

オブジェクト指向分析記述日本語 OOJ とその記述環境

梶山正行*, 加藤木和夫†, 石井義之‡

概要

ドメインユーザの低いプログラム開発力を向上させるため、自然日本語をベースとしてオブジェクト指向の記述が出来るオブジェクト指向日本語 OOJ を、分析段階の論理モデル用の記述言語として考案・開発した。/ OOJ は自然日本語の上位互換として、強い構造記述性を持つオブジェクト指向の枠組みを記述モデルと構文法の形で導入して構成された。また、オブジェクト指向記述環境も設計・実装した。記述例を作成すると共に数人のドメインユーザに記述実験を行った。その結果、自然言語の持つ強い記述力をそのまま維持しつつ、オブジェクト指向の構造化記述による記述力の高さに対する純向上が認められた。プログラミング言語フリー、自然言語の範疇であることから来る記述の容易性・高い理解性、日本語一貫記述言語系の一環であることのメリットが認められ、重大な欠点は見出せなかった。OOJ がドメインユーザにとって充分有用な分析記述言語であることが結論された。

Object-Oriented Japanese OOJ for Analysis Descriptions and its Description Environment

Masayuki Hatakeyama*, Katougi Kazuo† and Yoshiyuki Ishii‡

ABSTRACT

To improve the system (program) development ability of the domain users in various fields, we have developed a new description language called the OOJ (Object-Oriented Japanese) based on the native mother language (Japanese) in the analysis modeling processes. We have introduced and designed a new set of the description model and the syntax/grammar methodology as a description framework with the aid of the Object-oriented Description Environment. Some typical description examples have been made and evaluated by some domain users. The comparisons with other analysis description languages are performed mainly with the UML. As the conclusion, we have confirmed that the OOJ is one of the most useful and valid description language.

1 はじめに

プログラムを自身でよく作り・しばしば改訂しながら使うユーザに、「ドメインユーザ」*と呼ばれる代表的なユーザがある。彼らは自身の専門(流体や構造物の解析やシミュレーション、画像分析、建築設計等々)には深い知識と興味を持

つが、計算機システムは単に結果のデータ(数値)を出す“道具”に過ぎず、必要以上に“道具”に興味を持たない。むしろ、プログラミング言語(以下頻出するので PL と略す)は Fortran のみで新規な言語習得は避ける、エディタとコンパイラ、可視化ツール程度を動かせるだけ、対象ドメイン(専門分野)に対する分析記述やプログラムの設計経験はない、というドメインユーザも少なくない。この様なドメインユーザは個人或いは 2~3 人で、数百行から数千行程度のプログラムを作ることが殆どである。

しかしドメインユーザが作るプログラムは、膨大な種類の分野に多様な利用法を駆使し、作成プログラムの総量は膨大な量に上り、また、彼

*茨城大学工学部情報工学科

Ibaraki University

†日立プロセスコンピュータエンジニアリング(株)

Hitachi Process Computer Engineering Co. Ltd.

‡日本電気株式会社

NEC Corporation

*本論文では、各専門分野(例えば科学技術計算分野)の研究開発に計算機を道具として Fortran 等を用いて利用するエンドユーザの一種 [1].

らの計算結果のもたらす成果は科学技術の発展に大きな影響を及ぼす。それにも関わらず、彼ら自身はプログラム（ソフトウェア）の開発力は一般に低い。その様な状況の解決のため、生来 (native) の主用言語である母国語（本論文では自然日本語）の範疇で新規な記述言語（詳細は次節）を再構成した [1, 2, 3]。その様な狙いで前論文 [1] では自然日本語、OOJ、日本語設計記述言語 JSMDL、の複数種の自然日本語の記述言語のシリーズを提案したが、分析モデリング過程の日本語記述言語である OOJ の構成や機能には以下の問題点が残された。

1. 自然言語記述で特に構造記述法が不足。
2. 文法、記述法、変換法が殆ど無い。
3. 記述支援環境が不足。

そこで本論文では OOJ を改めて設計し直して再構築する。

2 オブジェクト指向記述言語の設計方針

まえがきでの記述言語の議論、JOMON[1, 2]の構築経験、オブジェクト指向 PL を参考にし、自然言語ベースのオブジェクト指向記述言語[†]という方針で、以下を決定した。

1. オブジェクト指向の PL-free な記述言語
2. 自然日本語の範疇の記述言語
3. オブジェクト指向記述支援環境

3 オブジェクト指向記述モデル

オブジェクト指向に基づくシングモデル[‡]の構成要素の一般形を表 1 の様に提案する。この記

[†]特定の記述方式（本論文においてはオブジェクト指向記述方式）実現のために、或いは、特定対象の記述のために、自然言語に対して新規な構文・文法・記述形式・用法というある種の「枠組み」を導入した言語。

[‡]モデル化単位”もの”の、論理モデリング段階での呼び名

表 1: 記述モデル構成

<p>1. シング名</p> <p>2. 属性</p> <p>2.1 単独属性（－他のシングや属性と直接相互関係を持たない属性）</p> <p>記述構成： （属性名，属性値，作用振る舞い名，属性種，制約）</p> <p>2.2 相互関係属性</p> <p>－相互関連属性と相互作用属性から成る</p> <p>2.2.1 相互関連属性：</p> <p>－汎化，集約，集合，一般関連の4種あり</p> <p>記述構成： 1) 汎化：（上位シング，下位シングリスト） 2) 集約：（上位シング，下位シングリスト） 3) 集合：（所属シング） 4) 一般関連： 記述構成： （相互関連属性名，相互関連相手シング名リスト）</p> <p>2.2.2 相互作用属性</p> <p>記述構成： （相互作用相手シング名，相互作用抽象名，相互関係するシング名リスト，制約）</p> <p>3. 振る舞い</p> <p>3.1 内部振る舞い</p> <p>記述構成： （内部振る舞い名，振る舞い記述文（OOJ 単文，複数可），相互関係シング名，制約）</p> <p>3.2 相互作用振る舞い</p> <p>－相互作用の起動手順・手続きを記述</p> <p>記述構成： （相互作用抽象名，振る舞い記述文（OOJ 単文，複数可），相互関係シング名，制約）</p>

述モデルの特徴は相互関連記述 (2.2.1) や相互作用記述 (2.2.2) を属性として扱うこととした点であり、その他は通常よく使われるオブジェクトモデルである [5]。

相互作用属性 (2.2.2) とはシング間の相互作用における相互関連（構造）との関わりを、複数のシングとの相互関連を記述した相互関連スロットと相互作用スロットとのセットという形で記述したものである。

表 2: シング及び属性の定義

1.OOJ 記述は<シング定義>の並びである.

```

<シング定義> ::= "# シング : " <シング名>
<説明> [ <属性記述> ] [ <振り舞い記述> ]
<シング名> ::= "自然日本語名詞"
<説明> ::= "%説明 : " <日本語単文>
               { <日本語単文> }
<日本語単文> ::= "自然日本語単文" . "
    
```

2. <属性記述> ::=

```

{ <単独属性> } { <相互関係属性> }
    
```

2.1 <単独属性> ::=

```

"%単独属性:"
  <属性名> [ "(=" <属性値>")" ]
  <作用振る舞い名> <属性種> [ <制約> ]
<属性種> ::= ("固有" |
  "%特定公開 (" <特定シングリスト> ")" |
  "%公開" |
  "%特殊作用制約" )
<作用振る舞い名> ::=
  "%作用振る舞い名 : " <内部振る舞い名>
<制約> ::= "%制約:" <日本語単文>
               { <日本語単文> }
    
```

表 3: 相互関係属性記述構文

2.2 <相互関係属性> ::=

```

{ <相互関係属性> } { <相互作用属性> }
2.2.1 <相互関係属性> ::= "%相互関係属性:"
  (<汎化> | <集約> | <集合> | <一般関連>)
<汎化> ::= "%汎化 : "
  [ "%上位シング名 : " <上位シング名> ]
  [ "%下位シング名 : " <下位シング名リスト> ]
<集約> ::= "%集約 : "
  [ "%上位シング名 : " <上位シング名> ]
  [ "%下位シング名 : " <下位シング名リスト> ]
<集合> ::= "%集合 : " <構成シング名リスト>
<一般関連> ::=
  ("一般関連 : " <相互関係属性名>
  "%関連先 : " <相互関係相手シング名リスト>)
2.2.2 <相互作用属性> ::=
  "%相互作用属性 : " <相手先シング名>
  <相互作用抽象名> [ <特別関係記述> ]
  [ <制約> ]
<相手先シング名> ::= "%相手先シング名 : "
  <シング名>
<相互作用抽象名> ::= "%相互作用名 : "
  <相互作用名>
<特別関係記述> ::= "%関係シング名 : "
  <シング名>
    
```

4 OOJ 構文法の設計

設計にはよく使われる BNF 記法の構文定義を必要に応じて拡張して利用し、前節で提案したオブジェクト指向モデルに沿ってそのモデルの記述法を表 2, 3, 4 の様に定義した。

構文記号

1. <> は非終端記号を表す。
2. "xyz" は終端記号を表す。
3. ::= は書き換え記号で、左辺が右辺で定義される (書き換えられる) ことを表す。
4. {A} は A が 0 個以上繰り返されることを示す。
5. [] は省略可能を示す。
6. | はどちらかを選択することを表す。
7. () はグループ化された記述単位を示す。
8. #, %, \$ は各々シング, 属性, 振り舞いのキーワードを表す。
9. "C : " は、ユーザのコメントを示す。

(2.2.2) <相互作用属性> は相互作用抽象名, 相互作用の相手先シング名の二者の関係を記述

し、構造部分 (シング名とそれらの相互関係) と相互作用に伴って起こる振り舞いの関係を記述する。相互作用を起動させる手順の記述は相互作用抽象名として名前のみをここに記述し、実際の手順及びその詳細記述は (3.2) で行う。

OOJ 変換構文法

OOJ 変換構文法とは、概念モデルの自然言語記述を論理モデル記述の OOJ 構文内に記述し終わるまでの構文の「変換」規則を定めたものである。この構文法は複文を OOJ の定める文法に従う単文に分解し、各単文の振り舞いに関する記述以外の構成各要素を OOJ 構文の各要素へと識別・変換・仕分け (振り分け) を行うための変換規則を指す。

5 OOJ 文法・OOJ 変換文法

OOJ は自然言語記述からの変換や、対象世界から新規に抽出・補充記述をされて OOJ 記述にする関係上、最終記述形が満たすべき通常の

表 4: 振る舞い記述構文

<p>3. <振る舞い記述> ::=</p> <p>{<内部振る舞い>} {<相互作用>}</p> <p>3.1 <内部振る舞い> ::= "\$ 内部振る舞い:"</p> <p><内部振る舞い名> <振る舞い記述></p> <p><振る舞い記述> ::=</p> <p>" \$ 振る舞い記述:" {<振る舞い記述文>}</p> <p><振る舞い記述文> ::=</p> <p>(<OOJ 単文>—<選択文>—</p> <p><繰り返し文> <多方向分岐文>)</p> <p><OOJ 単文> ::=</p> <p><OOJ 文法に従う日本語単文 >,"</p> <p><選択文> ::= "IF" <条件></p> <p>"THEN" {<振る舞い記述文>}</p> <p>"ELSE" {<振る舞い記述文>}</p> <p>"ENDIF"</p> <p><繰り返し文> ::= "LOOP" <条件></p> <p>"DO" {<振る舞い記述文>}</p> <p>"ENDLOOP"</p> <p><多方向分岐文> ::= "多方向分岐"</p> <p><条件> { "ケース:" {<振る舞い記述文>} }</p> <p><条件> ::= <OOJ 単文></p> <p>{ ("AND" "OR") <OOJ 単文> }</p> <p>3.2 <相互作用> ::= "\$ 相互作用:"</p> <p><相互作用名> <振る舞い記述文></p> <p>[<相互関係相手シング名></p> <p><相互関係属性名>] [<制約>]</p> <p>(非終端記号を用いた階層構造記述形式については省略した)</p>

「OOJ 文法」と、自然言語記述からの変換規則を定めた「OOJ 変換文法」の二つを定める。

OOJ 文法は主として論理モデリング段階の自然言語記述の各単文に対する記述法を定める。

OOJ 変換文法は論理的で一意性を持つ基本文型を持つ平述文の単文の接続への変換規則、及び記述補充規則を示したものである。

6 OOJ の記述環境の設計

OOJ 記述の処理手順・順序を4つのステップに区分して構成し、設計・実装した [3, 4].

(1) 対象識別・仕分けステップ

自然言語記述の概念モデル記述に対して、まずシングを識別・抽出し、その属性・振る舞い、シング間の相互関係を識別・抽出・記述する。未抽出シングを探索・抽出して新規に定義を行う。

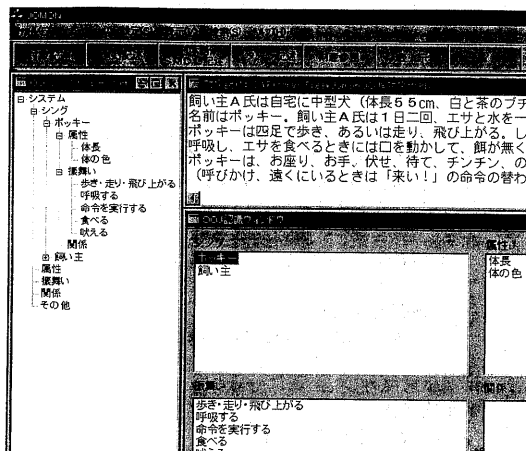


図 1: 対象識別・仕分けステップの記述環境画面

図 1 にその例の画面を示す。

(2) 構造記述・相互関係記述ステップ

識別・仕分けされた全ての構造記述要素である属性の詳細記述や相互関係の記述の追加・補完・訂正を行う。原則として構造に関係する全ての要素を揃える。

(3) 振る舞い記述ステップ

シング間の相互関係を基にその振る舞いや相互作用の起動・駆動順序の整合性や合理性をチェックする。各振る舞い記述の参照属性との対応関連とその参照処理も再確認する。振る舞いの詳細記述を行う。

(4) レビューステップ

JSM DL [1, 2] 移行への最終記述の完成のため、モデル記述全体についての相互関係などを再検討・探索・重複チェックを行う。

記述エディタの設計と実装

全ての実作業はドメインユーザ自身で手動記述・変換で実現するので、記述エディタではオブジェクト指向構造化記述形式の枠組みを分類・提示・表示して記述を要求し・誘導し・ガイドする以下の条件を実現した記述環境を設計・実装した。満たすべき条件は以下の様である。

- 振り分け・仕分けのためのドラッグ (コピー) & ドロップ (ペースト) 機能。
- シング単位及び記述要素の追加・削除・訂正機能。

表 5: 自然言語記述の概念モデル例

飼主A氏は自宅に中型犬を飼っている。それを見ると体長は55cm, 体色は茶色と白のブチで名前はポッキーである。飼主A氏は1日二回エサと水を一定量ずつ与える。ポッキーは四足で歩き, あるいは走り, 飛び上がる。しっぽを振り, 肺で呼吸し, エサを食べるときには口を動かして, エサが無くなるまで食べる。ポッキーは, お座り, お手, 伏せ, 待て, チンチン, その他, 「ポッキー!」(呼びかけ, 遠くにいるときは「来い!」の命令の替わり), 「ほーい」(投げたボールを取ってこい。)が出来る。ポッキーは首輪を付けている。室内では放し飼いで散歩に出るときにはひもを付ける。

表 6: OOJ シング及び属性記述例

```
#シング:ポッキー %説明:犬, 雑種
%単独属性:体の色 (=白と茶色のブチ)
%振る舞い名:見る %公開
%単独属性:体長 (=55cm)
%振る舞い名:見る %公開

%相互関連属性:
%一般関連:巻き付き %関連先:首輪
%説明:ポッキーの首には首輪が付いている
%相互関連属性:
%一般関連属性:常時接触 %関連先:空気
%説明:肺呼吸の必要上, 常に空気に接触.
%相互関連属性:
%集約:
%下位シング名:四本の足, 口, 肺, しっぽ
%説明:ポッキーは四本の足, 口, 肺, しっぽから成る.
```

3. 属性や振る舞いの関係先候補の表示や画面間移動機能.
4. 振る舞い記述の形式的な文法チェック機能.
5. レビューのための一覧表機能と, 相互関係チェックのためのツリー表示機能.
6. シング, 属性, 振る舞い間の参照・更新の関係や相互関連の表示・チェック機能.

7 OOJ の記述例とその評価

本論文では一般的で誰でも分かる飼い犬の散歩のシミュレーションを取り上げた。表5に自然日本語を用いた概念モデル記述を示し, 表6,

表 7: OOJ 相互作用属性の記述例

```
height %相互作用属性:
%相手シング名:地面
%相互作用名:歩き・走り・飛び上がる.
%関連シング名:四足の足
%説明:ポッキーは四足を使って歩き, 走り, 飛び上がる.
%制約:3日間餌が貰えないと筋肉は動かなくなる.

%相互作用属性:
%相手先シング名:空気
%相互作用名:呼吸する
%関連シング名:肺
%制約:肺呼吸

%相互作用属性:
%相手先シング名:餌
%相互作用名:エサを食べる
%関連シング名:口
%説明:エサに近づいたら口を開け, エサに触れたら口を締める. (判断機能やセンシング機能は省略. エサがある限り無限に食べ続ける.)
%制約:無し.

%相互作用属性:
%相手先シング名:飼い主
%相互作用名:命令を実行する
%関連シング名:四本の足
.....
```

表7, 表8にそのOOJへの変換及び追加記述の結果の例を示す。

8 評価と考察

8.1 記述実験と記述例の評価

記述実験は, 流体の数値解析を自身のドメイン(専門分野)とする三人の大学院学生, 及び情報工学専攻の大学院学生数人を対象にして実施された。

前章の論理モデル記述である OOJ 記述例と概念モデル記述例を比べると, 記述の正確さと完備さの程度は OOJ の方が当然優れており, 記述環境の提示とガイドの支援を受けることで必要記述要素を「ほぼ記述している(=記述の完備性が比較的高い)」と言えよう。

記述例から評価した OOJ の特性は, 記述し易さ, 理解のし易さ, 記述要素のより高い完備

表 8: OOJ 相互作用記述例

<p>\$相互作用：歩く・走る・飛び上がる。 \$振る舞い記述： 多方向分岐（動作の種類） ケース（歩く）：歩くの振る舞い記述 ケース（走る）：走るの振る舞い記述 ケース（飛び上がる）：……</p> <p>\$相互作用：呼吸する \$振る舞い記述： 肺を膨らませて空気を吸い込む。 酸素と炭酸ガスを交換する。 肺を縮めて空気を排出する。</p> <p>\$相互作用：エサを食べる \$振る舞い記述： 口を移動させ、エサに接触する。 LOOP {エサが無くなるまで} DO 口をあげエサを口に入れる。 口の中にエサが入ったら 口をきつく閉める。 エサをかみ砕いて飲み込む。</p> <p>ENDLOOP \$相互作用：飼い主の命令を実行する。 (省略)</p>

さ、等の全ての面で OOJ は優れた特性を見出せており、大きな欠点は見出せない。この特性はまえがきにも述べたソフトウェア専門家との間での議論や考察を共有できる意志疎通機能の高い言語であることも判明した。

8.2 構文・文法、変換構文・変換文法、記述モデルの評価

OOJ 構文法

OOJ の構文法は拡張 BNF 風の表現形式としたが、キーワードが日本語の単語であることもあって、馴れるのに支障は殆ど無かった。これは記述実験被験者のコメントでもある。従って、例題を提示して対応を明示しながら口頭での説明を併用すれば、(オブジェクト指向を「概念的には」理解しているドメインユーザが前提なので) 約 1~2 時間ほどで被験者のほぼ全員が十分な理解に達し、オブジェクト指向への理解も深まり、1 日以内には、例となる対象世界を試行錯誤を繰り返せば十分に記述可能になった。習得に必要な時間は PL の習得時間に比べれば無視できる程度に短く、違和感もなかつ易しい。

構文に従った記述の困難さ・煩雑さなどは被験ユーザからは聞かれなかった。

以上から OOJ 構文が記述モデルを十分記述できる事を指摘でき、記述例からも充分確認でき、更に概念モデル記述への逆変換が易々と可能であることをから、OOJ 構文法は基本機能としては十分な OOJ 記述を完成可能な構文法であることが結論できる。

OOJ 文法

OOJ 文法そのものは、基本文型に沿った単文の平述文の接続にすることがその内容の殆どであり、振る舞い記述に自然言語以外の OOJ 特有の特殊な文法は規定されていない。僅かな記述制約も十分に緩い。従ってその結果として自然言語の記述力は全く保持されたままであり、しかも一意に規定出来る単文の接続群が得られる。

OOJ 変換構文・変換文法

変換構文法と変換文法は簡潔で一意的な日本語へと変換することがその趣旨であり、これらの設定や規定も問題とすべき点は見出せなかった。両者の殆どは記述環境の設計と実装に反映されており、自然と利用・具現化する形になるように仕組みられている。

記述モデル

オブジェクト指向記述モデルとしての構成については、抽象的には十分な要素が用意されており、大きな不足や過剰な仕様は見出せず、記述に支障は無いことがユーザの記述例から明らかにされた。記述モデルに関連や相互作用を属性の一部として記述するように規定した方法も問題は見出せなかった。ただし、振る舞いの詳細を記述するための特別なモデルが欲しいとのコメントもあった。

8.3 特定ドメイン用の振る舞いの記述

各被験ドメインユーザの従来からの自身の記述を OOJ で記述し直すことにより、それを実行したドメインユーザ自身の専門分野の分析記述力がほぼ全員顕著に向上するという結果を得た。その結果、各ドメインユーザは各ドメインの詳細かつ膨大な概念体系や知識を殆ど制約無しで自在にかつ直接的にオブジェクト指向に基づい

て記述できる可能性が大きく広がったことは確かであり、被験ユーザ達もそれを高く評価した。

ただし、本論文で決めた OOJ 構文はいわば、共通基盤言語としてのオブジェクト指向の自然日本語を規定したものであり、各個別ドメイン用の（特に振る舞いに対する）記述構文や文法はその共通基盤言語の上に立って、個々の分野対応で設計・実装する必要がある。

8.4 オブジェクト指向一貫記述日本語系の一環としての OOJ の考察

OOJ の弱点を強いて取り出せば、唯一、計算機可読ではなくそのままでは直接実行不可能という点である。しかし、この弱点も JSMDL 及びオブジェクト指向言語へのトランスレータを加えた記述言語の”系” OODL[1, 2] で見ると、

1. 自然日本語（PL フリー）に近い記述を実現。
2. PL 記述への変換は JSMDL トランスレータ [1, 2] が自動変換を支援。

という言語系 OODL で見ればドメインユーザの理想に近い言語系であり、その一翼を担う OOJ を前節までに述べた特長を併せて評価すれば、上記の弱点を補って余りある記述言語であると言える。

8.5 仕様記述言語との比較

ソフトウェアの仕様記述には自然日本語がよく用いられるが、自然言語では曖昧さが生じることから、LOTOS[7]、Z 言語 [8] といった仕様記述言語が開発された。比較すべき仕様記述言語に共通の特徴として以下がある。

1. 当初から厳密で形式的な仕様定義記述要
2. PL への変換は全てユーザ自身

これらの両点についてはドメインユーザの実用記述には全くメリットはない。結論としては、仕様記述言語は本研究の目的には合わない言語である。

8.6 UML との比較

UML[6] を分析段階のオブジェクト指向記述言語であると考えて比較検討する。我々自身及びドメインユーザの被験者の、UML と OOJ 両方法での記述経験から UML の比較すべき特徴をドメインユーザの側からみる。まず第 1 点は PL 的表記法が見られる点は UML が不利な点である。

第 2 点は多様なダイアグラム表記法を持っていることについてのドメインユーザの評価は高い。そこで OOJ においても用語を全て日本語とし補助的な併用表現手段として利用することにした。トータルとして OOJ がより使い易くなり、この点で OOJ の表現力が UML のそれに近づいた。

第 3 は多様で完成度の高い機能である。即ち UML の記述モデルの多様さや記述環境の完成度においては現状 UML の方が高い。機能モデル (OMT)、動的モデル (OMT, UML) 等もそうである。またデザインパターンとのつながり、各種の支援記述環境等も商品化されて提供されている。分析記述力・表現力は現状では UML の方が OOJ よりも高い。

両言語の共通点は、オブジェクト指向に基づいている点である。各々の特徴は

- A OOJ が母国語の自然言語ベースの記述言語であり、それに由来する多くの特長を持つこと。
- B UML がダイアグラム記法を主体とした多様な記述形式と、完成度が高く詳細な言語仕様や記述環境。

その他の点を比較して挙げると、

- PL フリーという点では OOJ が有利。
- 詳細で豊かなダイアグラム表現法という点では現状で UML が有利。
- Native な自然言語仕様という点で OOJ が有利。
- 記述環境は UML の方が優れている。
- 習得にかかる負担という点で OOJ が有利。
- 成熟度・普及度という点で UML が圧倒的。

- UMLはある意味で完成システム, OOJは未だ試作システムで拡張の余地が大きい.

そこで, OOJの記述機能向上案として, 機能モデル/動的モデル, 及び対応する構文・文法の追加設計を行うと共に, 多様なダイアグラム表現法を標準補助表記法として採り入れることでOOJには基本的に無いUMLの特長を採り入れる. この追加の方法の採用は複雑な振り舞いをOOJ記述の「機能向上法として」大きく役立つであろう.

9 結論と今後の課題

自然日本語に対してオブジェクト指向構造化構文法の枠組みを付加して, 記述力の高さの純然たる向上を実現した記述言語を設計した. その様な記述言語の導入の結果, ドメインユーザが分析モデリング過程において十分なオブジェクト指向記述を可能にする記述の枠組みが構成され, 充分実用的な分析記述言語が確立された.

記述例及び記述実験の被験者のドメインユーザの評価として, 構文法の理解の容易さ, 記述力の高さの純向上が容易に得られること, 必要記述要素数のかなり高い完備さ, PLフリーであることの親近感, 等の高い評価が得られた. UMLとの詳細な比較も行われ, 改良案も提案された.

今後の課題としては, 動的モデル/機能モデルを導入しての複雑な階層構造の内部振り舞いや相互作用振り舞いを記述する構文や記述の手順・起動手順の開発, UMLのダイアグラム表記法の導入による高い記述力の強化, 記述環境の充実, 構文や文法の詳細化・充実・拡張等が挙げられる.

参考文献

- [1] 加藤木和夫, 畠山正行, オブジェクト指向日本語一貫プログラミング環境, 情報処理学会論文誌, Vol.40, No.7, pp.3016-3030 (1999).
- [2] 加藤木和夫, 畠山正行, オブジェクト指向日本語一貫プログラミング環境, 情報処理学会第118回ソフトウェア工学研究会研究報告, pp.15-22, 98-SE-118, 1998年3月10日.

- [3] 石井義之, 畠山正行, 加藤木和夫, オブジェクト指向日本語分析記述環境 OOJ の設計と開発, 情報処理学会第122回ソフトウェア工学研究会研究報告, 99-122, 55/62 (1999).
- [4] 石井義之, オブジェクト指向日本語分析記述環境 OOJ の開発, 茨城大学大学院博士前期課程情報工学専攻学位論文, 平成11年2月.
- [5] 本位田真一, 青山幹雄, 深澤良彰, 中谷多哉子, オブジェクト指向システム分析・設計”, 共立出版 (1995).
- [6] H.E., Erickson, and Mugnus Penker, UML Toolkit, John Wiley and Sons, 1998.
- [7] 高橋薫, 神長裕明, 仕様記述言語 LOTOS, カットシステム (1995).
- [8] B.Potter, et.al., An Introduction to Formal Specification and Z, Prentice Hall (1991).

A 用語定義

- 相互関連**: 複数要素間の静的な(時間の経過により変化しない固定的な)関わり.
- 相互作用**: 複数要素間の動的な(時間の経過に従って変化する)関わり.
- 相互関係**: 相互関連と相互作用をそのサブセットとする統合化概念. 属性の一種.
- 構造**: 複数の記述要素とそれらの間の相互関連記述の組.