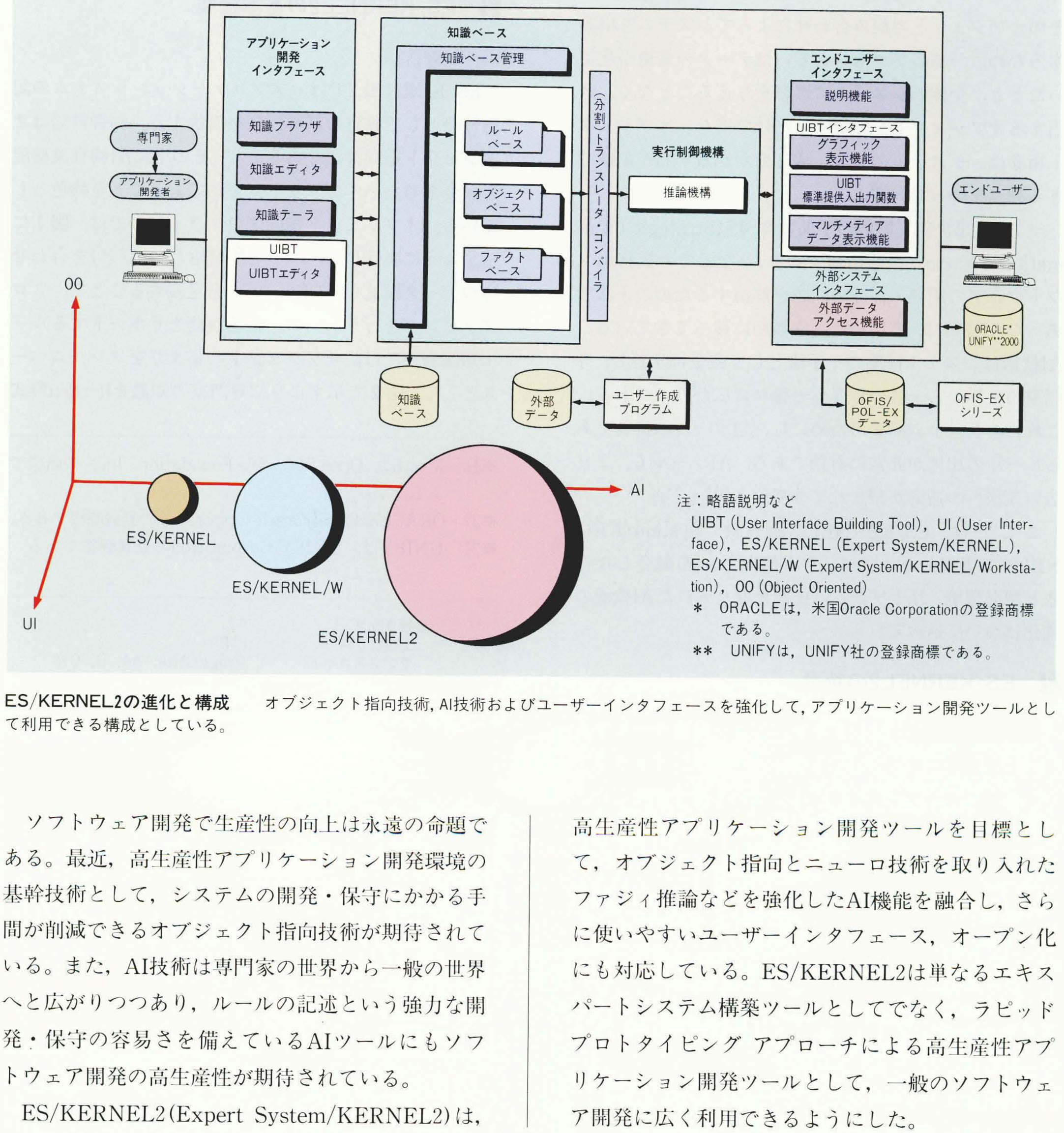
# 知的システムの構築を容易にするアプリケーション開発ツール

## Application System Development Tools for Developping Intelligent Systems Rapidly

堺原　健*　*Ken Sakaibara*　　塙　信弘*　*Nobuhiro Hanawa*
長田充弘*　*Mitsuhiro Nagata*　　荒砥偉浩*　*Yoshihiro Arato*




注：略語説明など
UIBT (User Interface Building Tool)，UI (User Interface)，ES/KERNEL (Expert System/KERNEL)，ES/KERNEL/W (Expert System/KERNEL/Workstation)，OO (Object Oriented)
＊　ORACLEは，米国Oracle Corporationの登録商標である。
＊＊　UNIFYは，UNIFY社の登録商標である。

**ES/KERNEL2の進化と構成**　オブジェクト指向技術，AI技術およびユーザーインタフェースを強化して，アプリケーション開発ツールとして利用できる構成としている。

ソフトウェア開発で生産性の向上は永遠の命題である。最近，高生産性アプリケーション開発環境の基幹技術として，システムの開発・保守にかかる手間が削減できるオブジェクト指向技術が期待されている。また，AI技術は専門家の世界から一般の世界へと広がりつつあり，ルールの記述という強力な開発・保守の容易さを備えているAIツールにもソフトウェア開発の高生産性が期待されている。

ES/KERNEL2(Expert System/KERNEL2)は，高生産性アプリケーション開発ツールを目標として，オブジェクト指向とニューロ技術を取り入れたファジィ推論などを強化したAI機能を融合し，さらに使いやすいユーザーインタフェース，オープン化にも対応している。ES/KERNEL2は単なるエキスパートシステム構築ツールとしてでなく，ラピッドプロトタイピング アプローチによる高生産性アプリケーション開発ツールとして，一般のソフトウェア開発に広く利用できるようにした。

＊ 日立製作所 ソフトウェア開発本部

## 1 はじめに

ハードウェアのダウンサイジングとともに，生産性の高いアプリケーション開発環境へのニーズはとどまるところを知らない。高生産性アプリケーション開発環境の基幹技術として，オブジェクト指向技術が期待されている。オブジェクト指向はデータとプロセスを一体化し，そのオブジェクトの組み合わせによってシステムを構築するため，一部のプロセスあるいはデータの変更が起こったとき，全体のシステムに影響を与えることなく，該当するオブジェクトを変更するだけで済む。オブジェクト指向は，システムの開発・保守にかかる手間が削減できる利点を持っている。

一方，意思決定・制御の手助けやSIS(Strategic Information System：戦略情報システム)などでの業務用ソフトウェアの開発・保守の問題を解消するための手法であったAI技術は，実用期から成熟期に移ってきている。AI技術はシステム構築の一手法として完全に定着し，専門家の世界から一般の世界へと徐々に広がりつつある。これらのシステム構築のためには，AIツールの特長であるルールの記述が非常に有効である。AIツールも，より広い業務への適応が望まれてきている[1)]。

ここでは，ES/KERNEL2(Expert System/KERNEL2)で実現したオブジェクト指向とAIを融合した言語と開発環境，および新しい技術を取り入れたAI機能の強化について述べる。

## 2 ES/KERNEL2の特徴

ES/KERNEL2は，アプリケーション開発ツールであることを基本にオブジェクト指向，ユーザーインタフェース，AI，オープン化を考慮し，次の方針で設計した。

(1) 一般のアプリケーション開発の一環として，AI機能を特別視しないで違和感なくAI機能が利用できること：オブジェクト指向による知識表現とルール知識表現を，同時に記述可能なハイブリッド型言語を提供する。

(2) オブジェクト指向の持つ高い生産性を引き出すこと：RAD(Rapid Application Development)モデル(アプリケーションを短期間に開発するための作業モデル)を想定した開発環境を提供する。

(3) 高度なユーザーインタフェースが簡単に作成できること：多彩な機能を持つUIBT(User Interface Building Tool)との連携機能を提供する。UIBTエディタによるユーザーインタフェース定義の自動生成も可能である。

(4) 問題に応じたAI(推論方式)が選べること：ファジィ推論，協調型推論，仮説推論も提供する。

(5) オープン時代に適応すること：日立Motif※1)で稼動する。ORACLE※2)，UNIFY2000※3)などの外部システムインタフェース機能を提供する。

## 3 ES/KERNEL2の基本機能

(1) 制御言語

ES/KERNEL2では，アプリケーションシステムの記述言語として独自の制御言語を提供する。制御言語はオブジェクト指向言語であること，その中にAI的推論機能を融合させたハイブリッド型言語であることを特色としている。オブジェクト指向プログラミングでは，**図1**に示すように属性(スロット)と手続き(メソッド)を合わせ持つデータ構造(オブジェクト)を定義することで，プログラミングを行う。一方，推論機能をサポートするルール知識表現では，オブジェクトの集まりをフレームベースとして，**図2**で示すように専門家の知識をif-then形式

※1) Motifは，Open Software Foundation, Inc. の商標である。

※2) ORACLEは，米国Oracle Corporationの登録商標である。

※3) UNIFYは，UNIFY Corporationの登録商標である。

```
(会社クラス
   上位クラス              {#PS}
   ランクスロット          (range {AAA, AA, B, C})
   資本スロット            (data_type int)
   従業員数スロット        (data_type int)
   上場スロット            (data_type string)
   .....
 #methods
 #method
   ランク取得メソッド (
     (Ask(ランク))
   )
   .....
 #methods_end
)

(会社A
   クラス                  会社クラス
   資本金                  20
   上場                    第一部
)

.....
```

図1 オブジェクト指向による知識表現 “会社クラス”クラスオブジェクトは，“ランクスロット”，“資本スロット”，“従業員数スロット”，“上場スロット”，…という属性とランクを参照する“ランクスロット取得メソッド”，…という手続きを持つ。“会社A”は，“会社クラス”クラスのインスタンスオブジェクトである。手続きの定義は，制御言語を用いて行う。

```
{信用度チェックルール群}
(信用度ルール1
   if （?会社 の @資本 が 10　以上であり
              @上場 が（第一部 である
              または　　第二部 である))
   then
      （?会社 の @ランク に AAA を代入する）
)
(信用度ルール2
.....
)
.....
```

図2　ルールによる知識表現　　if-then形式による知識表現で，制御言語を用いて記述される。このルールをもとに，前向き推論が実行される。推論の対象は，ルール群と呼ばれるルールの集まりを1単位とする。

で表現する。オブジェクト指向によってモジュラリティ，保守・拡張性が高いプログラミングが可能になるのと同時に，必要に応じてAIの推論機能を同じ文脈で用いることができる。

(2)　コンパイラ

制御言語で記述されたプログラムは基本的にはインタプリタで動作するが，高速実行のためにES/KERNEL2はコンパイラを提供する。開発・デバッグ環境と連携が十分にとれたインタプリタを用いてアプリケーションシステムの開発を行い，コンパイラの生成した高速なコンパイルドコードを用いて実行する。ES/KERNEL2のコンパイラは，ソースの部分的なコンパイルを可能にするのと同時に，生成されるコンパイルコードは，コンパイルされてないソースと混在してインタプリタ環境で動作可能なので，アプリケーションシステムのインクリメンタルな開発(まず基本部分を開発し，徐々に機能拡張していく開発方法)が可能となる。

コンパイラの最適化機能として，高速で省メモリの新しい推論方式CREST(Conflict Resolution Strategy Oriented Matching)をサポートした。これは競合解消戦略を意識することにより，推論中に増大する中間情報を最低限に抑えて従来の推論方式(データ駆動型照合方式：Rete)の欠点を補ったものである。

## 4　ES/KERNEL2の開発環境

オブジェクト指向とAI的推論を融合したES/KERNEL2を使ったアプリケーションシステム開発は，試作と評価を繰り返すことによって要求仕様を実現するプロトタイピング アプローチで行うことができる。このアプローチでは，短期間で試作を行うことが重要である。ES/KERNEL2の開発環境は，RAD[1]モデルを採用してアプリケーションを短期間で開発できるようにしている。

ES/KERNEL2の開発環境は，**図3**に示すようなコーディングサイクルとデバッグサイクルを，開発作業のRADモデルとして仮定している。種々のツールによって，おのおのの作業をサポートするとともに，ツール間の連携機能によって各作業の間をスムーズに移行できるようになっている。次に，おのおのの作業がどのような機能によってサポートされ，作業間の移行がどのように効率化されているかを順を追って述べる。

(1)　コーディングサイクル

(i) エディタを使ってソースを編集する。(ii) 編集中にハンドブック(オンラインマニュアル)機能を使ってメソッドの名前や引き数，その他の構文などを調べることができる。(iii) ソースコードの中で何回も使う部分は部品として登録し，再利用することができる。(iv) ソースコードの編集が一段落したら，トランスレータを使って構文のチェックができる。(v) もし，構文に誤りがある場合は，自動的に誤りの一覧が表示され，誤りのある個所にエディタでワンタッチで位置づけできる。

(2)　デバッグサイクル

ブラウザによる動的デバッグ機能とクロスリファレンスによる静的ソースコード解析機能がある。

(vi) ブラウザからアプリケーションの好きなところで実行を中断し，変数の値やオブジェクトのスロットの

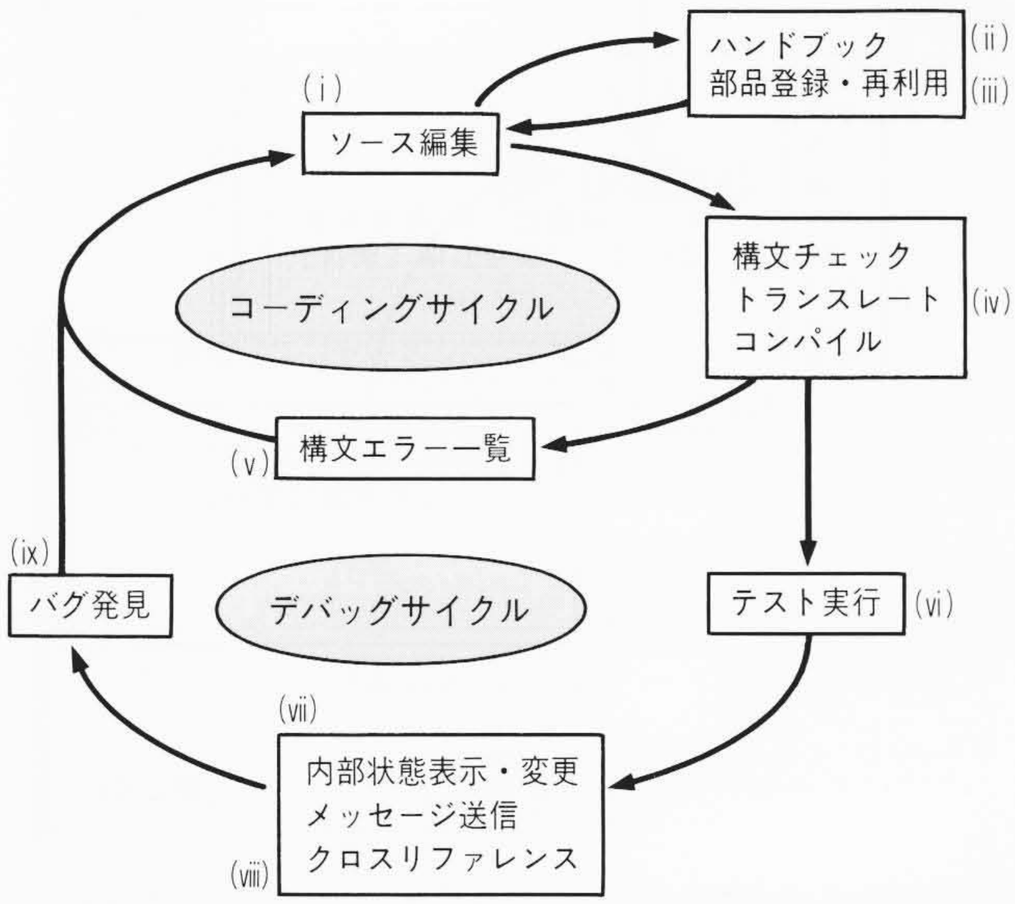



図3　アプリケーションの開発作業モデル　　コーディングサイクルとデバッグサイクルとに分かれ，それぞれの作業がスムーズに行えるように機能を提供している。

値，実行可能なルールの一覧，実行中断個所のソースコードなど，内部の状態をビジュアルに表示したり，一時的に変更したりできる。(vii) メッセージ送信機能によって，任意のメソッドを試し実行できる。(viii) オブジェクト名やメソッド名などの定義・参照関係を知りたいときにはクロスリファレンス機能を使う。(ix) ブラウザでもクロスリファレンスでも，バグを見つけた場合は即座にコーディングサイクルに戻り，バグを修正した後に再びテスト デバッグ フェーズに戻り，デバッグを継続できる。デバッグ完了部分だけをコンパイルし，デバッグ中の部分をインタプリタで実行すると，テスト実行時間を短縮し，デバッグしたい個所へ短時間で到達できる。

ブラウザ，エディタ，トランスレータの画面上のメニューを選択するだけで，デバッグサイクルに従って作業できる例を**図4**に示す。

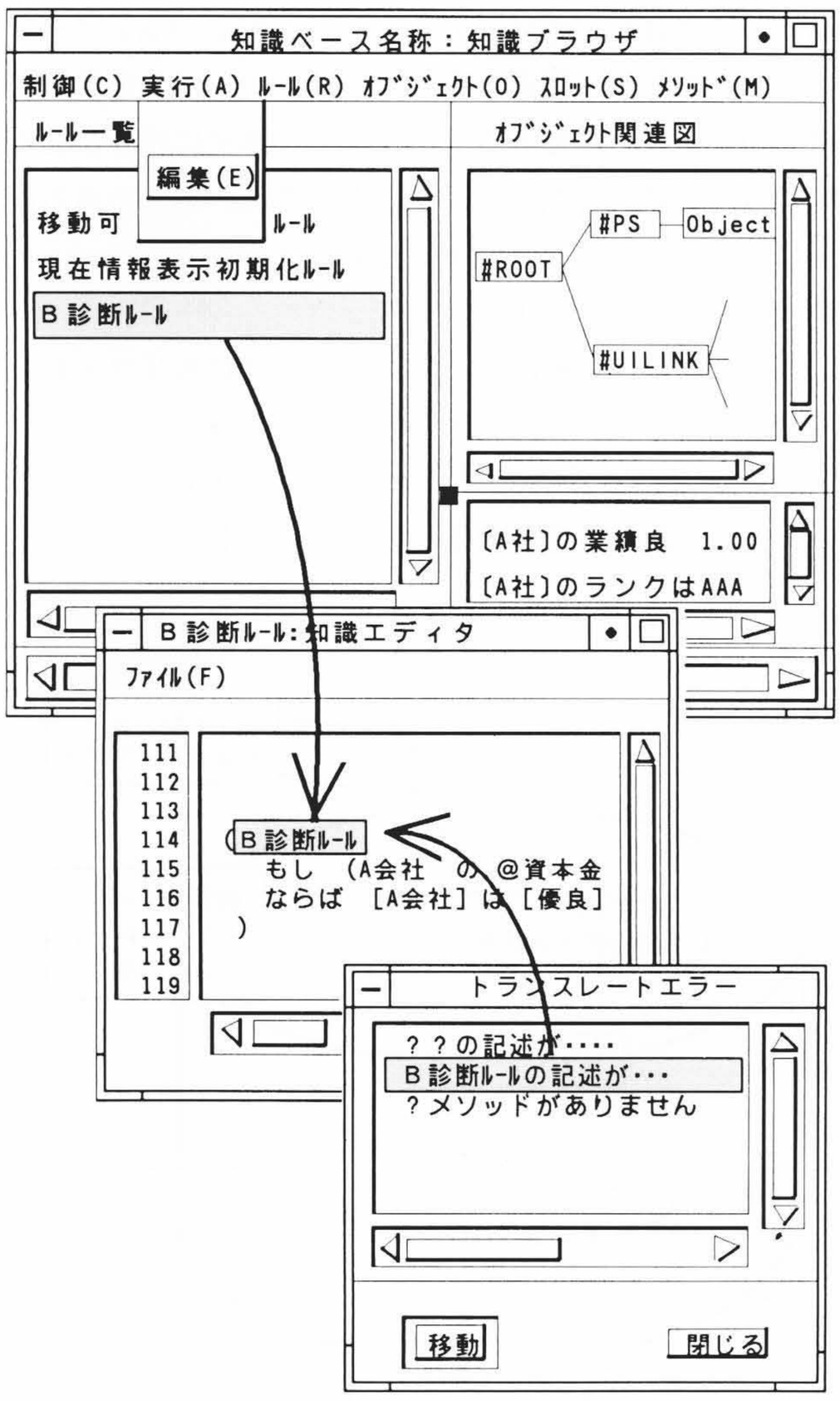



図4 デバッグ作業の例 ブラウザでB診断ルールを選択し，編集メニューの実行でB診断ルールがエディタに表示される。また，トランスレートエラーの一つを選択し移動ボタンを押すと，エラー発生個所が表示される。

このようにES/KERNEL2の開発環境では，各種のツールとそれらの連携によって，オブジェクト指向の持つ高いアプリケーション生産性を引き出すことができる。

## 5 UIBTによるユーザーインタフェースの開発

最近のアプリケーションシステムは，GUI(Graphical User Interface)の導入などによって，ユーザーインタフェースが年々高度化・高機能化する一方，その開発に要する工数はシステム全体の過半数を占めるに至っている。アプリケーションシステム開発での低コスト・短納期を実現するためには，GUI構築のための工数を低減することが重要な課題となっている。ES/KERNEL2では，対話的にGUIを構築できるユーザーインタフェース構築ツールUIBTをES/KERNEL2の開発環境と連動することで，GUI構築工数の大幅な低減を実現している。

UIBTは，

(1) 画面作成用標準部品(ボタンやラベルなどのパネル部品，幾何図形，画像，ビジネスグラフ，表)

(2) 標準部品の生成や属性(表示位置，色，文字列，線種など)

定義を対話的に行うためのエディタで構成する。

画面編集用エディタ(パネルエディタ)での画面編集作業例を**図5**に示す。マウスでの直接操作で，アプリケーションのユーザーインタフェース画面を設計・開発することができる。

一方，実際のアプリケーションでは，UIBTを用いて作成した画面上のGUI部品をプログラムから動的に制御したり，画面からの入力イベントをプログラム中の他のオブジェクトに通知するといった機能が必要となる。ES/KERNEL2では，こうしたプログラム本体と画面部品のインタフェースを**図6**に示すUILINKオブジェクトを介して実現している。UILINKオブジェクトを介することで，ソース中に定義したオブジェクトとGUIを構成する画面部品という実現形態の異なるオブジェクトを，制御言語によるメッセージ送信という単一のプログラミングパラダイムで開発することができる。

UILINKオブジェクトは，GUIを構築するすべての画面部品について1対1に対応しており，編集中の画面に部品が定義された時点で自動的に生成される。画面編集操作に連動したUILINKオブジェクト自動生成機能により，GUI構築に要するプログラミング工数が大幅に低減できる。

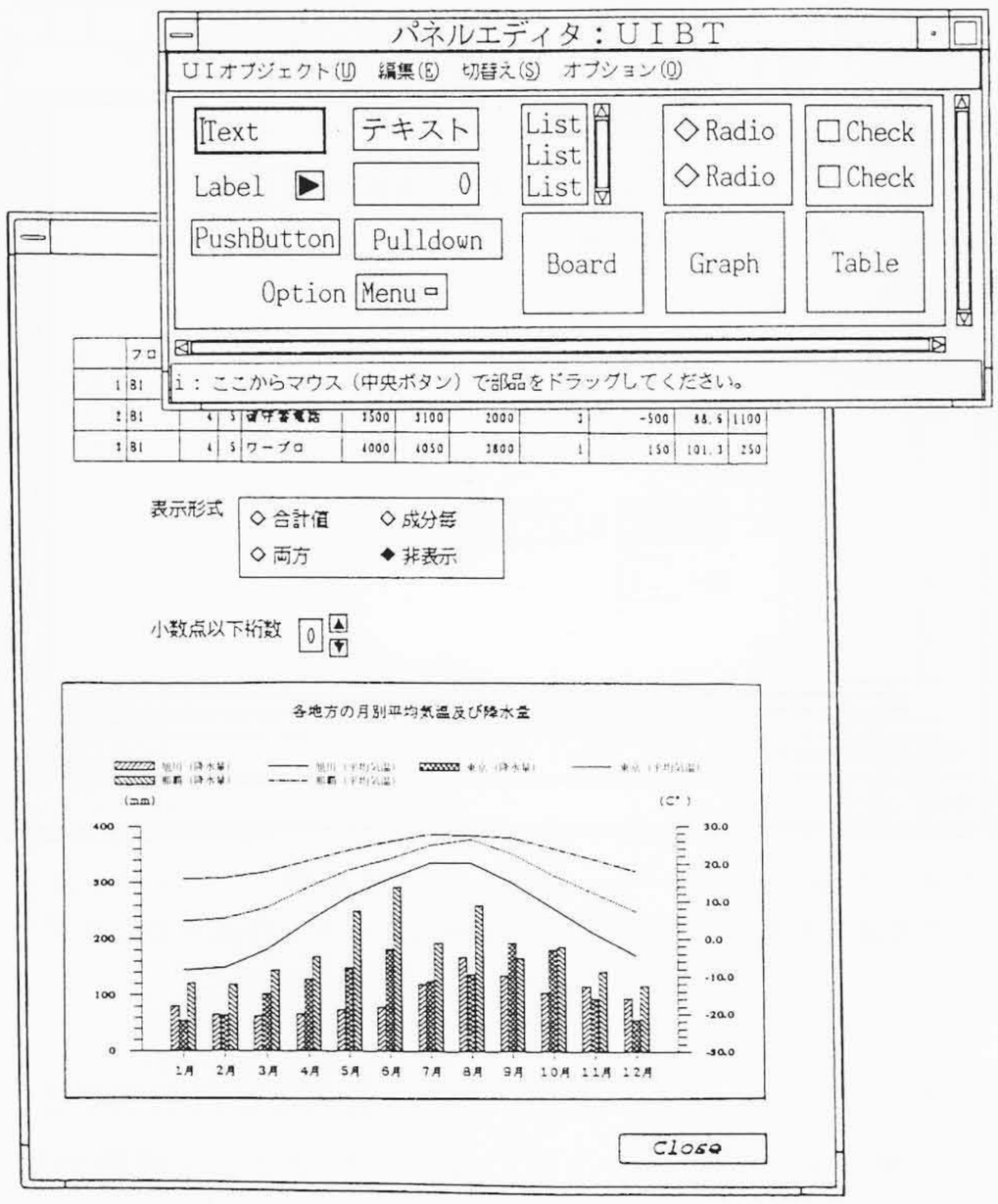



図5　UIBTのパネルエディタによる画面作成　　パネルエディタで編集中のGUI画面を示す。画面上には，右上部のパネルエディタウインドウから直接操作によって生成した表，ラジオボックス，アローボタン，グラフなどの画面部品が並んでいる。

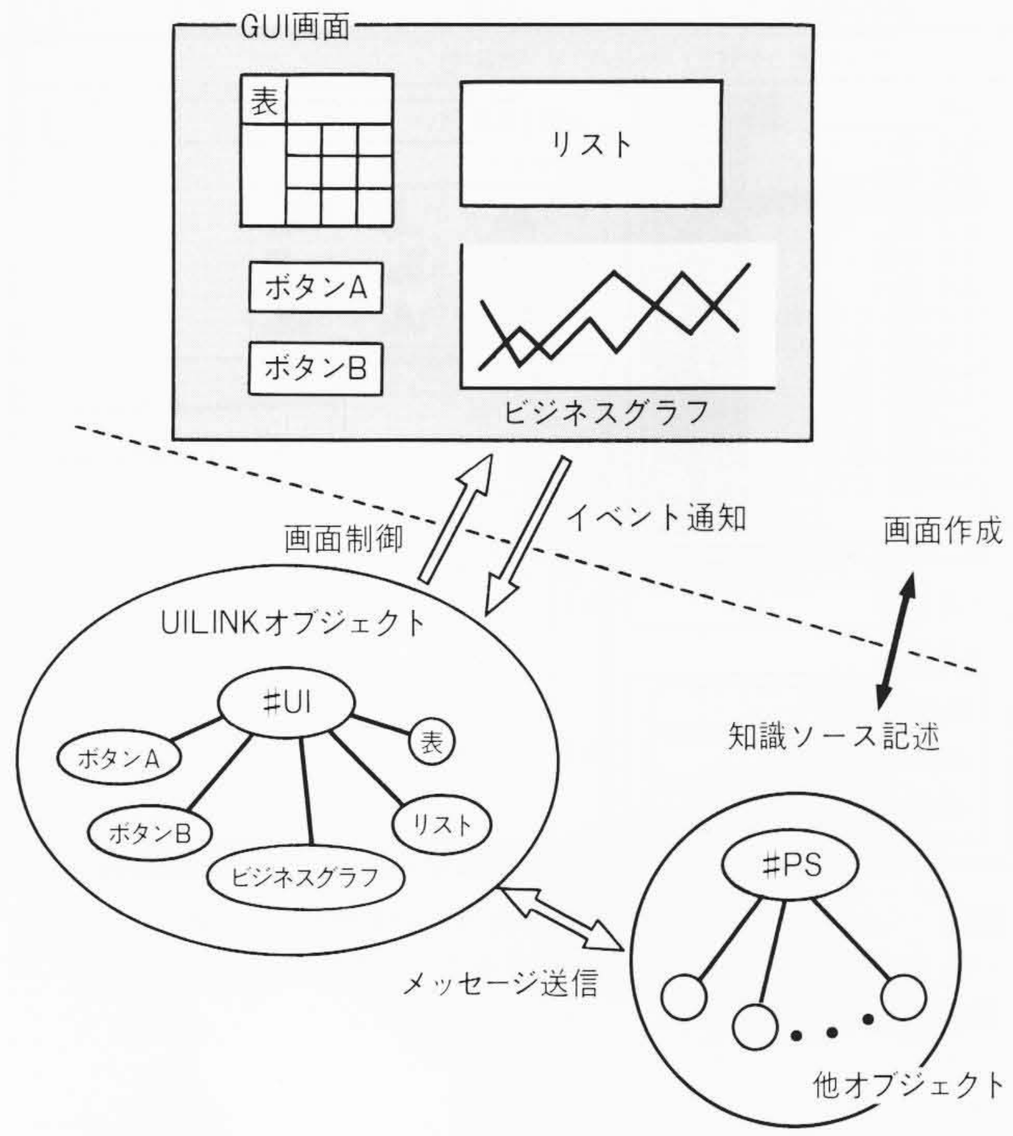



図6　UILINKオブジェクトによる画面と知識ソースの対応づけ　　インタフェース画面上に部品が生成されると，これに対応したUILINKオブジェクトが知識ソース中に自動生成される。プログラム中の他オブジェクトは，このUILINKオブジェクトを介して画面と連動する。

## 6　ES/KERNEL2の強化されたAI機能

ES/KERNEL2は，通常の前向き推論に加えて仮説推論，協調型推論，ファジィ推論の三つを備えている。ここでは，ニューロ技術などを用いてさらに使いやすくなったファジィ推論機能について述べる。

(1)　ファジィ知識の自動チューニング機能

従来，開発者が良好な推論結果を得るために行っていたメンバシップ関数のチューニング作業を迅速・簡便に行うために，ファジィ知識の自動チューニング機能を提供している。旧製品のES/KERNEL/W(ES/KERNEL/Workstation)では，推論結果が目標の結果になるようにユーザー自身がメンバシップ関数のパラメータ値を変えていた。

ES/KERNEL2/Wでは，過去の事例データ(教師データ)を設定するだけでメンバシップ関数の形状とルールの重みを自動的に調整することができる。教師データからの学習には，ニューラル ネットワーク シミュレータ(バックプロパゲーション法)を使用している。チューニング作業の例を**図7**に示す。

(2)　メンバシップ関数エディタ

メンバシップ関数を，ビジュアルに定義できるメンバシップ関数エディタを提供している。メンバシップ関数の定義は，パラメータの設定による定義に加えて，マウスによる定義も可能にした。マウスによる定義方法は，マウスでメンバシップ関数を直接ドラッグすることによ

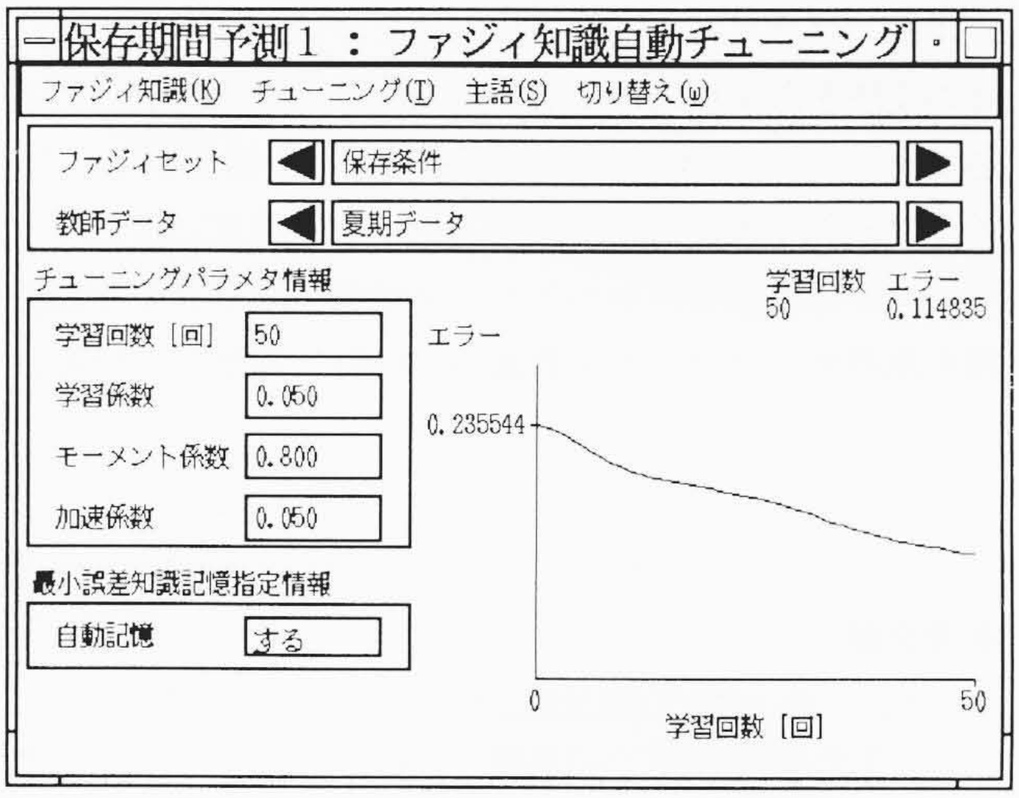



図7　ファジィ知識の自動チューニング作業の例　　バックプロパゲーション法を用いて，メンバシップ関数を50回学習させている様子を示す。チューニングの状態は画面上のグラフで表示されている。

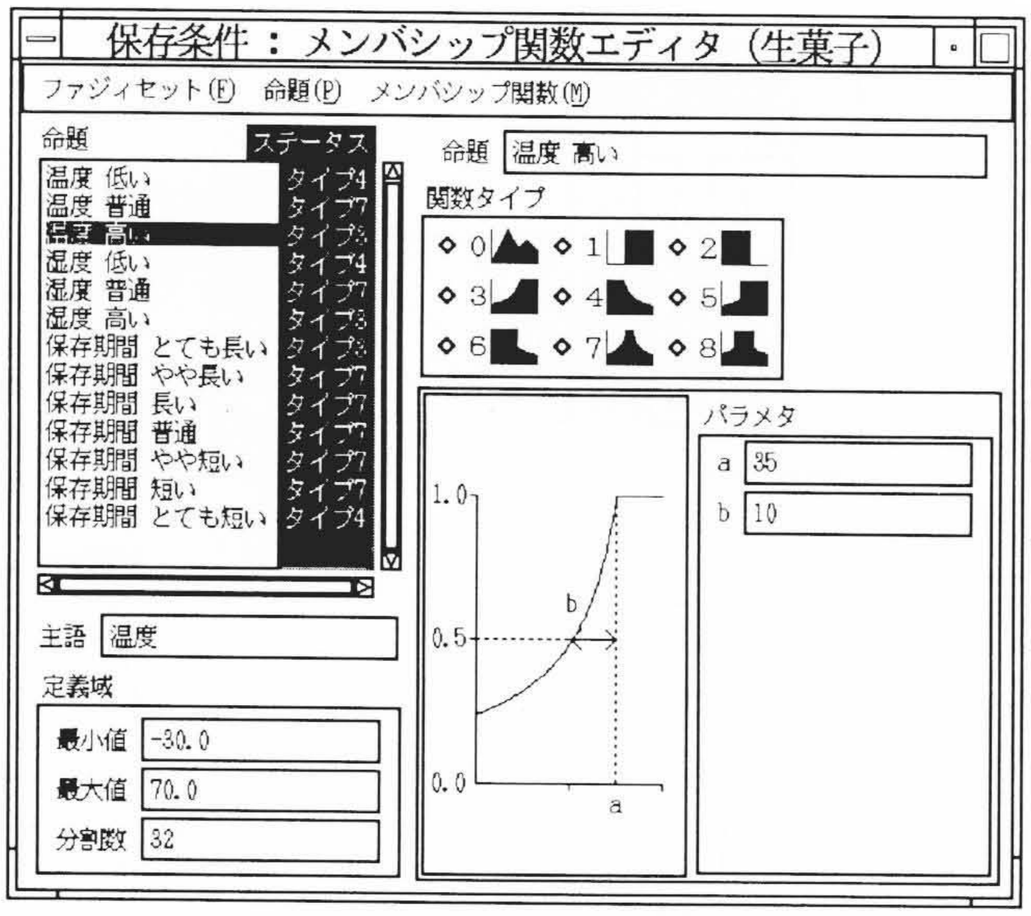



図8 メンバシップ関数エディタの使用例 マウスでメンバシップ関数を直接ドラッグできるので，感覚的にメンバシップ関数の形状を定義できる。

って，感覚的にメンバシップ関数の形状を定義できる。

パラメータによる定義方法は，マウスで定義できない微調整に有効である。メンバシップ関数エディタの使用例を**図8**に示す。

(3) ファジィ知識とメンバシップ関数の独立

ファジィ知識は，あいまいな規則を表すファジィルールと命題の適合度を表すメンバシップ関数を，それぞれ独立したカテゴリーとして管理するようにした。このため，ファジィルールとメンバシップ関数を組み合わせを変更することで状況に応じた推論が行えるようにしている。

また，ファジィルールの作成のために，ファジィルールの文法をあまり知らなくても，簡単にルールを作成できるファジィ ルール エディタ(**図9**参照)を提供している。

(4) ファジィ推論情報取得用メソッドを豊富に提供

ファジィ推論実行中のトレース情報，ファジィ知識情報を取得するメソッドを豊富(約30種)に提供している。

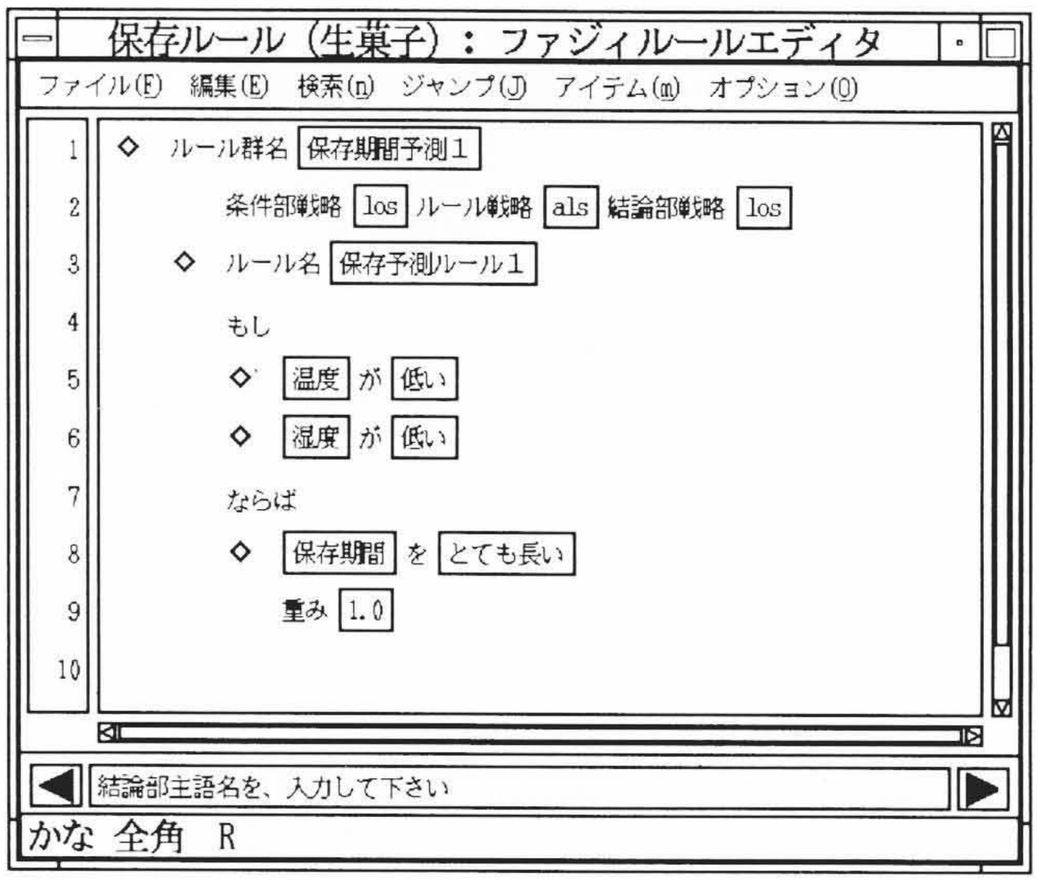



図9 ファジィ ルール エディタの使用例 ファジィルールの文法をあまり知らなくても，簡単にルールを作成することができる。

また，ファジィ推論の途中経過の表示などに利用でき，ユーザーアプリケーションで使用したファジィ推論の確かさを確認できる。

## 7 おわりに

ES/KERNEL2はオブジェクト指向型言語を基本とし，各種の推論機能はそれぞれ一つのクラスとして実現することで，オブジェクト指向とAIを融合した。オブジェクト指向の持つ高生産性を引き出すために，プログラムの部品化と再利用の支援，GUIを対話的に作成できるUIBT，コンパイラとインタプリタが共存した開発環境を提供した。また，オブジェクト指向技術だけでなく，ニューロ技術を取り入れたファジィ推論などのAI機能の強化も行った。

ES/KERNEL2は単なるエキスパートシステム構築ツールでなく，ラピッド プロトタイピング アプローチによるソフトウェア開発支援として，一般のアプリケーションの開発に広く利用できるようにした。

**参考文献**

1) 増位，外：ES/KERNEL2をベースにした知識ベース・システムの次世代への展開，日経インテリジェントシステム，別冊，78～87，1992-秋号(1992)