プログラム言語と自然言語の比較論

浅　井　邦　彦*

A study to compare Programming Languages with Natural Languages

Kunihiko Asai

The way to approach in this argument is the comparison of the programming languages as a good representative of artificial ones, on origin, syntax and semantics.

We can expect that the natural languages used in the present days replaces programming languages on the application of computer, that is to say, our every days language would be available for computer, thereby to contribute largely to the development of information of society.

はじめに

本論文は、代表的な人工言語であるプログラム言語と我々が日常使用している自然言語とを、成立起源、構造論、意味論等の面から比較したものである。

比較してわかったことは、両言語の相違性がきわめて大きいことである。それはコンピュータリーダブルな自然言語処理の困難さを表わすとともに、その研究の重要性も示している。

1．プログラム言語と自然言語の成立の比較

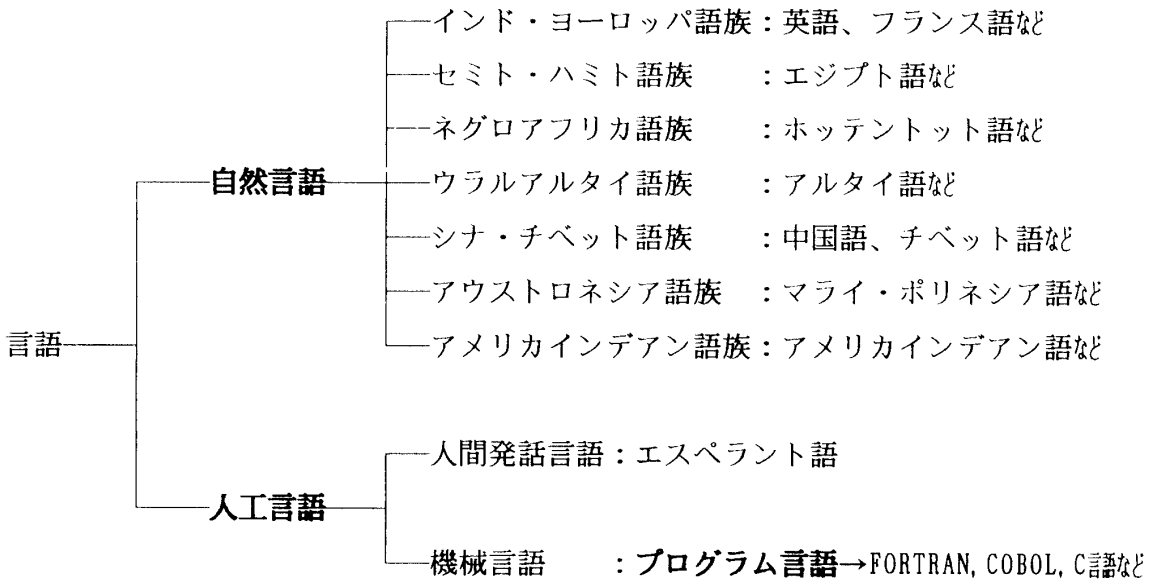
「言葉」を情報伝達の手段として考えたとき、人間以外でも「言葉」を持っていることがわかる。哺乳動物の多くは音声の違いによって仲間に危険などを知らせる。この場合、音声も一種の「言葉」と言える。また、蜂などの昆虫が身ぶりで餌のありかを仲間に知らせることが観察されているが、この場合は身ぶりが「言葉」になる。蜂を殺して潰すと、揮発性の特殊な物質が滲み出るが、これが仲間を殺した敵に対する攻撃指令という「言葉」になる。

このように、多くの生物は「言葉」を持っていると言える。ただし、本稿では「言葉」→言語をそこまで広げないで、「人間の使用と利用の言語」に絞ることにする。

このように限定した範囲という前提で、**自然言語**と**人工言語**に大きく２区分できる。更にそれ

＊経営工学科

それを分類したものが次の表である。(自然言語の分類は佐藤喜代治氏の「国語学要説」朝倉書店 による)



（表１－１）

日本語の帰属については今もって不明である。古代日本語において、すべての音節が母音で終わっているところなどはアウストロネシア語族の特徴とか、身体各部の名称の類似性からはシナ・チベット語族に近いなどとも言われたこともあったが決定的ではない。

日本語は古い時代に、色々な言語が多層的に混合したらしいので、それを解きほぐすのは、日本人自身の起源の探索と同様に、かなり困難なことである。本稿では、日本語の起源についてはこれ以上は触れないことにする。

以下にプログラム言語と自然言語とを比較していく。なお、（表１－１）から分かるように、同一レベルの比較対象という観点からは、人工言語対自然言語ということになるが、コンピュータ対応と人間が日常使用する言語という意味で、プログラム言語と自然言語との比較ということにする。

プログラム言語と自然言語との相違を一言で表現すれば、プログラム言語は「文法先にありき」に対して、自然言語は「使用先にありき」ということになる。近年、各国で国語文法の確定がなされ整備されてきた。わが国でも、国立国語研究所や国語審議会など公の機関で言葉使いの現状調査とその標準化などがはかられてきた。ただし、言葉の標準化であっても、文法が、使用している言葉の後追いであることには変わりない。

別の言葉で言えば、文法に関してプログラム言語は決定論的、自然言語は確率論的であるとも言える。

２．プログラム言語と自然言語における構造論の比較

構造論的な面でプログラム言語を考えてみると、プログラム対象の表現方法は、その自由度が小さく、多様性も少ない。特に、レベルの低いプログラム言語になるほど、表現方法は単一的になる。ＦＯＲＴＲＡＮやＣＯＢＯＬのような高級言語になると、記述の自由度が増す。たとえば、空白の置き方の制限は可なり緩い。次はＦＯＲＴＲＡＮプログラムの１ステップであるが（１）（２）（３）とも文法的に正しく同じ意味である。

（１）　　ＧＯ　ＴＯ　３０

（２）　　ＧＯＴＯ３０

（３）　　ＧＯＴ　Ｏ　３　　０

また、高級言語では文法規定にあってさえいれば、「奇妙な」表現も誤りにならない。次もＦＯＲＴＲＡＮの例である。（この例はコーディング的に望ましいということではない）

ＧＯＴＯ＝ＩＥＮＩ　　（「強盗は家に」というふざけた表現）

この例において、ＧＯＴＯ（実数型変数）もＩＥＮＩ（整数型変数）も、そして「＝」の使い方も文法的には正しい。ＦＯＲＴＲＡＮにとって「＝」はきわめて重要である。その有無によって算術代入文であるかどうかの判定になる。従って、ＣＯＢＯＬでは予約語（Reserved words）として禁止されているＧＯ　ＴＯやＩＦなどもＦＯＲＴＲＡＮでは変数名として使用することが可能となる。このようにＦＯＲＴＲＡＮは高級言語の中でも、相対的に、構造論的自由度は高い。

自然言語の構造論上の自由度を考えてみる。

「私は駅に行きます」「私は駅に行く」「駅に私は行きます」などいろいろと表現できるが、これは構造論的な多様性を表わしているのではない。ではＦＯＲＴＲＡＮ的に

「私　は　駅　に　行　きま　す」

と記したらどうであろうか。もとより、こんな表現は記述上も発話上も許されない。

次に「私は駅に行きます」を「私は駅に行くます」とした場合の誤りを調べる。「ます」（丁寧表現）という助動詞に接続する動詞は連用形に限るのに「行く」は終止形、だから誤りなのである。

自然言語においても、構造上の正さは一意であるといえる。つまり文法的に、あることが正しければ他は全部、誤りである。もっとも、同じ意味の内容をいろいろと表現ができるということとは別問題である。

プログラム言語も自然言語も構造論上では厳しい**きまり**がある。文法が存在しているので当然だとも言える。ただ、プログラム言語は、機械語からアセンブル語、さらにコンパイル語と高級化するにつれて「寛容性」は増すが、自然言語約３０００種について高級かどうかのレベル的分類は考えられないので言語の違いによる構造論的「寛容性」の相違もなさそうである。また、自然言語のなかで比較的に情緒的言語と言われている日本語は構造論的「寛容性」があるかというと必ずしもそうとは言えない。

3．プログラム言語と自然言語における意味論の比較

ＦＯＲＴＲＡＮなどプログラム言語で記述された文章（プログラム）の構造論（文法）的な誤りは、コンパイラやアセンブラなどのシステムが親切にチェックしてくれる。これは、前述したように構造論上の誤りは決定論的だからである。

次はＦＯＲＴＲＡＮプログラムの一部とする。

Ａ＝（Ｂ＋Ｃ＊Ｄ＋Ｅ

多くのコンパイラは以下のようなエラーメッセイジを出してくれるだろう。

Parentheses do not balance.

では次はどうなるだろうか

～

ＧＯ　ＴＯ　３０

Ａ＝（Ｂ＋Ｃ）＊Ｄ＋Ｅ

～

この場合、文法的には正しいからコンパイラは一般的にはエラーメッセイジを出さないが、プログラム上は不備である。無条件飛び越しのＧＯ　ＴＯ（unconditional GO TO)の後に始めて出てくる実行文に文番号が必要だからである。この例の「Ａ＝（Ｂ＋Ｃ）＊Ｄ＋Ｅ」にプログラムは絶対に流れない。「親切」なコンパイラならば、＊などの警告印（warning mark）を出力して注意を促すぐらいである。

コンピュータは意味があるかないかは指摘してくれない。だからこそ、意味論的（そのプログラムの目的との合致性を含んで）なチェックである**デバッグ**を人間が苦労して行なわなくてはならない。

一方、自然言語ではどうだろうか。

「私は**駅**に行きます」この文章のなかの単語「**駅**」を「**豆腐**」に変えて

「私は**豆腐**に行きます」とした時、なんとなく変であるが文法的には正しい文である。「駅」も豆腐も品詞が同じ（名詞）だからである。つまり、構造論上は正しいということになる。

しかし、普通に考えれば、意味をなしていない文章である。意味論上誤りとみなされよう。我々は、経験から豆腐が食べ物であって、そこへ向かって行く目的場所でないことを知っている。

ある文章の内容が正しいかどうかは、人間の脳にある知識データベースとその内容とをマッチングさせて判断した結果による。また、知識データベースは経験の積み重ねによって豊かになる。しかし、自然言語の意味論的正さは、多くの場合、決定論的でなく確率論的である。

「私は豆腐に行きます」が絶対に誤りかというと、そうは断定できない。詩の文章で比喩的に使われたり、小人が豆腐でできた家に遊びに行くという童話のなかの文章かもしれないからである。

自然言語における意味論の処理を考えるとき、コンピュータに比べて、人間がいかに優れてお

り柔軟性に富んでいるかが分かる。

4．プログラム言語と自然言語を修得する時の難易程度の比較

企業でのプログラマの育成を考えてみる。コンピュータ概要を１、２日で終わった後、プログラム言語コースに入る。普通、ＦＯＲＴＲＡＮ４日、ＣＯＢＯＬ５日程度である。ここにはコンピュータ（端末）操作の実習も含んでいる。１日は６時間が標準である。従ってプログラムコースは２４時間から３０時間で終了する。これでプログラム言語をマスターするわけではないが簡単なプログラムの読み書きができるようになる。あとはＯＪＴで修得していく。

筆者の経験によると初めてプログラムを組む時、難解、違和感、不安感をおぼえたものだった。ところが、１つのプログラム言語をマスターしてしまうと他のプログラム言語の修得は簡単で自習も可能になる。

ここで最初のプログラム学習の困難さの原因を考えてみる。この時の難しさは、真の難しさではなく自然言語との距離の隔たりによる。ＦＯＲＴＲＡＮやＣＯＢＯＬのように自然言語（英語数式表現）に近いと言われていても自然言語との距離の隔たりは小さくない。しかし、一方、マスターするまでの時間は比較的短いこと、一つマスターすると別の言語の修得は容易であるなどの特徴もある。この場合の容易さは、自然言語に比べて、「文法先にありき」の文法の規則が無矛盾であり例外がほとんどないこと、および、単語（ステートメント、インストラクション）も極端に少ないことによる。プログラム言語の単語の種類が自然言語に比べて少ないだけでなく、普通に使われる単語の数は限定的である。ＦＯＲＴＲＡＮの場合、３０程度のステートメントで大概のプログラムは組める。コンピュータ要員の苦労はプログラム作成よりもＳＥ（システムエンジニア）としての作業に移っていく。

次に自然言語修得の難しさを考察してみる。

日本人にとってギリシャ語の修得は楽ではない。

			〈発音〉	
単数	１人称	παιδεύσω	[paidéusɔ:]	
	２人称	παιδεύσεις	[paidéuseis]	
	３人称	παιδεύσει	[paidéusei]	
*双数	２人称	παιδεύσετον	[paidéuseton]	*双数とは2つのものが一体となっている概念
	３人称	παιδεύσετον	[paidéuseton]	
複数	１人称	παιδεύσομεν	[paidéusomen]	
	２人称	παιδεύσετε	[paidéusete]	
	３人称	παιδεύσουσι	[paidéusu:si]	

これは「教育する」という動詞の現在形直接法能動相の人称による変化である。ギリシャ語の動詞は、相（３種），法（５種），人称（３種），数（３種），時称（７種）によって様々に変化する。従って、ギリシャ語を学習するのに、文法にこだわると、きわめて効率が悪い。

英語はギリシャ語に比べて、動詞の変化は小さい。educate という動詞を考えてみるとわかる。また、英語の場合、一つの動詞に、その役割をあれもこれも負わせるのではなく、haveやtoなどを使うことで動詞の変化を相対的に小さくしている。

日本語の場合も同様である。「教育する」にしても、本来は「教育」という名詞に「する」という動詞（さ行変格活用）を結合させた複合語である。「教育した」と変化した場合も「た」という過去を表わす別の品詞（助動詞）を接続させたものである。

ギリシャへ行けば、３歳の子供でもギリシャ語を話す。もとより、ギリシャ語文法など知るはずはない。人間は、自然言語をまさに、自然におぼえていくものである。ある言語（外国語）と現在、使用している言語（母国語）との距離（発音、単語、文法仕様など）が離れた言葉を「難しい」と「感じる」。そして、あくまで「感じる」のであって本質的な難しさではない。

「処女等乎袖振山水垣乃久時由念来吾等者」

この漢字だけのフレーズをヘボン（J.C.Hepburn 1815~1912 糊）は十分に理解したという。これは「おとめらを、そでふるやまの、みずがきの、ひさしきときゆ、おもへりわれは」と詠む万葉集の和歌である。誰かがヘボンに、日本語を知るには、まず、中国語を学習すべきである、それは、日本語は中国語の方言である一方、中国語の語順は英語に近いのでマスターしやすいと言ったという。もとより、日本語は中国語の方言ではない。ヘボンは、この誤解にもとづき、まず中国語を学習した。それがため、漢字で構成されている万葉仮名を、あまり抵抗なく受け入れたようである。

自然言語にとって「慣れ」がいかに、重要であるかがわかる。

５．プログラム言語と自然言語における「意図」を表現する方法の比較

「意図」の表現方法について、プログラム言語は寛容性がほとんどない。ただし、寛容性がないということと、同じ目的を持った案件に対して、複数解答があるかどうかとは別問題である。たとえば、ある企業の給与計算をＣＯＢＯＬでプログラムするとき、給与明細書などの出力フォーマットが同じであっても、プログラマが違えば別々のプログラムを作成する。そういう点では、プログラム言語も自然言語と同様である。

ここで言う「寛容性の無さ」は、うっかりミスのチェックや「常識」が通じないことである。もとよりコンパイラは構造上（文法的）の誤りを細かく指摘する。しかしながら、明らかに、おかしいと思われる論理的なミスについては、コンピュータ（ソフトウエアのシステム）は何もしてくれない。だからこそ、前述したように、デバッグが必要なのである。

だから、ＣＯＢＯＬのＩＦ文やＰＥＲＦＯＲＭ文、ＦＯＲＴＲＡＮのＩＦ文やＤＯ文の設定を

間違えたために、無限ループに陥ったり、ループを抜け出した時とんでもない値になることがある。また、ＦＯＲＴＲＡＮにおける整数計算式の項の順序に対して注意が足りなかったがために精度の悪い結果になることがある。(I, J, K, Lを整数型変数とする, I=J/K*L と I=J*L/K のIの値が一致するとは限らない)

自然言語の場合、かなり、いい加減な表現をしても、受け手に発信者の意図を正しく伝えられるものである。のみならず、口頭表現の場合、「えーと」とか「あのー」などの「雑音」が入っていても、聞き手は「雑音」を排除して話し手の「意図」をくみとる。自然言語が持つこのような「寛容性」は、話し手と聞き手との間に、表現された言葉以外の暗黙の情報が共有されているからである。また、読み書きの場合でも、「暗黙の情報の共有」がある程度、認められる。これらの「暗黙の情報」は場面場面で異なってくる「背景情報」でもある。「背景情報」をもとにして、受け手は、発信者の少々の間違いを「自主的」補正する。もっとも、あまり、言外の背景情報に頼りすぎると、誤解が生じたり、「深読み」的な混乱が起きる場合もある。

６．将来の展望――コンピュータによる自然言語理解を目指して

プログラム言語は機械語、アセンブル語、コンパイル語と進歩するにつれて、人間の言葉（英語、数式表現）に形式上、近づき、確かに使いやすくなってきた。

一方、自然言語に関しては、約３０００種もある人間の言葉に共通する規則を見つけようとする努力がチョムスキー（Noam Chomsky 1928～　米国→「普遍文法」）などによって、なされてきた。

とは言え、前項までに述べたように、プログラム言語と自然言語との隔たりはまだまだ大きい。更に、言語処理に附随する事柄として、不特定話者の不特定単語の音声認識や手書き文字の認識、認識の奥にある「理解」など多くの問題が未解決である。

これらの問題の解決には、コンピュータ関連の技術、大脳生理学、言語学、音声学、心理学、哲学などいろいろな分野の一層の交流が必要である。

自然言語とコンピュータとの関連について、大学や民間の機関などで研究され、その成果の上に新しい理論が生まれている。将来、自然言語によるプログラムやデータ入力が可能になれば、コンピュータと人間との距離が更に接近しコンピュータ利用者の裾野の広がりが増すことが期待できる。

参考文献

１）　佐藤喜代治（編）「新版国語学要説」 朝倉書店 １９９３年３月版

２）　田中美智太郎、松平千秋（共著）「ギリシャ語入門」 岩波書店 １９５９年８月版

３）　望月洋子 「ヘボンの生涯と日本語」新潮選集 昭和６２年４月版

（平成10年10月21日受理）