4B−5　形態素解析ツールによるかな漢字プログラミングの実現

*間瀬久雄　*辻　洋　*絹川博之　**川村隆雄
*(株)日立製作所 システム開発研究所　**(株)日立製作所 ソフトウェア開発本部

1．はじめに

プログラム開発の効率向上の一方法である，かな漢字プログラミングの研究を進めている．これまでに，単語分かち書きのカナ文字列で記述可能なＣＯＢＯＬプログラミング用簡易言語ＣＯＲＡＬ（図１）を開発した．ＣＯＲＡＬは大型計算機ＶＯＳシリーズ上で稼動している．

我々は，開発効率をさらに向上させるべく，より可読性に優れた非単語分かち書きかな漢字文による記述（図１）を検討し，形態素解析によってかな漢字プログラムを既存のＣＯＲＡＬに変換するプリコンパイラのプロトタイプを開発した．これによりデバッグ効率が向上するほか，プログラムを仕様書として利用できる．なお，本プロトタイプにおけるかな漢字プログラムの構文は，従来のＣＯＲＡＬの構文をほぼ継承している．本稿では，本プロトタイプの構成および機能について述べ，また，テストプログラムを用いた評価結果について考察する．

2．プロトタイプの構成

ここでは，本プロトタイプによりかな漢字プログラムをＣＯＲＡＬに変換し，既存のコンパイラによりＣＯＲＡＬをＣＯＢＯＬに変換するという２段階方式を採用する．

本プロトタイプの構成を図２に示す．本プロトタイプはワークステーション上で稼動する．

（１）入力

①かな漢字プログラム（図１）

ワープロやエディタ等を用いて作成したかな漢字プログラムを入力とする．かな漢字プログラムは，Shift-JIS の２バイト文字で構成する．ただし，インデント等をしやすくするために，タブ・半角スペースの使用を許している．

②利用者定義データ

ＣＯＲＡＬでは，ＤＢのスキーマ情報や画面名称等のグローバルデータを定義するためのテーブルがいくつか用意されている．利用者はプログラム中で参照するデータを各テーブルに記述する．その際，かな漢字文字列とカナ文字列とを対応させて記述する．

（２）出力

①ＣＯＲＡＬプログラム

カナ文字列に変換されたＣＯＲＡＬプログラムを出力する．単語辞書に未登録の単語（未登録語）はかな漢字のまま出力する．

②エラーメッセージ

日本語形態素解析処理の結果，未登録語が存在する場合，その旨を表すエラーメッセージを出力する．本プリコンパイラでは，語彙エラー，すなわちタイプミスや未宣言の変数の使用等によるエラーのみを扱っている．その他の細かいエラー検出については，既存のコンパイラに任せる．

<かな漢字プログラム>	<ＣＯＲＡＬプログラム>
1　＃２０．	1　#20.
2　　画面１を読む。	2　　ガメン1　オ　ヨム．
3　　終わりなら終了。	3　　オワリ　ナラ　シュウリョウ．
4　　１０回ループ。	4　　10　カイ　ループ．
5　　＃３０を処理。	5　　#30　オ　ショリ．
6　　＃３０−１０へ行く。	6　　#30-10　ヘ　イク．

図１　かな漢字プログラムと従来のＣＯＲＡＬプログラム

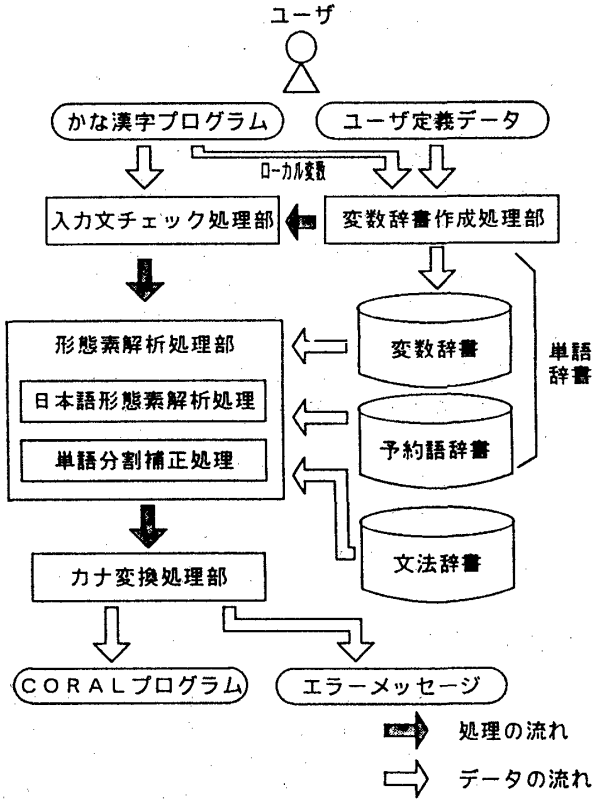

図２　本プロトタイプの構成

（３）予約語辞書

ＣＯＲＡＬで定義されている基本的な予約語は約１２０語である．予約語辞書にはこれらの予約語に関する形態素情報（品詞，活用等），およびカナ文字列が格納されている．

（４）変数辞書

変数辞書には，グローバル変数に関するデータとローカル変数に関するデータが格納されている．辞書アクセスの効率化のため，変数辞書は予約語辞書と同一構造をなす．変数は，動的に変化するため，実行時に変数辞書を自動的に更新する

Kana-kanji Character Programming Based on a Japanese Morph Analysis Toolkit.
HISAO MASE, HIROSHI TSUJI, HIROSHI KINUKAWA, TAKAO KAWAMURA
* Systems Development Laboratory, Hitachi, Ltd.
** Software Development Department, Hitachi, Ltd.

必要がある（（５）参照）．

（５）変数辞書作成処理部

ここでは，プログラムおよびユーザの定義したデータを用いて変数辞書を自動作成（更新）する．

①グローバル変数について

（１）で述べたように，グローバル変数に関しては，利用者が変数の見出しと対応するカナ文字列を定義するので，新たに定義された変数について，その見出しに関するデータを変数辞書に登録する．なお，グローバル変数の品詞はすべて名詞として登録する．

②ローカル変数について

ＣＯＲＡＬでは，ローカル変数を宣言する場合には行頭に識別子（" Ｔ"，" Ｖ" 等）をつけるので，ローカル変数はプログラムから自動抽出できる．また，ローカル変数に対応するカナ文字列を利用者に定義させることは使い勝手を悪くするので，システムが適当な変数コードを割り当てる．

（６）入力文チェック処理部

入力文の中に不正な文字が存在しないかを判別し，存在する場合には，エラーとしてその旨を報知する．また，１バイト文字のタブ，半角スペースを全角スペースに置き換える．

（７）形態素解析処理部

①日本語形態素解析処理 ''

日英機械翻訳システムＨＩＣＡＴＳ／ＪＥ（HItachi Computer Aided Translation System / Japanese to English）の形態素解析処理を切り出してモジュール化した日本語形態素解析ツールを利用して入力文を単語分割し(最長一致法)，各単語に対応するカナ文字列を取得する．

②単語分割補正処理

形態素解析ツールによる単語分割結果がＣＯＲＡＬの単語分割仕様と異なる部分について，単語分割結果を補正する．

（例１）" ＃ ２０ − １０" → " ＃２０−１０"

（例２）未登録語" Ａ" ＋未登録語" Ｂ" →未登録語" ＡＢ"

（８）カナ変換処理部

予約語辞書および変数辞書から取得したカナデータに基づいて各単語をカナ文字列に変換して出力ファイルに格納する．

３．評価と考察

（１）評価方法

テストプログラムを用いて本プロトタイプの性能を評価した．本評価で用いたかな漢字プログラムは，ＣＯＲＡＬの操作マニュアルに掲載されているサンプルプログラムを，ある人に頼んでかな漢字に変換して頂いたプログラム（１７６行）である．また，評価内容としては，①変換能力，②エラー検出能力（エラーメッセージの妥当性），③処理速度の３点とした．

（２）評価結果

テストプログラムの一部とその変換結果を図３に示す．

処理結果として，次の二つのエラーを検出した．その他の部分については，すべて正しくカナ文字列に変換できた．

①語彙エラー：４３行目「子」「位置付け」

ＣＯＲＡＬでは，「ピリオド＋スペース」以降の部分はコメントを表す．図３の４３行目の「子レコード位置付け」はコメントであるが，ピリオドの直後にスペースを挿入しなかったために，コメントとみなされず，その結果，「子レコード」が単語分割され，予約語「レコード」がカナに変換され，「子」「位置付け」が未登録語と判断されエラーとして表示された．これらがともに予約語であるとした場合，この箇所のエラーは検出されず，ＣＯＢＯＬへのコンパイル時まで検出が遅れることになる．

<かな漢字プログラム>

```
41  C  プロジェクト構成員一覧Ｉ−ＰＦＫ  ＝  ？
42  −          |ＰＦ９|      |.
43  Ａ  ？を処理  |＃４０|＃２０|. 子レコード位置付け
...
140      １６回ループする。
141          Ｉ  −−−>  添え字Ｉ。
142          ＃６０を処理する。
...
156  Ｖ  添え字１  Ｓ（４）
```

<変換後のＣＯＲＡＬプログラム>

```
41  C  ﾌﾟﾛｼﾞｪｸﾄｺｳｾｲｲﾝｲﾁﾗﾝI-PFK = ?
42  -            |PF9 |    |.
43  A  ?  ｵ  ｼｮﾘ  |#40 |#20 |.  子  ﾚｺｰﾄﾞ  位置付け
...
140      16  ｶｲ  ﾙｰﾌﾟ  ｽﾙ.
141        I  --->  添え字  I.
142        #60  ｵ  ｼｮﾘ  ｽﾙ.
...
156  V  ﾘｴｼﾞ1  S(4)
```

図３　評価に用いたテストプログラムの処理結果の一部

②語彙エラー：１４１行目「添え字」

１５６行目でローカル変数「添え字１」が定義されているので，変数辞書に登録される．しかし，１４１行目には「添え字Ｉ」と記述されている．「Ｉ」はループ変数を表すシステム変数であるので，「添え字Ｉ」は「添え字」と「Ｉ」に単語分割される．そして「添え字」が未登録語と判断され，エラーとして表示された．

全体としては処理精度は良好であった．ただ，上述したように，未登録語が予約語や変数データを含む場合，エラー検出能力が落ちる恐れがある．語彙エラーをすべて正しく検出するためには，構文・意味的な制約を利用する必要がある．

なお，テストプログラムの処理時間は約３秒であり，妥当な速度範囲内にあると言える．ただし，今回の評価では，変数辞書作成処理に要した時間を含んでいない．本プロトタイプの処理時間の大半は日本語形態素解析ツールが占めている．

４．おわりに

プログラム開発効率の向上を目的として，可読性に優れた非単語分かち書きかな漢字によるプログラミングを検討し，かな漢字プログラムをＣＯＢＯＬプログラミング用簡易言語ＣＯＲＡＬに変換するプリコンパイラのプロトタイプを開発した．今後は，ＣＯＢＯＬコンパイルの前までに語彙エラーをすべて解消できるように，エラー検出能力を向上させていくとともに，プロトタイプの評価をさらに進めていく．

参考文献

１）西森，木山，絹川：汎用日本語形態素解析ツールの開発，情報処理学会第４４回全国大会（1992）．

２）（株）日立製作所：ＥＡＧＬＥ２　ＣＯＲＡＬ入門編，ＶＯＳ３　６１８０−３−８２２（1991）．