Common ESP 上の LTB について*

5M−2

藤瀬 哲朗 西山 聡 歌代 豊
(株) AI 言語研究所 第3研究室

1 はじめに

AI 言語研究所では，逐次型マシン上で動作するオブジェクト指向型と論理型の機能を併せもつプログラミング言語 Common ESP (以下 CESP と略す) を開発中である ([1]). 第3研究室では CESP 言語仕様および言語処理系を実証評価することを目的に，各種応用システムを試作中である.

現在，我々は応用システムとして，ICOT (新世代コンピュータ技術開発機構) が逐次型推論マシン PSI 上の ESP/SIMPOS で開発した汎用日本語処理系 (Language Tool Box, 以下 LTB と略す [2]) を選択し，CESP/Unix 上に実現中である.

本稿では，この CESP/Unix 上の LTB (以下 CESP-LTB と略す) の実現方式とともに

- PSI 上のプログラムの CESP/Unix へ移植
- ESP/SIMPOS 上のツールの CESP/Unix への拡張

により CESP の基本機能，拡張機能を実証評価した中間結果について述べる. この結果は，数多く実現された PSI 上のソフトウェアを汎用ワークステーションへ移植する際の指針となり得る.

2 CESP-LTB の実現方式

LTB は，ICOT に蓄積された自然言語処理技術の中で基本的な部分をツール群化したものである. 種々の応用分野において必要とする自然言語 I/F を容易に実現するための開発環境を備えている. LTB に実現された機能は次の通りである.

- 日本語についての種々の情報が計算機処理できる辞書の枠組み
- 日本語の文字列を有意味な単語等の並びに変換する形態素処理系 (LAX)
- この並びから文の構造を得る構文解析系 (SAX)
- 日本語の文字列を生成する生成系
- 辞書や種々の処理系が必要とする日本語の意味を記述するための種々の基本機能を提供する意味記述言語 (CIL)

*Implementation of LTB on Common ESP
Tetsuro Fujise, Satoshi Nishiyama, Yutaka Utashiro
AI Language Research Institute, Ltd.

2.1 CESP-LTB の機能

PSI-LTB 処理系は，シングル・ユーザ向けの大容量メモリワークステーションである逐次型推論マシン PSI 上で動作することを前提として実現されている. CESP-LTB 処理系は汎用マシン (種々の計算機の意味) 上の CESP により実現される汎用日本語処理系である. PSI-LTB 処理系との機能の違いを中心に以下にまとめる.

- PSI-LTB が開発環境を中心に構成されているのに対して，日本語文処理 I/F 機能を中心に据え，PSI-LTB により開発された辞書等を Unix 上のツールとして利用できることを目指す.
- 基本機能として，CESP を利用する. CIL 処理系を CESP 上にデバッグ環境を含めて実現することは現在のところ難しい. それよりは CESP を CIL の仕様を包含する方向で処理系共々強化することに頼る方がより合理的である. CIL の部分項は CESP 提供ライブラリを，遅延評価は CESP の組込述語 bind_hook とマクロ機能を利用することになる.
- 開発環境は，LAX のみ実現する. 研究よりは日本語文処理向けの利用に重点を置いた結果である. ただし辞書記述を LTB データに変換する機能は SAX も実現する.
- 文字端末での利用を念頭に置いたため，グラフィック機能は削除した.
- 生成系は実現しない. 本機能が CIL により実現されているため今回は断念した.
- C 言語 I/F を利用して，SIMPOS 機能等を実現する.

CIL-ESP トランスレータや X-window I/F，また SAX による構文解析サーバについては今後の実現項目とした. 拡張機能としてさらに次の機能も実現する.

- 形態素解析機能を各種プログラムに提供するための形態素解析サーバを実現する. LAX を利用しるので，開発環境を利用して各サイト独自の辞書を開発することができる.

2.2 実現方式

CESP-LTB を，基本的には PSI-LTB から生成することで，CESP を実証評価する. 実現方針は次の通りである.

- PSI-LTB のソースプログラムをできる限り修正せずに移植する. CESP はオブジェクト指向およびマク

ロ機能, および C 言語 I/F を基本機能にもつ. 具体的には次の方針で実現する.

- 組込述語, データ型の違いはマクロを用いる.
- ツールクラスにより継承された SIMPOS クラスは, オブジェクト指向の枠組みで実現する.
- I/O 等のクラスは CESP の C 言語 I/F を用いる.

- CESP 処理系上の LAX エンジンを CESP のシステムコーディネータ ([3]) により形態素解析サーバを実現する. そして形態素解析サーバの Unix ツール化を試みる.

3 CESP-LTB 実現の障害とその解決法

現在, 来年 9 月の完成を目指し, CESP-LTB は実現途上である. CESP-LTB を上述の方式で実現する際の問題点および解決方法は以下の通りであった.

- 基本データの違い
  - 文字コードの相違

    問題点 PSI の文字コードは 16-bit JIS コードだが, CESP は EUC コードである.

    解決法 プログラム中で文字コードを数で直接記述している箇所は修正が必要であった. ESP ではこれらは文字マクロで表現することとになっているので, 問題が起る場合は ESP プログラムのバグと考えてよい.
  - アトムの相違

    問題点 CESP は, アトムを実現するデータ型として 16-bit と 8-bit の文字列型が用意されている. この結果表示上同じ字面の 2 種類のアトムが存在する.

    解決法 CESP プログラム中のアトムの文字列表現はコンパイラのパーザが 8-bit 文字列と決定している. プログラム中でアトムを生成するユーティリティを新たに実現し, CESP のパッケージにより重複した名前のクラスを定義することで解決する.
  - 32-bit 文字列型の取扱い

    問題点 PSI にあり, CESP に存在しない型である.

    解決法 32-bit 文字列の代わりに impure vector を用いた. マクロにより文字列処理用組込述語をマクロで置き換えることで解決する. もっとも文字列と impure vector を混在したプログラムは全面的に書き換える必要がある.
  - 整数の表現範囲の相違

    問題点 PSI は 32-bit で, CESP は 29-bit で整数を表現している. bit 操作する処理で問題になる.

    解決法 整数をハッシュのための bit table として利用しているため, 最も大きな障害であった. しかし PSI-LTB のプログラム自体が利用 bit 数をパラメータ化していたため, 若干の変更で済んだ. プログラムの性質がよかったためであり, 問題点となろう.

- SIMPOS 環境と Unix 環境の相違
  - I/O に関する相違

    問題点 マルチ・ウインドウを利用したプログラムを文字端末利用方式にどの程度変更できるかなど, またファイル等の違いが問題となる.

    解決法 SIMPOS ではこの辺はオブジェクト指向を意識した構成になっているのでクラスの最下層を Emacs I/F と C 言語 I/F を利用して実現することで部分的に解決できた. グラフィック I/F は, 将来的に X-window 上で実現することも可能であろう.
  - マルチプロセス環境とシングルプロセス環境の相違

    問題点 PSI 上のプログラムは, マルチプロセスのプログラムを実現できるが, CESP 基本版上のプログラムはシングルプロセスのプログラムである.

    解決法 現在の CESP 基本仕様版では非同期処理は断念する他ない. 次版の CESP では部分的に可能になるはずである.

4 おわりに

LTB のように巨大なプログラムを CESP 上に実現することによって CESP の核となる機能のひとつひとつが実証評価されつつある. また, このような規模なプログラムをほぼ書き換えることなく CESP 上で実現できることは ESP の仕様が汎用性を含んでいることを示している.

謝辞

本研究の機会を与えて下さった実近 AI 言語研究所所長, 山本第 3 研究室長に感謝致します. また LTB に関して様々な相談に応じて頂いた 田中 ICOT 第 6 研究室長をはじめ LTB 開発チームの皆様に感謝致します.

参考文献

[1] 中澤他 : ESPer への道, bit Vol.22, No.1-12, 共立出版 (1990).

[2] 杉村他 : 汎用日本語処理系 *LTB* の構成, 情報処理学会第 *37* 回全国大会予稿集, *pp.1072-1073.*

[3] 鳥居他 : *Common ESP* におけるシステムコーディネート方式, 情報処理学会第 *40* 回全国大会予稿集, *pp.714-715.*