

2C-6

多言語モジュラープログラミングとその処理系
LIGER の概念設計

中所武司 (日立製作所システム開発研究所)

Ⅰ. はじめに

ソフトウェアの信頼性, 生産性, 保守性の向上のために, 古くからプログラムのモジュール化, 記述言語の高級化などが行われてきた。プログラム開発の構成要素を対象, 工程, 道具と見た場合, モジュール化技法は工程の定式化と結果として生じる対象の標準化, 高級言語は道具の高級化を狙うものと考えることが出来る。

一方, 近年のデータ抽象化, プログラムの構造化, 複合設計などのプログラミング方法論も結果としてはモジュール化を促すものであり, それに関連した新しい言語も提案されている。

このようなモジュール化技法を徹底するためには,

- (1) モジュール分割の方法,
- (2) モジュール間関係とモジュール自身の記述言語,
- (3) 効率の良いオブジェクトを生成する処理系,

などが必要である。本論文では, 性能中心のモジュール分割を徹底した場合に適する言語系とその処理系について述べる。

Ⅱ. 多言語モジュラープログラミング

機能的な単位でモジュールが構成されるプログラムの記述言語は,

- (1) モジュール間関係をインターフェイスも含めて正確に記述できること,
- (2) 各モジュールの機能に合った記述が可能であること,

が重要である。前者に関しては, Modula, Alphard などこれまで提案されている言語の多くは, リソースに属するモジュール間の使用関係とモジュール自身の記述機能を一つの言語に含めているが, 言語仕様が複雑化する一方で大規模ソフトウェアの開発には両機能共に不十分になっている。やはり, (1) の為の専用言語が必要である。

(2) に関しては, 2つの全く逆の方向がある。一つは, PL/I, Fortran 77, Ada などのように必要な機能を積極的に採入れて万能言語にしていく方法であるが, このような言語のマンモス化は, 言語の修得と活用の困難, 処理系の複雑化と巨大化, 初歩的なオブジェクト生成の困難, などの欠点と生じ, 必ずしも良い方法とは言えない。むしろ, これとは逆にそれぞれのモジュールの機能に適した専用言語を幾つか用意すべきである。

このように, モジュール間関係およびモジュール自身をそれぞれに適した何々の言語で記述する方法をここでは多言語モジュラープログラ

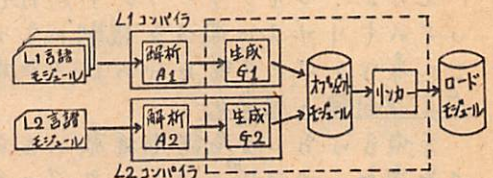


図1 従来の処理系

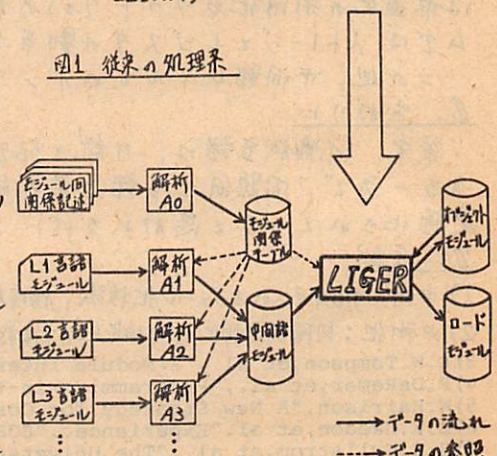


図2 MIPの処理系

ミング (MMP: Multilingual Modular Programming) と呼ぶ。

3. 処理系

MMPプログラムの図1のような従来方式で処理した場合、モジュール分割の徹底によって生じるリンカー・オーバーヘッド、冗長性、一時的エラーの増加などのオブジェクト効率の低下が問題となる他、異なる言語のモジュール間の結合が不可能な場合もある。そこで、MMPの処理系としては図2のように従来の言語処理系の生成部とリンカーの機能を併せ持つシステムが必要となる。ここではこれをLIGER^(注) (link-and-generator) と呼ぶ。即ち、各言語の処理系がソースの解析処理(字句、構文、意味解析)を行い、中間語モジュールやテーブルを出力した後、LIGERは

- (1) 各モジュール単位の最適化、
- (2) 引用箇所のないモジュールや短いモジュールのインライン展開、
- (3) 異なる言語のモジュール間での異なるパラメータ機構や異なるデータ型に関する処理コードの付加、

などを行ってオブジェクトの生成および通常のリンカーの処理を行う。

4. 本システムの利点

モジュラープログラミングによる利点の他に、MMPによるものとして、

- (1) モジュール間インターフェイスの検証による大規模ソフトウェアの信頼性向上、
- (2) 各モジュールをその機能に適した言語で効率良く記述でき、理解も容易、

などがある。また、LIGERによる利点としては次のようなものがある。

- (3) 異なる言語のモジュール間リンキーが容易。
- (4) インライン展開など、モジュール間の最適化によるオブジェクト効率の向上。

5. 実用化技法

本システム実現のため、現在、次のような項目について研究を進めている。

5.1 モジュール間関係記述言語

モジュールインターフェイス記述言語としてBridgeシステム³⁾やMIL⁴⁾があるが、いずれもリソースに関する規制が主である。本システムでは、その他に異なる言語間での異なるデータ型やパラメータ機構に関する記述機能も導入する予定である。

5.2 中間語の標準化

本項目の旨とは言語処理系の生成部を言語間で共通にするためのUNCOL⁵⁾、最近では最適化の汎用化⁶⁾やソフトウェアの移植性⁷⁾を目的として研究されている⁷⁾。本システムでは、ストレージレジスタの割当て直前のレベルでの標準化を予定している。²⁾ この他、中間語レベルでのインライン展開は既に他の言語で実施している。

6. おわりに

従来、計算機言語は、単純な記述言語は次々と機能拡張が行われ、マンモス化するという方向で、問題のある言語は柔軟性が無く、適用範囲が限定されるという2方向分極化されていると思われるが、ここでは次の方向を探ってみた。

7. 文献

- 1) 中沢: プログラムのモジュール化技法, 情報会誌, Vol. 62, No. 1, pp. 91~93 (1979).
- 2) 中沢他: 段階的詳細化, データ抽象化を機構言語SPLに適用した技法, 情報処理論文誌, Vol. 21, No. 3 (1980年).
- 3) E.W. Thompson, et al., "A Module Interface..", Proc. 12th Design Auto. Conf., 42-49, 1975.
- 4) F. DeRemer, et al., "Programming-in-the-large Versus...", IEEE TSE, SE-2, 4, 80-86, 1976.
- 5) W. Harrison, "A New Strategy for Cord...", Proc. 4th ACM Symp. POPL, 29-37, Jan. 1977.
- 6) B.K. Haddon, et al., "Experience...", SOFTWARE-PRACTICE AND EXPERIENCE, 8, 5, 601-616 1978.
- 7) H.C. Gyllstrom, et al., "The Universal Compiling...", ACM SIGPLAN, 14, 12, 64-70, 1979.

(注) ligerとはlion(♂)とtiger(♀)の交配種。