LISPマシン LIMEのソフト開発(I)

2P-4 —LISP記述LISPシステム—

越前 孝* 冨田一則* 越前章子* 南田英輝* 横田 実**
*神戸日本電気ソフトウェア株式会社 **日本電気株式会社

1. はじめに

試作機として研究開発されたLISPﾏｼﾝ LIME (LIsp Machine Engine) [1] のｿﾌﾄｳｪｱ (LISP処理系) を開発し、実用的規模の応用ﾌﾟﾛｸﾞﾗﾑ に対しても充分適用可能な処理系が作成されたのでここに報告する。LIMEは、通産省第五世代計算機ﾌﾟﾛｼﾞｪｸﾄの成果[2] を利用して試作されたもので、実用的なﾂｰﾙ を早期に提供することを主眼として開発されたものである。また、ﾜｰｸｽﾃｰｼｮﾝEWS4800をﾎｽﾄﾌﾟﾛｾｯｻとし、全二重ﾁｬﾈﾙで結合されﾊﾞｯｸｴﾝﾄﾞﾏｼﾝ として利用される。これにより、ﾎｽﾄOS であるUNIX上で開発されたｿﾌﾄｳｪｱ財産を充分に活用でき、さらに大容量のﾒﾓﾘ 空間 (最大 64M語) を使用することが可能となった。しかし、良好なﾌﾟﾛｸﾞﾗﾐﾝｸﾞ 環境を提供するためには、LIME側にもｲﾝﾀﾌﾟﾘﾀ、ﾃﾞﾊﾞｯｶﾞ、ｺﾝﾊﾟｲﾗ などを搭載する必要があり、これらのﾌﾟﾛｸﾞﾗﾐﾝｸﾞｼｽﾃﾑ およびその実行管理を行うためにLIME上にｶｰﾈﾙﾚﾍﾞﾙ のOSを搭載した。本稿では、これらｿﾌﾄｳｪｱ (LISP処理系) の開発方針、開発手法、およびｲﾝﾀﾌﾟﾘﾀの構成とその評価について述べる。

2. 開発方針

(1) LISP言語仕様

以下のような理由によりLISP言語仕様として、東京大学工学部和田研究室で作成されたUTILISP[3]を採用し、既にﾎｽﾄ 系ｺﾝﾋﾟｭｰﾀで開発されているUTILISP 処理系と完全互換を保つようにした。

①比較的言語仕様が小さく、処理系が作りやすいため早期に実現可能である。
②当社では、その速度性能が高いことから、ｼｽﾃﾑ記述言語としてLISP応用ﾌﾟﾛｸﾞﾗﾑ (たとえば、ｴｷｽﾊﾟｰﾄｼｽﾃﾑ構築支援ﾂｰﾙ EXCORE等) の開発において、既に他方面で使用されている。
③我が国では各ﾒｰｶｰが標準的にｻﾎﾟｰﾄしている。

ただし、LIME上のｵﾍﾟﾚｰﾃｨﾝｸﾞｼｽﾃﾑもLISPで記述できるようにしたため、多重ﾌﾟﾛｾｽの導入、ﾊｰﾄﾞｳｪｱ資源へのｱｸｾｽおよびﾒﾀ機能の実現のための組込関数を導入した。これにより、ｼｽﾃﾑ記述用のLISP言語としては、現UTILISP と上位互換を保ったLISP言語仕様となった。

(2) LISP記述LISPｼｽﾃﾑ

R.Brooksら[4] が指摘しているようにLISPｼｽﾃﾑをLISPで記述することには多くのﾒﾘｯﾄがある。そのうち可搬性に関しては、本開発ではさほど重要としなかったが、正しい処理系を容易に作成できるﾒﾘｯﾄは非常に大きい。そこで、ｺﾝﾊﾟｲﾗ 、ｲﾝﾀﾌﾟﾘﾀ、OSのほとんど全てをLISPで記述することにした。ただし、上記でも述べた一部OSが使用する組込関数はﾌｧｰﾑｳｪｱ 化して呼び、ｽﾀｯｸｱｸｾｽが中心となり高速性が要求されるｲﾝﾀﾌﾟﾘﾀの使用する関数、および不定個引数を持つ組込関数は、ｺﾝﾊﾟｲﾗ の中間言語としても使用されるLIMEｱｾﾝﾌﾞﾗ (機械語) で記述することにした。

(3) ｺﾝﾊﾟｲﾗ 主体ｼｽﾃﾑ

LISPは本来ｲﾝﾀﾌﾟﾘﾀとして考案された言語であるが、昨今そのｺﾝﾊﾟｲﾙ 手法がかなり確立されてきており、処理速度を上げるために、ほとんどのLISP処理系ではｺﾝﾊﾟｲﾗ を装備している。ﾕｰｻﾞ はﾌﾟﾛｸﾞﾗﾑ を最終的にはｺﾝﾊﾟｲﾙ して実行するのが普通であり、また、その処理系の性能もｺﾝﾊﾟｲﾙ 後のﾌﾟﾛｸﾞﾗﾑ の実行性能を挙げて実際の性能とすることが一般的である。

さらに、ｲﾝﾀﾌﾟﾘﾀをﾏｲｸﾛｺｰﾄﾞ 化する方式よりも、ｲﾝﾀﾌﾟﾘﾀをLISPで記述し、ｺﾝﾊﾟｲﾗ の生成する機械語の機能設計とそれを最適に生成するｺﾝﾊﾟｲﾗ の最適化を活用する方式の方が、開発も容易であり性能もそれ程悪くはならない。そして、処理系開発完了と同時にﾕｰｻﾞ にｺﾝﾊﾟｲﾗ を提供できるﾒﾘｯﾄがある。

これらのことから、開発時にもまたﾕｰｻﾞ 使用時にもｺﾝﾊﾟｲﾗ を主体としたｼｽﾃﾑとし、ｲﾝﾀﾌﾟﾘﾀはﾘｽﾅ およびﾃﾞﾊﾞｯｶﾞとしての機能を果たすだけのものとして捉えることとした。

3. 開発手法

各LISPｼｽﾃﾑｺﾝﾎﾟｰﾈﾝﾄ はｸﾛｽ 系で開発した。ﾎｽﾄﾌﾟﾛｾｯｻであるEWS には、同仕様のUTILISP が存在するために、ｸﾛｽ 系の開発ﾂｰﾙ をEWS UTILISP で記述することが可能となり、さらに、LISPのもつ開発環境を活用することで、開発効率を高めるとともに、各ｺﾝﾎﾟｰﾈﾝﾄ の信頼性を得ることに成功した。

LIME UTILISPではｺﾝﾊﾟｲﾙﾄﾞｺｰﾄﾞ の実行単位を個々の関数ではなく、それらの集まりであるﾓｼﾞｭｰﾙ とした。これは実行性能を向上させるために、ﾛｰｶﾙ関数 ([5] 参照。ｼﾝﾎﾞﾙ経由の参照ではなく、直接的なｱﾄﾞﾚｽ参照となる関数) を導入したためで、そのﾛｰｶﾙ関数の参照の有効範囲となる複数の関数の集まりをﾓｼﾞｭｰﾙ とした。

ｸﾛｽ 系開発ﾂｰﾙ として、以下のものを用意した。

①ｸﾛｽｺﾝﾊﾟｲﾗ ｿｰｽﾌｧｲﾙ 中の個々の関数をｺﾝﾊﾟｲﾙ し、それらを一つﾓｼﾞｭｰﾙ に構成して、fixnumをその主要な要素とするﾘｽﾄﾃﾞｰﾀとして出力する。またｺﾝﾊﾟｲﾗ は中間言語としてLAP へ引き渡すﾆｰﾓﾆｯｸﾚﾍﾞﾙ の機械語ﾘｽﾄ (LIMEｱｾﾝﾌﾞﾗ) も副次的に出力する。このﾘｽﾄﾃﾞｰﾀはｼﾐｭﾚｰﾀでの実行可能であり、ｼﾐｭﾚｰﾀもEWS UTILISP で記述されており、強力なﾃﾞﾊﾞｯｸﾞﾂｰﾙ となった。
②ｸﾛｽﾊﾞｲﾝﾀﾞ 複数ﾌｧｲﾙ間のﾛｰｶﾙ関数の参照を可能にするため、ｸﾛｽｺﾝﾊﾟｲﾗの出力ﾘｽﾄ を複数ﾌｧｲﾙから受け取り、それらを、複数ﾌｧｲﾙ間のﾛｰｶﾙ関数の参照を可能とする一つのﾓｼﾞｭｰﾙ へと再統合する。
③ｸﾛｽﾘﾝｶ ﾛｰｶﾙ関数間の参照を解決する。
④ｸﾛｽﾛｰﾀﾞ 実行可能なﾛｰﾄﾞﾓｼﾞｭｰﾙを組み立てる。

完成したﾛｰﾄﾞﾓｼﾞｭｰﾙは、LIME本体へﾀﾞｳﾝﾛｰﾄﾞされる。

以下にその処理の流れを示す。

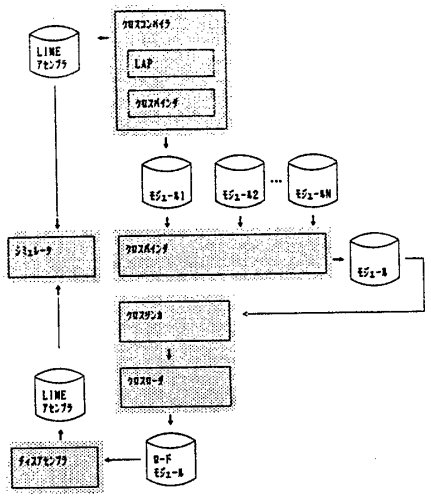

図1 LISPｼｽﾃﾑ開発手法

Software Development on a LISP machine LIME (1)
-LISP System written in LISP-
Takashi KOSHIMAE*, Kazunori TOMITA*, Ayako KOSHIMAE*,
Hideki MINAMIDA*, Minoru YOKOTA**
*NEC Software Kobe. Ltd. **NEC Corporation

4．インタプリタの特徴と構成およびその評価

(1) LIME UTILISPの名前空間

LISPプログラム の実行方式は、他のLISP処理系と基本的に同一であり、スタックとシンボルを用いて制御される。ただし、多重プロセス機能を導入するために、システム全体の名前空間を、ローカル空間とグローバル 空間に分けた。インタプリタは一つのユーザプロセス であり、原則として個々の名前空間を所有する。従って二つのユーザプロセス 内では名前の衝突は生じない。一方、システムでのみ使用する関数の名前やシステム制御変数のような、全プロセスで共有すべき名前はグローバル 空間にある。このような名前を共有シンボルと呼び、全プロセスから接頭語"SYS:"を付けることにより参照可能とした（図２）。ただし、ローカル性を保証するためローカル空間に置かれた構造体データ を共有シンボルに代入することは禁止される。本方式により、ユーザ 間の独立性を保証し、かつシステムの共有化が実現された。

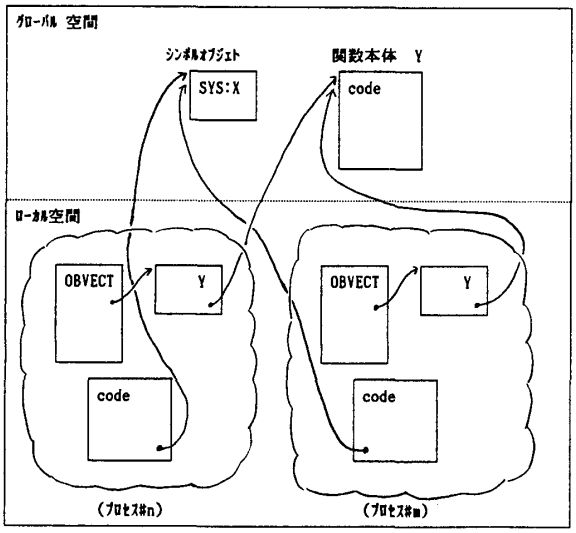

図２ LIMEの名前空間

(2) インタプリタの構成

インタプリタは、リーダ 、プリンタ、エバリュエータ 、ローダ 、エラーシステム （デバッガ）、組込関数で構成される。LIMEシステム内では図３のような位置にある。特に、組込関数のうちで入出力、GC等、OSおよびファームウェア と密接に関連する部分は、アウタOS 機能と称し、リスト 処理系の組込関数とは別分類とし、作成者のノウハウ分散を極力少なくした。

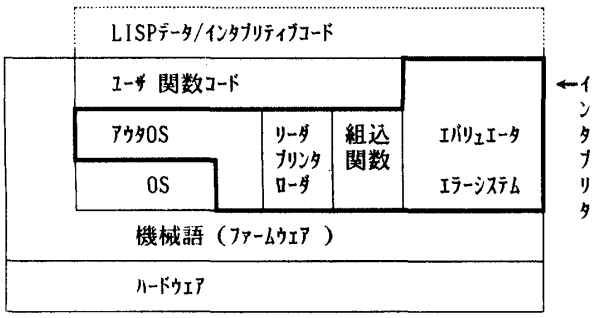

図３ インタプリタの構成

(3) インタプリタの評価

第三回LISPコンテスト [6] 関数をインタプリタで実行した結果を以下に示す。いずれもGC時間を含んでいない。

表１ LISPコンテスト 実行結果 （単位：ms）

インタプリティブコード	LIME UTILISP
TARAI-4	1855
TAK-18-12-6	10422
LIST-TARAI-4	3202
SREV-5	103
QSORT-50	108
NREV-30	114
STRING-TARAI-4	3037
FLONUM-TARAI-4	2098
SEQ-100	70

ただし、LIMEのクロックは200ns で測定した。

以上のように、他のLISP処理系[7] と比較しても、インタプリタとしてユーザプログラム の開発時等に使用するには充分な性能が得られ、当初の目標は達せられた。

5．おわりに

LISP言語仕様をUTILISP にしたことから、比較的少ない工数で、品質、性能の両面からも満足のいく処理系が製作された。また、LIME上のUTILISP はユーザインタフェースを含めて、ホストプロセッサEWS4800 上のUTILISP と完全互換を保っており、EWS4800 上で開発された応用プログラム はそのままLIME上で実行可能である。そのために全二重チャネルを経由して、ホスト 上のC 言語等の他言語連絡機能も備えている。

現在、このように実用システムとなった LIME UTILISP システムに対して、実用的規模のエキスパートシステムによる評価を進めている。

参考文献

[1] 横田，越前他: LISPマシン LIME，情処学会計算機アーキテクチャ 研究会資料，1989.
[2] 幅田 他: 逐次型推論マシンCHI小型化版のハードウェア，情処学会第33回全国大会，5B-8，1986.
[3] Chikayama, T.: UTILISP MANUAL, METR 81-6, Univ of Tokyo, 1981.
[4] Brooks, R. et al: Lisp-in-Lisp: High Performance and Portability, Proc. 8th Int.Jt.Conf.Artif. Intell.1983, Vol.2, pp.845-849, 1983.
[5] 越前 他: LISPマシン LIMEのソフト 開発（Ⅱ）-コンパイル技法-， 情処学会第38会全国大会，1989.
[6] 奥乃: 第三回Lispコンテスト 及び第一回Prologコンテスト の課題案，情処学会，記号処理28-4，1984.
[7] 奥乃: 第三回LISPコンテスト と第一回Prologコンテスト 報告，情処学会，記号処理33-4，1985.