プログラミング言語 16－12
（１９８８．５．１３）

Ｃｏｍｍｏｎ　Ｌｉｓｐにおける日本語化

元吉文男
（電子技術総合研究所）

Ｃｏｍｍｏｎ　Ｌｉｓｐにおける日本語化について述べる。まず、Ｌｉｓｐが他のプログラミング言語に対して持っている特殊性を説明し、Ｌｉｓｐにおいては日本語文字をASCII文字と区別せずに、日本語の１文字をＬｉｓｐでも１文字として扱う方法を述べる。この方法は日本語に限ったものではなく、多バイト文字集合を扱う場合一般に適用できるものである。

Japanese Character Set Handling in Common Lisp

Fumio MOTOYOSHI
Electrotechnical Laboratory
1-1-4, Umezono, Tsukuba, 305 JAPAN.

This paper describes the way of handling Japanese character set in Common Lisp. Lisp is compared to other programming languages and its characteristics are explained. We make a proposal for embedding Japanese characters in Common Lisp, where each Japanese character is treated as a character rather than consecutive characters and no declaration is needed to use Japanese characters. This method can be applied to any muti-byte character set.

１．はじめに

プログラミング言語が日本において一般に使用されるようになると、その日本語化の要求が生じる。ここでは、Lispの１つであるCommon Lispの日本語化について述べる。

Common Lisp は米国で標準的なLispとして設計されたものであるが、現在では多くの国において使用されており、日本においてもKCLを始めとして国産のシステムも作成されている。またその利用者の数も増大し、国内の産業界においても実質的な標準として使用されている。このように広く使用されるようになるに従い、Common Lispで日本語をも扱えるようにしたいという要求が起こるのは当然である。

このような状況下において、電子協では1986年度からLisp技術専門委員会の下に漢字WGを設けて、Common Lisp の日本語化を統一的に行うための議論を重ねてきた。また1987年には、米国のANSIにおいてCommon Lisp の標準化の検討を行っている X3J13に対して、一般の多種文字集合を扱うための提案を行い、これに対応して X3J13 にそのためのWGが発足し、新しい提案も出されている。

ここでは、電子協の漢字WGにおいて議論して合意に達したことを中心に紹介する。なお、既にいくつかのシステムではこの合意に基づいていて、実際にCommon Lispの日本語化が行われていることを付け加えておく。

２．Ｌｉｓｐの特殊性

Common Lispに限らず Lispは他のプログラミング言語とは性質を異にする部分があり、日本語化にあたっても特別の注意をはらう必要があるため、まずその特殊性を説明する。また、日本語化といっても様々なレベルが考えられるが、Lispにおいてはどの程度まで日本語化するかも上の特殊性と関係してくる問題であるのでここで述べる。

Lispは記号処理用の言語として開発されたものであり、そこでは当然日本語を含むデータも扱えるべきである。文字や文字列はLispのデータの基本をなすものであるので、単にコメントや入出力の中で日本語を扱えるだけでは不十分である。また、Lispは対話的に使用することが前提となっており、（コンパイルコードの効率をはかるために用いることはあるが）型宣言のいらない言語である。そこで日本語化を実現する際にも宣言なしに日本語の処理を行えるのが望ましいことになる。

また、Lispで日本語を（入出力としてではなく）記号として扱うときには、文字を最小の処理単位として扱うのが自然であるし、理にかなっている。実際に日本語を含むデータの処理を行う場合でも、バイト単位で扱うよりも、文字単位で扱う利点の方が多い。バイト単位で扱うのが便利なのは文字の出力に関係している部分であるが、文字の印字幅が占めるバイト数に等しいというのは「偶然の一致」であり、このためだけに処理単位をバイトにするのは変である。

さらにLispには、プログラムとデータが同じ構造をしており、同じ構文で記述されるという特徴がある。従ってLispにおいてはデータを日本語化すれば、同時にプログラムの日本語化ができることになる。すなわち、データの日本語化が行われれば変数名、関数名などの識別子にも日本語文字が使用でき、望みとあれば日本語の予約語を入れることすら可能である。このことは逆に、データの日本語化は中途半端にはできないことを意味するものでもある。

またCommon Lisp にだけ言えることで

はあるが、文字のコードとして取り得る大きさが処理系によって異なっていてもよいことになっている事情がある。これは本来は多バイト文字を扱えるようにという目的ではないにせよ、現在のCommon Lispの枠組みの中でも日本語化を行なう余地があることを示している。

またここでは日本語化といっているが、特に日本語に限ったわけではなく、ほかの文字集合、特に多バイト文字集合についても全く同様のことが言える。逆に、Common Lisp の日本語化に際しては、他の国での拡張にも同じ手法が使えるように考慮すべきである。

3．Common Lispの日本語化

上に述べたことを考慮すると Common Lispの日本語化は

文字を書くことのできる場所にはどこでも日本語文字を書けるようにする

べきであるということになる。これは日本語化としては極端なものであるが、記号処理言語であるLispにおいてはこの程度のことは行う必要がある。

上の原則がCommon Lisp における日本語化の全てであるが、この内容を少し詳しく説明することにする。この原則が言わんとしていることをいくつか抜き出してみると次のようになる：

a．何の宣言もせずに日本語を扱えるようにする。

b．最小処理単位は文字であり、日本語文字もASCII 文字も同じ１文字のデータとして扱う。

これを実現するためには、入出力では常に文字のストリームを扱うようにし、日本語文字も１単位として処理すればよい。すなわち、入出力においては、ファイルあるいは端末でのデータ表現を「文字」というデータのストリームに変換するためのフィルタを通して、それに対して処理を行うようにするということである。この様子を次の図に示す。こうすることによって処理系による日本語文字表現法の違いを吸収することも可能になる。なお。コードを直接操作したりするために、フィルタをかけないモードを用意しておくことはいうまでもない。この場合には文字の意味の解釈は利用者の責任にまかせることになる。

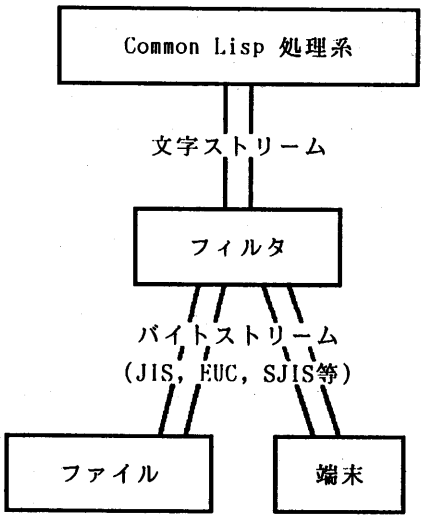

図１．ストリームの変換

以上のことを実現するためには、処理系において全ての文字を区別できる必要がある。そこで、使用する全ての文字に（ASCII文字にも、日本語文字にも）ユニークなコードを割り当てて区別することにする。このコードをCommon Lisp におけるcharacter codeであると考えれば、Common Lisp の仕様を変更せずに日本語化が行えることになる。なお、このとき当然ながら変数 char-code-limitの値は全ての文字の数より大きくしておく必要がある。

以上で日本語化は実現できるが、日本語を全く使用しない場合には、（時間的、特に空間的には）効率が悪くなることが予想される。これに対しては次のように対処することにする。すなわち、日本語を使用しないと最初から分かっているものについては、利用者が宣言をすることによってシステムに最適化の機会を与えるようにする。ここでシステムが（特にコンパイラが）賢ければ、効率良く実行できることになる。また、特に宣言がない場合でもシステムの判断で最適化を行うことは可能である。その際に注意すべきことは、最適化を行ったために（エラーを生じるなどの）それまでとは（利用者に対して）異なった動作をしてはならないことである。この点にさえ注意していれば、（システムが）内部で異なる表現法を使い分けることも自由である。

4．3区の問題

Common Lisp の日本語化を行った際に問題になるのは、日本語コード中にある、standard characterに対応する文字の扱いである。すなわち、日本語文字の空白、括弧などにも構文的な意味を持たせるかとか、日本語文字のAとstandard characterのAを同じものとして扱うかどうかなどという問題である。もちろん、処理系の作り方によってはどちらか1つしか存在しないようにすることも可能ではあるが、現実の問題として考えると、現在のところでは（いわゆる全角、半角の）2種類の文字を扱わざるを得ない状況にある。

これについては、まだ確固たるものが定まっておらず、とりあえずは、少なくとも日本語文字を特別な構文的意味を持たないconstituent文字として扱うモードを持つようにすることにしている。なおANSIにおいては、これらの問題も含めて統一的に解決しようとequivalence classという概念を導入する提案がなされている。

5．まとめ

以上述べてきたことを要約すると、次のようにまとめることができる：

Common Lispの日本語化においては
- 日本語の1文字をLispの1文字として扱う。
- 日本語を扱う部分に特別な宣言を入れなくても動作する。
- 日本語を扱わないと（宣言あるいは推論によって）分かるものにはシステムが最適化を行ってもよい。
- standard characterに対応する文字がある日本語文字は、とりあえずはconstituentとして扱う。

この案をもとに、いくつかの Common Lispの処理系では実際に日本語化が行われている。

6．参考文献

Ida, et al., "JEIDA Common LISP Committee Proposal on embedding Multi-Byte Characters", ANSI X3J13 document 87-022, 1987.

Linden,"Common LISP - Proposed Extensions for International Character Set Handling" (Ver 01.11.87), IBM Almaden Research Center, 1987.

元吉、"Common Lispにおける日本語処理方式の提案"、情報処理学会記号処理研究会 87-SYM-40、1987.

元吉、"Common Lisp アラカルト ―― 日本語化"、bit、Vol 19、No 5、1987.