

CL: ドメイン言語構築機能を強化した フレーム型知識表現システム

渡辺正信*, 岩本雅彦*, 出口幸子**

*日本電気㈱ C & C システム研究所, **日本電気ソフトウェア㈱

1. はじめに

エキスパートシステム(MYCIN, DENDRAL, R1, PROSPECTOR, CASNET, MOLGEN等、特にこれらを第1世代エキスパートシステムと呼ぶ)[1]に代表される知識ベースシステム(応用人工知能システム)を開発するために各種の知識表現方式(プロダクションルール、意味ネット、フレーム、述語論理等)が採用された。このことにより各種表現方式の利点及び問題点が、現実の具体的な適用領域において指摘され、明確に認識されてくるに至った。即ち、“知識表現において万能薬はない”という認識である。

一方、浅い推論を行なう第1世代エキスパートシステムとは異なり、深い推論を目指す第2世代エキスパートシステムの開発を可能とする知識表現方式が必要となってきた。そこでは、因果関係モデルや構造化された知識の表現と、関係に基づくインヘリタンス(性質の継承)の実現等がキーとなる。

又、我々は、概念ネットワークに基づくフレーム型知識表現システム(COMET)[2]の開発、及び、それを利用したCMOSレイアウト設計支援エキスパートシステム(CHRIS)[3]の開発を行なった経験を通して、“今後の知識表現システムは、ドメイン言語構築機能を強化すべき”という結論に至った。これは、CL(Conceptual-Network-Based Language)開発の大きな動機となったと同時に、先に述べた第2世代エキスパートシステムの開発を支える知識表現システムの基本思想でもある。

以下、CLの設計思想、システム構成、ドメイン言語構築機能について述べる。

2. 設計思想

CLの設計思想を①ドメイン言語構築機能の強化、②複数の表現パラダイムの統合化、③概念の構造化、④有意な処理の効率化という4つの観点から述べる。

2.1 ドメイン言語構築機能の強化

“知識表現に万能薬はない”を換言すれば、“知識表現は、特定応用ドメインの問題構造にうまく適合するものを使うのが一番良い”ということになる。又、このことは、現在有効と考えられている各種表現方式(ルール、フレーム等)を単に可能とすれば良いというわけではない。つまり、各応用ドメイン固有の知識を記述するための言語(これを、ドメイン言語と呼ぶ)を構築することが本質的に必要となることを示している。ドメイン言語を構築するとは、その言語のインタプリタを構築することを意味する。CHRIS[3]においては、ドメイン言語は基本的にLISPで構築された。CLは、ドメイン言語のインタプリタの構築を効率化するための表現素(ドメイン言語構築プリミティブ)を提供し、ドメイン言語構築機能を強化する。つまり、CLはLispをベースにした強力なプロトタイピングツールを目指す。

2.2 複数の表現パラダイムの統合化

知識表現に万能薬はないといっても、現在、表現方式として有効と認識されているいくつかの一般的表現枠組(表現パラダイム)がある。この代表的なものがオブジェクト、データ、ルール、手続き、論理指向である。CLは、フレーム表現形式(オブジェクト、スロット、ファセット、バリュウ)の4つ組表現と概念ネットワークをベースに上記の複数の表現パラダイムを統合化する。この思想は、STROBE[4], LOOPS[5], KEE[6], SRL+[7]等に共通のものである。

2.3 概念の構造化

応用ドメインの有能な問題解決プログラムを構築するためには、そのドメインの問題構造をうまく、

そのプログラムの中に反映させることが必要である。このためには、そのドメインの問題解決に必要な概念をプログラムとして効果的に構造化しなければならない。

ここで、概念とは、“思考の単位”である。思考の単位は、その思考に付随したデータと手続きで表現される。換言すると、WhatとHowの知識を基に概念（思考単位）が形成される。概念を構造化するポイントは、複数の概念の意味的体系化と概念を構成するデータと手続きの共有化にある。

(I) 意味的体系化

CLは、概念の意味的体系化を概念ネットワークの拡充として行なう。概念ネットワークは、概念を示すオブジェクトと呼ぶノードと、オブジェクト間の関係を示すスロットと呼ぶリンクからなるネットワークである。特に、CLは、オブジェクト間の汎化関係を軸に、意味的体系化（又は、概念の抽象化）を支援する。

(II) 共有化

計算機上のプログラムを現実的な環境下で稼働させるためには、概念を構成するデータや手続きを共有化することは必須である。又、利用者から見ると、共有化は、冗長なデータ/手続きを極力減らすことによる変更処理の効率化につながる。共有化のこのメリットは、データを利用する側、格納する側の両者から歓迎される。

CLにおいて、データや手続きの共有化は、概念の意味的体系化を通して行なわれる。つまり、汎化階層において上位オブジェクト（上位概念）は、それより下位にあるオブジェクトのデフォルトとなるデータ、手続きを保持するということで共有化を支援する。

2.4 有意な処理の効率化

概念の構造化では、意味的体系化と共有化というメリットを強調した。しかし、このために、①アクセス時間の増大、②情報選別の手間の増大というデメリットが発生することを注意しなければならない。

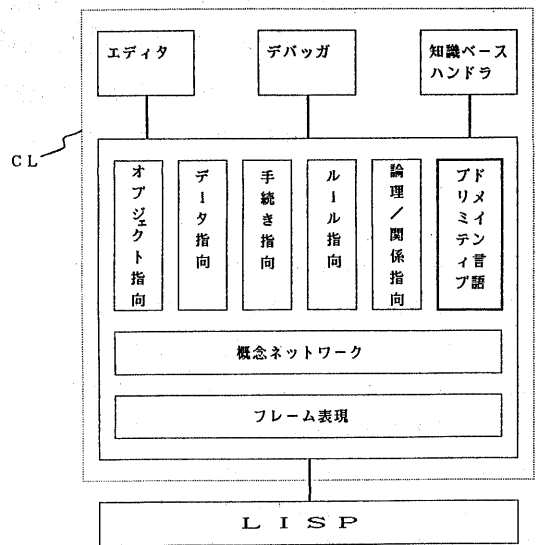


図1 CLの構成

CLは、このデメリットを減少させるため以下の対策を講ずる。

(I) アクセス時間の効率化

マルチインヘリタンスを許す汎化階層において、下位オブジェクトに必要な情報（データ又は手続き）を持つ上位オブジェクトのサーチを高速化するために、オブジェクトの各スロットは、そのスロットの情報を保持するオブジェクトへのポインタを Links Upファセットの値に持つ[4]。これは、オブジェクトは、種々の見え方があり、各見え方はスロット毎に対応する他のオブジェクトとの関係によって表現されるという考え方に基づく。

さらに、サーチを省略化する対策として、利用者要求に基づくスロットのキャッシングを行なう[4]。

これは、オブジェクトにスロットそのものが存在しない時は、上記の LinksUpの恩恵をうけられないため、利用者の要求に従って、必要なスロットを生成し、その後のサーチを高速化する。

(II) 情報選別の効率化

情報選別の手間とは、あるオブジェクトから他のオブジェクトへ伝播する情報を選別又は限定するため必要とする手間のことである。例えば、汎化階層において、上位オブジェクトから下位オブジェク

トへの情報の伝播の仕方に制約を加えるための手間がある。CLでは、オブジェクト間での情報伝播、情報選別は、基本的にオブジェクト間の”関係”に基づき決定されるものとする。つまり、オブジェクト間の任意の関係（スロット）に関係オブジェクトを定義して、情報の伝播の仕方に制約／規約を与えるわけである[7]。

3. CLの構成

図1は、CLの構成を示す。現在、CLはVAX-11/780の Frantz-Lisp上に開発中である。CLの核となるフレーム表現、概念ネットワーク、オブジェクト指向、データ指向の機能仕様は、COMET[2], STROBE[4] のものをほぼ踏襲している。

3.1 基本要素

フレーム表現：概念を（オブジェクト、スロット、ファセット、バリュー）の4つ組で表現すること。

オブジェクトの定義：オブジェクトは、スロットの集合で表現される。
スロットは、オブジェクトの属性、構造、他のオブジェクトとの関係、オブジェクト自身のメタ情報を示す。
ファセットは、スロットの種々の側面（例えば、構造的な性質とか属性、スロットやバリューのメタ情報等）を表現する。バリューは、ファセットに対する具体値（実際の値）を示す。

(例) (DefObject ; オブジェクト定義関数
Bluebird ; オブジェクト名
Class ; オブジェクトタイプ
(SportsCar FamilyCar); 汎化オブジェクト
NissanCar ; グループ名

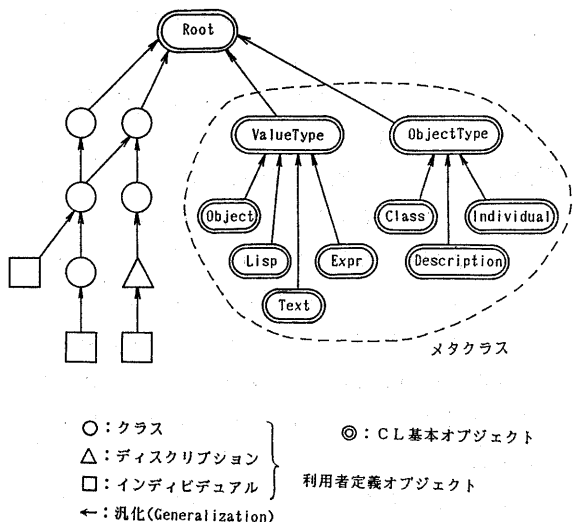


図2 汎化階層からみたCLの概念ネットワーク

((Style (ValueType . Object);利用者定義
スロット
(Value . BoxStyle))
(Width (ValueType . Expr));利用者定義
スロット
(Body (ValueType . Object));利用者定義
スロット
(Comment (ValueType . Text);利用者定義
スロット
(Value . "This ia a test"))))

オブジェクトタイプ：次の3つのタイプがある。

- Individual ; 自分より特殊化オブジェクトがないもの。
- Class ; 基本的にIndividualでないオブジェクト。
- Description ; クラスオブジェクトを例示化 (Instantiation)するための補助的なオブジェクト。

バリュータイプ：次の4つのタイプがある。

- Object ; CLのオブジェクト
- Lisp ; Lisp関数
- Expr ; LispのS式（アトムカリスト）
- Text ; ドキュメント

グループ名：CLのオブジェクトの任意の集合に与えられた名前。

グループはサーチの範囲等を限定する場合に使用される。

概念ネットワーク：概念の意味的体系化は、概念ネットワークを拡充することにより行なわれる。この基本枠組は、概念単位を示すオブジェクトをノード、オブジェクト間の任意の関係をリンクとするネットワークである。オブジェクト間の関係は、汎化関係を軸に陽に管理される。さらに、利用者定義関係オブジェクトを通してその他の関係が管理される。

図2は、汎化階層から見たCLの概念ネットワークを示す。

CL基本オブジェクト：

Root；CLの全てのオブジェクトはRootの特殊化オブジェクトとなる。

ObjectType, Class, Description, Individual；各オブジェクトタイプに関するデフォルトのデータと手続き（そのオブジェクトタイプのオブジェクト、スロットの生成／削除等）を規定するオブジェクト。

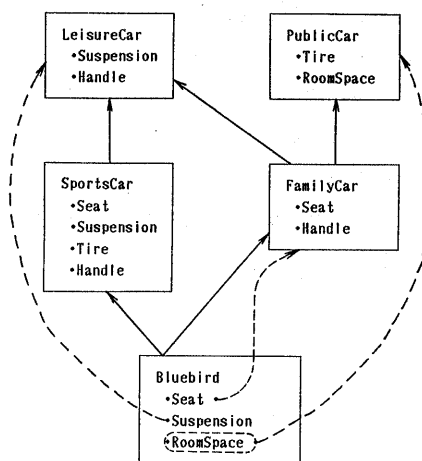
ValueType, Lisp, Object, Expr, Text；各バリエータタイプに関するデフォルトのデータと手続き（ファセットの生成／削除／参照／変更／プリント等）を規定するオブジェクト。

（注）上記Root以外のオブジェクトは、全てメタクラスの役割を果たす。しかし、CLでは、それぞれのオブジェクトタイプはMetaClassではなく、Classとして扱う。

インヘリタンス：

①デフォルト継承；CLにおいて、汎化階層の上位オブジェクトは下位オブジェクトのデフォルト情報を保持することが原則である。故に、上位オブジェクトの全ての情報は全て下位オブジェクトにデフォルトとして継承される。

②マルチインヘリタンス；複数の上位オブジェクトの情報をデフォルトとして利用できる。但し、スロットには、高々1個の上位オブジェク



---; スロットキャッシング

図3 マルチインヘリタンスとキャッシング

トが指示される（そのスロットの LinksUpファセットに明示される）。図3は、Bluebirdオブジェクトが複数の上位オブジェクトの情報を、継承すること、各スロットは明示的にそれぞれの上位オブジェクトを保持することを示している。LinksUp の情報を使ってアクセス時間が高速化される。

③継承の制約；上位オブジェクトから下位オブジェクトへの情報の継承に関する制約を規定するためには、次の2つの方法がある。

（Ⅰ）各スロットのデモンを利用する。例えば、スロット値の変更前デモンに制約を記述する。これは、情報を受理するオブジェクトのスロットに自分が受理できるものは何かを明示する方法である。

（Ⅱ）関係オブジェクトを利用する。これは、オブジェクト間の情報の流れの中で、そのオブジェクトを結合する関係（スロット）に付随するものは、関係自身の記述の中で宣言されるべきとする考え方に基づいている。具体的には、CLでは、関係オブジェクトを生成することにより実現される。

尚、デモン、関係オブジェクトは、後で説明する。

キャッシング：

概念ネットの中に構造化／共有化された情報をサーチするための高速化対策のひとつとして、CLはスロットキャッシングを提供する。図3を使ってこの方式を説明する。まず、BluebirdオブジェクトにRoomSpaceスロットがない場合を想定する。この時、BluebirdのRoomSpaceの値を問われたら、CLはサーチを実行する。もし、このスロットに対してキャッシング指示があれば、CLは、このサーチ結果を基に、RoomSpaceスロットをBluebirdオブジェクトに作成する。以後、このスロットに対する問いに対して当初のサーチは不要となり、高速化が図られる。

3.2 表現パラダイム

図1に示すように、CLは、複数の表現パラダイム（オブジェクト指向、データ指向、手続き指向、ルール指向、論理／関係指向）及びドメイン言語プリミティブを、概念ネットワーク／フレーム表現ベースに提供する。ここでは、特に、CLでのオブジェクト指向、データ指向を説明する。ドメイン言語プリミティブ等は、次節で述べる。

オブジェクト指向：

現在、オブジェクト指向プログラミングの定義がはっきりしているわけではない。フレーム表現自身、データ、手続きを内包させてモジュール化を進めるという点において、もともとオブジェクト指向である。つまり、CLでの概念ネットワーク自身がオブジェクト指向のひとつの大きな要素である。オブジェクト指向のもうひとつの主要な要素は、メッセージによるオブジェクト間の通信である。

ここでは、特に、CLでのメッセージ処理について説明する。CLは、次のメッセージ関数を提供する。

(Message Object Slot Facet MSG)

但し、MSG=(Arg1 ... Arg n)；変数リスト、又は=Arg；変数。Objectは、メッセージのレシーバ、Slot, Facetはそのセクタ、MSGはメッセージ変数となる。

(Message \$Obj1 'S1 'F1 'M1) は、次のように処理される。

(I) オブジェクトObj1にスロットS1が存在する時、

①S1にファセットF1が存在する時、

F1の値が関数ならばこれが起動され、関数でなければ、その値がリターンされる。

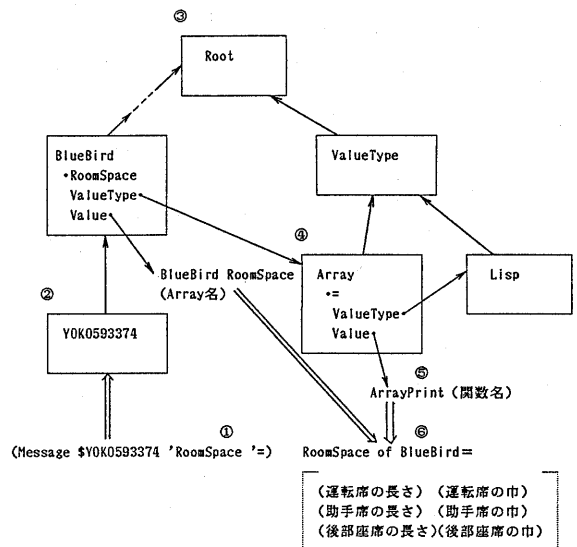
②S1にF1が、Rootまでのサーチの結果、存在しない時、

このメッセージは、S1スロットのValueTypeファセットの値に記述されたオブジェクト(ValT)に再送される。但し、ValTオブジェクトのF1スロットのValueファセットへ再送される。

(II) オブジェクトObj1にS1が存在しない時、

このメッセージは、Obj1のオブジェクトタイプスロットに記述されたオブジェクトのスロットS1のValueファセットに、再送される。

図4は、CLでのメッセージ処理を示す例である。ここで、データタイプオブジェクトArrayは次のように定義されているとする。



処理手順

- ①YOK0593374にRoomSpaceをプリントせよ('='で示される)というメッセージが送られる。ここで、#は名前をオブジェクトポインタに変換するためのread macroである。
- ②'RoomSpace'スロット、=ファセットを持つオブジェクトをYOK0593374、及び、その汎化オブジェクトを対象として探す。
- ③探索が失敗する。
- ④ValueTypeオブジェクトへ再送される。
- ⑤手続きArrayPrintを評価。
- ⑥Arrayのプリント。

図4 メッセージ処理

(I) ドメイン用語生成/定義プリミティブ

① CLでのオブジェクト生成プリミティブ: DefObject関数とは異なり、既存のオブジェクトを汎化又は特殊化することによって新たなオブジェクトを生成する。

・制限オブジェクト

(SpecializeObject Object Slot Val)

: ObjectのSlotの値を Valに制限して、新しいオブジェクトを生成する。

(NameObject Array

(RestrictObject ValueType

((= (Value . PrintArray)

(Value . Lisp))))))

これは、図4で示した Arrayオブジェクトの別の生成法である。

・緩和オブジェクト

(GeneralizeObject Object Slot Val)

: ObjectのSlotの値を Valに緩和して新しいオブジェクトを生成する。

②関係オブジェクト生成プリミティブ:

(DefRelation

Address ; 関係名

Relation ; 汎化オブジェクト

AddressOf ; 逆関係

CarOf ; インヘリタンスのバイパス

(bypass)関係名

((Inclusion ; 継承する情報
(Value . PlatePublisher)) の制約。ここ

ではPlatePublisherの情報のみが Address関係を通して継承される。

(Domain

(Value . Things)) ; ドメイン制限(Addressをスロットとして持つオブジェクトの

(Range 制限)

(Value . Places))))); レンジ制限(Address

スロットの Value値の制限)

この Address関係オブジェクトの定義により、あるオブジェクトの Address値は、汎化オブジェクトからではなく CarOfの関係を通して継承される。

③制約オブジェクト生成プリミティブ:

(DefConstraint

Adder ; 制約オブジェクト名

nil ; 汎化オブジェクト

(X Y Z) ; 制約変数リスト

((eq X ; 導出関数リスト

(- Z Y))

(eq Y

(- Z X))

(eq Z

(+ X Y))))

; X+Y=Z という制約条件式を管理するオブジェクト (Adder) を生成する。

Adderは、変数 X, Y, Zの任意の2つの変数の値から残りの変数の値を決定する。

(SpecializeConstraint

#C1; 制約オブジェクト名

Adder; 汎化制約オブジェクト

(A.a A.b 10)); 制約変数リスト

; オブジェクトAのスロット a, bの値に対して a+b=10の制約を Adderの特殊オブジェクトとして生成する。

(II) オブジェクト間関係導出プリミティブ

① (RelBetween Obj1 Obj2): オブジェクト Obj1 からオブジェクト Obj2 を見た関係 (スロット) 名を導出。

② (MoreSpecial Obj1 Obj2): Obj2がObj1の汎化オブジェクトの時、Obj1をリターンする。

③ (MoreGeneral Obj1 Obj2): Obj1がObj2の汎化オブジェクトの時、Obj1をリターンする。

(Ⅲ) スロット値導出プリミティブ

- ①(The Slot Object) : オブジェクトObjectのスロットSlotの Value値をリターンする。

(注) (The Slot ...SlotK Object) : オブジェクトObjectのスロットSlotK に記述されているオブジェクトの...スロットSlot1 の Value値をリターンする。

- ②(>> Obj Slot Val) : オブジェクトObj のスロットSlotの Value値に Valをセットする。

- ③(Value? Obj Slot Val)=(#= (The Obj Slot) Val)

(Ⅳ) ドメイン理論(ルール)記述プリミティブ

ドメイン知識は、基本的に、そのドメインの用語と用語を用いたルールにより記述できる。ドメイン理論は、これらルール集合からなる。以下、ドメインルールを記述するプリミティブ例を示す。

- ①(ThereExist :x Class s.t. (.....))

: 条件(.....)を満足するクラスClass の特殊化オブジェクトを変数: x にバインドする

- ②(ThereExistClass Ind :y s.t. (.....))

: オブジェクトInd の汎化オブジェクトで、条件(.....)を満足するクラスオブジェクトを変数: y にバインドする。

- ③論理演算プリミティブ: #AND, #OR, #NOT

この他、ルール制御用のプリミティブとして、推論方向制御プリミティブ(前向き、後向き指定等)、パターンマッチング記述プリミティブ、コンフリクト解消記述プリミティブ等がある。

(Ⅴ) その他プリミティブ

- ・エラー処理記述プリミティブ(利用者定義可能なエラー処理、例外処理記述用)
- ・アジェンダ記述プリミティブ
- ・ドメインエディター作成用プリミティブ

以上、Lispでのドメイン言語インタプリタ構築を効率化するために、CLが提供するプリミティブの例を示した。

5. おわりに

CL(フレームと概念ネットワークに基づく知識表現システム)の設計思想、構成、ドメイン言語プリミティブについて述べた。特に、ドメイン言語構築機能を強化することが今後の知識表現システムにとって重要となる。これは、Krypton[8]システムでの機能的アプローチ、KEE[6]でのTellAndAskオペレータの提供等と共通するものである。さらに、このアプローチはLispプログラミングの生産性を向上させるツールを求めていくものと同時に、今後の新しい表現パラダイム/プログラミングパラダイム(例えば、Taxonomic Programming[9]等)を求めていくためのベースとなる。

CLは、現在開発中で、オブジェクト指向、データ指向等が稼働している。ルール指向/論理指向は現在設計中である。又、CLのドメイン言語プリミティブの評価及び強化は、各種応用ドメインでの使用を通して行なっていく。

最後に本研究の機会を与えていただいた当研究所箱崎勝也主管研究員、山本昌弘部長、小池誠彦主任にそれぞれ感謝いたします。

参考文献

- [1] Hayes-Roth, F., Waterman, D.A., and Lenat, B., "Building Expert Systems," Addison-Wesley Pub. Co., Inc., 1983.
- [2] 渡辺, "COMET: 概念ネットワークによる知識表現システム," 情報, 知識工学と人工知能研究会30-3, 1983.6.
- [3] 渡辺, "CHRIS=CMOSレイアウト設計支援知識ベースシステム," 情報第30回全国大会, 1985.3., pp1413-1414.
- [4] Smith, R.G., "STROBE: Support for Structured Object Knowledge Representation," Proceedings of IJCAI-83, August, 1983, pp855-858.
- [5] Bobrow, D.G. and Stefik, M., "The LOOPS Manual," Knowledge-Based VLSI Design Group, Xerox PARC, August, 1981.
- [6] 塩沢, "エキスパートシステム開発支援ツール"KEE", "Expert Systems, Vol.1, No.1, (株)CSK総合研究所, Winter, 1985, pp9-12.
- [7] Wright, J.M. and Fox, M.S., "SRL/1.5 User Manual, Technical Report, The Robotics Institute, CMU, December, 1983.
- [8] Brachman, R.J., Fikes, R.E., and Levesque, H.J., "Krypton: A Functional Approach to Knowledge Representation," IEEE Computer, October, 1983, pp67-73.
- [9] Woods, W.A., "What's Important About Knowledge Representation?," IEEE Computer, October, 1983, pp22-27.