

Vocalogの視覚的プログラミング環境

後藤 文太郎

北見工業大学 情報処理センター

VPEは、拡張論理型プログラミング言語Vocalogに対する視覚的統合プログラミング環境である。VPEでは、プログラムの作成、デバッグ、そしてプログラムを利用するインターフェースの3つのモードを統合してつなぎめなく扱えるように視覚化を行った。統合してつなぎめなく扱えるような視覚化とは、各モードにおける視覚オブジェクトの表現形態が同一であり、かつモード間で視覚的な連携が行えるものである。VPEはシンセティック・メディア・システムIntelligentPadを用いて作られている。VPEにおける視覚オブジェクトは、IntelligentPadにおけるパッドである。異なる表示パッドを用いることで、同一のオブジェクトに対してモードに応じた種々の見せ方を与えた。

A Visual Programming Environment for Vocalog

Fumitaro GOTO

Information Processing Center, Kitami Institute of Technology

Koen-cho 165, Kitami 090, Japan

VPE is a visually integrated programming environment for an extended logic programming language Vocalog. In VPE, objects has been visualized so as to seamlessly integrate three modes, i.e., editing mode, debugging mode, and interface mode for using a program. In VPE, visual objects have the same representation form among those modes and there are visual cooperation among modes. VPE has been developed by using the synthetic media system IntelligentPad. Visual objects in VPE are pads in IntelligentPad. VPE gives the different presentations of the same object by using different view pads.

1. はじめに

プログラミングにおける視覚化には、

- (1) プログラムの作成の視覚化
- (2) プログラムのデバッグの視覚化
- (3) プログラムを利用するインターフェースの視覚化

の3つがある。プログラムの作成、デバッグ、そしてプログラムを利用するインターフェースは、それぞれ独立したものではなく、相互に関連している。したがって、これら3つのモードを個々に視覚化するのではなく、統合してつなぎめなく扱えるように視覚化すべきである。

統合してつなぎめなく扱えるような視覚化とは、

- (a) 3つのモードにおける視覚オブジェクトの表現形態が同一であり、かつ
- (b) 各モードの視覚化が独立していない

ものである。(a)は各モードにおける視覚オブジェクトを統一的に扱うための条件である。(b)は各モード間での視覚的な連携を行うための条件である。モード間での視覚的な連携の例としては、プログラム利用のための視覚的インターフェースを使っている状況の視覚的なデバッグや、視覚的デバッグを行う際に、プログラム作成時における視覚オブジェクトを利用することなどがある。

本論文では、論理プログラミングにおいて、上記の3つの視覚化を統合してつなぎめなく扱えるプログラミング環境VPE(Vocalog Programming Environment)を提案する。VPEは、後藤と田中により提案されている拡張論理型プログラミング言語Vocalog^{3),14)}に対する視覚的統合プログラミング環境である。語彙をデータベースへの問い合わせならびに一階述語論理で記述された知識ベースへ導入することがTanakaにより行われてきた^{7),8),9),10)}。Vocalogは、それに基づき、Prologに語彙を導入し利用できるようにしたものである。Vocalogでは、基本的な概念をコンテキストから取り出して名詞や形容詞としてジェネリックに定義し、これらの合成により、より高次の概念をコンテキストから独立に定義することができる。

Prologの視覚化を行ったシステムとしては、LadretとTueherのVLP⁴⁾、EisenstadtとBrayshawのTPM²⁾、NumaoらによるPROEDIT2⁵⁾、PauとOlasonのVPP⁶⁾、DewarとClearyのDEWLAP¹¹⁾などがある。VLPはアイコンを用いたプログラムの作成が行え、かつ実行の流れをグラフィカルに表示できる。TPMとPROEDIT2とDEWLAPはプログラムのデバッグを視覚的に支援するシステムであるが、プログラムの作成はテキスト形式を用いる。VPPはラベル付きの有向グラフ表現とテキスト表現の両方を用いたプログラムの作成が行える。表1にあるように、これらのシステムは、前述の3つの視覚化のうち、1つまたは2つを支援しているのに対して、VPEは3つの視覚化を統合して支援するものである。

以下本論文では、2章でVPEの概略を、3章でVPEの基盤アーキテクチャを述べる。4章で実現方法を説明する。5章でVPEの使用例をあげる。

2. VPEの概略

VPEは、拡張論理型プログラミング言語Vocalogの視覚的プログラミング環境である。そこで、本章ではVocalogの特徴を簡単に紹介し、そのあとで、VPEの特徴に関して説明する。

2.1 Vocalogの概略^{3),14)}

Prologは述語表現 $p(t_1, \dots, t_n)$ を使って関係を表現するが、Vocalogは属性名 n_i と値 t_i の対 $n_i:=t_i$ のならば $[n_1:=t_1, \dots, n_k:=t_k]$ により関係を表現することができる。 n_i のことを名詞と呼び、 $[n_1:=t_1, \dots, n_k:=t_k]$ をリスト述語表現と呼ぶ。“:=”は名詞と値を区切るための区切り子である。名詞が述語の引数位置の概念に相当し、名詞の集合 $\{n_1, \dots, n_k\}$ が述語記号 p に相当する。名詞の集合 $\{n_1, \dots, n_k\}$ でオブジェクトの概念(クラス、タイプ)を表しているときとみなすことができる。例えば、Prologにおける

```
person( taro, male, "1955.07.28", sapporo)
```

という述語表現に対しては、Vocalogの

```
[ name:=taro, sex:=male,
  birthday:="1955.07.28",
  birthplace:=sapporo]
```

	VLP	TPM	PROEDIT2	VPP	DEWLAP	VPE
プログラムの作成の視覚化	○	×	×	○	×	○
プログラムのデバッグの視覚化	○	○	○	×	○	○
プログラムを利用するインターフェースの視覚化	×	×	×	×	×	○

表1 論理プログラミングにおける視覚化システムの比較

というリスト述語表現が対応する。

名詞を用いることで、オブジェクトが格納されている構造とは独立に、オブジェクトの属性に制約を加えることができる。例えば、「オブジェクトの出生地に関する属性値にsapporoという制限を加える」ということは、Prologでは、

```
person(_,_,sapporo)
```

のように述語表現を使わなくてもいいが、Vocalogでは

```
birthplace:=sapporo
```

のように、名詞とその値を対応付けるだけでよい。

Vocalogでは、オブジェクトの属性値間に制約を付加することで、オブジェクト間の関係をジェネリックに定義でき、それを形容詞と呼んでいる。aを形容詞、nとn'を名詞とすると、

```
[n:=t, a@n':=t]
```

というリスト述語は、属性nの属性値がtであるオブジェクトと、属性n'の属性値がtであるオブジェクトの間には、aという関係が成り立っていることを表している。例えば、

```
[name:='Taro', of_the_parent@name:='Jiro']
```

というリスト述語は、名前が'Jiro'という人は名前が'Taro'という人の親であることを表している。ここで、of_the_parentが形容詞であり、“@”は形容詞と名詞を区切るための区切り子である。

Vocalogでは、語構成子というものを使って名詞や形容詞を組み合わせることで、複雑な概念を記述できる。例えば、親であるという関係を表す形容詞of_the_parent、子供であるという関係を表す形容詞of_the_child、自分自身であるという関係を表すof_the_selfという形容詞があれば、兄弟であるという関係は「親の子供であって、自分自身ではない」というものである。

(of_the_child:of_the_parent)&(～of_the_self)
という形容詞句で記述できる。ここで、“:”と“&”と“～”が語構成子である。

2.2 VPEの特徴

VPEは、Tanakaらにより提案されているシンセティック・メディア・システムIntelligentPad^{11),12),13)}を用いて開発されている。IntelligentPadでは、すべての視覚オブジェクトは、機能を持った紙として表現される。この紙をパッドと呼んでいる。パッドの上には、複数のパッドをはることができる。パッドの上にパッドをはることで、レイアウトを行うとともに、機能の合成を行うことができる。

VPEでは、プログラムの作成の視覚化、プログラムのデバッグの視覚化、プログラムを利用するインターフェースの視覚化の3つのモードにおいて、パッドという単一の表現形態を用いている。

VPEでは、Vocalogにおける名詞や形容詞など、

プログラムの部品となるものを全てをパッドとして視覚化している。それらのパッドを視覚的に組み合わせることでプログラムを構成するシンセティック・プログラミングを行うことができる。例として、形容詞of_the_parent, of_the_child, of_the_selfと語構成子“&”, “-”のそれぞれを視覚化したパッドを組み合わせることで得られる合成パッドの例を図1に示す。図1において、四角い枠で示されるものがすべてパッドである。図1の合成パッドは前節の最後に示した、兄弟であるという関係を表す形容詞句に対応する。なお、図1には語構成子“-”に対応するパッドはないが、これは形容詞に対応するパッドどうしを図のように組み合わせること自体が、“-”の機能を実現しているからである。

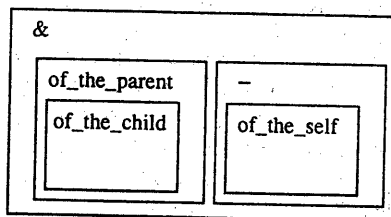


図1 VPEにおける視覚的な合成例

VPEでは、プログラムの部品となるパッドは、各モードに応じた見せ方を持つことができる。変数Xを表すパッドの各モードにおける見せ方の例を図2に示す。プログラムの作成時には、変数名を表示する(図2-a)。プログラムのデバッグ時には、その変数が束縛されていれば束縛された値を、それ以外では変数名をそのまま表示する(図2-b)。プログラムを利用するインターフェースのモードでは、変数Xに束縛された数値をバーメータで表示したり、バーメータを操作して変数Xを適当な数値に束縛することができる(図2-c)。図2の3つのパッドは、パッドの大きさが異なったり、表示の仕方が異なるが、すべて同一のパッドである。

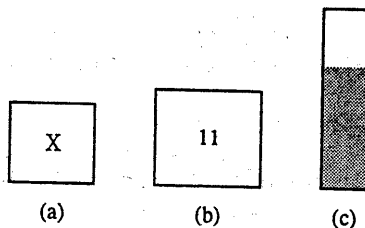


図2 異なる表示パッドを使った
同一変数の種々の見せ方

3. 基盤アーキテクチャ

本章では、まずVPEにおいてパッドとして視覚化されるものを明らかにする。次に、パッドのは

りあわせにより得られる合成オブジェクトについて説明する。そして、モードに応じたオブジェクトの見せ方について説明する。

3.1 視覚化されるオブジェクト

VPEでは、プログラムを構成する全ての部品をパッドとして視覚化する。すなわち、定数、変数、複合項、名詞、形容詞、語構成子、リスト述語、節の頭部、節の本体、節、形容詞を定義するための式をパッドとして視覚化する。それぞれ定数パッド、変数パッド、複合項パッド、名詞パッド、形容詞パッド、語構成子パッド、リスト述語パッド、ヘッドパッド、ボディパッド、節パッド、形容詞定義パッドと呼ぶ。これらのパッドをまとめて、VPE基本パッドと呼ぶ。

それに加えて、名詞や形容詞の定義を格納するルールベースと、Vocalogインタープリタを含むVocalogシステムといったシステムを構成するものも視覚化される。それぞれルールベースパッドとVocalogパッドと呼ぶ。これらのパッドをまとめて、VPEシステムパッドと呼ぶ。

3.2 合成オブジェクト

VPEにおけるパッドのはりあわせがどのような操作に相当するかを、主な組み合わせに関して説明する。

●変数パッドへの貼付

変数パッドXの上に、定数パッドCをはることで、変数Xを定数Cに単一化できる。同様に、変数パッドの上に複合項パッドや他の変数パッドをはることで、それらと単一化できる。

●複合項パッドへの貼付

複合項パッドは、それ自体だけではファンクタと引数の数が指定されているだけである。複合項パッドの上に、変数パッドや定数パッド、または別の複合項パッドをはることで引数を指定できる。

●名詞パッドへの貼付

名詞パッドの上に、定数パッドや変数パッド、または複合項パッドをはることで名詞の値を指定できる。

●形容詞パッドへの貼付

形容詞パッドの上に、名詞パッドをはることで、その名詞を修飾することができる。形容詞パッド A_1 の上に、別の形容詞パッド A_2 をはることで、合成演算子を使った形容詞句 $A_2:A_1$ を表すことができる。

●語構成子パッドへの貼付

語構成子パッドの上に、名詞パッドをはることで、名詞句を構成できる。さらに、その上に定数パッドなどの項を表すパッドをはることで、構成された名詞句の値を指定できる。

語構成子パッドの上に、形容詞パッドをはることで、形容詞句を構成できる。さらに、その上に名詞パッドをはることで、その名詞を修飾することができる。

語構成子パッドの上に、別の語構成子パッドをはることで、より複雑な名詞句や形容詞句に対応するパッドを構成することができる。

●リスト述語パッドへの貼付

リスト述語パッドの上に、名詞パッドや、名詞パッドを修飾している形容詞パッドをはることで、それらに対応する語を並べたリスト述語を構成できる。

●節パッドへの貼付

節パッドの上に、ヘッドパッドとボディパッドをはることで、節が構成される。ヘッドパッドの上には、複数の名詞パッドがはられる。ボディパッドの上には、複数のリスト述語パッドがはられる。

●形容詞定義パッドへの貼付

形容詞定義パッドの上には、2枚の名詞パッド N_1, N_2 とリスト述語パッドLをはることで、形容詞の定義を構成できる。

●ルールベースパッドへの貼付

ルールベースパッドは、節集合を格納する機能を持つ。ルールベースパッドの上に節パッドや形容詞定義パッドをはると、それらに対応する節や形容詞がルールベースに追加される。ルールベースパッドの上にすでにはられているパッドをはがすと、そのパッドに対応する節や形容詞の定義がルールベースから削除される。

●Vocalogパッドへの貼付

Vocalogパッドは質問を実行する機能を持つ。Vocalogパッドの上には、ルールベースパッドとリスト述語パッドがはられる。貼付されたルールベースパッドに格納されている節の集合が質問の実行に使われる。貼付されたリスト述語パッドに対応するリスト述語が質問として実行される。

3.3 モードに応じた見せ方

VPEには、プログラムの編集モードとデバッグモード、そしてプログラムを利用するためのインターフェースモードがある。VPE基本パッドとVPEシステムパッドは、どのモードに設定されているかというモード情報を持つ。デフォルトでは編集モードである。モードの変更に関する機能はVocalogパッドが持つ。Vocalogパッドの上に貼付されたパッドは、Vocalogパッドで設定されたモードとなる。

VPE基本パッドとVPEシステムパッドは、複数の見せ方を持つことができる。この見せ方のことを表示スタイルと呼ぶ。プログラムの編集、デバッグ、プログラムを利用するインターフェースの各モードに対して適当な表示スタイルがデ

フォールトで対応づけられている。モードと連動して、表示スタイルが切り替わる。

例えば、変数パッドはデフォルトで次の3つの表示スタイルを持つ。

- (1) 変数名を見せるスタイル
- (2) 変数の束縛状況を見せるスタイル
- (3) 変数への束縛をキーボードから与えることができるスタイル

(2)のスタイルでは、変数が束縛されていなければ変数名を、束縛されていれば束縛された値を表示する。(3)のスタイルには、(2)のスタイルが持つ機能も含まれている。各モードと表示スタイルとの対応はデフォルトで次のようになっている。プログラム作成モードには(1)の表示スタイルが設定される。デバッグモードには、(2)の表示スタイルが設定される。インターフェースモードには、(3)の表示スタイルが設定される。

VPE基本パッドおよびVPEシステムパッドの表示スタイルを変更するには、表示スタイル変更パッドを用いる(図3)。変更を行いたいパッドの上に表示スタイル変更パッドをはる。図3の矢印のボタンを操作することで、表示スタイルを変更する。スタイルに対して名前をつけることができ、それも表示スタイル変更パッドで行われる。図3のLockスイッチは、表示スタイルの変更を禁止するためのものである。Lockスイッチがonになっていると、モードと連動した表示スタイルの変更も禁止される。Editスイッチ、Debugスイッチ、Interfaceスイッチは、モードと表示スタイルとの対応関係の変更を行うためのものである。例えば、Editスイッチをonにすると、その表示スタイルが編集モードと対応づけられる。

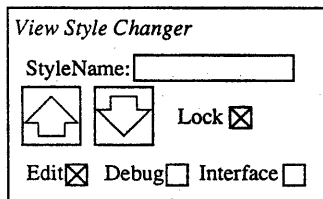


図3 表示スタイル変更パッド

VPEでは、これらの見せ方も種々のパッドのはりあわせで実現している。デフォルトでは、各モードに対応した3つの表示スタイルが用意される。IntelligentPadシステムに、プリミティブとして用意されているパッドや、新たにユーザーが作ったパッドを、表示スタイルとして追加できる。デフォルトの3つの表示スタイルの他に、表示スタイルを追加するための特殊スタイルがある。表示スタイルの追加は、表示スタイルを追加するスタイルに変更してから、追加したい表示スタイル用のパッドをはるにより行われる。

4 実現手法

この章では、3章で述べた基盤アーキテクチャの実現手法を説明する。最初に、IntelligentPadの動作機構を説明する。

4.1 IntelligentPadの動作機構

IntelligentPadにおけるパッドは、Smalltalk-80のMVC構造を単純化した構造を持つ。Mでパッドの内部状態が定義され、Vで表示の仕方が定義され、Cでマウス操作などに対する反応が定義される。Mにおけるいくつかの内部状態がスロットとして公開される。

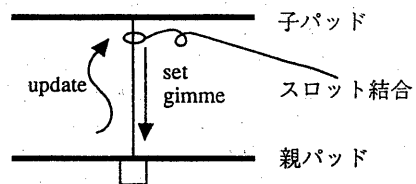


図4 パッド間の標準インターフェース

パッド間でのデータ通信は、はりあわせ構造を介した、set、update、gimmeという3つのメッセージとパッドの幾何属性を変更するジオメトリメッセージに限られている。この様子を図4に示す。

パッドのはりあわせ構造において、下のパッドを親パッド、上のパッドを子パッドと呼ぶ。子パッドは、自分より下のパッドのスロットのうちの一つだけに結合できる。setメッセージにより、子パッドは自分が結合しているスロットに値を入れることができる。親パッドの状態が変わると、子パッドにupdateメッセージが送られる。updateメッセージが送られると、子パッドは親パッドにgimmeメッセージを送り、自分が結合しているスロットの値を読み出す。setメッセージ、gimmeメッセージとも、親パッドが対応するスロットを持たなければ、そのメッセージは親の親に送られる。

パッドは共有コピーをとることができる。単一のパッドの場合、共有コピーをとると、M部が共有されて、VとCは個別に持つパッドが作られる。複数のパッドがはりあわせた合成パッドの共有コピーをとると、はりあわせ構造の一番下にあるパッドのM部が共有され、他の部分は個別に持つ合成パッドが作られる。

4.2 質問の構成と実行機構

質問を実行する際のパッドのはりあわせ構造は図5ようになる。一番下に、Vocalogパッドがあり、その上に質問に相当するパッドがはられており、Vocalogパッドのqueryスロットに結合してい

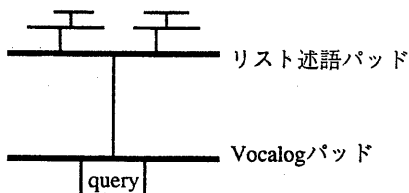


図5 質問実行時のはりあわせ構造

る。

質問の実行は次の手順で行われる。

- (1) queryスロットに初期化のための特殊シンボル#initializeがセットされる。
- (2) Vocalogパッドからupdateメッセージが子パッドに送られる。
- (3) Vocalogパッドの上にあるすべてのパッドにupdateメッセージが伝搬して、初期化フェーズであることが伝わる。
- (4) はりあわせ構造の一番上にあるパッドから順々にsetメッセージが親パッドに送られていき、質問が構成されていく。
- (5) Vocalogパッドのすぐ上のパッドに、質問に相当する字面が構成され、それがVocalogパッドのqueryスロットにセットされる。
- (6) 質問を処理した結果が成功の場合は、成功したことを示す特殊シンボル#yesと正解代入がVocalogパッドのqueryスロットに入れられる。失敗した場合は、失敗したことを示す特殊シンボル#noがVocalogパッドのqueryスロットにセットされる。
- (7) Vocalogパッドから子パッドにupdateが送られる。質問を処理して得られた情報(成功か失敗かということと、成功の場合は正解代入)が子パッドに伝搬する。
- (8) 各パッドで表示が更新される。例えば、変数パッドの表示が、束縛された値になる。

4.3 表示モードの切り替え機構

VPE基本パッドとVPEシステムパッドは、複数のパッドのはりあわせにより構成されている。各パッドの内部状態を保持するパッドが一番下にあり、その上にマルチファセットパッドと呼ばれる種類のパッドがはられる。マルチファセットパッドの上に、各表示スタイル用のパッドがはられている。各表示スタイル用のパッドは、内部状態を保持している一番下のパッドのスロットと結合する。

マルチファセットパッドと呼ばれる種類のパッドは、その上にははられている複数のパッドのなかから、一つだけを表示し、かつ自分自身の大きさ

を表示したパッドにあわせる。どのパッドを表示するかを選択できる。

VPEで用いているマルチファセットパッドには、表示スタイルとパッドとの対応を格納するスロットと、モードと表示スタイルとの対応関係を格納するスロットと、モード切り替えのロックに関するフラグを格納するスロットがある。マルチファセットパッドでは、これらのスロットを参照して、表示スタイルを切り替える。

変数パッドの例を図6に示す。図6の一番下のパッドは、変数に関する情報を保持するパッドである。変数名がスロットvarNameに、束縛された値がスロットvarValueに入る。vocalogDataは正解代入などのデータをやり取りするためのスロットである。モードに関するデータも、このスロットに格納される。マルチファセットパッドでは、スロットvocalogDataからモードに関するデータを得て、表示スタイルとモードとの対応を調べて、そのモードに相当する表示スタイルに切り替える。図6のパッド P_1 、 P_2 、 P_3 が表示スタイル用のパッドである。 P_1 、 P_2 、 P_3 はそれぞれ、変数名を表示するパッド、変数の束縛状況を表示するパッド、変数への束縛をキーボードから与えることができるパッドである。 P_1 が編集モードに、 P_2 がデバッグモードに、 P_3 がインターフェースモードに対応づけられている。マルチファセットパッドにより、一度に P_1 、 P_2 、 P_3 のうち、一つだけが表示される。

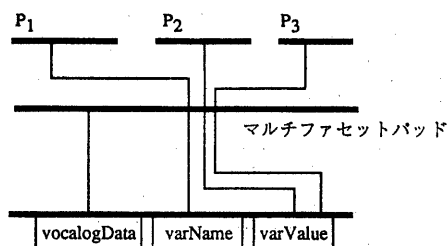


図6 変数パッドの見せ方の切り替え機構

VPE基本パッドとVPEシステムパッドの共有コピーを取った場合、実際に共有されるのは、一番下のパッドが持つ内部状態である。その上にあるマルチファセットパッドが保持する情報に関しては、それぞれのパッドが個別に持つ。図6でいうと、変数に関する情報を保持するパッドだけが共有される。したがって、それぞれのマルチファセットパッドでは、別々の表示スタイル用のパッドを選択できる。例えば、デバッグモードにおいて変数パッドの共有コピーをとって、一方のパッドをインターフェース用の表示に切り替えることができる。

5 例

世界の都市の名前と緯度・経度を格納するデー

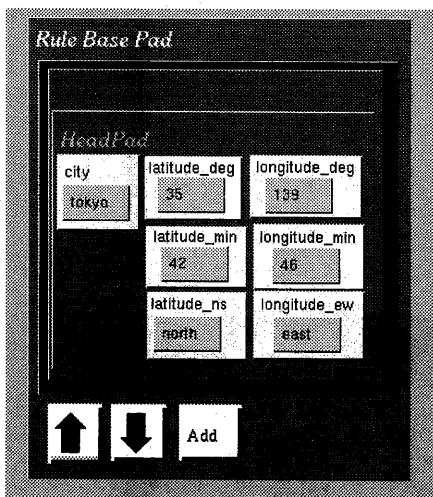


図7 編集モードにおける表示例

データベースに関する例をあげる。図7, 図8, 図9は、プログラムの編集モード、デバッグモード、インターフェースモードにおけるハードコピーである。

図7において、一番下にあるパッドはVocalogパッドである。編集モードと対応づけられている、Vocalogパッドに貼付されているルールベースパッドが表示されている。ルールベースパッドでは、格納されているファクトの一つを表示している。

図8の左側のパッドでも、一番下にあるパッドはVocalogパッドである。デバッグモードでは、デフォルトで表示されるのは、質問に相当するリスト述語パッドである。図8の左側のパッド上のstepボタンは実行を1ステップ進めるために使う。図8の右側のパッドは、図8の左側のパッドの共有コピーをとり、表示スタイルを変更して、ルールベースパッドを表示したものである。ルールベースパッドでは、現在選択されているファクトを表示している。ルールベースパッドの下の方にある表示器が?を示しているのは、この節の選択が成功したかどうかが未定であることを示している。成功すると○に、失敗すると×に変わる。図8の右側のパッド上のstepボタンを押して、実行が成功すると、変数の束縛状況が示される。

図9のパッドでも一番下にあるパッドはVocalogパッドである。インターフェースモードに対応づけられている、リスト述語パッドが表示されている。このリスト述語パッドは図8の左側のパッドと同一のものであるが、表示スタイルが異なっている。都市名"sapporo"をテキストで入力して、Searchボタンを押した後の状態が図9である。Searchボタンを押すことで質問が実行されている。緯度と経度がメーターを使って表示されている。地図上でsapporoの場所が*印で示されている。地図上の*と・は同じ種類のパッドである。それらのパッドは都市名を保持しており、リスト述語パッド中の都市名に関する内部状態を参照して、それと同じ都市名のものが*を表示し、他のものは・を表示する。

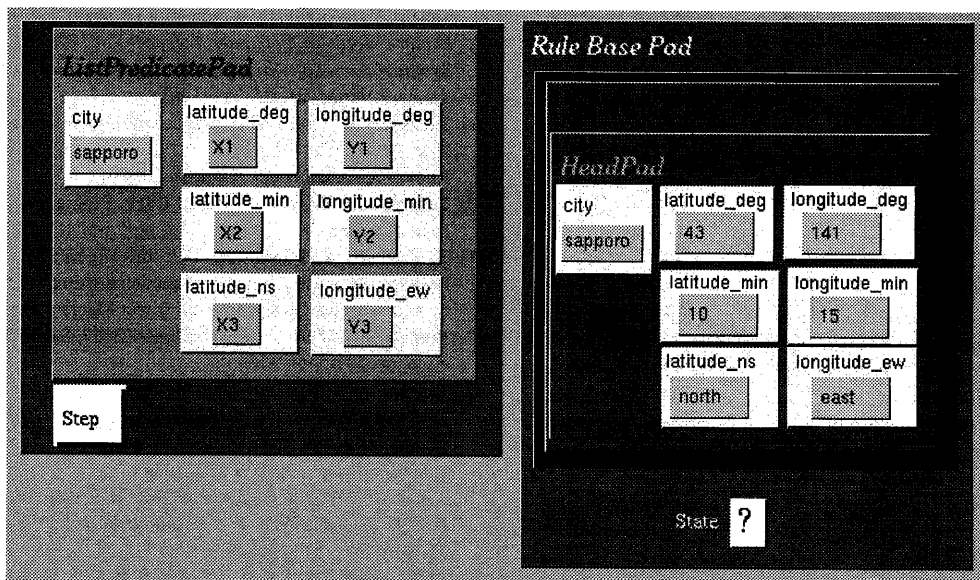


図8 デバッグモードの表示例

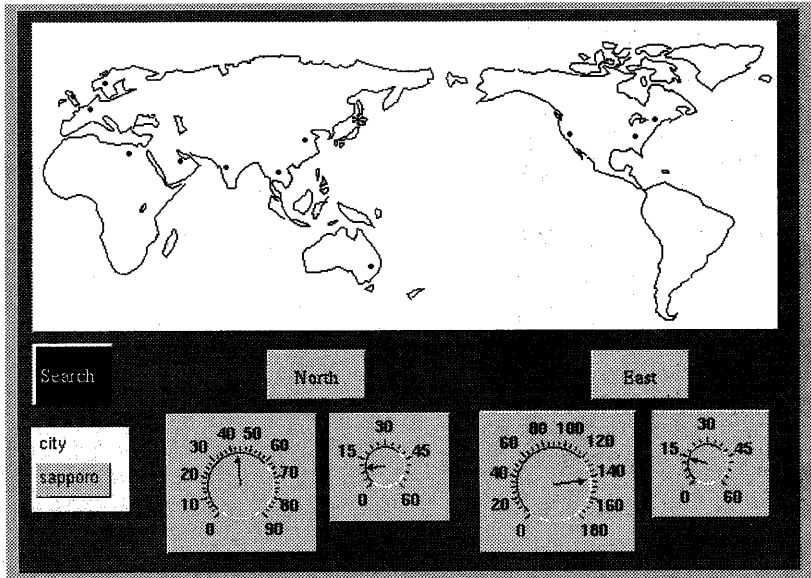


図9 インターフェースモードの表示例

6 おわりに

拡張論理型プログラミング言語Vocalogの、視覚的統合プログラミング環境VPEを示した。VPEはシンセティック・メディア・システムIntelligentPadを用いて開発されている。

VPEでは、プログラムの作成、デバッグ、プログラムを利用するインターフェースの3つのモードを統合してつなぎめなく扱えるようにするために、次の二点を行った。第一に、3つのモード全てにおいて視覚オブジェクトを、パッドという同一の表現形態で統一した。第二に、同一のオブジェクトに対して、種々の見せ方(表示スタイル)を与えることができるようにした。これにより、同一オブジェクトを複数のモードで共有して、なおかつ、それぞれのモードに適した見せ方を与えることができた。

参考文献

- 1) Dewar, A. D. and Cleary, J. G. : Graphical display of complex information within a Prolog Debugger, *Int. J. Man-Machine Studies*, Vol.25, pp. 503-521 (1986)
- 2) Eisenstadt, M. and Brayshaw, M. : The Transparent Prolog Machine (TPM): An Execution Model and Graphical Debugger for Logic Programming, *J. Logic Programming*, Vol. 5, pp. 227-342 (1988)
- 3) Goto, F. and Tanaka, Y. : Introducing a Large Vocabulary into Prolog, *Proc. Pacific Rim International Conference on Artificial Intelligence '90*, pp. 816-821 (1990)
- 4) Ladret, D. and Rueher, M. : VLP: a Visual Logic Programming Language, *Journal of Visual Languages and Computing*, Vol. 2, pp. 163-188 (1991)
- 5) Numao, M, Morishita, S. and Maruyama, H. : How Should Prolog Computation Be Represented for Practical Use ?, *New Generation Computing*, Vol. 8, pp. 95-112 (1990)
- 6) Pau, L. F. and Olason, H. : Visual Logic Programming, *Journal of Visual Languages and Computing*, Vol. 2, pp. 3-15 (1991)
- 7) Tanaka, Y. : Information Space Model, *Proc. 2nd Workshop on Formal Bases for Databases*, Toulouse (1979).
- 8) Tanaka, Y. : Vocabulary Building for Database Queries, *Lecture Notes in Computer Science*, vol.147: *RIMS Symposia on Software Science and Engineering*, Springer-Verlag, pp.215-232 (1983).
- 9) Tanaka, Y. : Roles of a Vocabulary in Knowledge-Based Systems, *IFIP WG 10.1 Int'l Workshop on Knowledge Architecture*, Gotenba (1987).
- 10) Tanaka, Y. : Vocabulary-Based Logic Programming, *Proc. InfoJapan'90*, Tokyo, Vol.2, pp. 25-32 (1990).
- 11) Tanaka, Y. and Imataki, T. : IntelligentPad: A Hypermedia System allowing Functional Composition of Active Media Objects through Direct Manipulations, *Proc. of IFIP World Computer Congress '89*, San Francisco, pp. 541-546 (1989)
- 12) Tanaka, Y. : A Toolkit System for the Synthesis and the Management of Active Media Objects, *Proc. of 1st International Conference on Deductive and Object-Oriented Databases*, pp. 259-277 (1989)
- 13) Tanaka, Y. : A Synthetic Dynamic-Media System, *Proc. International Conference on Multimedia Information Systems*, Singapore, pp. 299-310 (1991)
- 14) 後藤、田中: Vocalog: 語彙を用いたジェネリックな概念記述が可能な拡張論理型プログラミング言語, *情報処理学会論文誌第33巻第4号*, pp. 512-520 (1992).