Lispにおける最適システム生成方式の検討

5Q-3

杉山 広幸
NTT ヒューマンインタフェース研究所

1．はじめに

Lispの適応範囲はエキスパートシステムの普及により、プログラム開発のためのワークステーションでの利用形態だけではなく、デリバリーマシンとしての特定AP専用端末、さらには機器組み込み型での利用形態が求められつつある。デリバリーマシンでは、利用できるリソースが限られているため、Lispシステムの必要な部分とアプリケーション・プログラムのみで構成される、特定AP向きのコンパクトなLispシステムを容易に生成できることが望まれる。現在われわれが開発している新ELIS[1](ELIS-8200シリーズと呼ぶ)用の、デリバリーマシン向きシステム生成方式の検討を行ったので報告する。

デリバリーマシン向きのシステム生成においては、システムに組み込むべき言語機能の抽出と、抽出した言語機能とアプリケーションを組み込んだオブジェクト・ファイルの生成、それを用いた起動の各方式を明かにしなければならない。本稿では、Lispの標準言語として広まっているCommonLispを対象に、そのデータ型に基づく言語機能の抽出方式を提案する。

2．データ型に基づく言語機能抽出方式

従来のLispにおいては、言語機能の抽出を行おうとした場合、関数単位に行うのが自然であった。しかし、CommonLispでは、共通の性質を持つデータ型を一つのデータ型として取り扱うジェネリックなデータ型と、その型を扱う関数が導入され、関数単位の言語機能の抽出だけでは十分であるといえなくなってきた。たとえば、シーケンス型は、リスト型、文字列型のスーパー型であり、シーケンス型を扱うconcatenate関数はリスト、文字列のどちらも扱うことができる。このため、関数単位で機能抽出を行った場合、リストの結合に用いたconcatenate関数のために、文字列を結合する機能も一緒に抽出してしまい問題である。

データ型に基づくモジュール分割、選択を行い、さらに関数単位での機能抽出を行うことにより、上記の問題点を解決できるものと考える。機能抽出の流れを図1に示す。

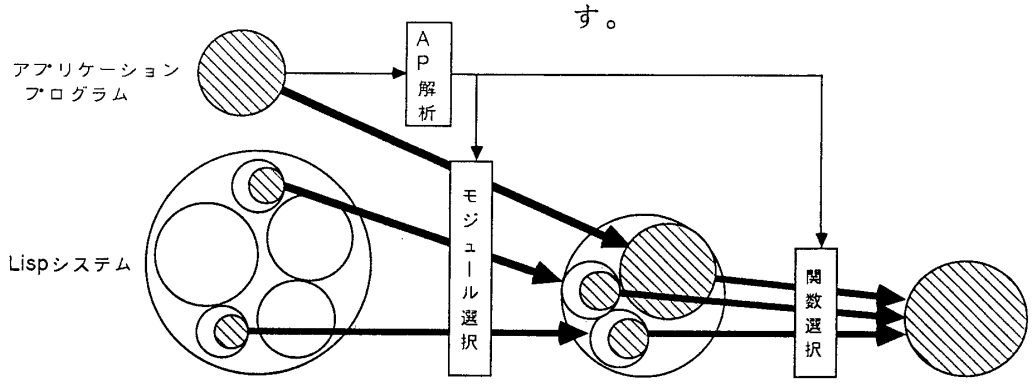
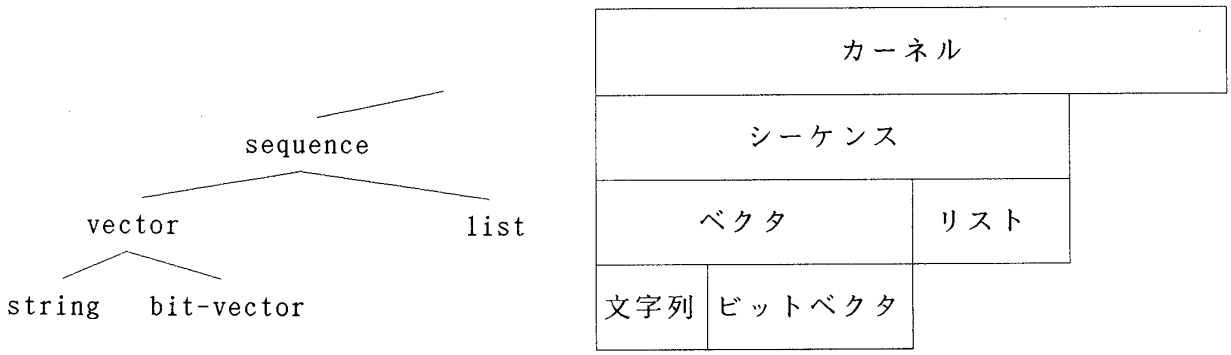


図1 機能抽出の流れ

A Study on Genaration Method for Lisp System
Hiroyuki SUGIYAMA
NTT Human Interface Laboratories



CommonLispの型階層(一部)　　モジュール構成(一部)

図2 型階層とモジュール構成の対応

3．データ型に基づくモジュール分割

CommonLispの各型階層において、そのデータ型を操作する関数または、そのスーパー型を操作する関数の下請け関数、つまりそのデータ型を直接操作する機能をモジュールと定義する。CommonLispの型階層とモジュール構成の対応の一部を図2に示す。このようなモジュール構成をとることにより、型階層でモジュール間の依存関係を表現でき、特別に各モジュール間の依存関係を記述するための機構を用意する必要が無い。たとえば、文字列型に対応するモジュールが必要である場合、ベクタ型、シーケンス型が文字型のスーパー型であることより、さらにベクタ型、シーケンス型に対応するモジュールが必要であることが分る。

4．データ型に基づくモジュール選択

分割されたLispシステムから必要なモジュールを選択するためには、対象とするアプリケーション・プログラム中で用いられているデータ型を指定する。指定されたデータ型とそのスーパー型に対応するモジュールを必要なモジュールとして選択する。たとえば、シーケンスの副型ではリスト型のみ指定された場合、リスト型、シーケンス型に対応するモジュールを選択し、その他のシーケンス型の全ての副型に対応するモジュールは選択されない。

データ型の指定を容易にするために、アプリケーション・プログラムからデータ型の抽出を支援する機能が必要である。このため、プログラムの静的(declare、theなどの宣言、さらにシステム関数の機能)、動的な解析(評価機構にデータ型を調べるフックの組み込み)からデータ型を抽出することとする。

5．関数単位での機能抽出

選択されたモジュールはデータ構造単位なので、同じデータ構造に対する操作であれば、使用しないものまで含んでいる。関数単位での機能抽出を行うことで、さらに最適な機能抽出を行う。

6．まとめ

CommonLispのデータ型に基づいた言語機能抽出方式について提案した。この方式に従い、オブジェクト・ファイルの生成、それを用いた起動の各方式も含めたLispシステム生成システムをELIS-8200シリーズ上に実現する。

さらに、このデータ型によるモジュール分割は、これらの各データ型をクラスと対応づけると、各モジュールに含まれる操作群は各クラスのメソッドと対応し、将来CommonLispにCLOSが導入された場合にも、同様な手法が適応できるものと思われる。

謝辞

常日頃御指導をいただいている日比野主幹研究員に感謝いたします。

[参考文献]

[1] 鈴木他:新ELISシステム概念,信学全大,1989.