にはならない。Prolog+ では通信機能をＵＮＩＸ４．２ＢＳＤの機能であるソケットを用いて実現している。この方法によると、外部プロセスとの通信も、通常のファイルストリームとして扱える。その結果、Prologプログラムは他プロセスとの通信も、read, write等の述語を使った、ファイル入出力と全く同等な、自然な記述になった。

5 Ｐｒｏｌｏｇ＋インタプリタの割り込み機能

Prologtoolが、既存システムとインタラクティブな処理を行うために、Prolog+ には、Prolog言語レベルの割り込み処理機能がある。通常、ＵＮＩＸプログラミングでは、割り込み処理の識別をシグナルで行うが、このシグナルは、1 から31までしかなく外部プロセスからの要求に対する分岐が限られている。この為，多分岐を行うために、外部プロセスは割り込みのベクトルをProlog+ に送っている。この割り込み処理の要求の管理は、Prolog+ の子プロセスが行っている。

Prologtool : A Programming Environment for Prolog on Engineering Work Stations

Masaki Kondo [1], Yoshiyuki Koseki [2]
[1] NEC Scientific Information System development Ltd. [2] C&C Systems Research Laboratories, NEC corp.

５－１　割り込み要求管理用子プロセス

割り込み要求のデータは、次のような構造を持つ。

(1) 割り込みを要求したプロセスのＩＤ

(2) 割り込みのベクトル

(3) 割り込み処理の優先順位

このデータを、Prologプロセスが管理すると、割り込みが煩雑になった場合、通常のプログラムの進行が妨げられる可能性がある。この為，Prolog+ は、割り込み要求を管理する子プロセスを持つ。

この子プロセスは、割り込み要求を、キューに貯め、一定時間毎に、Prolog+ に対して割り込み要求を送る。

外部プロセスが、Prolog+ に対して割り込みを要求する場合、この子プロセスに対して割り込み要求のデータを送らなくてはならない。この手続きは、Ｃ言語のライブラリとしてまとめてあり、外部プロセスのプログラムでは、必要なパラメータを与えるだけである。

５－２　Ｐｒｏｌｏｇ＋の割り込み処理

Prolog+ の割り込み処理機構は、インタプリタのループの先頭において、割り込み要求があるかどうかをチェックして、割り込みが可能な状態であれば割り込み処理に分岐する。割り込み処理には次の二種類がある。

(1) マスク可能な割り込み処理

(2) マスク不可能な割り込み処理

(1) は，Prolog+ の子プロセスが管理している割り込みである。外部プロセスが、割り込みを要求し、Prolog+ がこれを受け入れた時、Prologの述語の、

```
user_interrupt(Vector)
```

が実行される。この割り込みは、多重割り込みが起きない。次の割り込み要求は、Prolog+ が、割り込み処理を終了するまで、子プロセスのキューに留まる。又、(2) はProlog+ に直接シグナルの30，31が送られたときに実行される割り込みで、

```
system_interrupt(Signal_Id)
```

が実行される。この割り込みは、マスク可能な割り込みの処理中でも割り込み処理を行うが、マスク不可能な割り込み処理中の場合、割り込みは起こらず無視される。

Prolog+ と子プロセス、外部プロセスの関係を、図２に示す。

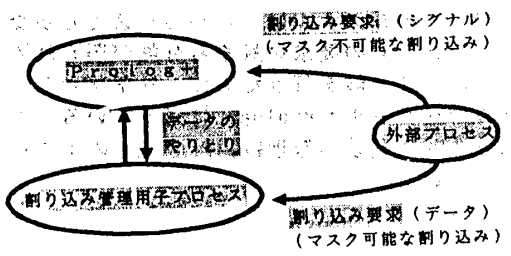

図２　プロセスの関係

６　割り込み処理と通信機能の応用

割り込み処理と通信機能によって、Prologの処理と、外部プロセスの処理とをうまく機能分散するとProlog処理系に負担を掛けない効率的な環境が実現できる。

６－１　デモンの記述

割り込み処理と通信機能を使うと、外部プロセスの状態が変わったときに、Prolog側で、その状態を知るというようなデモンの記述ができる。

例えば、外部プロセスは、その状態が変わった時自分の 'status' というソケットに状態をセットしProlog+ に対して、割り込みベクトル100 を送るようになっている場合、Prolog+ では、次のようにすれば良い。

```
/* System Status Daemon */
user_interrupt(100) :-
    (retract(sys_stat(_));true),
    seeing(CSee), see('status'),
    read(Status),
    seen, see(CSee),
    asserta(sys_stat(Status)).
```

ここでは、割り込み処理として 'status' というソケットから状態を読み、その状態を'sys_stat'という述語としてアサートしている。

このように使うことで、ポーリングの手間を掛けずに常に最新の状態を述語として保持することが出来る。

６－２　インタラクティブなインターフェース

ユーザーの立場から、エキスパートシステムのユーザーインターフェースをみると、エキスパートシステムが入力を必要としなければ、質問してこないというような不満がある。しかし、Prolog+ の割り込み機能とメニュー等を組み合わせると、ユーザーオリエンテッドなインターフェースを構築できる。

７　終わりに

本稿では、Prologtoolと、その実現のために開発したシステム記述性の高いProlog+ について述べた。エキスパートシステムにとって、操作性の良いマンマシンインターフェースは、重要な要素である。現在、我々は、このPrologtoolの上で、知識プログラミングシステムPeace[2]を用いて、電子交換機の故障診断エキスパートシステムSHOOTX[3] を開発している。

参考文献

[1] F.Pereira, "C-Prolog User's Manual Version 1.5", EdCAAD (1984)

[2] 若杉他,"Prolog言語を用いた知識プログラミングシステムPeace", 情報処理学会第33回全国大会 4M-2(1986) 1349-1350

[3] 古関他,"マルチプル知識利用故障診断エキスパートシステムSHOOTX",情報処理学会第33回全国大会 1L-6(1986) 1175-1176

[*] UNIXは、米国ベル研究所で開発したソフトウェアである。