プログラミング言語 16－11
（１９８８．５．１３）

Ａｄａにおける日本語化の現状と今後の方向

伊集院　正　　石畑　清

ＮＴＴ　　東京大学
ソフトウェア研究所　　理学部情報科学科

本報告では，情報規格調査会ＳＣ２２のＡｄａＷＧで検討を進めている，Ａｄａの文字セットの拡張案について述べる．現在，Ａｄａでは，プログラムテキストに表現できる文字セットを，ＩＳＯ７ビットコードと規定しているため，文字列リテラル中や注釈中でも片仮名や日本語を記述することができない．本拡張案では，現行仕様との上方互換性を保ちつつ，文字セットをＩＳＯ８ビットコードに拡張するとともに，処理系によってはマルチオクテット図形文字セット（日本語）を追加できる．この拡張によって，日本語を処理するプログラムの記述性を大幅に改善することができる．

A PROPOSAL FOR FACILITIES OF MULTI-OCTET CHARACTERS HANDLING IN Ada PROGRAMMING LANGUAGE

Sei IJUIN　　and　　Kiyoshi ISHIHATA

NTT Software Laboratories,
NTT Corporation
1-9-1, Kounan, Minato-ku,
Tokyo, 108 Japan

Department of Information Science,
Faculty of Science, Tokyo University
7-3-1, hongou, bunkyou-ku,
Tokyo, 113 Japan

This report proposes facilities of Multi-octet and 8-bit coded characters handling in Ada programming language. Japanese 8-bit coded and multi-octet characters are not allowed to be written in Ada programs, even in comments, since Ada allows only the ISO 7-bit coded characters to appear in a program text. In order to allow these characters to appear in a program text, extention of character set are proposed. And also, supprement of a predefined character type and input-output facilities for extended character set are proposed. Compatibility with the present language specification is considered.

１．背景

近年，日本語の処理が必要なプログラムの需要の増大に対応するため，ＣＯＢＯＬやＦＯＲＴＲＡＮ等の標準プログラミング言語への日本語処理機能の導入が盛んに行われてきた．Ａｄａではプログラムテキストに表現できる文字セットを，ＩＳ０７ビットコードと規定しているため，文字列リテラル中や注釈中でも片仮名及び日本語を記述することはできない．また，Ａｄａの言語仕様に完全に準拠した処理系だけを「Ａｄａ処理系」と認めるというアメリカ合衆国政府の政策があり，処理系で独自の仕様拡張を行うことも困難な状況であった．このため，この文字セットの仕様は，日本等の非西欧語圏でＡｄａを使用する上で重要な問題になると考えられた．

このため，国内において，文字セットの改良案の検討が開始された[1],[2]．また，１９８６年に設立された情報規格調査会のＳＣ２２のＡｄａＷＧでも，検討が開始された．１９８６年のＡｄａの国際規格原案の登録に対する投票において，将来の規格の改訂の際には，文字セットを８ビットコードとする検討を行うこととの付帯意見が付けられた．現在，ＡｄａＷＧでは，ISO/TC97/SC22/WG9に日本語も含む文字セットの拡張案を提案すべく，検討を進めている．ここでは，その拡張案の概要について報告する．

２．拡張の原則

仕様の拡張案を検討するに当たって，次に示す原則を設けている．

（１）現行仕様との上方互換性及び言語設計思想との調和を保つ．

（２）符号化法や実現方式から独立した仕様とする．（但し，ＩＳ０４８７３－１９９　８ビットコードには準拠する．）

（３）日本語機能専門委員会で検討されているプログラミング言語における日本語の統一的取扱い方法[3]を尊重する．

３．拡張仕様案

３．１．文字セット

現行仕様で，プログラムテキストに現れることが許されている文字は，ＩＳ０６４６の７ビットコード文字セットの内の図形文字と一部の制御文字（水平タブ，垂直タブ，復帰，改行，書式送り）である．これを，ＩＳ０４８７３　８ビットコード文字セットの内の図形文字と現行仕様と同じ一部の制御文字とする．但し，８ビットコード文字セットの列１０～列１５に割り当てられる図形文字は国別に異なるため，規定しない．この位置に図形文字が割り当てられていなくてもよい．

国によっては，さらに，マルチオクテット図形文字セット（日本語）を追加してもよいこととする．このマルチオクテット図形文字セットを指示する方法は規定しない．しかし，この指示が，ソーステキスト上に現れる複数個のビットの組合せによって行われる場合は，このビットの組合せは，ＩＳ０２０２２－１９８２符号拡張法で規定されているエスケープシーケンスでなければならない．

３．２．日本語が使用できる場所

プログラムを読みやすくするため，識別子にも８ビットコード文字セットの片仮名や日本語が使用できることが望ましい．しかし，現行の処理系への影響を少なくし，欧米各国が受け入れやすい拡張案とするため，これらは，文字リテラル，文字列リテラル及び注釈の中だけに現れることが許されるとする．

例：

```
KANA :=  "ｱｲｳｴ" & 'ｵ';          --ﾘﾃﾗﾙ ｵﾖﾋﾞ ﾁｭｳｼｬｸ ﾁｭｳ ﾆ ｶﾀｶﾅ ｶﾞ ｶｹﾙ
NIPPON_GO :=  "日本" & '語';  --リテラル及び注釈中に日本語が書ける
```

なお，日本語を含む文字リテラルや文字列リテラルを区別するために，これらのリテラルの表記法を変える案もある．しかし，Ａｄａでは，これらのリテラルの型は，文脈から決定できなければならないため，表記法を変えて区別する必要はない．したがって，表記法は変えないこととする．

３．３．文字及び文字列を表すデータ型

（１）現行仕様

Ａｄａでは，文字型は，一つ以上の列挙リテラルが文字リテラルである列挙型と定義されている．このため，文字型に属する型を自由に定義できる．また，既定義の文字型として，７ビットコード文字セットの１２８文字を表すCHARACTERがある．文字列は文字型を要素とする一次元配列で表す．型CHARACTERの一次元配列である既定義の型STRINGがある．

（２）拡張案

既定義の型CHARACTERは，８ビットコード文字セットの文字を表す文字型とする．その型定義は，列０～列７の文字については現行と同じ（ＡＳＣＩＩ）とし，列８～列１５の文字については，国別で異なるため規定しない（なくてもよい）．　日本語を許す処理系の場合，次に示すように，１つの文字型に８ビットコード文字と日本語の混在を許す案と許さない案があり，検討を行っている．

案１：混在を許す案

一つの列挙型定義の列挙リテラルに，８ビットコード文字の文字リテラルと日本語の文字リテラルを混在させることで，次の例のように，容易に混在が実現できる．この場合，８ビットコード文字も日本語もそれぞれ１文字と数える．

例：

```
type MY_CHAR is ('ｶ','ﾞ','ﾃ','ｷ','ﾙ','混','在');
type MY_STR is array(POSITIVE range<>) of MY_CHAR;
S : MY_STR(1..8);

S :=  "混在ｶﾞﾃﾞｷﾙ";
```

既定義型のCHARACTERに対応して，８ビットコード文字及び日本語のすべての文字を表す既定義の文字型LONG_CHARACTERを次の例に示すように定義する．また，型LONG_CHARACTERの一次元配列である既定義の型LONG_STRINGを定義する．

例：

```
type LONG_CHARACTER is (nul, soh, ····· 'ﾞ', del,--ＪＩＳ８ビット文字と漢字
      '　', '、', '。', ····· '槇', '遥', '瑤');  --が混在する場合の定義例
```

```
for LONG_CHARACTER use (16#0000#,16#0001#, ····· 16#00FF#,
                        16#2121#,16#2122#, ·····
                        16#7422#,16#7423#,16#7424#);
type LONG_STRING is array(POSITIVE range <>) of LONG_CHARACTER;
```

案２：混在を許さない案

一つの列挙型定義の列挙リテラルに，８ビットコード文字の文字リテラルと日本語の文字リテラルが混在して現われてはならないこととする．このため，８ビットコード文字と日本語が混在した文字型及び文字列は許されない．

既定義型のCHARACTERに対応して，日本語のすべての文字を表す既定義の文字型LONG_CHARACTERを次の例に示すように定義する．また，型LONG_CHARACTERの一次元配列である既定義の型LONG_STRINGを定義する．

例：

```
type LONG_CHARACTER is (' ', '、', '。', ·····  --ＪＩＳ漢字の場合の
                        '槇', '遥', '瑤');        --定義例
for LONG_CHARACTER use (16#2121#,16#2122#, ·····
                        16#7422#,16#7423#,16#7424#);
type LONG_STRING is array(POSITIVE range <>) of LONG_CHARACTER;
```

案１は処理系が若干複雑となるが，混在した文字列やテキストファイルの処理の記述が容易に行える．案２は案１とちょうど逆の利点欠点となる．

３．４．入出力パッケージ

現行仕様では，文字が入出力されるファイル（テキストファイル）に対して入出力や改行・改頁等を行うサブプログラムを集めた既定義のパッケージTEXT_IOが定義されている．

拡張案でも同様に，パッケージTEXT_IOを定義する．日本語を許す処理系の場合，型LONG_CHARACTERの文字を含むファイルに対しても入出力機能を規定するが，規定方法等については，検討中である．

４．今後の予定

記述実験等によって更に検討を進め，言語仕様案を取りまとめて，ISO/TC97/SC22/WG9へ提案する予定である．

<謝辞>

本報告作成に当り，貴重なご意見をいただいた情報規格調査会ＳＣ２２／ＡｄａＷＧの皆様に深謝いたします．

<参考文献>

(1)福山，小林，伊集院他：Ａｄａ日本語処理機能の実現法，情報処理学会第３０回全国大会論文集，p.459(1985)

(2)松尾，牛島：Ａｄａにおける日本語テキスト処理パッケージの構築と利用，情報処理学会第３２回全国大会論文集，p.437(1986)

(3)日本語機能専門委員会：情報処理における日本語機能の標準化に関する調査研究報告書，情報処理学会(1988)