4B-9

知識工学の手法を用いたソフトウェア作成支援システム

吉田　敦　　田中　英彦　　元岡　達

(東京大学工学部)

1．はじめに

長期化し、ますます深刻化しつつあるソフトウェア危機の打開策の一つに、いわゆるソフトウェアCADがある。これは、プログラマが、作成しようとするソフトウェアへの要求・仕様を計算機に与え、計算機と対話しながらソフトウェアを作成していくものである。このソフトウェアCADの近年の動向として、対話型グラフィックス端末の利用や、知識工学の応用［1］［2］などがあげられる。そこで我々は、知識工学の応用分野としてソフトウェアCADを取りあげてみることにし、現在、そのためのシステムを開発中である。本稿では、このシステムの概要を述べることにする。

2．作成支援対象となるソフトウェア

本システムは、ソフトウェアCADという分野にどのように知識工学を適用していくべきかという方向づけのためのものである。従って、今日までに発表された多くのソフトウェアCADシステムのように作成を支援する対象を、ビジネス・ソフトウェアとはしなかった。そのかわり、対象として、グラフィック・サブルーチンパッケージを用いた、小規模な対話型図形処理プログラムを選んだ。その理由は、以下の通りである。

1)　対話型のプログラムや、図形処理プログラムは、仕様から性能が予測しにくく、その設計には、対話形式や経験的知識が必要となる。

2)　大規模なシステム・ソフトウェアのCADでは、マンマシンインタフェース部分など、ソフトウェアの一部を試作し、単体でテストする、プロトタイピングが重要視されつつある。図形処理は、プロトタイピングの対象の一つである。

3)　サブルーチンや基本的手続きに関する知識表現の検討は重要な課題の一つであり、サブルーチン・パッケージを用いるプログラムは丁度よい例題である。

今回、グラフィック・サブルーチンパッケージとして、GSPCが標準化のために提案したCORE［3］に準拠して、当研究室で開発したMT－CORE等をとりあげることにした。尚、ターゲット言語には、FORTRANを選ぶことにした。

3．システムの構成

3．1　システム設計の方針

ソフトウェアCADにおいては、プログラマの要求に適合する処理手続きの決定の計算機による支援が1つの主要な仕事である。また図形処理プログラムの場合、要求に適合する図形を表示するためのデータ生成も、計算機側で行なう必要がある。また、一般に、ソフトウェアを作成する場合、手続き決定が困難であるのは、ユーザ・インタフェースの部分である。ユーザ・インタフェースの手続き決定には、経験的知識を用いた、計算機による支援が必要になる。

以上のような観点に立って、我々は、出力図形のためのデータ生成部、ユーザ・インタフェース部分の手続き決定を中心に、知識工学的手法を用いることにした。次に、システムの構成について述べる。

3．2　システム構成

本システムは、以下のような構成要素から成る。(図1参照)

1) ユーザ・インタフェース
2) プログラム生成部
3) 図形データ生成部
4) 通信機構

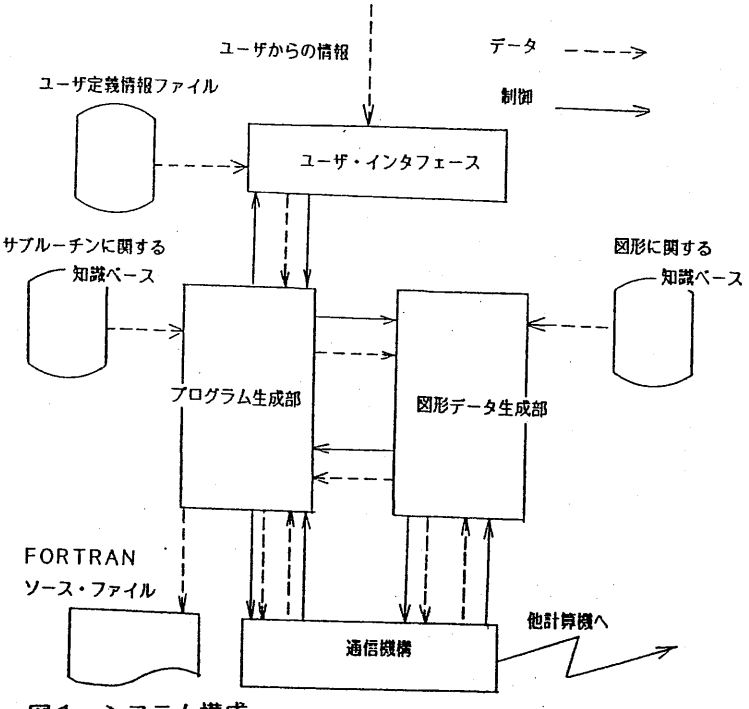

図1　システム構成

ユーザ・インタフェースは、質問・応答の形式で、プログラマから、プログラム生成に必要な情報を受け取る部分である。

プログラム生成部は、サブルーチン等に関する記述（図２）の知識ベースを持ち、これと、ユーザの与える情報をもとに目的とするプログラムを作成していく。図形表示のためのデータは、次に述べる図形データ生成部を起動して得る。

図形データ生成部は、プログラム生成部から起動され、基本図形や図形の組み合わせ方に関する知識を格納した知識ベースと、ユーザ・インタフェースからプログラム生成部を介して受け取った、図形に対する要求から、その図形の表示手順をプロダクション・ルールを用いて推論し、さらに、図形の表示に必要なデータをも生成する。

通信機構は、他の計算機の持つ情報を利用する場合に、プログラム生成部または図形データ生成部により起動される。

図形データ生成部は、ＰＲＯＬＯＧ／ＫＲ［４］で、他の部分は、ＬＩＳＰで記述する。

３．３　システムの特徴

本システムでは、さらに、以下に述べるような特徴をもつ。

1)　プログラムの作成に必要な情報のうち、特に仕様頻度の高いものは、一つのファイルとして扱えるようにし、ユーザの入力の手間が省けるようにした。（図１　ユーザ定義情報ファイル）

2)　ユーザの作成した有用なサブルーチンの記述も、プログラム生成部の知識ベースに登録できるようにした。

3)　図形データ生成部では、作成されたデータによる図形を出力し、これを対話形式で評価、修正できるようにした。

４．おわりに

本システムは、現在、試作・評価の途中である。なお、知識ベース内に納めた、サブルーチンに関する記述及び、図形に関する知識の表現形式は、必ずしも十分なものとは言えない。一般的なプログラムの作成支援システムへ拡張していく場合に不都合な点があるかも知れない。従って、今後も十分な評価と検討を行なっていくつもりであり、場合によっては、知識表現の大幅な変更もありうる。

本システムの将来の拡張方針として、次のようなものがあげられる。

1) ＣＯＲＥ以外のサブルーチンパッケージを利用するプログラムの作成も可能にする。

2) 既に作成されたプログラムを解析する機能の追加。

3) 異なるサブルーチンパッケージ間の変換機能。

4) プログラム生成部への、説明機能の追加。

5) 自然言語入力の採用によるマンマシンインタフェースの改善。

```
(crttsg (args nil)
        (func (put-c-v 'tsg-flag parmlist 1))
        (able (and (eq core-state 'init)
                   (eq vs-state '(init select))
                   (eq tsg-flag 0))))

(clstsg (args nil)
        (func (put-c-v 'tsg-flag parmlist 0))
        (able (and (eq core-state 'init)
                   (eq vs-state '(init select))
                   (eq tsg-flag 1))))

(stlnx (args ((linecolor int)))
       (func (setq line-index linecolor))
       (able (and (eq core-state 'init)
                  (eq vs-state '(init select))
                  (eq tsg-flag 1))))

(movea (args ((dist-x real) (dist-y real)))
       (func (setq current-point '(dist-x dist-y))
             (setq currntp-x dist-x)
             (setq currntp-y dist-y))
       (able (and (eq core-state 'init)
                  (eq vs-state '(init select))
                  (eq tsg-flag 1))))
```

図２　サブルーチンの記述例（一部抜粋）

```
(c declear)
(dimension x00001 3 y00001 3)
(dimension x00002 4 y00002 4)
(dimension x00003 4 y00003 4)
(dimension x00004 4 y00004 4)
(c data-def)
(data x00001 (9.0 12.5 9.0) y00001 (9.0 12.0 9.0))
(data x00002 (10.0 10.0 15.0 15.0) y00002 (9.0 5.0 5.0 9.0))
(data x00003 (13.0 13.0 14.0 14.0) y00003 (7.0 6.0 6.0 7.0))
(data x00004 (11.0 11.0 12.0 12.0) y00004 (5.0 7.0 7.0 5.0))
(c initialize)
(call initcr (bufferd synchronous dim3 explicit))
(call initvs (1 1))
(call selvs (1))
(c set-parms)
(call stwind (0.0 20.0 0.0 20.0))
(call stvup (0.0 1.0))
(call stndc (1.0 1.0))
(call stvpt (0.0 0.8 0.0 0.8))
(call crttsg nil)
(c output-pic)
(call stlnx (31))
(call movea ((9.0) (9.0)))
(call pllina (x00001 y00001 3))
(call stlnx (252))
(call movea ((15.0) (9.0)))
(call pllina (x00002 y00002 4))
(call stlnx (3))
(call movea ((14.0) (7.0)))
(call pllina (x00003 y00003 4))
(call stlnx (227))
(call movea ((12.0) (5.0)))
(call pllina (x00004 y00004 4))
(c cerminate)
(call clstsg nil)
(call dselvs (1))
(call termvs (1))
(call termcr nil)
(c eof)))
```

図３　プログラム作成の途中結果

５．参考文献

[1] Richard C Waters : The Programming Apprentice : Knowledge Based Program Editting , IEEE Trans. on SE vol SE-8 No . 1 Jan. 82

[2] D Barstow et al : An Automatic Programming System to Support an Experimental Science : Proc of 6th ICSE Sep. 82

[3] Status Report of the GSPC Computer Graphics vol 13 no . 3 Aug. 79

[4] 中島 : Prolog ／KR Manual , METR82-4, 東京大学工学部