

SIMPOSのプログラミング環境

4D-1

概要

近山 隆 内田俊一 上田尚順 中澤 修
(財) ICOT 三菱電機(株) 沖電気工業(株)

1. はじめに

SIMPOS (Sequential Inference Machine Programming and Operating System) は ICOT で開発された個人使用向逐次型推論マシン PSI のプログラミング/オペレーティング・システムである。PSI は Prolog の主要な機能 (ユニフィケーション, バックトラッキングとそれに伴う基本的なメモリ管理機能など) を機械語 KLO の機能として備える, 高級言語専用計算機である。本稿では, SIMPOS の提供する機能とその開発進行状況, および, その過程で明らかになってきた高レベルの単一言語で OS からアプリケーションまですべてを記述することの得失について報告する。

2. 開発の経緯

PSI および SIMPOS の開発プロジェクトは ICOT 研究所の発足 (1982年6月) の直後, まず高レベル機械語 KLO の設計から開始された。ハードウェアの1号機の完成が 1983 年末, 並行して開発されてきたファームウェアの完成が 1984 年春, やはり並行開発してきた SIMPOS は同年11月には FGC S' 84 国際会議の会場でデモンストレーションを行うことができるまで開発が進んだ。この版はまだセルフ・コンパイラが動かないなど実用に耐えるものではなかったが, 同年末には 0.7 版, 翌 1985 年春 1.0 版, 夏には 1.5 版, 1986 年春 2.0 版, 夏には 2.5 版と, 着実に改良を重ね, 機能面でも拡充していった。現在では PSI/SIMPOS は ICOT を中心に各所で種々のアプリケーション・プログラムの開発に用いられている。

現在の SIMPOS の規模は ESP ソースプログラムの行数にして約 20 万行, クラスの数が約 1500, プログラムの最小構成単位である述語定義の総数で約 18,000 述語である。その機能の大きさを考えると, この開発期間は従来の OS 開発に比してかなり短く済んでいると言えるであろう。これには開発に用いた言語 ESP/KLO の提供するプログラム生産効率の高さが大きく貢献していることは間違いないところである。

3. SIMPOS の特徴

PSI および SIMPOS は, 論理型プログラミング言語による良好なプログラム開発環境を提供することを目的として開発された。利用者 (アプリケーション・プログラムの開発者) から見た SIMPOS の主な特徴は以下のようである。

(1) 開かれたオペレーティング・システム

全面的に論理型プログラミング言語 ESP で記述されており, ESP の提供する継承の機構を用いて, OS の機能をアプリケーション・プログラムからまったく自由に活用することができる。

(2) 高度なマンマシン・インタフェースの提供

容易にカスタマイズできるマルチウィンドウ・システムなど, 高度なインタフェース機能を備えている。たとえば, 豊富なメニュー機能や, 編集機能を備えた (PMACS 機能付き) ウィンドウなど, SIMPOS の実現に用いている部品をユーザに開放している。文字フォントやアイコンの編集には, ビットマップ・画面とマウスを活用したフォント・エディタが用意されている。また文書作成には WYSIWYG 型の文書処理システムも提供している。

(3) プログラム開発のサポート

論理型/オブジェクト指向プログラム言語 ESP によるプログラミングを容易にするため, 強力なマクロ機能を備えたコンパイラ, プログラムの管理を容易にするためのライブラリアン, シンボリックなステップ実行ができるデバッガ, 複雑なデータ構造を解析しやすいインスペクタ, プログラムの実行時の振舞を解析するためのコール・カウンタなどの豊富なツール群を提供している。

(4) ネットワーク機能のサポート

個人使用向けのワークステーションであることの弱点を補うために, ネットワーク機能を用いて, ファイルサーバ, プリントサーバ, メールサーバ, PSI 同士や汎用機との間のファイル転送など, 資源の共用を促進する機能を提供している。

Programming Environment of SIMPOS — Overview

Takashi Chikayama (ICOT) Shun-ichi Uchida (ICOT) Takasumi Ueda (Mitsubishi Electric Co.) Osamu Nakazawa (Oki Electric Industry Co.)

4. 高級言語とOSとの整合性

従来の汎用計算機の汎用オペレーティング・システムの上に構成された高級言語システム、特に非手続的な高級言語のシステムを利用する場合、その言語の基本的な考え方や方針（たとえばメモリ領域の管理方針）はOSの基本的な方針と整合しないのが普通である。このため、OSの提供する機能をそのままの形でユーザ・プログラムに見せると、言語システム内で不整合を生ずることになる。そこで、言語システムでは必要と考えられるOSとのインタフェースをシステム内に用意するのが普通である。この方式ではあらかじめ想定された以外の方法でOSを利用することはできない。OSの機能を完全に使いこなすために、アプリケーション・プログラムとOSのインタフェース部分を他のより低級な言語（Cやアセンブラなど）で記述し、これとリンクして用いる方式もとられているが、こうした他言語とのインタフェースも考え方・方針の不一致からある程度制限せざるをえないため、特に複雑な制御を必要とする柔軟なユーザ・インタフェースを記述するのに困難が生ずる場合も多かった。

この点SIMPOSでは最低レベルのプロセス、メモリ、入出力機器の管理から、最高レベルのユーザ・インタフェース部分、さらにはアプリケーション・プログラムに至るまで、すべて単一の言語E S Pで記述する方針をとったので、このような不都合が生じることはない。さらに、E S Pのもつ継承(inheritance)の機構を用いることにより、従来のサブルーチン的なOSの利用方法だけではなく、OS機能を包含し、必要に応じて機能の拡張や修正を加えたユーザ定義のオブジェクトを定義して用いることも可能に、開かれたOSを実現している。

5. 高レベル言語専用システムの利害得失

前述のようにOSからアプリケーションまですべてをひとつの言語で記述することには有利な点が多いのであるが、その際にどの程度のレベルの言語を選ぶかが問題になる。

E S Pのような高いレベルの言語を用いる利点は、高レベルのアプリケーションの書きやすさにあることは言うまでもない。

一方、そのマイナス面は、そこまで高いレベルの機能を必要としない部分（たとえばOSの低レベル部分の記述）にもそうした高いレベルの言語を用いることによる実行効率上の不利益である。これを補うためには記述したい内容のレベルに応じて使用する言語を変えるのがひとつの手段で、これが複数の言語を併用する場合の主要な動機のひとつになっている。しかし、効率が要求される部分には、別の意味で高いレベルの言語を用いる利点がある。これについては次項に述べる。

6. 実行時に効率のよい言語とは...

PSI/SIMPOSの開発の当初疑問視されていたことのひとつに、近年改善が著しいとはいえ、必ずしも高速とは言えない論理型プログラム言語に、これまた実行速度面で有利ではないオブジェクト指向機能までも取り入れた言語E S Pで全体を記述して、実用に耐えうる効率のOSにできるかどうかという点があった。現在のPSIの40 KLIPSという速度はPrologの処理系としては比較的高速な部類に入るが、ごく小さなループの実行にも1回最低25 μ 秒かかるということであり、同程度の規模の汎用プロセッサに較べてひと桁程度低速なのである。現在開発中のPSI-IIでは3倍程度の速度向上が期待されているが、それでも論理型プログラム言語の高い機能を生かせる問題以外では決して高速であるとは言えない。

実際、SIMPOS開発の当初の版は、1文字を画面に表示するのに数秒かかるなど、かなり低速であった。しかし、その後、効率上のネックとなっていた部分に対して使用するアルゴリズムやデータ構造などを大幅に改良し、対象となるソフトウェアの規模の大きさ（数万行）にもかかわらず、ごく短期間（約3ヶ月）の内にふた桁程度（うち3倍程度はファームウェアのサポートによる）の速度向上を得、実用レベルの実行効率を実現することができた。

このような大幅な改修が短期間の内に可能であったのは、記述言語E S Pの論理型/オブジェクト指向機能による、プログラムのモジュールリティの高さとモジュール間インタフェースの簡潔さ、デバッグ作業の容易さに負うところが大きい。さらにファームウェアのサポートを得て動的な述語の実行頻度の計測も容易になり、効率上のネックを見付けること自体も容易になっている。

実行速度を支配する要因は、プリミティブなレベルでの実行速度よりも、むしろ高レベルのアルゴリズムやデータ構造の設計にある、ということは当然のことにすぎない。しかし、プログラムの仕様変更は不可避であること、したがって最初から最適なプログラムを設計・実装することが事実上不可能であることを考えると、プログラムの書き直しが容易であるかどうか最終的なプログラムの効率を大きく左右することは重要な事実である。このことを考えると、こうした改修が容易に可能な言語は、たとえ一見実行効率が低くとも、結局は高い効率を提供しているのだとも言えそうである。

7. おわりに

SIMPOSは現在も本年末に完成予定の第3版に向けて開発を続行中である。第3版では、アプリケーション・ユーザの要望もあって、これまでともすれば不統一になりがちだった各サブシステムのユーザ・インタフェースの統一、改修を重ねて複雑化したシステムの整理、などを主たる目標に置いている。