自然言語インタフェースの開発方針について

2C-5

増山顕成，妹尾栄子，神保寿久
富士通㈱　　㈱富士通テクノシステム

1．はじめに

近年，自然言語インタフェースの商品化が行われるようになってきたが，従来の自然言語インタフェースには，次のような問題がある．

①ユーザの期待する性能と実際の性能の間に，大きなギャップがある．
⇒ユーザは，「自然言語インターフェース」という言葉を聞くと，人間と会話するように，計算機と会話できると考える．ところが，現状の技術レベルでは，そのようなレベルの汎用的な自然言語インターフェースを提供することは出来ない．さらに，現状のパーサは機構が大変複雑であるため，出来ることと出来ないことを一言で言うことができず，運用担当者に大変理解しにくいシステムになってしまっている．

②運用・構築の負担が大きすぎる．
⇒自然言語インタフェースの構築・運用を行うためには，複雑なシステムの内容を理解することが必要であり，特に，自然言語特有の問題の性質といったものを理解しなければ完全にはできない．さらに，この自然言語特有の問題により，従来のプログラムの保守の数倍の工数がかかる．

③自然言語インタフェースの使う資源（ＣＰＵ時間，メモリ）が大きいため，応用プログラムが圧迫される．
⇒インタフェースは応用プログラムのフロントエンドであるのに，インタフェース自身が応用プログラム並みの資源を使うのはおかしい．

我々が現在開発中の自然言語インタフェースは①の問題を解決するために，応用分野をデータベースに限ることにした．
しかし，データベースに限っても，汎用の自然言語インタフェースを提供することは難しい．そこで，特定のデータベースに限るならば，質問文や語彙も限定され，必要な性能が得られるだろうと考え，特定データベースへのチューニングツールを提供することにした．

次節で，我々が現在開発中の自然言語インタフェースの開発方針について，具体的に示す．

2．開発の方針

自然言語インタフェースの開発にあたって，次のような方針を立てた．

①提供形態をサブルーチン提供とする．

一般に，自然言語インタフェースは次図のような構成となる．

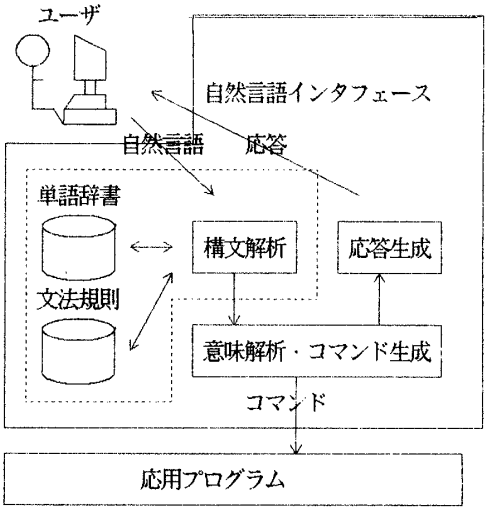


図1．1

我々は，次のような理由から，自然言語インタフェースの内から，上図の…で囲った構文解析部のみを提供することにした．

(1)自然言語のみでユーザと会話を行うのは現状の技術レベルでは難しい．

(2)応答生成も自然言語インタフェースで行うと，応用プログラムの応答と自然言語インタフェースの応答の２種類が存在することになり，エンドユーザを混乱させる．応答生成は応用プログラム作成の担当者がトータルな観点から考えるべきである．

従って，このシステムの提供先は，各応用プログラムの構築者であり，エンドユーザが直接このシステムを使うことはない．応用プログラム構築者は，このシステムと，エキスパートシステムや応用プログラム自身のフロントエンドなどを組み合わせて，システムのフロントエンドを構築することになる．

②個々のユーザに合わせた文法規則が容易に開発できるようにする．

第１節で自然言語インタフェースが十分な性能を持つためには，個別のシステムにチューニングできること，及び構築・運用の負担を軽減することが重要であることを示した．
この要件を満足させるため次の２つの対処を行った．

A design of the natural language interface
Akinari MASUYAMA, Senoo EIKO, Jinbo TOSHIHISA
FUJITSU,Ltd.　　FUJITSU Technosystem,Ltd.

(1)構文解析を単なるストリング変換と考え，その変換メカニズムは単純なパターンマッチとした．

変換メカニズムを単純なパターンマッチとすることは運用管理者がシステムを容易に理解できるという利点と資源を多く使わないという利点がある．

出力を単純なストリング列とすれば，構文解析は単なるストリング変換のサブルーチンであると捉えることができる（下図）．

日本語ストリング→ [構文解析] →生成ストリング

図2

この生成ストリングをシステム構築者が自由に設定できるようにすれば，応用範囲も広がる．

メカニズムの詳細については，別稿[1]に譲る．

(2)辞書・パターンの構築・運用を容易にするようなツール群を用意した．

第1節で述べたように，構築・運用を行うためには，自然言語特有の問題を理解することが重要である．ところが，これをすべての運用管理者に期待するのは無理がある．そこで，開発元で構築・運用のストーリーを組み立て，そのストーリーに沿って構築・運用ができるよう誘導するツール群を提供することにした．

このストーリーの基本として，辞書・パターンの構築はボトムアップに行うことを想定したストーリーを組み立てた．

機械翻訳システムの開発の経験から，辞書・パターンの構築をトップダウンに行うことはできないことが分かっている．つまり，自然言語について特に経験のない運用担当者が，いきなり文法を書き下すとか，辞書登録を行うということは非常に考えにくい．

そこで，すべてのパターンは例文に基づいて登録することにした．つまり，できない例文があったとき，その例文と生成ストリングをシステムに与え，多少のシステムからの質問に答えることにより，パターンが登録できるようにした．

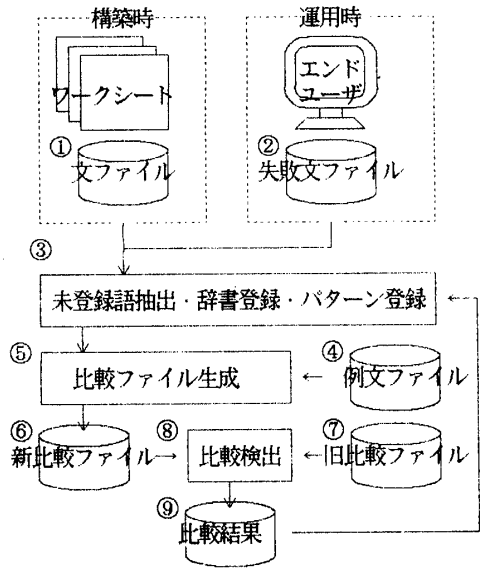
また，辞書・パターンの保守おいて，特に問題となるのは，ある修正を加えたときに，どの範囲まで影響するのかを調べるのが非常に難しいという問題である．

一方，図2に示したように，入力と出力の関係が正しければ，中身はどうなっていても，エンドユーザにとっては，特に問題はない．

そこで，日本語ストリングと，生成ストリングを対応付けたファイル（比較ファイル）を生成し，辞書・パターンの修正後，再び同様のファイルを作成し，比較検出するツールを提供することにした．これにより，運用担当者は少なくとも押さえている例文については，影響がないことを確認することができる．

このように，我々の自然言語インタフェースは，例文を中心にして，構築・運用のストーリーを組み立てている．以下に，構築・運用のストーリーを示す．

①文ファイル：運用担当者はシステムの構築時に，ワークシートなどにより，例文をできるだけ多く集める．これを入力したものが文ファイルである．



②失敗文ファイル：運用時に，正しい結果が出ない文はエンドユーザにより，失敗文ファイルに記録される．

③未登録語抽出・辞書登録・パターン登録：文ファイル，失敗文ファイルを基に，運用担当者は辞書・パターンの修正を行う．このために，未登録語抽出，一括辞書登録ツール，パターン登録ツールなどが用意されている．

④例文ファイル：文ファイル，失敗文ファイルなどにより入力された文のすべてを記録しておくファイル．

⑤比較ファイル生成：最新の辞書・パターンと例文ファイルを使い，入力文と出力ストリングのペアを記録した比較ファイルを生成するツールである．これは比較検出の入力となる．

⑥新比較ファイル：最新の辞書・パターンから生成された比較ファイル．辞書・パターンに問題がなければ，これが旧比較ファイルになる．

⑦旧比較ファイル：一世代前の辞書・パターンから生成された比較ファイル．

⑧比較検出：新・旧比較ファイルどうしを比較して，辞書・パターンの修正により，出力ストリングに変化がないかを調べるツール．

⑨比較結果：辞書・パターンの修正により，同一の文で出力が変わったものを出力したファイル．運用担当者はこれを見て，辞書・パターンを再度修正する．

3．おわりに

構築・運用ツールにより，各ユーザが自分の持つデータベースに自然言語インタフェースを容易にチューニングできるようになった．

今後は，ツールとして，より使い易いものを目指すとともに，談話管理のツールも考えて行く．

〔参考文献〕

[1]「自然言語による意志決定支援」，妹尾他，本大会予稿集