4H-1 目的に沿った手続きの流れを考慮したインタフェースの研究

御宿哲也　永田守男
(慶応義塾大学)

1．はじめに

計算機システムの進歩には著しいものがあり、計算機のエキスパートでないひとたちが計算機を利用し始めている。そのため、人間と計算機との対話における使いやすさの問題が重要視されているが、満足のいく解答はまだ得られていない。本稿では、ヒューマン・インタフェースの向上のために、目的を達成するときに利用者が行わなければならない手続きを中心においた、目的指向のインタフェースを提案する。具体的には、コンパイラ系言語を用いたプログラム開発をケース・スタディとした、試作システムについて述べる。

2．研究方針

コンピュータを問題解決の手段として用いるとき、その達成しようとする目的には目的固有の手続きが存在し、利用者はその手続きのある流れに沿って、目的を達成しようとする2)。ところが、現在のインタフェースではコマンド言語の個々の結び付きは無機的であり、目的を反映しているとはいえない。そこで、本システムでは、利用者のその時々の目的に従った手続きの流れを意識して、手続きの中のサブ・ゴールの充足、再充足の有機的なつながりを表現できるコマンドを導入した。

それらのコマンドは、「次に進む」(next)、「前に戻る」(back)、「今のをもう一度」(again)といった簡単なもので、利用者の作業概念となっている定型的操作の中での、メンタルモデルであろうと考えられるものをコマンド化した。

しかし、それらが使われたときの利用者の置かれた状況によって意味が異なってくるので、ソフトウェアの側でそのコマンドの示す利用者の意図を推論する必要性が生じる。この推論を核にすることによって、利用者の目的に適応した知的なインタフェースを作成した。

3．推論方法

プログラム開発において、エディット、コンパイル、リンク、ランという、一連の流れが存在する。このため、これらの大きな流れを捉えれば、利用者の置かれた状況をある程度限定できる。例えば、コンパイルに失敗した状態で、利用者が「前に戻る」という意志を示せば、それはエディットに戻ることであろうと推測される。このように過去の利用状況から、流れの中での現在の位置を推測することができる。これが、推論におけるマクロ情報の利用である。

しかし、対象としているファイルが一つの場合は手続きの流れがはっきりしているが、複数の場合は一意に限定することは難しい。それは、プログラムの意味によるものと利用者がそれぞれのファイルにもたせている役割によるものとがあるため、大きな流れでは捉えきれないのである。そこで、本システムでは、ファイル間の呼び出し関係をミクロ情報として利用した。具体的には、関数型言語の入出力に着目して、関数や引数の型、引数の数の変化が流れの変化に及ぼす影響を考える。1)

4．システムの実現

対象とする言語として、関数型言語の一つであるC言語を採用した。本システムは、流れ管理部、ファイル管理部、推論部から構成されている。

流れ管理部が、今までの手続きの流れを整理して保存している。マクロ情報を用いた推論では、この二つの関係から利用者の意志と思われるものに優先順位をつける。このとき、最近に関心があるものの方に高い優先順位をつける。

ファイル管理部では、ファイルの中で定義された関数に関する情報が記述されてあり、その変更が流れを変えるのであれば、マクロ情報による推論結果を修正する。

5．システムの実行例

```
/* test.c already compiled successfully */
>edit test2.c
>cc test2            /* compile success */
>next    link test,test2<y/n>y
>back    edit test2.c <y/n>y
>next    cc test2 <y/n>y    /* compile fail */
>back    edit test2.c <y/n>y
/* change specification of function
   affecting on test.c */
>next    cc test2 <y/n>y    /* compile success */
>next    edit test.c<y/n>y
/* use micro information */
```

6．評価

手続きの表層的な流れだけを考えただけでは、推論しきれなかった状況が、ファイルの役割というミクロ情報を抽出して利用することによって可能となった。実行例の場合でいえば、ミクロ情報なしでは、最後のnextはeditではなくlinkと捉えてしまう。

本来の使いやすいインタフェースという点からは、まず定型的な操作のガイドになるということが評価される。一括して処理を行うバッチ処理的なものと比較すると、本システムの方が柔軟性という点で優れているといえる。

また、ファイルの役割を抽出することで、make的な特徴も存在する。システムが自動的に呼び出し関係を解析してくれるので、利用者の負担が軽減され、プログラムの保守という点からも有効である。

7．おわりに

今後、プログラムの意味といった情報の抽出などを考え、ミクロ情報の充実を図る予定である。

[参考文献]

1)Gail E.Kaiser,Peter H.Feller,Steven S.Popovich: Intelligence Assistance for Software Development and Maintenance,IEEE Software 1988 5
2)浜田礼、横井尚子、川越恭二：適応型ユーザインタフェース実現のための対話操作分析法とその評価、情報処理学会第３４回全国大会1987

An Interface based on Objective Oriented Procedures

Tetsuya MISHUKU and Morio NAGATA
Keio Univ.