法律エキスパートシステム・実験システム Ver.1.2

吉野 一
明治学院大学 法学部
yoshino@mh.meijigakuin.ac.jp
桜井 成一朗
東京工業大学 大学院総合理工学研究科
古関 義幸
日本電気株式会社 C&C 研究所
近藤 浩康
日本電気株式会社 第一C&C システム事業本部
元山 沢、久保田 健一
株式会社アスキング システム開発部

1 はじめに

法律エキスパートシステムの中核は法的推論システムである。それは、法律家が行う法的推論の実際にできるだけ近似な推論をコンピュータ上に実現するものである。これを実現していくためには、現行の法律の分析を通して法対系を明らかにしながら、法律知識ベースを記述し、それを説例に対して推論実行を行なう実験を繰り返すことを通して問題点を明らかしていくとともに、知識ベースと推論方式を改良していくことが重要である。これを行なうためのシステムとして、**法律エキスパートシステム・実験システム**Ver.1.2を開発した。

本稿では、開発されたシステムについて、昨年度の成果報告との重複を避け、本年度に開発した部分を中心に報告する。読者におかれては昨年の報告を併せて参照されたい。

参照 平成7年3月発行 研究成果報告書「法律エキスパートシステムの開発研究」 P.12

2 システム構成

本システムは、知識ベース作成支援と推論実行システムとから構成される。本年度は、昨年度に開発した知識ベースエディタ（図1～図2）を中心とする知識ベース作成支援システムを充実させると共に、CPF表記された知識ベースを解釈して法的推論を実行する推論実行システムを開発した。

なお、知識ベース作成支援システムはTcl/Tkが動作する環境で、推論システムはSICStus Prologが動作する環境で、利用可能である。

図1の左側のダイアログ(Legal Expert System Supervisor)は、このシステム全体のメインメニューである。メインメニューの「Edit Data」ボタンをクリックすると、左側のダイアログ(Edit Data)が現れる。Edit Dataダイアログの「Rule Editor」ボタンをクリックすると図2の知識エディターメイン画面が表示される。

ソフトウエア構成

OS	UNIX
メインメニュー	Tcl/Tk
知識エディタ	Tcl/Tk (一部機能にC言語)
推論エンジン	SICStus Prolog2.1

3 推論実行システム

推論実行システムは、CPFで記述された知識ベースを利用し、同じくCPF表記された説例ファクトファイルに対して、推論実行を行なう。また、デバッグおよび推論の理解を目的として、トレース機能、スパイ機能、ビジュアルトレーサ機能を持つ。これらの表示は、日本語テンプレート機能を利用することにより、簡易自然言語文として表示することができるため、CPFないしはPrologでそのまま表示する場合に比べてはるかに可読性の高い形式で表示ができる。また、述語の概念階層を用いて知識ベースをマクロ展開することにより、簡潔な表現で知識ベースの記述が可能になっている。また、各種推論パラメータの設定、実行の制御などを、GUIを用いた推論用ダイアログウィンドウにより行なえるので、使いやすくなっている。

3.1 推論エンジン

推論エンジンは、CPFを直接実行するPrologメタインタプリタとして実装されている。SICStus Prologで記述されているのでポータビリティがよい。ここでは、推論ダイアログウィンドウ、簡易自然言語表示機能、階層展開、ファクトファイルフラット化について説明する。

3.1.1 推論ダイアログウィンドウ（図3）

図3は、知識エディターの「Debugger」メニュー内の「Files fot Inference」ボタンをクリックすることにより表示されるダイアログである。
このGUIを通して、以下のような機能が制御できる。

- ルール、ゴール、ファクト、Taxonomyなどの各ファイルをあらかじめ推論用に指定しておく。

・使用する推論エンジンのタイプを選択する。但し、現在のところ一種類のエンジンのみ。

・推論トレース実行時のトレースの深さを設定できる。数値により任意のレベルが指定できる。

・ルール中の全コンセプトを予めサーチしておき、スパイポイント設定ダイアログで選択して、スパイポイントを設定する。設定は、あらかじめルールファイルを検索して作られたメニューを通して行なえるので、マウスにより選択するだけで、タイプをせずに入力できる。

・「Quick run」ボタンを押すと、ゴールファイルで指定されているゴールが一覧表示されるのでそれを選択すると、そのゴールについての推論が実行される。(図4) これも、あらかじめルールファイルを検索して作られたメニューを通して行なえるので、マウスにより選択するだけで、タイプをせずに入力できる。

図4は、「Quick run」ボタンを押してゴールメニューを表示させた画面である。
ゴールIDとトップゴールコンセプト名が表示されるので、それを選択してクリックすると推論がスタートする。

・ビジュアルトレーサを起動して、推論実行のトレースを、ツリー表示機能を使ってビジュアルに表示する。(図5)

図5はビジュアルトレーサーの実行画面である。
図中の箱は推論実行をトレースする形で次々に描画される。

・実行制御スイッチにより、以下のような実行時制御を行なえる。

- トレースのON/OFF
- スパイのON/OFF
- フラット化ファクトの参照のON/OFF
- マクロ化ルールの参照のON/OFF
- 解答一時停止スイッチ (推論実行時に答えが1つ出るごとに一時停止) のON/OFF
- 自然言語表示のON/OFF

3.1.2 簡易自然言語表示機能 (図6～図8)

CPFで記述されたルール、ゴール、ファクトを、自然言語テンプレートを利用して簡易自然言語に変換して表示する機能を開発した。これにより、CPFそのものでの表示に比べて、格段に可読性が向上した。これを推論システムに組み込み、推論エンジン実行時のトレースの表示、ゴールの表示などに利用している。また、説明のためのビジュアルトレーサの表示にも利用する予定である。例えば、下記の例題のようにCPFが入れ子になっている場合でも簡易自然言語による表示が可能である。

<元のルール>

```
'is_valid' (IS_VALID, [
    abj:sen(SEN31, [
        'is_obligatory' (IS_OBLIGATORY, [
            goa:A,
            obj:X,
            tim:T])
        ]),
    goa:A,
    tim:T
]).
```

<自然言語テーブル>

```
nl_table('is_obligatory' (IS_OBLIGATORY,
            [goa:GOA, obj:OBJ, tim:TIM]),
['「', (OBJ),'義務がある」']).

nl_table('is_valid' (IS_VALID,
            [abj:ABJ, goa:GOA, tim:TIM]),
[('時 点', TIM,'に'), (ABJ,'が'),'効力がある']).
```

<ルールから生成される自然言語>

時点Tに「X義務がある」が効力がある

ここで、自然言語テーブルで定義される表示形式内で変数を扱うことができることに注意されたい。変換実行時に変数のままである場合は変数名表示を行ない、インスタンシエートされている場合は、その値を表示する。なお、変数がインスタンシエートされていない場合にはその部分を表示しないモードも選択できるようにしている。これにより不要な変数を表示しないようにできる。

図6は、ルール毎に自然言語を表示する機能の実行画面である。
コンセプト名をマウスで選択し、「Debugger」メニュー内の「NaturalLanguage (for test)」コマンドを選択すると、そのルールについて登録されている自然言語が図6のように表示される。

図7は、「生成文表示」ボタンをONにしてQuick Runのゴールを選択したあとの画面である。「条文表示」「法ルール文表示」「生成文表示」のいずれかのボタンがONになっている時にゴールを選択すると、選択されたゴールについての自然言語を表示してから推論実行に移る。

図8は推論画面 (自然言語表示モード) である。左側の推論実行ウィンドウ上に成功したタームが現れた時に、それに対応する自然言語を右側のウィンドウに表示する。
条文ウインドウは、ルールに対応する条文

を表示する。法ルール文ウインドウは、ルールに付随しているコメントを表示する。生成文ウインドウは、ルールの自然言語テーブルをもとにしてシステムが生成した文を表示する。

3.1.3 階層展開

ＣＰＦで使用される述語に対して、概念階層を定義できるように拡張することによって、一種の継承機能が実現できる。このようにすれば、共通の論理構造を持つ複数の述語それぞれに対して複数のルールを用意することなしに、共通の一般的ルールを記述するだけで、簡潔に知識ベースを記述することが可能になる。

これを実現する方法としては、順序ソートロジックの採用などが考えられるが、本推論システムでは、現状の推論エンジンに手を入れずにそのままこの機能を実現する方法として、概念階層で定義された述語を含むＣＰＦ記述を、概念階層に従ってあらかじめ展開しておくおくことにより、同様の機能を実現する方式を採り入れた。このような実現方式をとったことにより、推論エンジンにおける実行時の変数の管理などの複雑な処理を導入せずに、概念階層を利用した簡潔な知識ベースを利用することが可能になった。

本機能は以下のような規則に従って、ＣＰＦルールを辞書ファイルに記述された概念階層をもとにして階層展開行なう。

1. ＨＥＡＤ部だけに現れる述語は置き換えない。また、ＢＯＤＹ部だけに現れる述語も置き換えない。
2. ＨＥＡＤ部とＢＯＤＹ部に共通に現れる述語のうち、ＩＤが同じものについてのみ、概念階層に定義された子概念の数だけ置き換える。
3. 複数の述語が一つのルールの中で展開される場合は，両方の子概念の組み合わせの数だけ置き換える。
4. ＩＤが同じでも共通の下位概念が存在しなければ展開しない。

例題１（一述語の展開）

```
[input]
tax(申込, 意志表示). %申し込みは意思表示である
tax(承諾, 意志表示). %承諾は意思表示である

sen('3aa1', [効力発生(RID, [
        obj:意志表示(ID_1,_),
        tim:T])
    <- 意志表示(ID_1, [..]) &
        到達(ID_2, [tim:T])]).


[output]
sen('mac_001_3aa1', [効力発生(RID, [
        obj:申込(ID_1,_),
        tim:T])
    <- 申込(ID_1, [..]) &
        到達(ID_2, [tim:T])]).
sen('mac_002_3aa1', [効力発生(RID, [
        obj:承諾(ID_1,_),
        tim:T])
    <- 承諾(ID_1, [..]) &
        到達(ID_2, [tim:T])]).
```

例題２（複数述語の組合せ展開）

```
[input]
tax(申込, 意志表示).
tax(承諾, 意志表示).
tax(smp1, smp0).
tax(smp2, smp0).

sen('3aa1', [効力発生(RID,
    [obj:意志表示(ID_1,_),abj:smp0(ID_2,_)])
    <- 意志表示(ID_1, [..]) & smp0(ID_2,_)]).

[output]
sen('mac_001_3aa1', [効力発生(RID,
    [obj:意志表示(ID_1,_),abj:smp1(ID_2,_)])
    <- 意志表示(ID_1, [..]) & smp1(ID_2,_)]).
sen('mac_002_3aa1', [効力発生(RID,
    [obj:意志表示(ID_1,_),abj:smp2(ID_2,_)])
    <- 意志表示(ID_1, [..]) & smp2(ID_2,_)]).
sen('mac_003_3aa1', [効力発生(RID,
    [obj:申込(ID_1,_),abj:smp0(ID_2,_)])
    <- 申込(ID_1, [..]) & smp0(ID_2,_)]).
sen('mac_004_3aa1', [効力発生(RID,
    [obj:申込(ID_1,_),abj:smp1(ID_2,_)])
    <- 申込(ID_1, [..]) & smp1(ID_2,_)]).
sen('mac_005_3aa1', [効力発生(RID,
    [obj:申込(ID_1,_),abj:smp2(ID_2,_)])
    <- 申込(ID_1, [..]) & smp2(ID_2,_)]).
sen('mac_006_3aa1', [効力発生(RID,
    [obj:承諾(ID_1,_),abj:smp0(ID_2,_)])
    <- 承諾(ID_1, [..]) & smp0(ID_2,_)]).
sen('mac_007_3aa1', [効力発生(RID,
    [obj:承諾(ID_1,_),abj:smp1(ID_2,_)])
    <- 承諾(ID_1, [..]) & smp1(ID_2,_)]).
sen('mac_008_3aa1', [効力発生(RID,
    [obj:承諾(ID_1,_),abj:smp2(ID_2,_)])
    <- 承諾(ID_1, [..]) & smp2(ID_2,_)]).
```

3.1.4 ファクトファイルフラット化

ＣＰＦは入れ子構造で記述ができるが、入れ子の中で記述された部分をファクトとして参照することはできない。これを可能にするために、以下のような条件にある述語を、ファクトファイルの中から抽出しフラット化する機能を実現している。

1. 格リスト中に定義されている述語のみを再帰的に展開する。
2. 展開する述語はＣＰＦの記述に準拠しているものを対象とする。

ファクトデータのフラット化の例

```
[input]
sen(c7a_9_1, [
  'stop' (stop_c7a_9, [
        abj:'a construction machinery',
        cau:'inferior' (inferior_c7a_9, [
              abj:'the engine' (engine_c7a_9, [
                    obj:'a construction machinery'
                    ])
        ]),
        man:'often',
        obj:'work' (work_c7a_9,
              [abj:'a construction machinery']),
        tim:'8_10'
    ])
]).

[output]
sen('c7a_9_1_001', ['inferior' (inferior_c7a_9, [
            abj:'the engine' (engine_c7a_9, [
                  obj:'a construction machinery'
                  ])
             ]).

sen('c7a_9__002', ['the engine' (engine_c7a_9, [
            obj:'a construction machinery'])]).

sen('c7a_9_1_003', ['work' (work_c7a_9, [
            abj:'a construction machinery'])]).
```

4　知識ベース作成支援システム

昨年度に開発した知識ベース作成支援システムをより使いやすいものとするために、以下のような機能拡張を行なった。

4.1　辞書からのルールタームの生成（図９～図１０）

マウスで辞書ファイルの一部を選択し、この機能を実行することにより、CPFのタームの枠組を自動作成し、ルールエディタに挿入する。この機能により、ルールの作成が迅速化されるだけでなく、タイプミスなどによる誤りが防止できるようになった。

実行例

＜辞書例＞

```
dic(d1,
    (e, is_obligatory,
      is_obligatory,
      _,
      [[nec, date(95:11:7:15:6:46)]]),
    cases_of_concept([[goa, obj, tim], [_]]),
    _,
    _,
    super_sub_concept([
       sup:[_], sub:[_], syn:[_], ant:[_]])).
```

↓ 変換

```
'is_obligatory' (IS_OBLIGATORY, [
      goa:GOA_IS_OBLIGATORY,
      obj:OBJ_IS_OBLIGATORY,
      tim:TIM_IS_OBLIGATORY
      ])
```

図９は、ターム生成コマンド「Apply dictionary to rule」を起動する画面である。使用する辞書の「dic」の部分をマウスで選択し、このコマンドをクリックする。

図１０は実行結果である。図９で選択したis_obligatoryについてのルールが画面右側のルールエディタ上に生成されている。

4.2　知識ベースチェック機能

迅速に正確な知識ベースを記述するためのいくつかの機能拡張を知識ベースエディタに対して行なった。

4.2.1　ルール／ファクトチェック（図１１）

知識ベースエディタ上に読み込まれているルールまたはファクトの文法チェックを行う機能を付加した。

図１１は、ルールチェックが終了した時の画面である。知識エディターの「Debugger」メニュー内の「Rule/Fact check」コマンドを起動すると図１１のようなチェッカーウィンドウが現れて、シンタックスチェックを行う。

4.2.2　辞書チェック

知識ベースエディタ上に読み込まれている辞書の文法チェックを行う機能を付加した。

4.2.3　環境ファイル読み込み機能（図１２）

ルール、ファクトなどの推論用ファイルを一括して設定、選択する機能を実現した。これにより、複数ファイルにまたがる推論用ファイルを一括して指定できるので、推論実行実験が容易になった。

図１２は環境ファイル読み込み用ダイアログである。ホームディレクトリに拡張子「.env」の環境ファイルがある場合は知識エディター起動時に自動的に表示される。環境ファイルを選択すると、それに記述されている推論関係のファイルが読み込まれる。

環境ファイル例

```
# Environment File
#   推論用ファイルと、起動時にエディタに読み込むファイルの指定
# 書式:
#   ID ファイル名
#
#   ID の種類
# RULE : ルールファイル指定とルールエディタ表示ファイル指定
```

```
# FACT : ファクトファイル指定とファクトエディタ
表示ファイル指定
# GOAL : ゴールファイル指定
# TAX  : タクソノミーファイル指定
# DIC  : 辞書エディタ表示ファイル指定
# TXT1 : テキストエディタ１表示ファイル指定
# TXT2 : テキストエディタ２表示ファイル指定
# ENG  : 使用する推論エンジンタイプ
#
CMT  2月13日付最新データ
RULE ~/newpro/work5/0213/q5ruly.euc
FACT ~/newpro/work5/0213/c7b_q5y.euc
GOAL ~/newpro/work5/0213/q5goay.euc
ENG t_eng3
DIC ~/newpro/work5/0213/q5dic.euc
TXT1 ~/newpro/work5/0213/q5goay.euc
```

5　おわりに

以上で、知識ベースを記述しそれを設例に対して推論実験を行なう法律エキスパートシステム・実験システムVer1.2の紹介を終わる。われわれは本システムを利用して、国際統一売買法の一部についてCPFによる知識ベースの開発を行ない、設例に対して法的推論を行なった。この実験を通して、本アプローチの有効性を確認できたとともに、いくつかの将来の研究課題が明らかになった。本システムを基に、知識を一段と追加し、推論方式を改良・追加していくことにより本格的な法律エキスパートシステムを展開していくことが可能になったと思われる。

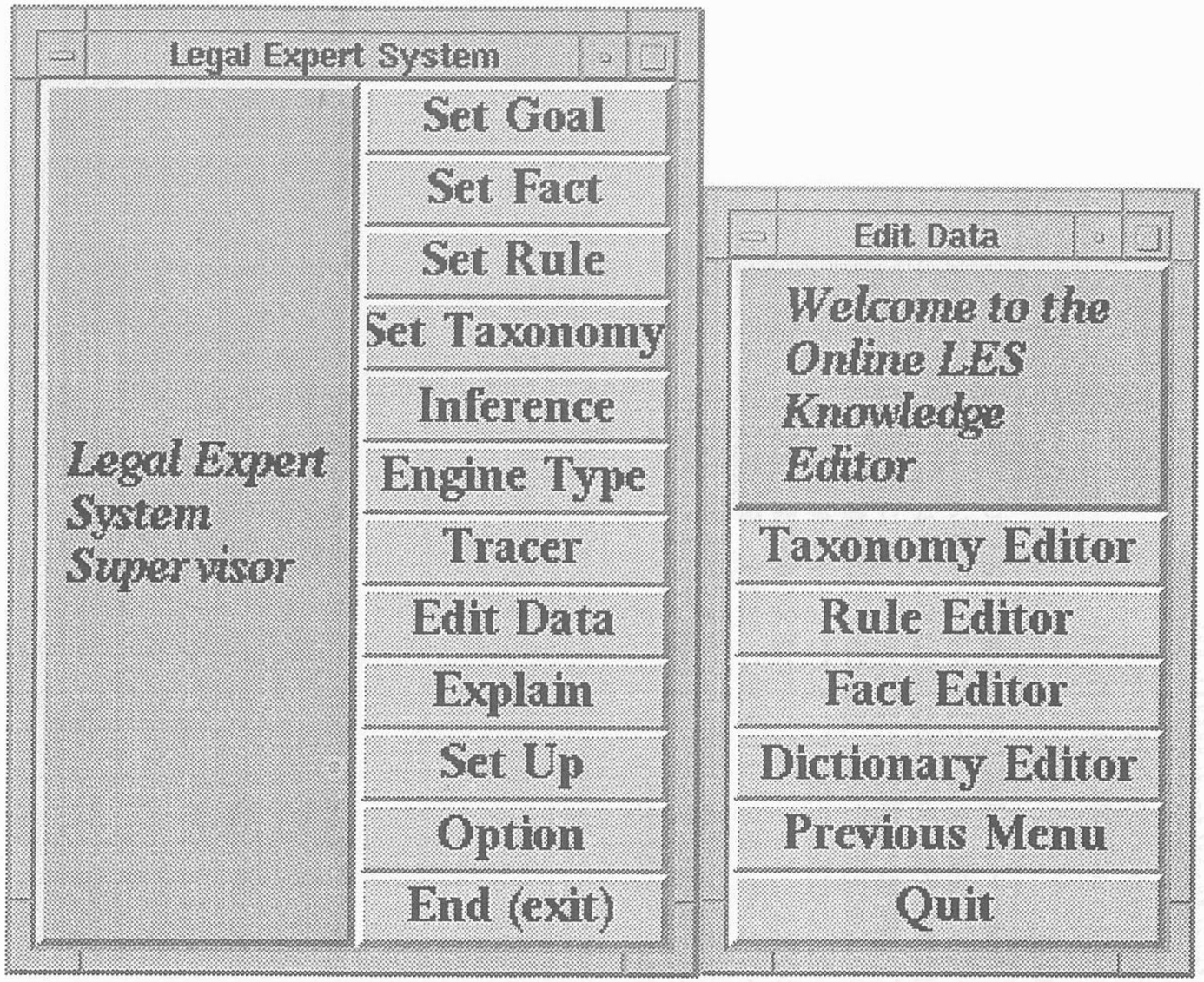

Legal Expert System
Legal Expert System Supervisor
Set Goal
Set Fact
Set Rule
Set Taxonomy
Inference
Engine Type
Tracer
Edit Data
Explain
Set Up
Option
End (exit)
Edit Data
Welcome to the Online LES Knowledge Editor
Taxonomy Editor
Rule Editor
Fact Editor
Dictionary Editor
Previous Menu
Quit

図1

Knowledge Editor Version 1.0

Change Text | Dictionary | Fact Editor | Debugger | Language

FILE | EDIT | SEARCH | OPTION | HELP

Textual Representation 1 < q5goay.euc > (1:1)

```
%%
goal('perform to pay',[
'perform'(PERFORM,[
      agt:A,
      obj:'obligation'(OBLIGATION,[
          goa:A,
          obj:'pay'(PAY,[
                  agt:AGT_PAY,
                  goa:B,
                  obj:OBJ_PAY,
                  tim:T
          ]),
          tim:T
```

Textual Representation 2 < q5ruly.nl > (1:1)

```
nl_table('is_obligatory'(IS_OBLIGATORY,[goa:GOA_IS_OBLIGATORY,obj:OBJ_
IS_OBLIGATORY,tim:TIM_IS_OBLIGATORY]),
        ['時点',TIM_IS_OBLIGATORY,'に'),(OBJ_IS_OBLIGATORY,'の'),'義
務がある']).
nl_table('is_valid'(IS_VALID,[abj:ABJ_IS_VALID,goa:GOA_IS_VALID,tim:TI
M_IS_VALID]),
        ['時点',TIM_IS_VALID,'に'),(ABJ_IS_VALID,'という'),'効力がある']).
nl_table('has_right'(HAS_RIGHT,[goa:GOA_HAS_RIGHT,obj:OBJ_HAS_RIGHT,ti
m:TIM_HAS_RIGHT]),
        ['時点',TIM_HAS_RIGHT,'に',OBJ_HAS_RIGHT,'という権利がある']).
nl_table('become_valid'(BECOME_VALID,[abj:ABJ_BECOME_VALID,goa:GOA_BEC
OME_VALID,tim:TIM_BECOME_VALID]),
```

Dictionary Editor < q5dic.euc > (1:1)

```
% q1rul 歴辞書%   1995/11/10(Fri)
%!
%! $Header: /home/swada/mac/work5/q5.dic,v 1.3 1996/01/12 15:38:11 swa
da Exp swada $
%!

dic(d1,
    (e,is_obligatory,is_obligatory,_,[[nec,date(95:11:7:15:6:46)]]),
    cases_of_concept([[goa,obj,tim],[_]]),
    -,
    -,
    super_sub_concept([sup:[_],sub:[_],syn:[_],ant:[_]])).
```

Rule Editor < q5ruly.euc > (1:1)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
%   3 義務の存在       revised at 95.08.01, 10.05
%
sen(3,[
%   [3]時点Tに X義務がある
%   X is obligatory or A at time T(in legal world)
'is_obligatory'(IS_OBLIGATORY,[
    goa:A,
    obj:X,
    tim:T
])
<-
% 31 時点Tに文「X義務がある」が効力がある
% "X is obligatory for A" is valid at time T.
'is_valid'(IS_VALID,[
    abj:sen(SEN31,[
        'is_obligatory'(IS_OBLIGATORY,[
            goa:A,
            obj:X,
            tim:T
        ])
        ]),
    goa:A,
    tim:T])
]).

%%%%%%%%%% is_obligatory と must を結び付けるルールを作った
%%%%%%%%%% 義務がある」と「ねばならない」を結び付けるルールAdded by
HY on 1995/12/ 4(Mon) 16:12

% 以下のルール(31)はis_obligatory < mustという階層辞書にしたがってルー
ルの具体かが行われるといらなくなる。
% 31
sen(31,[
'is_valid'(IS_VALID,[
    abj:sen(SEN_IS_OBLIGATRORY,[
        'is_obligatory'(HAS_DUTY,[
            goa:A,
            obj:X,
            tim:T
        ])
    ]),
    goa:A,
```

図2

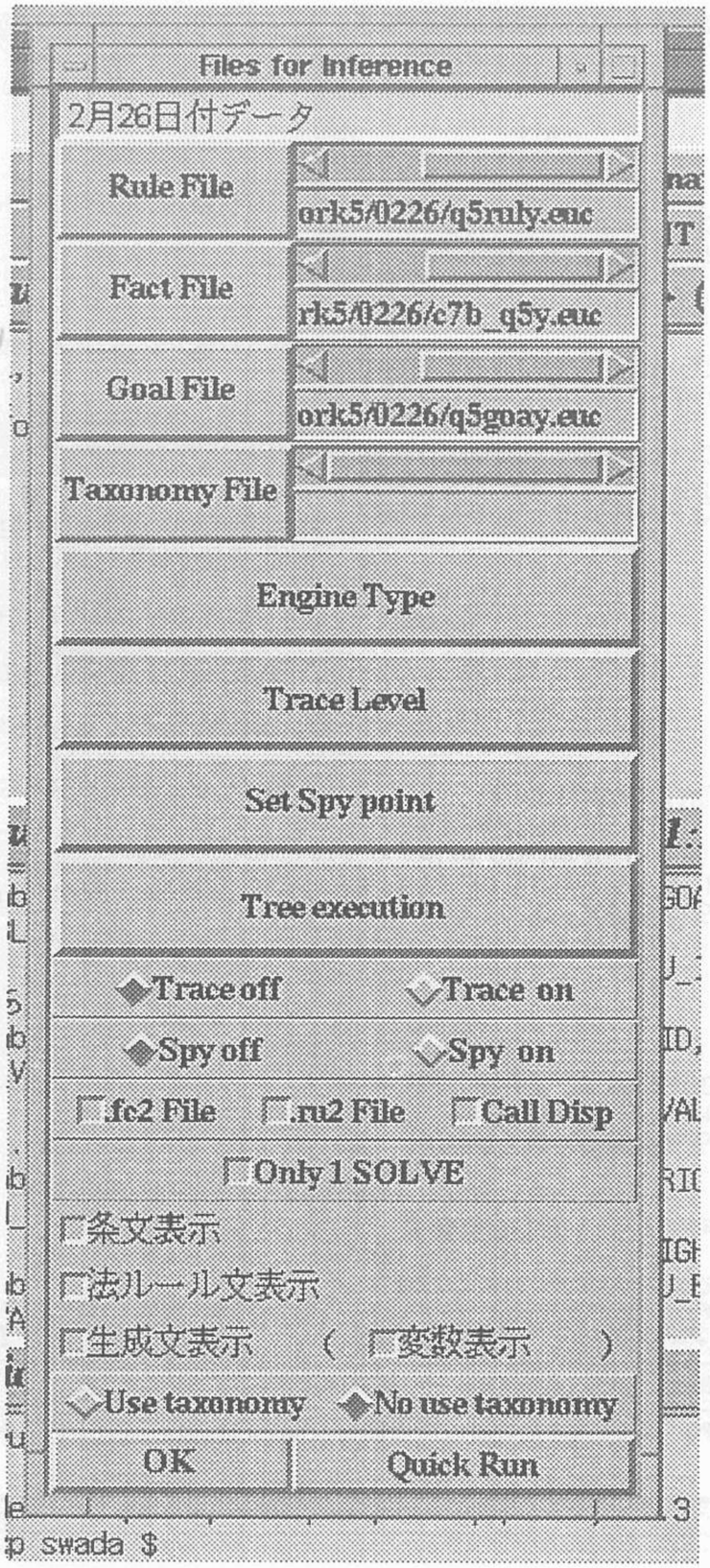

Files for Inference
2月26日付データ
Rule File
ork5/0226/q5ruly.euc
Fact File
rk5/0226/c7b_q5y.euc
Goal File
ork5/0226/q5goay.euc
Taxonomy File
Engine Type
Trace Level
Set Spy point
Tree execution
Trace off
Trace on
Spy off
Spy on
.fc2 File
.ru2 File
Call Disp
Only 1 SOLVE
条文表示
法ルール文表示
生成文表示
(変数表示)
Use taxonomy
No use taxonomy
OK
Quick Run
swada $

図3

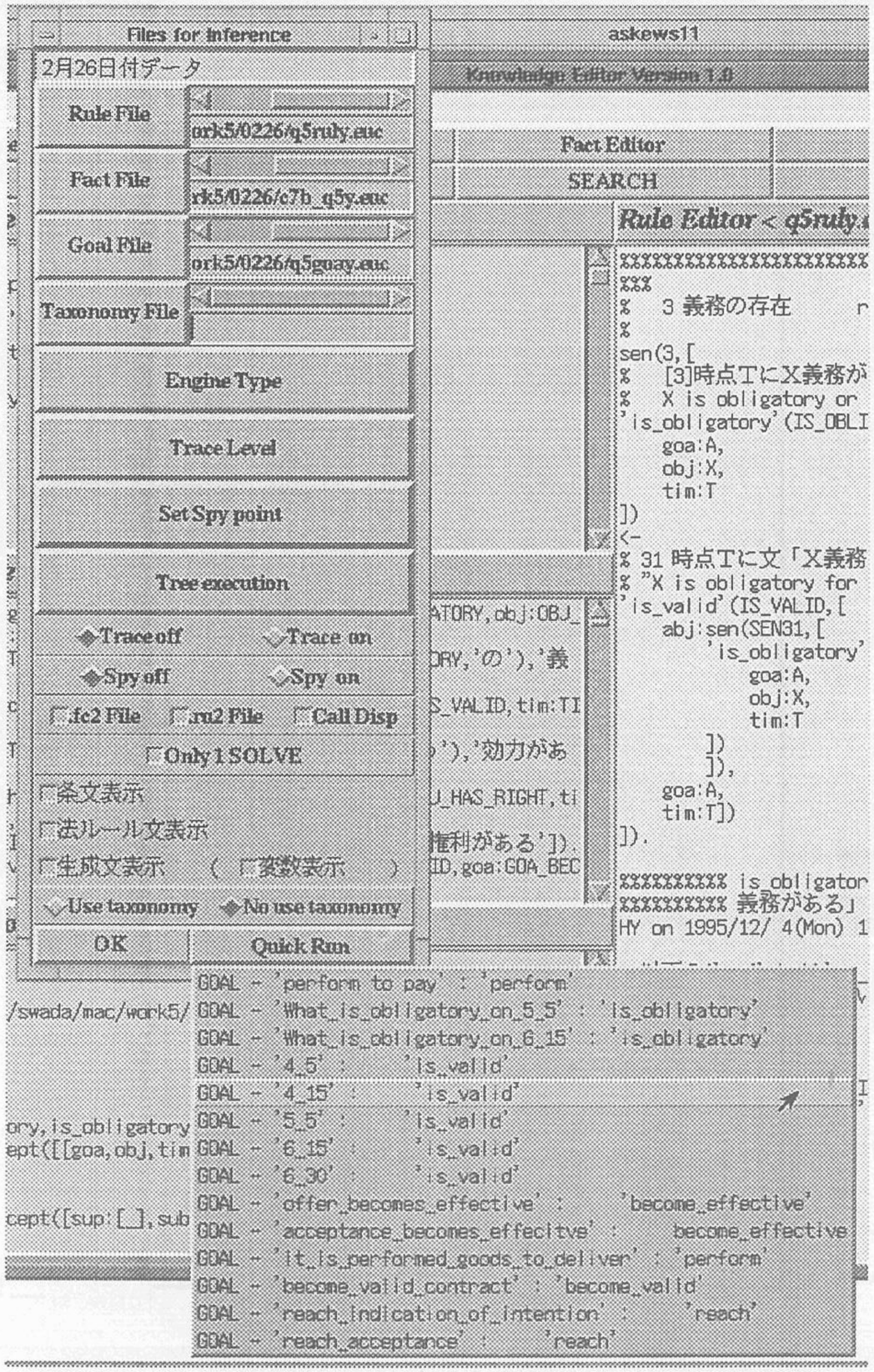

Files for inference
2月26日付データ
Rule File
ork5/0226/q5ruly.euc
Fact File
rk5/0226/c7b_q5y.euc
Goal File
ork5/0226/q5goay.euc
Taxonomy File
Engine Type
Trace Level
Set Spy point
Tree execution
Trace off
Trace on
Spy off
Spy on
.fc2 File
.ru2 File
Call Disp
Only 1 SOLVE
条文表示
法ルール文表示
生成文表示
変数表示
Use taxonomy
No use taxonomy
OK
Quick Run
askews11
Knowledge Editor Version 1.0
Fact Editor
SEARCH
Rule Editor < q5ruly.
%%%
% 3 義務の存在
%
sen(3,[
% [3]時点Tに X義務が
% X is obligatory or
'is_obligatory'(IS_OBLI
goa:A,
obj:X,
tim:T
])
<-
% 31 時点Tに文「X義務
% "X is obligatory for
'is_valid'(IS_VALID,[
abj:sen(SEN31,[
'is_obligatory'
goa:A,
obj:X,
tim:T
])
]),
goa:A,
tim:T])
]).
%%%%%%%%%% is_obligator
%%%%%%%%%% 義務がある」
HY on 1995/12/ 4(Mon) 1
/swada/mac/work5/
ory,is_obligatory
ept([[goa,obj,tim
cept([sup:[_],sub
GOAL - 'perform to pay' : 'perform'
GOAL - 'What_is_obligatory_on_5_5' : 'is_obligatory'
GOAL - 'What_is_obligatory_on_6_15' : 'is_obligatory'
GOAL - '4_5' : 'is_valid'
GOAL - '4_15' : 'is_valid'
GOAL - '5_5' : 'is_valid'
GOAL - '6_15' : 'is_valid'
GOAL - '6_30' : 'is_valid'
GOAL - 'offer_becomes_effective' : 'become_effective'
GOAL - 'acceptance_becomes_effecitve' : become_effective
GOAL - 'it_is_performed_goods_to_deliver' : 'perform'
GOAL - 'become_valid_contract' : 'become_valid'
GOAL - 'reach_indication_of_intention' : 'reach'
GOAL - 'reach_acceptance' : 'reach'

図4

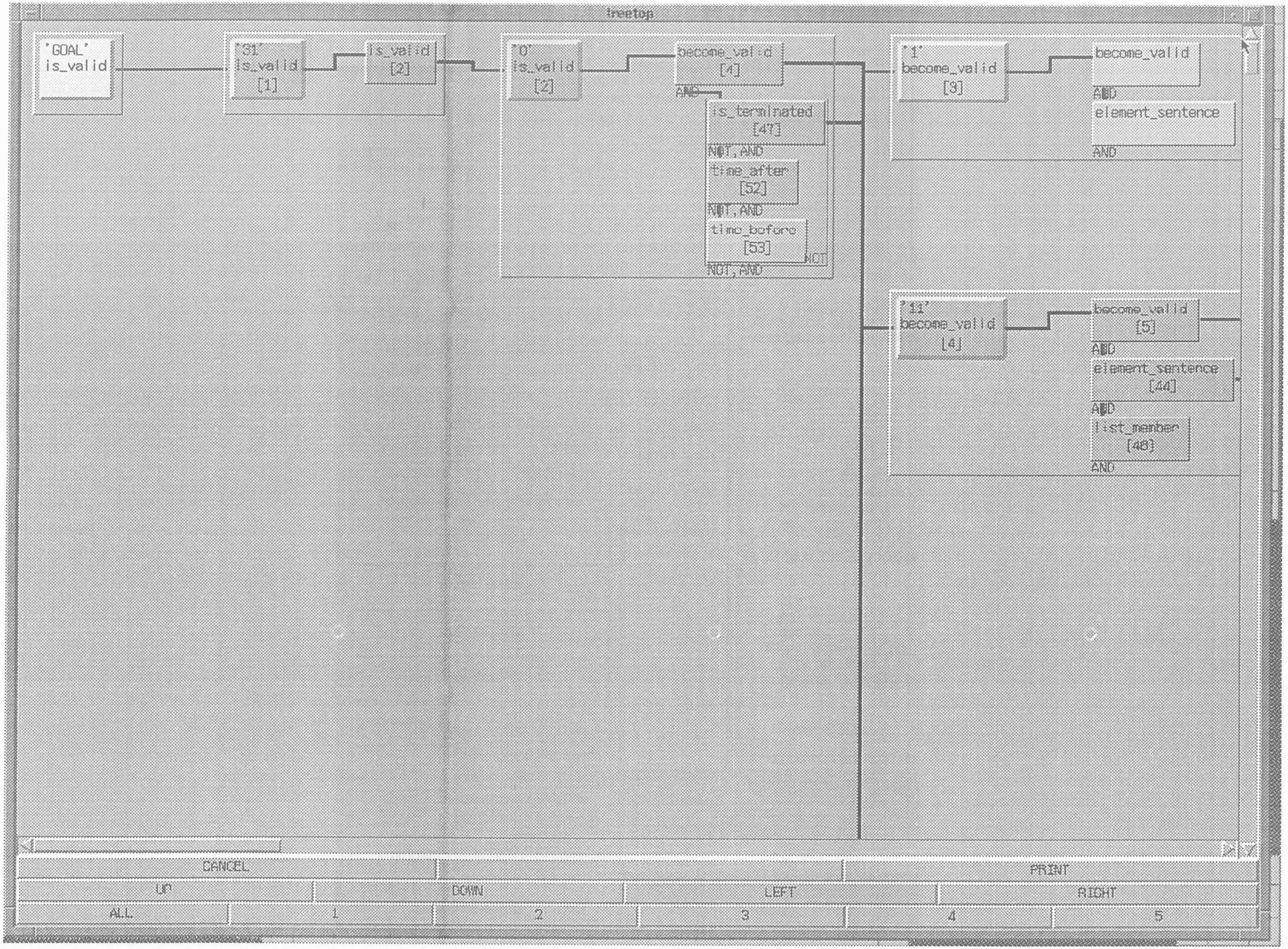

treetop
'GOAL'
is_valid
'31'
is_valid
[1]
is_valid
[2]
'0'
is_valid
[2]
become_valid
[4]
AND
is_terminated
[47]
NOT,AND
time_after
[52]
NOT,AND
time_before
[53]
NOT
NOT,AND
'1'
become_valid
[3]
become_valid
AND
element_sentence
AND
'11'
become_valid
[4]
become_valid
[5]
AND
element_sentence
[44]
AND
list_member
[48]
AND
CANCEL
PRINT
UP
DOWN
LEFT
RIGHT
ALL
1
2
3
4
5

図5

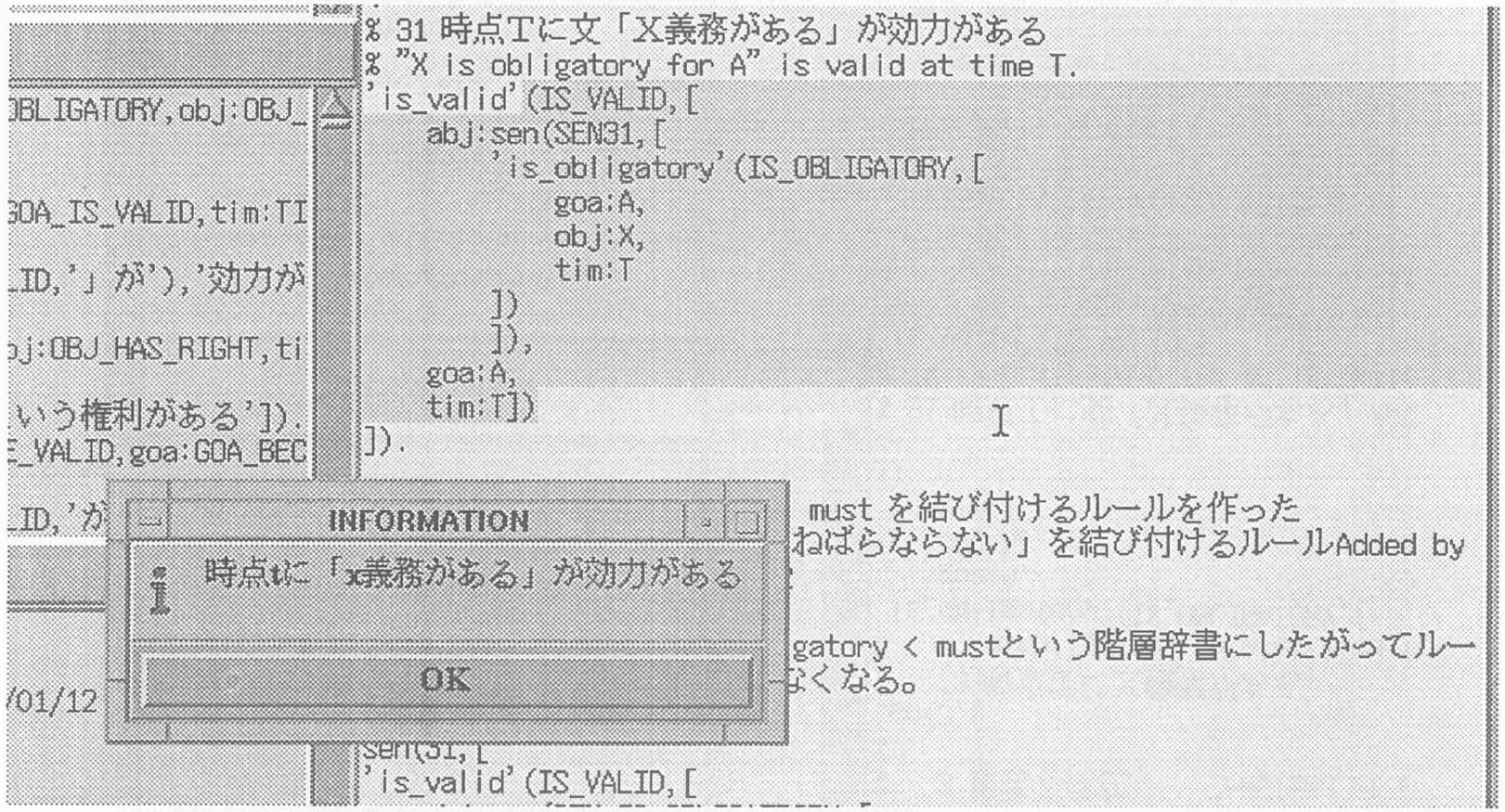
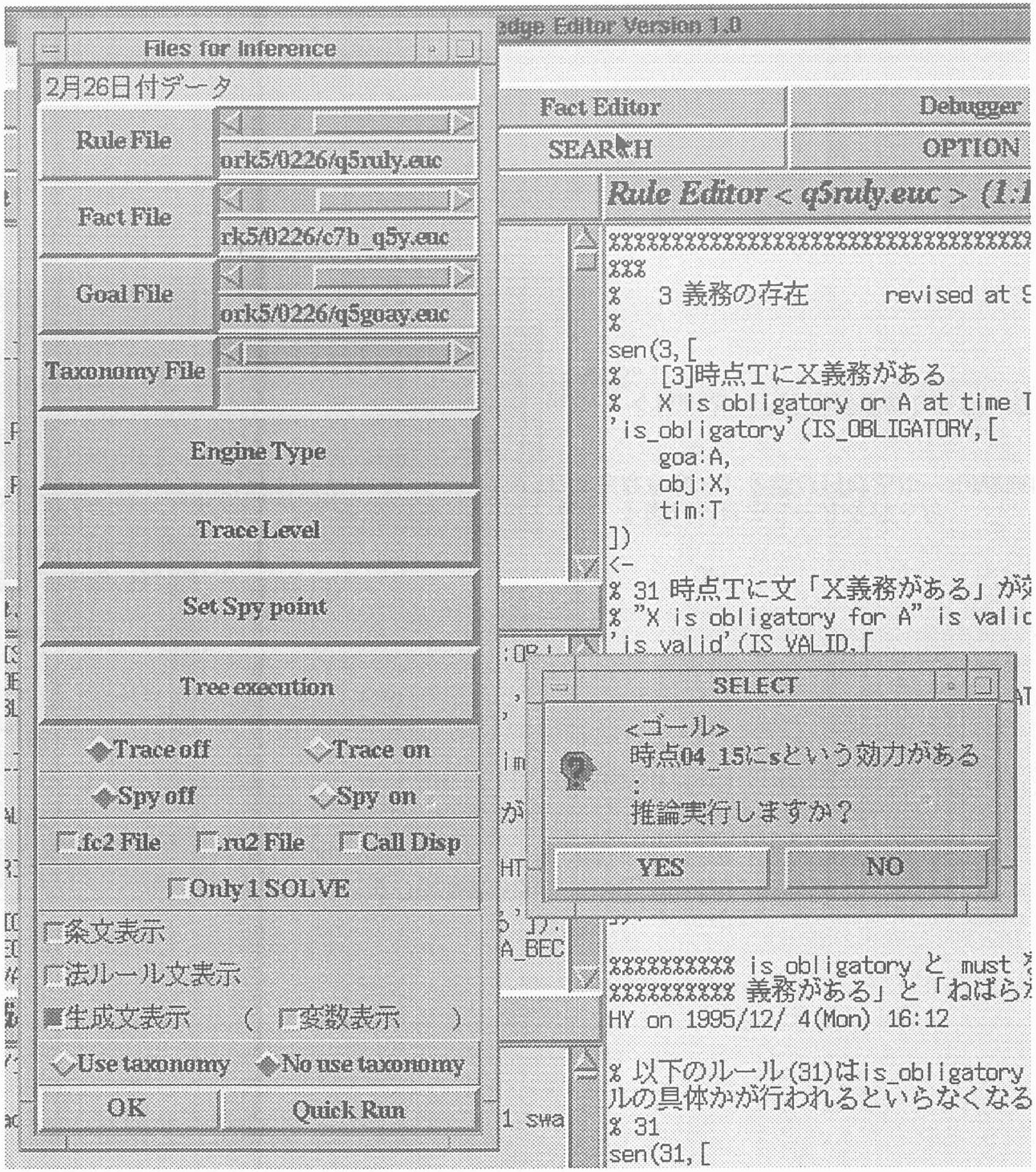

% 31 時点Tに文「X義務がある」が効力がある
% "X is obligatory for A" is valid at time T.
'is_valid'(IS_VALID,[
abj:sen(SEN31,[
'is_obligatory'(IS_OBLIGATORY,[
goa:A,
obj:X,
tim:T
])
]),
goa:A,
tim:T])
]).
INFORMATION
時点tに「x義務がある」が効力がある
OK
must を結び付けるルールを作った
ねばならない」を結び付けるルールAdded by
gatory < mustという階層辞書にしたがってルー
なくなる。
sen(31,[
'is_valid'(IS_VALID,[

図6


dge Editor Version 1.0
Files for Inference
2月26日付データ
Rule File
ork5/0226/q5ruly.euc
Fact File
rk5/0226/c7b_q5y.euc
Goal File
ork5/0226/q5goay.euc
Taxonomy File
Engine Type
Trace Level
Set Spy point
Tree execution
Trace off
Trace on
Spy off
Spy on
.fc2 File
.ru2 File
Call Disp
Only 1 SOLVE
条文表示
法ルール文表示
生成文表示　（　変数表示　）
Use taxonomy
No use taxonomy
OK
Quick Run
Fact Editor
Debugger
SEARCH
OPTION
Rule Editor < q5ruly.euc >
%　3 義務の存在　　revised at
sen(3,[
%　[3]時点Tに X義務がある
%　X is obligatory or A at time T
'is_obligatory'(IS_OBLIGATORY,[
goa:A,
obj:X,
tim:T
])
<-
% 31 時点Tに文「X義務がある」が
% "X is obligatory for A" is valid
'is_valid'(IS_VALID,[
SELECT
<ゴール>
時点04_15にsという効力がある
推論実行しますか？
YES
NO
%%%%%%%%%% is_obligatory と must
%%%%%%%%%% 義務がある」と「ねばら
HY on 1995/12/ 4(Mon) 16:12
% 以下のルール(31)はis_obligatory
ルの具体かが行われるといらなくなる
% 31
sen(31,[
図7

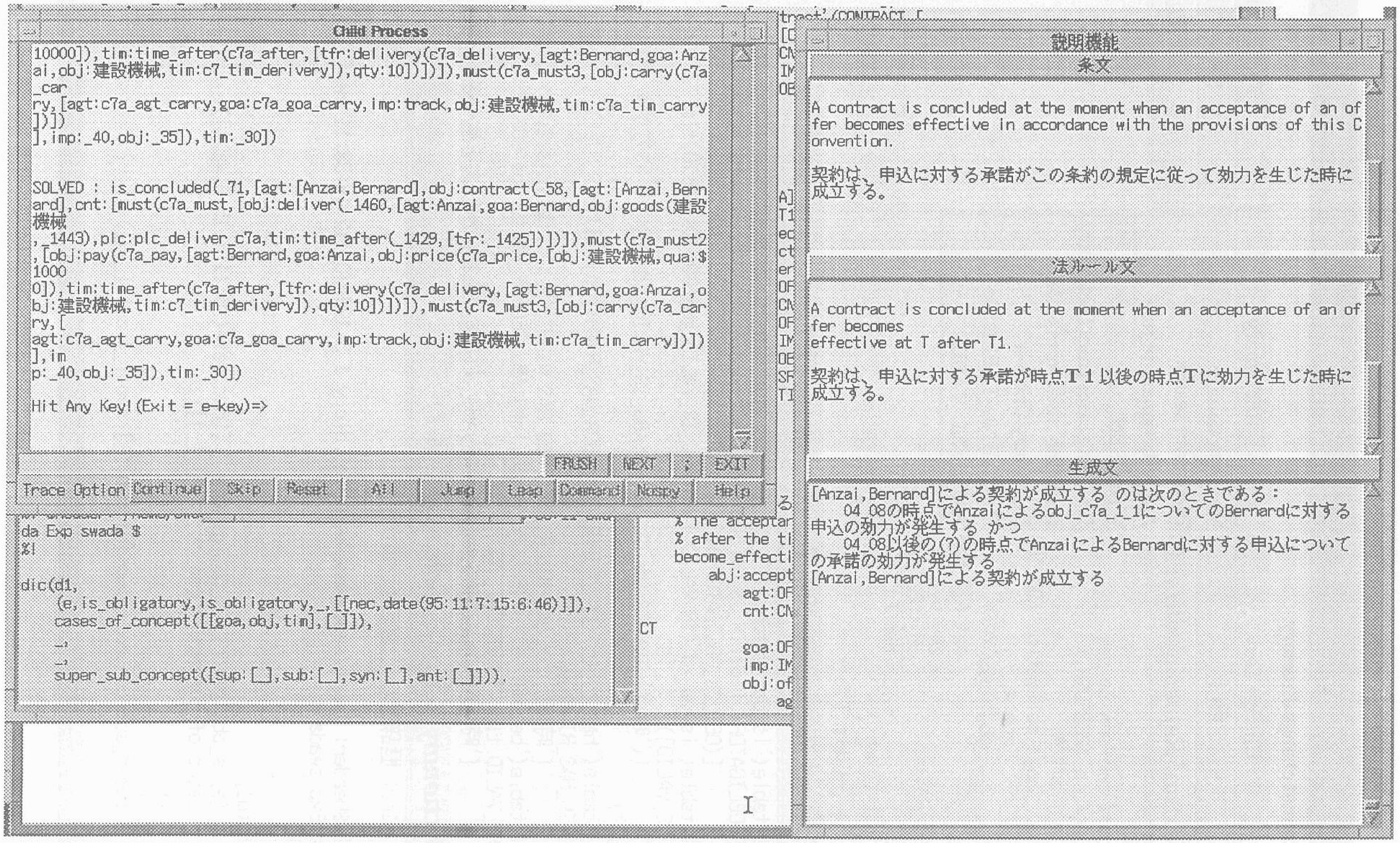

Child Process
10000]),tim:time_after(c7a_after,[tfr:delivery(c7a_delivery,[agt:Bernard,goa:Anz
ai,obj:建設機械,tim:c7_tim_derivery]),qty:10])])]),must(c7a_must3,[obj:carry(c7a
_car
ry,[agt:c7a_agt_carry,goa:c7a_goa_carry,imp:track,obj:建設機械,tim:c7a_tim_carry
])])
],imp:_40,obj:_35]),tim:_30])
SOLVED : is_concluded(_71,[agt:[Anzai,Bernard],obj:contract(_58,[agt:[Anzai,Bern
ard],cnt:[must(c7a_must,[obj:deliver(_1460,[agt:Anzai,goa:Bernard,obj:goods(建設
機械
,_1443),plc:plc_deliver_c7a,tim:time_after(_1429,[tfr:_1425])])]),must(c7a_must2
,[obj:pay(c7a_pay,[agt:Bernard,goa:Anzai,obj:price(c7a_price,[obj:建設機械,qua:$
1000
0]),tim:time_after(c7a_after,[tfr:delivery(c7a_delivery,[agt:Bernard,goa:Anzai,o
bj:建設機械,tim:c7_tim_derivery]),qty:10])])]),must(c7a_must3,[obj:carry(c7a_car
ry,[
agt:c7a_agt_carry,goa:c7a_goa_carry,imp:track,obj:建設機械,tim:c7a_tim_carry])])
],im
p:_40,obj:_35]),tim:_30])
Hit Any Key!(Exit = e-key)=>
FRUSH
NEXT
;
EXIT
Trace Option
Continue
Skip
Reset
All
Jump
Leap
Command
Nospy
Help
説明機能
条文
A contract is concluded at the moment when an acceptance of an of
fer becomes effective in accordance with the provisions of this C
onvention.
契約は、申込に対する承諾がこの条約の規定に従って効力を生じた時に
成立する。
法ルール文
A contract is concluded at the moment when an acceptance of an of
fer becomes
effective at T after T1.
契約は、申込に対する承諾が時点Ｔ１以後の時点Ｔに効力を生じた時に
成立する。
生成文
[Anzai,Bernard]による契約が成立する のは次のときである：
04_08の時点でAnzaiによるobj_c7a_1_1についてのBernardに対する
申込の効力が発生する かつ
04_08以後の(?)の時点でAnzaiによるBernardに対する申込について
の承諾の効力が発生する
[Anzai,Bernard]による契約が成立する

図 8

Knowledge Editor Version 1

Cannot change directory.

Change Text | Dictionary | Fact Editor

FILE | Make Dictionary | SEARCH

Dictionary search | Rule Edi

Apply dictionary to rule

Write 'SEN'

Checking cases from Dictionary

Checking cases from Rule

Dictionary Checker

Make all NL_TABLE file

Add one NL_TABLE

Textual Representation 1 .

```
%%
goal('perform to pay',[
'perform'(PERFORM,[
        agt:A,
        obj:'obligation'(OBLIGA
            goa:A,
            obj:'pay'(PAY,[
                    agt:AGT_PAY
                    goa:B,
                    obj:OBJ_PAY,
                    tim:T
            ]),
          tim:T
```

Textual Representation 2 < q5ruly.nl > (1:1)

```
nl_table('is_obligatory'(IS_OBLIGATORY,[goa:GOA_IS_OBLIGATORY,obj:OBJ_
IS_OBLIGATORY,tim:TIM_IS_OBLIGATORY]),
        [(OBJ_IS_OBLIGATORY),'義務がある']).
nl_table('is_valid'(IS_VALID,[abj:ABJ_IS_VALID,goa:GOA_IS_VALID,tim:TI
M_IS_VALID]),
        ['時点',TIM_IS_VALID,'に'),(ABJ_IS_VALID,'という'),'効力があ
る']).
nl_table('has_right'(HAS_RIGHT,[goa:GOA_HAS_RIGHT,obj:OBJ_HAS_RIGHT,ti
m:TIM_HAS_RIGHT]),
        ['時点',TIM_HAS_RIGHT,'に',OBJ_HAS_RIGHT,'という権利がある']).
nl_table('become_valid'(BECOME_VALID,[abj:ABJ_BECOME_VALID,goa:GOA_BEC
OME_VALID,tim:TIM_BECOME_VALID]),
        ['時点',TIM_BECOME_VALID,'に',ABJ_BECOME_VALID,'が効力を生じる
```

Dictionary Editor < q5dic.euc > (6:1)

```
% q1rul 歴辞書%   1995/11/10(Fri)
%!
%! $Header: /home/swada/mac/work5/q5.dic,v 1.3 1996/01/12 15:38:11 swa
da Exp swada $
%!

did(d1,
    (e,is_obligatory,is_obligatory,_,[[nec,date(95:11:7:15:6:46)]]),
    cases_of_concept([[goa,obj,tim],[_]]),
    _,
    _,
    super_sub_concept([sup:[_],sub:[_],syn:[_],ant:[_]])).
```

図 9

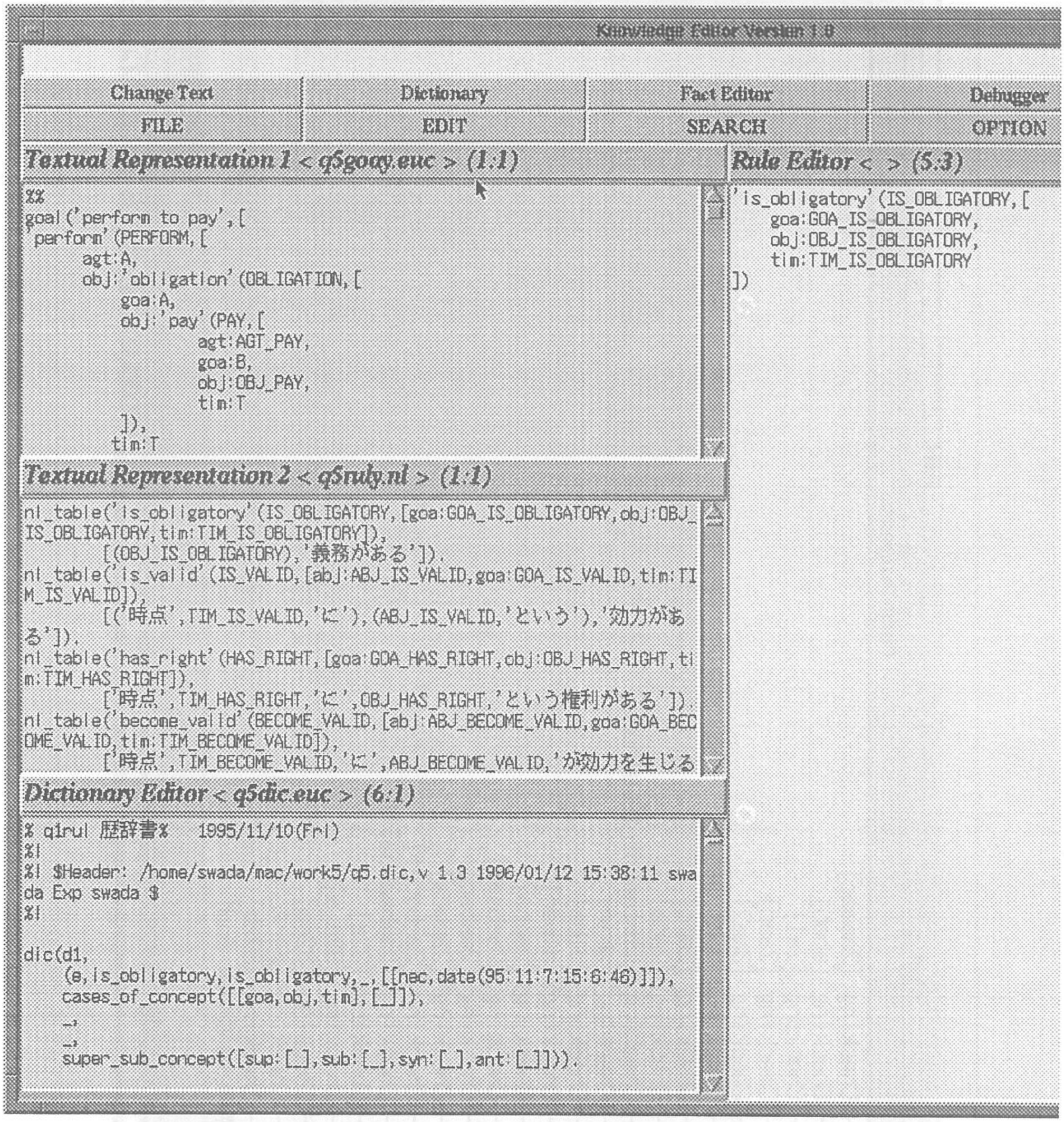

Knowledge Editor Version 1.0
Change Text
Dictionary
Fact Editor
Debugger
FILE
EDIT
SEARCH
OPTION
Textual Representation 1 < q5goay.euc > (1:1)
%%
goal('perform to pay',[
perform'(PERFORM,[
agt:A,
obj:'obligation'(OBLIGATION,[
goa:A,
obj:'pay'(PAY,[
agt:AGT_PAY,
goa:B,
obj:OBJ_PAY,
tim:T
]),
tim:T
Rule Editor < > (5:3)
'is_obligatory'(IS_OBLIGATORY,[
goa:GOA_IS_OBLIGATORY,
obj:OBJ_IS_OBLIGATORY,
tim:TIM_IS_OBLIGATORY
])
Textual Representation 2 < q5ruly.nl > (1:1)
nl_table('is_obligatory'(IS_OBLIGATORY,[goa:GOA_IS_OBLIGATORY,obj:OBJ_
IS_OBLIGATORY,tim:TIM_IS_OBLIGATORY]),
[(OBJ_IS_OBLIGATORY),'義務がある']).
nl_table('is_valid'(IS_VALID,[abj:ABJ_IS_VALID,goa:GOA_IS_VALID,tim:TI
M_IS_VALID]),
['時点',TIM_IS_VALID,'に'),(ABJ_IS_VALID,'という'),'効力がある']).
nl_table('has_right'(HAS_RIGHT,[goa:GOA_HAS_RIGHT,obj:OBJ_HAS_RIGHT,ti
m:TIM_HAS_RIGHT]),
['時点',TIM_HAS_RIGHT,'に',OBJ_HAS_RIGHT,'という権利がある']).
nl_table('become_valid'(BECOME_VALID,[abj:ABJ_BECOME_VALID,goa:GOA_BEC
OME_VALID,tim:TIM_BECOME_VALID]),
['時点',TIM_BECOME_VALID,'に',ABJ_BECOME_VALID,'が効力を生じる
Dictionary Editor < q5dic.euc > (6:1)
% q1rul 歴辞書% 1995/11/10(Fri)
%!
%! $Header: /home/swada/mac/work5/q5.dic,v 1.3 1996/01/12 15:38:11 swa
da Exp swada $
%!
dic(d1,
(e,is_obligatory,is_obligatory,_,[[nec,date(95:11:7:15:6:46)]]),
cases_of_concept([[goa,obj,tim],[_]]),
_,
_,
super_sub_concept([sup:[_],sub:[_],syn:[_],ant:[_]])).

図10

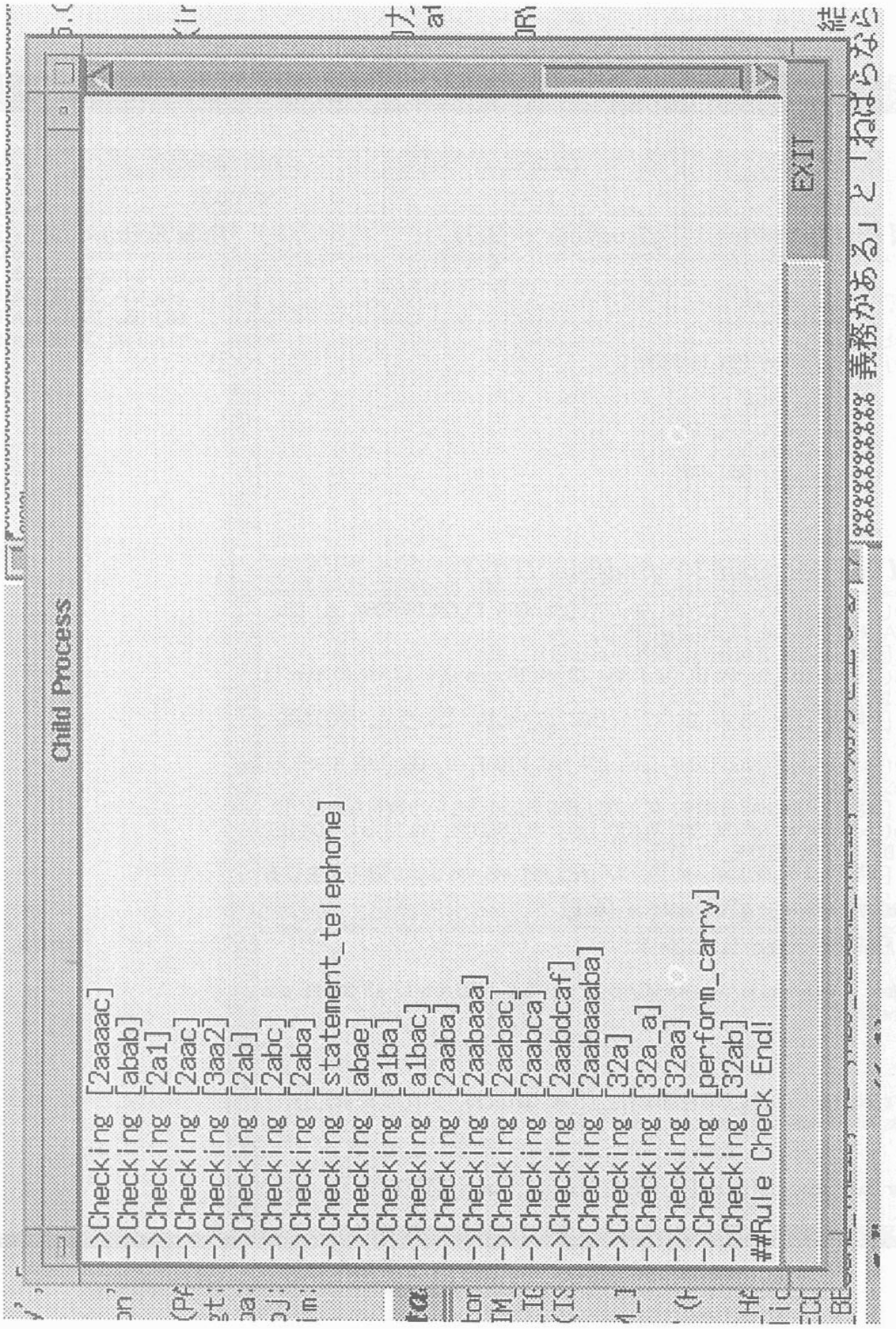

Child Process
->Checking [2aaaac]
->Checking [abab]
->Checking [2a1]
->Checking [2aac]
->Checking [3aa2]
->Checking [2ab]
->Checking [2abc]
->Checking [2aba]
->Checking [statement_telephone]
->Checking [abae]
->Checking [a1ba]
->Checking [a1bac]
->Checking [2aaba]
->Checking [2aabaaa]
->Checking [2aabac]
->Checking [2aabca]
->Checking [2aabdcaf]
->Checking [2aabaaaba]
->Checking [32a]
->Checking [32a_a]
->Checking [32aa]
->Checking [perform_carry]
->Checking [32ab]
##Rule Check End!
EXIT

図11

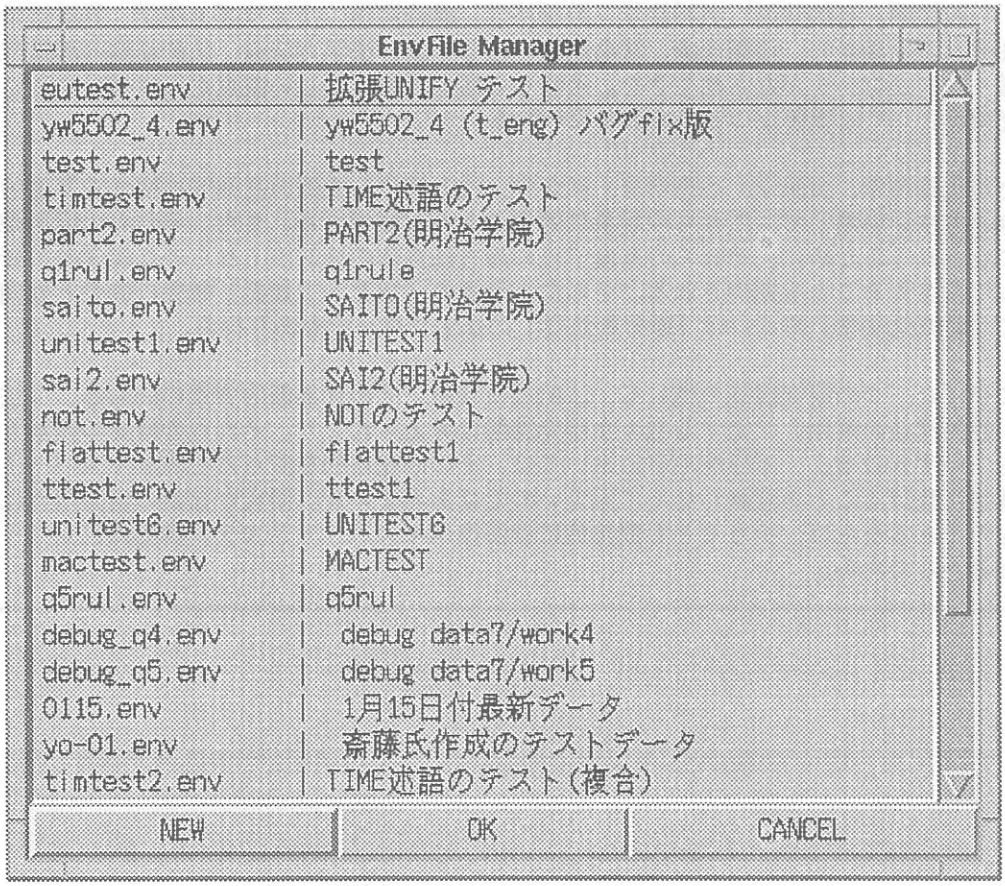

EnvFile Manager
eutest.env | 拡張UNIFY テスト
yw5502_4.env | yw5502_4 (t_eng) バグfix版
test.env | test
timtest.env | TIME述語のテスト
part2.env | PART2(明治学院)
q1rul.env | q1rule
saito.env | SAITO(明治学院)
unitest1.env | UNITEST1
sai2.env | SAI2(明治学院)
not.env | NOTのテスト
flattest.env | flattest1
ttest.env | ttest1
unitest6.env | UNITEST6
mactest.env | MACTEST
q5rul.env | q5rul
debug_q4.env | debug data7/work4
debug_q5.env | debug data7/work5
0115.env | 1月15日付最新データ
yo-01.env | 斎藤氏作成のテストデータ
timtest2.env | TIME述語のテスト(複合)
NEW
OK
CANCEL

図１２

研究項目Ａ０１
法的知識の一般的構造の解明

この研究項目は、民法や国際統一売買法といった個々の実定法ではなく、それらに共通する法一般を対象として取り扱う。法一般の知識構造の解明に努力することは、個々の実定法の知識構造の解明に役立つはずである。（またその逆も成り立つ。）

この研究項目には、例えば、次の研究課題が考えられる。

（１）法命題の構造と機能、（２）法体系の構造、そして（３）法的推論の構造。これらは、法哲学や法社会学といった基礎法学の研究対象である。これらの対象に対しては、従来、概念法学、一般法学、目的法学、利益法学、分析法学、純粋法学、リアリズム法学、経験法学などがそれぞれの立場から、法概念論、法命題論、法体系論、法学方法論、法的推論の理論等として、様々なアプローチがなされてきた。それらの研究成果を新しい観点から再検討することも本研究項目としては重要なことであると思われる。

また現代論理学、言語学、心理学等、社会学、法と経済、人間の認知等に関する基礎的学科の観点と方法を応用して、これらの対象に迫ることも重要である。また、それらの応用がより有効になされうるために、それらの基礎学科の諸方法自体の検討がそれらの基礎学科の研究者によってなされることも期待される。さらに、最近の「法と人工知能」研究の成果と動向を参照して「新概念法学」あるいは「新分析法学」的なアプローチもあり得る。

上に挙げた三つの研究課題の他にも、法的価値判断、その世界観的背景の問題等、「法的知識の一般的構造の解明」に関する様々な研究課題は考えられよう。それらの研究の提案も歓迎される。いずれにせよ、本研究項目においては、その研究目的を直ちに法律エキスパートシステム構築に繋げることを意識しなくてもよいのではあるまいか。研究が知識の構造解明に関連するものであるならば、少なくとも間接的には法律エキスパートシステム構築に役立ちうるものとなるからである。

“Clarification of the General Structure of Legal Knowledge”

Group A01