4－4

# MENDELにおける意味ネットを用いた部品結合

# Software Reuse using Semantic Network in MENDEL

伊藤美香子　内平直志
Hikako ITO　Naoshi UCHIHIRA
㈱東芝　システム・ソフトウェア技術研究所
Systems & Software Engineering Lab. TOSHIBA Corp.

In this paper, we describe a new binding mechanism for software reuse, in which reusable components written in the language MENDEL are interconnected automatically. This mechanism enables semantic matching of software components by using a semantic network.

Every software component has the input/output attributes. For each i/o attribute, the semantic structure is defined on the semantic network. By using this semantic structure, bindable components are definded and total ordering is introduced into them. Therefore, the most adequate component is selected from the reusable software library.

MENDEL is a Prolog-based Concurrent Programming Language, which we have developed as a kernel language for Intelligent Programming Environment, called MENDELS ZONE.

## まえがき

本稿では、Prologベースの並列プログラミング言語MENDEL/87 [1](以下、単にMENDEL) におけるプログラム部品再利用という観点から、意味ネットを利用した部品の結合について述べる。MENDELは、リアルタイムシステムのプロトタイプ記述を目的とする言語であり、我々は再利用をプロトタイピングの重要な手段として位置づけている。

本稿では、まず部品検索・結合と、それを利用したプログラミング言語MENDELについて簡単に説明し、次に部品検索・結合のための意味ネット、及び意味ネットを知識として用いた部品結合について述べる。

## 部品検索・結合

まず、ここで考える部品とは、タスクやプロセスのような並行に動きうる単位のブラックボックス型のプログラム部品についてである。

部品検索は、部品が蓄積されているデータベースから欲しい部品を検索する方法である。我々は、入出力仕様だけに限定しても有効な部品検索・結合が可能であると考え、MENDELにおける部品結合では、入出力仕様による部品の仕様表現を採用した。また、与えた仕様を満たす単一の部品は存在しないが、複数の部品の組合わせでなら仕様を満たせるといった場合がある。このような複数個の部品の組合わせの検索に対してもアプローチが比較的容易である。

用意された部品と部品、あるいは部品とそれを埋め込む親プログラムとのインタフェースを調節するのが部品結合である。MENDELでは、部品間のインタフェースの調節を同一化(identification)によって行う。MENDELでは、入出力属性が部品間のインタフェースにあたる。同一化とは、部品間のパラメタを関連づけることである。パラメタの関連づけ及びその整合性のチェックは、辞書やフレーム表現による"知識"を参照することによって機械的に行われる。[2] この同一化のために参照する知識として、入出力属性の意味ネットによる表現を提案する。

## MENDEL /87における部品結合（自動配管）

MENDELでは、オブジェクトの内部と外部は、メッセージの通信パイプ栓であるポートを通してしか情報交換できず、各ポートは入力か出力かどちらか一

方しか許されない．各ポートは固有のポート名と、オブジェクトに出入りするメッセージの属性を持つ．属性において、その仕様を要素であるクラスに分解して表している．オブジェクト間の関係づけは、属性名によってポートとポートを通信パイプで配管することによって成される．

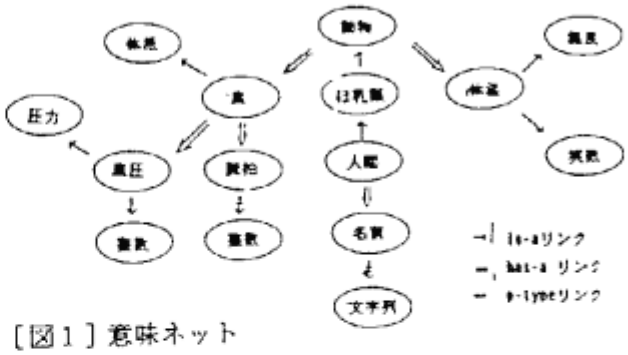
HENDELでは、オブジェクト（部品）群に対して入出力属性で表された仕様を与えると、その仕様（入出力属性）を満たすために必要なオブジェクト（部品）を推論によって検索・選択し、ポート間に通信パイプを配管（結合）する．この自動配管は、各オブジェクトが持つ出力ポートと入力ポートの属性の同一化にほかならない．そのために参照する知識として入出力属性を表現する意味ネットを用いる．

意味ネットを用いた部品結合

属性を“意味的”に組織化することにより属性の表現に秩序ができ、より柔軟性のある配管が可能になると考えられる．例えば、 <体温> という属性の取る値（物理的データ型）は、<36.5>などの <実数> であるし、 <温度> の下位概念と考えることができる． <人間の体温> という属性は、 <体温> に <人間の> という１つの限定を加えた形になっているし、 <ある（名前の）人の体温> という属性は、実は <ある人（名前）> という属性と <人の体温> という属性の組と考えることができ、組になって初めて意味のあるものである．そこで <体温><人間><実数> というもの（クラス）を節点(node)とする意味ネット(semantic network)で組織化する．

枝(link)の関係としては、とりあえずis-a（上位・下位概念の関係）とhas-a(構成要素の関係)とp-type（構成要素の物理的データ型）の３種類とする．この属性を意味ネットによって表現するために、部品結合サブシステムGARNET(Generalized Attribute Representig NETwork) を用いている．関連のある複数の属性の扱いは、構成要素に構造を導入することによって行う．属性の要素がリストの形式になっている場合も、構造を導入する場合と同様の表現を用いることにする．

上記の体温の例は、意味ネットにより次のように表現される．




［図１］意味ネット

変数を使用して属性の決定を遅らせる（結合する時点で属性を決める）ことにより、ある程度の入力と出力の間の関係が記述できるようにする．この方法により、属性の伝搬がなされる．

例えば、リストをつなぐプログラムであるオブジェクト"append"を考えてみよう．appendには、二つのリストが入力され、これらのリストをつなげたリストが出力される．これらのリストは、要素は何でもよいが、出力されるリストの属性は入力したリストによって決まってしまう．この場合は出力属性を入力属性と同じ変数を用いたリストの形で表せば、出力属性が結合する時点で決まった入力属性と同じ属性で表現される．この様に変数を用いることにより、入力属性と出力属性の関係を表し、属性を伝搬させることができる．

属性の記述形式

属性の記述形式をExtended BNFで定義する．

［定義］属性の記述形式

```
<属性名>       ::=<構造属性>{"of" <クラス名>}
<構造属性>     ::=["("]<非構造属性>{,<非構造属性>}[")"] |
                  ["["]<非構造属性>{,<非構造属性>}["]"]
<非構造属性>   ::=<物理的データ型構造>"type-of" <クラス名>
<クラス名>     ::=<クラス名>"of"<クラス名> |
                  <Prologのアトム名> | <変数>
<物理的データ型構造>
  ::=["("]<物理的データ型>{,<物理的データ型>}[")"] |
     ["["]<物理的データ型>{,<物理的データ型>}["]"]
<物理的データ型>
  ::= signal  | integer  | character |
      real    | string   | boolean   | <変数>
```

ofはhas-a 関係を、type-of はp-type関係を表す関係である．

データの構造化は、定義のように、構造化したい属性の組を括弧（）でくくることによって行う．データをリスト形式にしたいときは、括弧［］でくく

る．括弧でくくる範囲は、括弧でくくられる要素の次の上位クラスが一致するクラスとする．ただし、今の段階では、括弧は一度しか使わないものとする．

例
人間の名前と体温と人間に関する何かのデータの組
(string type-of name,real type-of body-temp,
integer type-of X)of man　　(Xは変数)
人間の名前のリスト
[string type-of name]of man

入出力属性表現の制約

入出力属性を表現する場合、属性の要素（クラス）は、次の制約を受ける．

① is-a関係に関してはsingle inheritanceとする．
② p-type関係は、is-a関係によって継承しない．
③ is-a関係と、has-a 関係にまたがる属性表現は上位概念が持つ属性を下位概念が持つことはできるし、直接持つもののis-a関係で結ばれた上位概念をも、持つことができるが、その逆はできない．

配管の方向

自動配管する場合は、出力属性にとって、配管可能で最も似ている入力属性を検索する．すなわち、出力属性側から入力属性を探す．これは、後に述べるように、属性の類似度に関して全順序を入れるためである．

配管可能な属性の抽出方法

ある出力属性にとって配管可能な属性を定義しよう．その原則は『ある出力属性にとって、意味的に自分を包含する（限定度が弱い）入力属性は、配管可能である』ということである．しかし、構造を用いている場合は、構造の型（構造の要素の数と順番、括弧の形）が完全に一致していなければならない．

例えば、①実数 type-of 体温 of 人間と②実数 type-of 体温 of 動物は、動物の下位クラスが人間であるので、出力属性①から入力属性②には配管可能だが、その逆は受け手（入力属性）のほうが限定度が強いので配管できない．

このような限定度の強さの違いを考えるために、クラス間に距離の概念を導入し、構造を用いた場合と、構造を用いない場合とに分けて定義すると、次のようになる．

［定義］クラス間の距離

クラスaとクラスbがis-a関係で結ばれた上下関係であるとき、クラスaがクラスbに比べてどれだけ上のクラスにあるかを距離｜a－b｜で表す．

クラスaとクラスbが等しい場合
｜a－b｜＝｜b－a｜＝0

is-a関係で直接結ばれるクラスの場合
（上位クラスをa、下位クラスをcとする）
｜a－c｜＝1　　｜c－a｜＝－1

is-a関係で結ばれた上下関係であるクラスaとクラスdにおいて
｜d－a｜＝k　　｜a－c｜＝1　とすると
｜d－c｜＝k＋1

便宜上、入出力属性A，B，Cを次の様に定義する

○構造を用いない場合
A（≡a0 type-of a1 ……of am）
B（≡b0 type-of b1 ……of bn）
C（≡c0 type-of c1 ……of cn）
○構造を用いた場合
A（≡（α1,α2 ……αs）of a1 …of am）
B（≡（β1,β2 ……βs）of b1 …of bn）
C（≡（γ1,γ2 ……γs）of c1 …of cn）
＜αi,βi,γi は非構造属性＞

［定義］配管可能な集合

出力属性Aに対して配管可能な入力属性の集合をU（A）と書く．

○構造を用いない場合
出力属性Aに対して、入力属性B　∈U（A）
<==> m≧n　かつ　a0＝b0 かつ
1≦∀i≦n　に対し　｜bi－ai｜≧0

○構造を用いた場合
出力属性Aに対して、入力属性B　∈U（A）
<==> m≧n　かつ
1≦∀j≦sに対しβj∈U（αj）かつ
1≦∀i≦n　に対し　｜bi－ai｜≧0

○変数を使用している場合

変数は、物理的データ型だけでなく、クラス名や構造属性や物理的データ型構造とも同一視できる．従って配管可能かどうかは、変数以外の属性の表現部で、上記の定義を適用すればよい．変数で属性を表現している部分については、ユニファイして、同一表現がなされている（各クラスが同じ）とみなす．

類似に関する全順序

配管可能な属性の集合の中で、最も似ている属性を配管の第一候補とするわけだが、その"最も似ている"属性を決めるために、類似に関する全順序を配管可能な属性の集合の中に導入した．

配管可能な属性について類似度を求めるための、類似に関する全順序は、次の定義により求められる。変数については、配管可能性の判定と同じである。

［定義］構造属性間の距離

構造を用いた場合は、括弧でくくられた構造属性部にEuclid距離を導入することにより、構造属性部全体をクラスと同様に考えることができる。

出力属性A、入力属性B　∈U（A）　のとき

$$|b0-a0| := \sqrt{\Sigma_{i=1}^{S}|\beta i-\alpha i|^2}$$

［定義］類似に関する順序

出力属性A、入力属性B、Cに対して、

B，C∈U（A）のとき、AにとってCよりBのほうが似ていることを、dA(B) ＜dA(C) と書く。

dA(B) ＜dA(C)

<--> ┌0≦ ∃i≦min（m，n）　s．t．
　　 │|bi－ai|＞|ci－ai|　かつ
　　 │0≦ ∀j＜iに対して　|bj－cj|＝0
　　 └（構造を用い、i＝0の時jは存在しない）

or ┌m＞n　かつ
　 └1≦ ∀i≦nに対して　|bi－ci|＝0

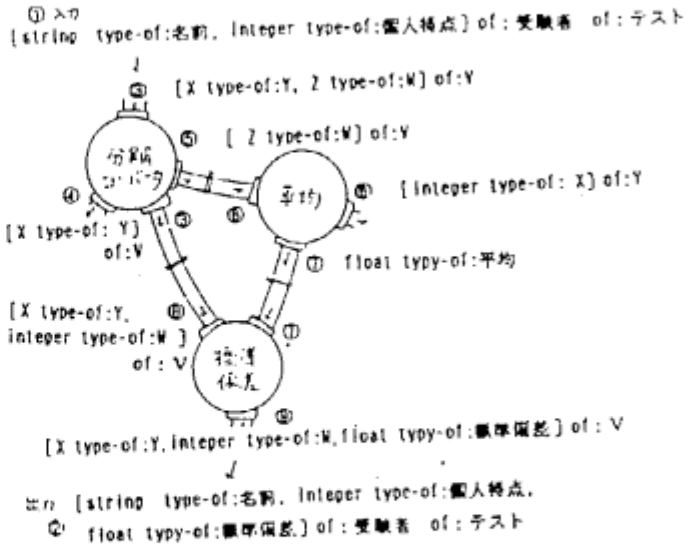
つまり、物理的データ型に近いクラスに差がある方が類似度は小さいという観点から、配管可能な属性の集合の中に、属性の類似に関する順序を入れたわけである。この順序は全順序になっている。

例 A＝(real type-of 体温,real type-of 血圧 of 血)of 人間
B＝(real type-of 温度,real type-of 圧力 of 血)of 人間
C＝(real type-of 温度,real type-of 血圧 of 血)of 動物
D＝(real type-of 体温,real type-of 圧力 of 血)of 動物
E＝(real type-of 体温,real type-of 血圧 of 血)of 動物
の順序関係は次のようになる。
dA(E)＜[dA(C)＝dA(D)]＜dA(B)．

属性名に変数を使用した場合、そのオブジェクトは選択されやすくなり、使用頻度が高いと思われるオブジェクトには有効である。しかし、変数を多用すると、検索に時間がかかり、属性の表現の秩序が乱れやすくなるので、注意しなければならない。

例題

ある試験の受験者の名前と点数のリストを入力した時、その試験の標準偏差と名前と点数を返す場合を考える。HENDELの自動配管機能によりオブジェクトを配管した例が図２である。




［図２］HENDELにおける自動配管の例

おわりに

意味ネットの導入により、配管に柔軟性ができ、秩序のある属性表現が可能となった。また、属性表現に変数を用いることにより、入力と出力のつながりが表現でき、構造の導入により、関係のある複数の属性の扱いが容易となった。

今後、理想の部品合成を達成するためには、オブジェクトの仕様をどれだけ入出力属性で表現できるかがポイントになる。本稿では、部品結合のために参照する知識として入出力属性を意味ネットで表現することにより、この点を向上させた。

現在、GARNETをＰＳＩ上で実装中である。

謝辞

本研究は、第５世代コンピュータプロジェクトの一環として行われた。研究の機会を与えて下さった新世代コンピュータ技術開発機構の方々に感謝する。

参考文献

1) N.Uchihira et.al.
Concurrent Program Synthesis with Reusable Components Using Temporal Logic Proc. IEEE1987
2) S.Hon-iden,N.Sueda et.al.
Software Prototyping with Reusable Components Journal of Informating Proc.Vol.9,No.3,1986