ソフトウェア工学 118－4
（1998．3．10）

命題文における言い廻しの違いの体系化

佐藤 匡正
総合理工学部 島根大学

梗概

本論文では，処理記述の言い廻しの違いを整理する方法について論ずる．プログラム処理を自然語で書くにあたって，表現方法の違いや省略などによって多様な言い廻しが用いられる．処理の理解や伝達を考えると，こうした言い廻しの多様さを，何らかの手段で整理することが望まれる．ここでは，この整理方法として，形式的に体系化する方法を試みる．つまり，フラグを使った命題文に着目して，文を形式化して表現しその基本形を確定させる．言い廻しの違いを，この基本形からの変形であると捉えることによって体系化する方法を考案する．この方法を実際の記述について適用してみて，この妥当性を確かめた．

Systematization of Descriptive Varieties of Proposition Written in the Natural Language

Satou Tadamasa
Interdisciplinary Faculty of Science and Engineering
Shimane University

ABSTRACT

This paper discusses a method to hold descriptive varieties which occurred in writing processes in the natural language. It is expected to establish a method which enables to control such varieties as lead to misunderstandings and/or impede to design machine processing. Here, it is proposed a method that provides to establish a reference form by formalizing proposition description to relate varieties to the reference through formal transformation.

1．序論

本論文では，自然語で書かれた命題文の言い廻しの違いを体系的に把握する方法について論ずる．

自然語での処理記述については，数学的な背景をもつ形式記述に比べて厳密でなく方法としては不十分と指摘されている[1]．しかし，ソフトウェア開発の実務の観点からは逆の見方も成り立つ．一般の技術者には非形式の故になじみ易くその経験によって培われた勘や知識を活用できるという特徴がある．現実的な手法として，こうした特徴を生かした方法は有意義と考えられる．設計や保守における文書化技術としてはもちろん，リバース・エンジニアリングにおける資産活用のための基礎技術となし得る．

こうした技術の確立には，自然語の記述上の特性を知ることが必要である．このひとつとして，言い廻しの多様さがあげられる．プログラム処理を自然語で書くにあたって，表現方法の違いや省略などによって多様な言い廻しが用いられる．処理として記述されている文は，自然語として見た場合，短く簡単である．にもかかわらず，表現の異なる文が用いられることが少なくない．

処理の理解や伝達を考えると，こうした言い廻しの多様さを，何らかの手段で整理することが望まれる．ここでは，この整理方法としの違いがどの程度であるか，どの様に整理すればよいかに着目する．

こうした変形は，処理記述の様々な文においても見られるが，ここではとくに命題の文に着目し，どの様なものであるかを把握する．命題は，処理の条件付けや繰り返しの制御などの規定に用いられていて重要である．また，変形には様々な種類があるが，ここでは文成分が同一でその組立て方に違いのある同成分変形[2]を取り上げる．

こうした命題を整理する手法として，よく知られている述語による方法の応用が考えられる．しかし，この考えによって表現上の変化，つまり変形を把握することには適さない．命題の表現は唯一化されてしまうからである．変形を体系的に表現できる別の方法が必要である．ここでは，言い廻しの違いを表わす方法を工夫し，この方法を用いて変形の体系化を試みる．この方法の妥当性について述べる．

2. 変形の形式化

プログラム処理を記述する文は処理文と命題文に分けられる．このうち，命題文は実行する処理に条件を付すために使われ，処理構成上の重要な要素である．この命題文についてその特徴を述べ，この代表的な文であるフラグ文に着目してこの文の変形の形式化した整理を試みる．このために，まず，ここで用いる用語から始める．

2.1 用語

(1)文法用語

文や文節などといった基本的な文法用語は橋本文法に添う．なお，メタ記号として「」を用いる．

(2)構文の型[2]

文をその成分に分解したとき，分解した成分の構成の違いの分類を構文の型と言う．構文の型は次のように表す．文の成分として存在するものを記号「#」とその格を代表する助詞，つまり主格は「は」，対格は「を」，与格を「に」を付して表す．また，動詞は「する」を付ける．例えば，「パラメータにエラーの出口が指定されている」は「#は#に#する」とする．

(3)変形[2]

変形は相異なる複数の文において意味が同じと判断される文をいう．構文の型から次の三種類がある．

①形式変形：構文の型が同一で，かつその文成分が一致する，

②同成分変形：構文の型は異なるがその全ての文成分が対応する．
ただし，空の文成分も含む．

③言い廻し変形：文成分は異なるが，文の意味が同じ．

2.2 フラグ文の形式化

フラグ文の形式化にあたって，まず，命題文の種類を示し，フラグ文の位置づけを明らかにする．次に，フラグの特徴を示し，この特徴を形式化して書く方法を述べる．

2.2:1 フラグ文の特徴

フラグ文の形式化をするにあたって，フラグ自体の特徴やこれを含む文の特徴を述べる．

(1)フラグ文の位置づけ

命題と命題文におけるフラグ文の位置づけを横観する．命題は，処理の条件付けとして，ある変数の値について書き表わされる．値は，大小などの関係と状態に分けられる．また，書き表す形態は，自然語の文によるものと，便宜的な式によるものがある．更に，命題としての表現上の鍵となる単語の種類は限定されているからこれらを要因とすれば，ひとつの分類ができる．これを文種と言うことにする．命題文の文種は，表１に示す名称を用いて，指定文とかフラグ文と言うことにする．

表１　命題文の分類

対象	形態	表現方法	文種	例　　文
値の関係	文	大小	大小	ﾌｧｲﾙ名は8より大である
		同等 (等しい,同じ ,一致する)	同等	論理ｱﾄﾞﾚｽはSCTの先頭と一致する
	式	不等号	式	a>b,　a<>b
		等号		a=b
状態	文 または 式	量限定 (全て，だけ)	量	SCTの全ｴﾝﾄﾘを見る
		可能	可能	実ｱﾄﾞﾚｽに変換できる
		指定	指定	保護が指定されている
		フラグ	フラグ	SVIのSVC表示ﾋﾞｯﾄがｵﾝである

(2) フラグの役目

プログラム処理の構築においてはフラグがよく用いられる．フラグは，基本的には二値をとる変数で，この二値に存否，有無，生滅といった意味付けを与える．例えば，ある事態の生じたときフラグをオンにしてこの事態の生じたことを記憶させておくと，後でこのフラグを参照すれば，オン，オフの値によってその事態の生じたか否かが判断できる．

このフラグを制御表中の項目として設けておき，様々なモジュールによって設定や参照を行なって制御する．このフラグ値の参照に命題文が用いられる．

(3) フラグの構造

フラグは構造をもたない変数としても，また何個かを組にしたレコード型としても使用される．このフラグを表す用語の別名として，表示やスイッチなどと呼ばれることもある．この多様な用語使いは混乱を招くので整理を試みる．ここでは，フラグを三階層の構造体(Ｃ言語などの)と捉える．最上層にフラグ，標識(インディケータ)を，中位層に表示を，最下層にビット，スイッチを，それぞれ位置づける．この構造によってフラグに関する記述は整理される．例えば，「ＳＶＴのタスク管理インディケータのＳＶＣ表示 ビットはオンである」となる．

この階層化によって，いくつかのビットを束ねられフラグは多値がとれるようになる．

(4)構文の型

構文の型は述語の動詞部分によって決まる．フラグは変数の一種だから，扱いは設定と参照による．命題は参照操作の文であるが，この参照の言い方は設定操作の言い方による．設定の操作では，「オン(オフ)にする」，「立てる(消す)」，「表示する(消す)」と言う．因みに，この「立つ」は起きるではなく，「夢枕に立つ」の「立つ」で，現れるという意味である．この構文の文の成分について考える．

■ 述語と動詞部分

参照では，主として値の状態や存否を言う．状態に対しては，「オンである」「立っている」「設定する」「表示する」が用いられ，また，「表示／設定する」に対しては受け身形も使われる．存否には，「有る」

が用いられる.

■ 層と動詞の関係

フラグ構造における層に対する動詞部分は，層の本来の意味に配慮すれば正統と変則に分けられる．これを表２に示す．表では，縦に動詞(部分)を列挙し，横にフラグの階層を並べ，両者の正統な関係を○で示している．「表示／設定する」には受け身形も含む．この表以外の書き方は変則である．例えば，「ビットが有る」は変則で，「ビットがオンである」か，「表示が有る」が正統である．

■ 構文の型

以上の観察からフラグ文では構文の成分として，フラグの構成，その所在，操作が必要であることが分かる．これを書くための構文の型は，「#は#である」と「#に#は#する」である．これに型名を付す．前者をDE型，後者をSU型ということにする．

表２ 述語の動詞部分とフラグ

フラグ 階層 / 動詞部分	フラグ	表示	ビット
オンである			○
立つ		○	
有る		○	
設定する		○	
表示する	○		

2.2.2 形式化

フラグ文の形式化では，文の成分，変化などが，構文の型におけるより更に詳細に書き分けられねばならない．このために，主部と述部の分離，述部の分離(与格，対格，これらを除いた動詞部分)，フラグ構造などの書き方が必要となる．この書き方について述べる．

(1) 主述分離記号～

記号「～」を使って主部と述部を分ける．この記号の左側部分は主部で，右側部分は述部である．

(2) 動詞部分

■「である」

DE型の動詞部分「である」は空とする．つまり，主述分離記号「～」に含める．SU型の動詞には，「有る(無し)」「立つ(消える)」「表示する(消す)」「設定する(消す)」が考えられる．「有る」は記号「＄」で表す．有ると漢字で書くのは「である」との混乱を避けるためである．「立つ」は，「有る」と同意かつ同構文の文である．

■態

「表示する」は受け身で使われることがある．この区別をするために，動詞の態を添字$_{\vee}$を使うことにする．右下の添字$_{\vee}$は能動態を，右肩の^{V}は受動態を表す．いま，「表示」をＥとすると，Ｅ$_{\vee}$は「表示する」で，Ｅ^{V}は「表示される」である．なお，Ｅ$_{\vee}$では「～」の左辺は対格を表すものとする．これは，文の成分順序を式表現上に保つためである．

(3) 与格分離記号＜，＞

命題では，与格は値の記憶場所の明記に使う．主部と述部を記号「＜」と「＞」で括り，与格はこの外の左側に置く．

(4) 構造名

構造名は，最上位層をＦで，中位層をＥで，最下位層をＢで表す．項目名はギリシャ文字の一字で表す．例えば，「ＳＶＣ表示ビット」において，構造名は「表示」と「ビット」で，項目名は「ＳＶＣ」であるから，これをαで表すと，「αＥＢ」となる．

(5) ビット値

オン，オフなどの値は記号「＊」で表す．文章表現の「設定済み」や「消去済み」はビット値を表すのでこれに含めて考える．

(6) 格助詞「の」

文の成分として，助詞「の」で複合しているものを記号「･」で表して分ける．

(7) 例示

本表記方法の適用を例題によって表す．

［例1］「ＪＦＢフラグにＪＳＩポインタ表示が有る」

β Ｆ＜γ Ｅ～＄＞

ここで，β，γは項目名で，βは「ＪＦＢ」で，γは「ＪＳＩポインタ」である．

［例2］「ＪＢＴにＭＴＴＧＡＰ要が表示されている」

動詞が受動態なので次のようになる．

α＜γ～E^{V}＞

［例3］「ＪＢＴのＭＴＴＧＡＰ要表示がオンである」

α･γ Ｅ～＊

2.3 フラグ文の変形

上の形式を用いて変形を基本形からの変化と捉え，この変化の体系化を計る．このために，文の性質から基本形を確定させ，これに対する主要な変化を生じさせる規則を見つけ出し，整理する．最後に，関連する変形についての観察を述べる．

2.3.1 基本形

構文の型にはDE型とSU型が使われるので，このいずれを基本として考えることもできる．いま，DE型を基本形とする．

さて，DE型において全ての文成分をもつものを考える．つまり，フラグの所在名をα，フラグの項目名をβ，表示項目名をγとする．所在は，DE型では「の」によるフラグの修飾によって，また，SU型では与格によって書き表される．そこで，次の形式となる．

α･β Ｆ･Ｅ～γ　　････式①

この具体例は次となる．

「ＳＶＴエントリのタスク管理インディケータの表示はＳＶＣである」

α　　β　　Ｆ　　Ｅ　　γ

2.3.2 変形の体系化

変形の生ずる要因には，構文の型と文成分の構成の違いがある．まず，構文の型を固定して考える．

(1) DE型における変形規則

基本形の構文の型における変形は次の規則に整理できる．

■規則１：「表示」の移項

…式①のＥが主述分離記号の右辺の末尾に移項する．

α･β Ｆ～γ Ｅ　　････式②

基本形では，表示値について言っているのに対して，この形式では，フラグ値が「表示」であることを表現している言い廻しである．

■規則２：表示項目名の移項

...式①のγが主述分離記号の左辺のＥの直前に移項する．

α・βＦ・γＥＢ～＊　　・・・・式③

基本形の表示値に対して，この形式では，ビット値を表現している言い廻しである．

(2)フラグの与格化

フラグを与格で表現すると構文の型が変わりSU型になる．この変形は次の規則に整理できる．

■規則３：表示関係項の移項

...式②で，フラグの与格化と表示が移項し，右辺は有無が値として残る．

α・βＦ＜γＥ～＄＞　　・・・・式④

この場合，　α・βＦ・γＥ～＄ や　α・βＦ＜γＥＢ～＊＞ も考えられるが，自然語としては舌足らずの言い方なので変則とする．

■規則４：動詞化

...式④で，表示が動詞化して右辺に移項する．

α・βＦ＜γ～Ｅ$_{v}$＞　　・・・・式⑤

この動詞は，過去形表現が普通である．

■規則５：受け身化

...式⑤で，動詞が受け身になる言い廻しである．

α・βＦ＜γ～Ｅ^{v}＞　　・・・・式⑥

この式⑥の言い方は，式⑤よりも自然である．なお，Ｅ^{v}を名詞化した，　α・βＦ＜Ｅ^{v}～γ＞の言い方も考えられるが，正統ではないので考えない．

(3)形式変形

変形の基本形において，αやＥなどの要素が省略されることが想定できる．特に，フラグの所在は省略される．これは，文脈から推定されることが多いためと考えられる．この変形は形式変形である．

第2.2.2節における例について，例１では式①における所在名が，例②では式⑤におけるβＦが，例３では式③におけるβＦとＢが，それぞれ省略されている．

2.3.3 特記事項

(1) 式化

文を簡略化した便宜的な式で書き表すことがある．これは言い回し変形である．具体的な例としては，「･･･表示ビット＝オン」とか，「PROTECT=ＳＶＣ表示」などである．

(2) 指定文の識別

指定文は対象外であるが，指定とフラグは実際には混同して使われることが少なくない．そこで，その類似性を述べておく．例えば，「ファイル･ラベルの指定はＮＳＴである」と，「ファイル･ラベルのＮＳＴ指定はオンである」を考える．前者は指定文であるが，背景の実現方法(how)としてＮＳＴ表示ビットが推定される．後者は，「･･･指定がオンである」と言っているが，これは，「･･･指定の表示」が省略さ

れているのか，動詞が「オン」でなく「有る」かのいずれかと考えられる．前者の立場であればフラグ文形と考えられる．

(3) 変形の複合化

複数の文種を組み合わせた複合文種がある．フラグ文と同等文の組み合わせの「αフラグのＳＶＣ表示はβフラグと同じ」などである．また，「全てのフラグが同じ‥‥」のように，量限定文(全て)の言い廻しが考えられる．こうした変形は明らかであるので対象外とする．

3. 考察

以上の変形の形式化の妥当性を，具体的な事例の観察によって確かめる．この対象及びその結果について述べる．

3.1 観察の対象

(1)文

観察の対象を，フラグ処理を多く含んでいるｵﾍﾟﾚｰﾃｨﾝｸﾞ・ｼｽﾃﾑの処理記述とし，この内の実行管理サブシステムの一部とする．この流れ図に用いられている命題文を対象とする．量はA3判用紙で352枚である．全体4940文のうち命題文の出現頻度は1655文で，33.5%を占める．

(2) 構文の表し方

流れ図の記号で菱型の箱内に記述されている文の，平叙文の形式を命題文として観察する．この文に対する註釈は含めない．

記述されている文には誤りや表現文字(かな/漢字/英字)の違い，動詞の活用などといった違いが含まれている．変形を知るには，本来の表現は保ったままでこうした違いは吸収してしまうように整形する．この整形後に文を解析する．

3.2 変形と出現頻度

表３にフラグ文の出現状況を示す．表では，比較的出現例が多いので文成分の一部を省略した形式

表３　フラグ命題変型の出現頻度

文の表現形式		出現頻度	[%]
基本形	形式変形		
α・βＦ・Ｅ～γ	α・βＦ・Ｅ～γ	9.7	
	α・βＦ～γ	38.7	59.0
	～γ	10.6	
α・βＦ～γＥ	α・βＦ～γＥ	20.0	20.4
	～γＥ	0.4	
α・βＦ・γＥＢ～*	α・βＦ・γＥＢ～*	0.2	
	α・βＦ・γＥ～*	1.6	
	α・βＦ・γＢ～*	1.8	
	α・βＦ～*	2.5	16.5
	γＥＢ～*	1.8	
	γＥ～*	7.9	
	βＦ～*	0.7	
α・βＦ<γＥ～$>	α・βＦ<γＥ～$>	1.8	2.0
	α・βＦ<γＢ～*>	0.2	
α・βＦ<γ～Ｅ'>	γ～Ｅ'	1.8	1.8

変形についても分類してある.

変形は，式①～⑤の五種類の基本形の出現例を得た．この基本形に対する形式変形は，全部で１６種類である．特に，式③に関する変形が出現種類の多い(７種類)点で特異である．他は，要素を多く省略した特定のものに集中している傾向が見られる.

構文の型は，DE型が大部分(96%)である．出現頻度を式番号(第2.3.1節～.2節)で示すと，式①は全体の六割(59.0%)であり，式②が二割(20.4%)，式③が16.5%と続く．これに対してSU型では式④と式⑤が出現しているが，合わせて全体の4%弱である．あまり使われていないといえる.

「設定する」に関しては，「～表示が設定されている」の様な表現も含め，今回は出現例はなかった.

以上をまとめて，ここでの形式化は実態を的確に表しており，変形が特定のパターンに集中している様子がよく分かる.

4. 結論

本論文では，フラグを書き示す命題文についてこの変形を形式的に表現する方法を示した．対象とする変形は同成分変形に分類されるものである.

この方法の妥当性を確かめるために，具体的な事例としてオペレーティング・システムの処理記述に用いられている流れ図のフラグを用いている命題文について観察した．その結果，次が得られ形式化の方法の妥当性が得られた.

すなわち，変形は五種類の基本形が使われている．この基本形に対して，さらに１６種類の細分化した形式変形が使われていて，この内の２種類に対する出現頻度は約６割である．これは，特定の変形形式に集中して出現しているといえる.

参考文献

1. Ian Sommerville:SOFTWARE ENGINEERING 4th ed. [CH 7 Formal Specification], Addison-Wesley Publishing Company (1992)

2. 佐藤　匡正：流れ図文の性質--文の違いに着目した分析，情報処理論文誌 Vol.35 No.5 (1995)

以上