情 報 シ ス テ ム 37－5
（1992. 1. 21）

クライアント・サーバによる機械翻訳システムの設計

諸橋 正幸　丸山 宏　岡野 裕之　野美山 浩　渡辺 日出雄　荻野 紫穂

日本アイ・ビー・エム株式会社 東京基礎研究所

日英翻訳システムを、コンピュータによって翻訳を行う翻訳サーバ、ユーザによる翻訳結果の検討・加工の場を提供する翻訳ワークベンチ、その両者に語い情報を提供する辞書サーバの結合体として設計した。本システムは、翻訳サーバとしてインタラクティブな環境で動く機械翻訳システムＪＥＴＳを導入した。そのため、ユーザとサーバはクライアントを介してあいまいさ解消のための情報交換をする必要が生じた。翻訳処理では、あいまいさが生じる個所が多数存在しうるので、日本語の解析終了後、解析結果の中にあいまいさの残る個所と選択肢をパックした選言付き素性構造をやり取りすることで、クライアント・サーバ間の交信を単純化した。

A Design of A Machine Translation System based on Client-Server Computing

Masayuki Morohashi, Hiroshi Maruyama,Hiroyuki Okano,
Hiroshi Nomiyama, Hideo Watanabe and Shiho Ogino

IBM Research, Tokyo Research Laboratory
5-19 Sanbancho, Chiyoda-ku,
Tokyo 102 Japan

We developed a prototype translation system as an integration of a translation server, a translator's workbench, and a dictionary server. As a translation server, we used JETS, an interactive machine translation system from Japanese into English, which led us to develop more sophisticated data exchange for ambiguity resolution between the translation server and the translator's workbench. We propose disjunctive feature structures as a carrier of data exchange, which can keep a number of ambiguous interpretations of a sentence in a packed form.

1　はじめに

翻訳作業に機械翻訳システムを利用しようとする場合、システムが単に原文を入力とし訳文を出力する単純な機能しか備えていないのでは使いものにならないことは明らかである[1]。したがって、翻訳システムを設計する際は、システムの翻訳精度を追求するばかりでなく、利用者＝翻訳者の作業（原文の意味を吟味し訳文が正しくそれを反映しているか否かを判断し、場合によっては、辞書によって適切な訳語かどうかを確認するといった作業）を援助する機能をどう組み込むかが重要な課題となる。

この課題に取り組んだ多くの翻訳システムにほぼ共通する設計方針は、機能を大きく４つ（原文の入力編集、翻訳処理、翻訳文の後処理、辞書編集）に分割し、各機能の間のインターフェースをなるべく単純なものにする（各機能を実現するモジュールの独立性を高める）ことであるように思われる[2]。

こうした形で分割されたモジュールを個々に見てみると、高速のＣＰＵパワーを必要とする翻訳処理部、ユーザインターフェースが設計の中心となる入力編集・後処理部・辞書編集主要部分、集中管理が必要となる辞書管理など、プログラムの性質の違いによる分類とほぼ対応する形で分けられていることが分かる。このことから、翻訳システムはクライアント・サーバ型のシステム構築に適したアプリケーションといえる。

2 翻訳システムＪＥＴＳの概要

著者らは、日本語から英語への機械翻訳の研究に取り組んでおり、以下のような機械翻訳の基礎技術を用いた日英機械翻訳プロトタイプ・システムを開発している。

（１）制約依存文法に基づくインタラクティブな日本語解析[3]

（２）用例主導による日英構文構造変換[4]

（３）最終的な英文構造の決定プロセスを独立させた英語生成[5]

このうち、（１）の解析技術は文献[6] で提唱された「翻訳すべき原文を作成する過程で翻訳を行い、原文におけるあいまいさ、意味のとりにくさなどをシステムから原文作成者に問いかけるなどして解消しながら翻訳をする、いわゆる対話型翻訳」を可能にする技術の一つである。さらに、機械翻訳におけるボトルネックである原文解析部分[6] にユーザの介入を許し、システムとユーザが協調して解析を行うことで、翻訳精度をあげようという狙いも持っている。

しかし、サーバが解析処理の途中でユーザと対話するという機構をシステムに組み込もうという試みは、当然のことながら、クライアント・サーバ間に、より複雑なプロトコルや、より多くの情報交換を必要とすることになり、翻訳処理と対話機能が一体化した従来のシステムと同様の設計方法は適用しにくくなる。なお、この問題に対する我々の解決策は、次章で述べる。

図１に、我々の研究プロトタイプを翻訳処理部としてもつ、日英翻訳システムＪＥＴＳ（Japanese-to-English Translation System）の構造を示す。

図中、ＪＥＴＳクライアントの辞書編集機能は、辞書サーバの提供するサービスによりその機能を果たすが、ＪＥＴＳ翻訳サーバも翻訳処理に必要な語いの情報をこの辞書サーバによって得る。したがって、辞書サーバから見れば、ＪＥＴＳクライアント同様、翻訳サーバもクライアントの１つとなる。また、ＪＥＴＳクライアントからみればサーバは２つ存在する。すなわち、翻訳処理のためには翻訳サーバを使い、ユーザからの辞書編集要求に応えるためには、辞書サーバと直接やりとりをする。

なお、マシン間の情報交換はストリーム型のソケットを用いて行う。ソケットを採用すると、ＲＰＣによる情報交換に比べ低レベルのプロトコルを使わねばならないという制約はあるが、逆に、受け渡しされるデータがＪＩＳＣＩＩの文字列なので可視性は高い。

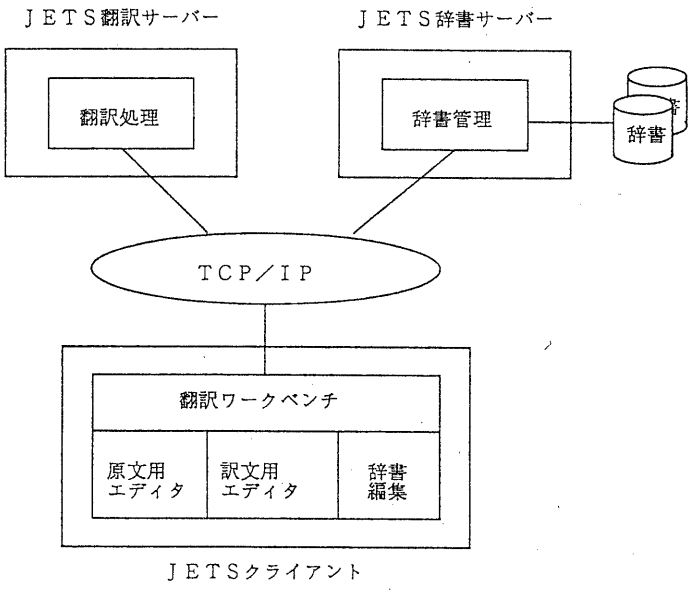

図１　日英翻訳支援システムＪＥＴＳの構造

３　クライアント・サーバ間の交換情報

図１に示すように、ＪＥＴＳでは翻訳および翻訳支援機能は３つのマシンが協調する形で行われる。ここでは、これらのマシン上で行われる処理を概観した後、それらがどのような情報を交換しながら動くかについて述べる。

３．１　ＪＥＴＳ翻訳サーバ

翻訳サーバにおける翻訳処理は、トランスファー方式の手順に従って、解析・変換・生成の順に行われる。翻訳結果には、最終的にシステムが選んだ訳語のほか、それと交換可能な訳語がすべて含まれる。例えば、『牛』に対応する英語として、システムが何らかの基準で "cow" を選んだとしても、訳語の候補として "bull", "ox" などを翻訳結果の中に残しておく。また、ユーザとのインタラクションを可能にするために、日本語の解析結果を、これもシステムが選んだ最終的な結果と同時に、可能な解釈のすべてを含んだまま、翻訳要求を出したクライアントに渡す。サーバはクライアントの応答を待たずに、システムが選択した解釈をもとにして翻訳プロセスを続行するが、クライアント側でシステムと異なる解釈を選択しサーバに戻した場合には、クライアント側の指定した解釈を優先して変換・生成のプロセ

スをやり直す。

さらに、システムの翻訳処理の可視性を高める目的で、日本語生成プロセスを用いて解析結果を再び日本語に戻し、クライアントに渡す。日本語解析のプロセスが、入力された日本語文の情報だけで解析を行うならば、生成プロセスで戻された日本語は入力文とまったく同じになるはずであるが、実際は、しばしば異なる文を生成する。これは、日本語解析が、自身でもつ文法知識やヒューリスティックスなどで原文に現れなかった情報を補ってあいまいな表現を解消しようとするために起きる現象である。

例えば、『コートは取らなかった』という入力文に対して『（誰か）がコートを取らなかった』という解釈をシステムがした場合は『コートを取らなかった』という文が生成されるが、『コートが（何か）を取らなかった』と解釈すれば『コートが取らなかった』という文が生成されることになる。したがって、ユーザはシステムの書き直した文と原文とを比べることで、システムのとった解釈をうかがい知ることができる。

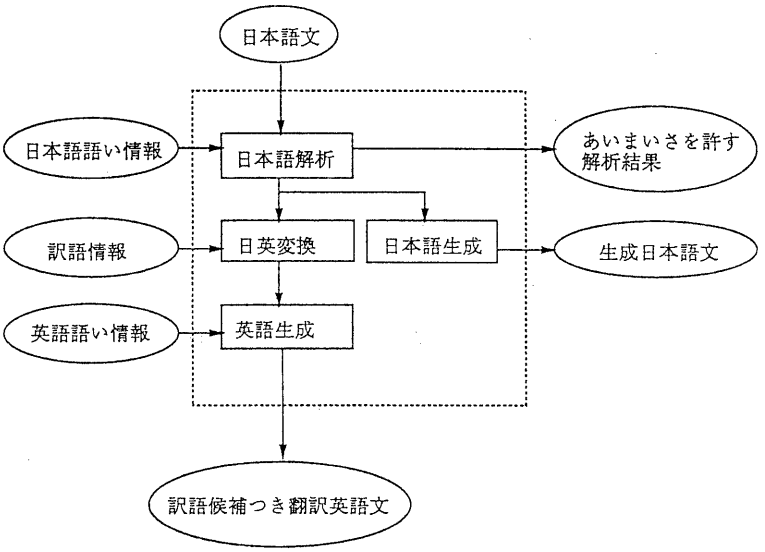

図2　ＪＥＴＳ翻訳サーバのプロセスとマシン間の交換情報

3.2　ＪＥＴＳクライアント

ユーザからの翻訳要求に基づき、クライアントは翻訳すべき文を翻訳サーバに送るが、サーバからは、解析結果、生成日本語文、翻訳英語文の３つが戻ってくる。クライアント・マシンで動いている翻訳ワークベンチは、こうした情報をユーザに分かりやすい形で表示し、システムの誤りを発見しやすくさせる必要がある。

〈解析結果の表示〉

翻訳対象となっている文はすでに原文用エディタに表示されているから、その表示された対象文の文

字列に色をつけたりハイフンを挿入したりして、解析結果に含まれる情報を表示する方法をとった。まずは、２色を使って単語を交互に色づけする。さらに、文節の境界にハイフンを挿入する。これらによって、単語切りや文節切りの間違いを発見しやすくする。

文の構造（係り受けの関係）は、マウスを用いたときのみ表示することにする。すなわち、係り受けを調べたい文節にマウスを移動し、ボタンをクリックするとその文節が修飾している文節をハイライトする。このように、表示すべき情報をいっぺんに出さないようにすることは、複雑な構造をもつデータの表示に当たっては必要な配慮である。

あいまいさの表示は、解析結果のなかでももっとも重要なものなので、なるべく目立つような工夫が必要である。我々は、係り先が複数個ある文節全体を反転した色によって指摘することにした。解析結果のなかには、その文節が修飾しうるすべての候補とその中からシステムが選んだ係り先が含まれているが、ここでも必要最小限の情報のみ表示するという方針に従って、マウスを利用しないかぎり、そうした情報は表示しないことにした。

〈生成日本語文〉

解析結果から生成した日本語は翻訳対象文のすぐ下に挿入することによって、原文との比較をしやすくした。また、この生成文は、システムが推奨する原文の書き換えにもなっているから（もちろん、生成された文が正しい場合のみ）こうした書き換えが表示されることで、システムに受け入れられやすい文をユーザに教えるという効果も期待される。

〈翻訳結果〉

解析結果の表示方法と同様、内部で保持している情報をすべて表示することは避け、重要な部分だけ目立つように表示する。ここで重要なものは交換可能な訳語の候補であるから、そうした候補を持つ単語を反転させた色づけでマークすることにした。交換可能な候補の表示については、マウスでクリックされてから行うようにしている。

３．３　翻訳処理における交換情報

どのような種類の情報交換がなされるかについては既に述べてあるので、ここではその中で最も複雑な構造を持ち情報量も多い解析結果の受け渡しに使われるデータ構造を紹介する。

あいまいな構造（構造が一意に決められないもの）を正確に表現するには、可能な構造をすべて列挙し、それらを線形リストにして送るのが一番簡単である。ただし、そうした構造をとると、データ量が多くなると同時に、ユーザにとっても次々と表示される構造の間の違いを探すのが煩わしいという欠点がある。

我々の翻訳サーバで用いている日本語解析では、制約依存文法[7] に基づいており、解析の際に使われる内部のデータ構造は基本的に係り受けの二項関係のみを保持し、他の二項関係との関連（制約）は制約伝播アルゴリズムによって維持される。したがって、このデータ構造をそのままクライアントとの交信に用いることで、データの圧縮が実現すると同時に ３．２で述べたようなあいまいさの表示が簡単に実現できるようになる。

図３に、簡単な例文を解析した結果を保持する素性構造を図示する。図中の[....]はそれぞれの素性構造を表す。また、{....}はリストを、{%....%}は素性構造における選言（disjunction）を表す。

```
{[id=0,words={[id=0,syn={%[str='私',pos=代名詞],[str='渡し',pos=名詞]%},
                str='わたし'],
               [id=1,syn=[str='は',pos=係助詞],str='は']},
 str="私は",modifiee={%1,2%}],
[id=1,words={[id=0,syn=[str='魚',pos=名詞],str='魚'],
               [id=1,syn=[str='が',pos=格助詞ガ],str='が']},
 str="魚が",modifiee=2],
[id=2,words={[id=0,syn=[str='食べ',pos=動詞],str='食べ'],
               [id=1,syn=[str='た',pos=助動詞],str='た',fe={hope}],
               [id=2,syn=[str='い',pos=活用語尾],str='い'],
               [id=3,syn=[str='。',pos=句点],str='。']},
 str='食べたい。']}
```

図３　「わたしは魚が食べたい。」の素性構造

３．４　ＪＥＴＳ辞書サーバと辞書に関する交換情報

図２に示すように、翻訳サーバがクライアントとして辞書サーバに情報を求めるときは原文に現れる語いの文法情報とそれに対応する訳語の要求は、別々の時点で発生する。さらに、英語に固有の語い情報（例えば、特殊な複数形を持つとか、不加算名詞であるとかいう情報）も独立して扱われる。しかしながら、辞書項目の参照・追加・変更といったクライアント側からの要求に対しては、少なくとも日本語の文法情報と対訳は同時に扱う必要がある。

そのため、実ＤＢとしての辞書は、日本語の語いをキーとして持つ項目に日本語の語い情報と対訳を持ち（これを和英辞書とよぶ）、英語の語いをキーとして持つ項目については英語固有の語い情報を持つ（これを英語辞書とよぶ）。そして、翻訳サーバからの要求に対応したビューを設けて、必要な情報だけにアクセスできるようにしている。

さらに、語いのセットに名前をつけることを許し（いわゆる、専門用語辞書とか個人辞書とよばれるもの）、アクセスにあたっては任意の辞書の組み合わせで検索できるようにしている。

また、辞書サーバは辞書中には存在しない語やフィーチャー値を動的に生成するためのプログラムの存在を許している。現時点では、カタカナを英語に直すプログラムや、名詞の複数形を推定するプログラム（これは動詞の三単現にも使える）が組み込まれているので、例えば、辞書にないカタカナ語が文中に現れても、プログラムが正しく推定する限り対応する訳語を供給できる。

４　ＪＥＴＳの利用形態の拡充

現時点のシステムは、翻訳サーバと辞書サーバがＡＩＸ上で稼働し、ＯＳ／２上で動く翻訳ワークベンチから翻訳依頼や辞書の参照・編集をする形態をとっている。ＪＥＴＳのオープンシステムとしての利点を生かすためには、まず、翻訳ワークベンチをＡＩＸなど、ＯＳ／２以外のＯＳに移植し、ＬＡＮに参加しているどのシステムからでも同じ翻訳サービスが受けられるという状況に近づけることを計画

している。

また、システムの利用範囲を広げるもう１つの可能性として、翻訳ワークベンチを既存のビジネスアプリケーションに結合して、ユーザが普段使い慣れているアプリケーションに翻訳機能が付加できるという形で利用形態を広げる試みをしており、その成果の１つとして、ＯＳ／２上のＣＡＤシステムとして市販されているマイクロＣＡＤＡＭ／２* とＪＥＴＳの翻訳ワークベンチがパイプを経由して結合されている。したがって、このアプリケーション・ユーザはＣＡＤ図面上に書かれた注記や部品名を翻訳する際にＪＥＴＳの翻訳サービスが受けられるようになっている。

５　おわりに

現在、ＪＥＴＳは我々の施設内のＬＡＮを用いて、比較的小規模な試用を行っているが、ユーザが増え、翻訳サーバや辞書サーバも複数個稼働するような状況に対処するためには、付け加えなければならない機能や検討を要する問題（サーバの負荷の分散や辞書のセキュリティの問題など）が多い。さらに、機械翻訳自体の研究の進歩により、異なる翻訳方式をもった翻訳サーバがＬＡＮ上に存在するようになったり、翻訳サーバに学習機能がついて個々のサーバの翻訳能力が異なった発展をしていくようなことが起きるとすれば、翻訳サーバの選択に関する戦略をもった賢いクライアントの開発という興味のある研究に発展していく可能性もある。

参考文献

[1] 「翻訳の世界」1991年1月号

[2] 野村・田中編「機械翻訳」bit別冊, 1988

[3] Maruyama, H., Watanabe, H., Ogino, S., "An Interactive Japanese Parser for Machine Translation," COLING '90

[4] Watanabe, H., "A Model of a Transfer Process Using Combinations of Translation Rules," PRICAI '90

[5] Johnson, D.E., Watanabe, H., "Relational-Grammar-Based Generation in the JETS Japanese-English Machine Translation System," Machine Translation 6, 1991

[6] 長尾「機械翻訳の新たな展開に向けて」人工知能学会誌 Vol.4, No.6, 1989

[7] Maruyama, H., "Constraint Dependency Grammar and Its Weak Generative Capacity," Advances in Software Science and Technology,

---

*) マイクロＣＡＤＡＭ／２はＣＡＤＡＭ社の商標