Vol.2010-MPS-78 No.16
2010/5/21

日本語を主用したオブジェクト指向分析記述言語 OONJ の設計主題とその記述法

畠山正行[†1]　池田陽祐[†2]
三塚恵嗣[†3]　加藤木和夫[†4]

オブジェクト指向 (OO) に基づき日本語を用いた記述方法として OO 構造化フレーム (OOSF) が提案された．しかしそれ単独では表現法や記述力が不足であり，OO 記述言語 OONJ へと発展的な設計を行った．そのため，まずは OO 記述用の要素種類を設計し，対象全世界から最小離散単位までの要素を一意特定的に特定できる OOSF の仕組みを大幅に拡張し，初期/境界条件やシナリオ，共有属性等を十分詳細に記述する方法や表現法を提案した．OOSF の最小離散単位の記述の実体は日本語 (日本語の単文) により行う．その記述実体である日本語 (日本語の単文) を分析し OO 構造化表現法を設計した．これ等を総合して OO 記述言語 OONJ という形式仕様に組み込んだ．その結果，OONJ は記述例からのデータも援用すると，当初の狙い通り OOSF よりも強い記述力/表現力を持ち，日本語記述言語や DEQSOL 等やとの比較の結果も OONJ が良好な記述力特性を示した．

A Primary Theme of Design and its Description Method for the Object-oriented Analysis Description Language OONJ based on the Japanese

Masayuki Hatakeyama,[†1] Yousuke Ikeda,[†2]
Keishi Mitsuka[†3] and Kazuo Katougi[†4]

An Object-oriented (OO) structured frame OOSF hes been proposed and developed for the OO descriptions of the various target worlds. The OOSF has, though, not enough power for the target world expressions. Then, the aim of the present paper is to design and provide an Object-oriented (OO) description language OONJ that has been extended out of the OOSF. Therefore, we have designed the OO element kinds, and extended the OOSF mechanism so as to specify uniquely the every OO description elements from the whole target world up to the minimum discrete unit. And also, the initial/boundary conditions, the scenario, the whole world descriptions, and the common attribute frame have newly been applied and designed. The tangible description of the individual "minimum discrete unit" are described using the natural Japanese. Then, the "natural Japanese unit sentence" in the minimum discrete unit has also been structured. All these elements have been built in into the formal framework of the OONJ. The results have shown that the description power of the OONJ is far stronger than those of the OOSF, and the comparisons with other languages or the development environments DEQSOL are rather favorable. The OONJ has successfully been constructed.

1. はじめに

我々の研究グループでは，以前からオブジェクト指向パラダイム (Object Oriented，以降 OO と略) に基づく記述言語系 OONJ[1)]，ODDJ[2)]，OPDJ[3)] そしてそれらを統合化した OOJ[4),5)] と呼ぶ記述言語系を研究開発して来ている．それらの研究開発は**図 1** に示すように対象世界を OO パラダイムに基づいて OONJ を用いて分析記述し，その記述を設計段階の ODDJ 記述に変換して計算機世界に持ち込み，OPDJ で特定の OO プログラム言語 (以降，**OOPL**) のプログラムを生成して実行する，というシステムである．

しかしこれまでに開発した OONJ，ODDJ，OPDJ，そして OOJ は結果的には半ば先行試作システムであり，本来の主題を実現するための準備としてその設計データ取得に終始したと言っても良い．何故ならば，開発の目的の 1 つであるプログラムの生成に至る過程内部に内在する様々な問題も解決に手間取り，三つの記述言語と一つの OOPL 間の半自動変換やプログラムの自動生成には不十分な成果しか得られていないからである．

しかし一方で，完全な仕様の実現ではなくとも，わずかに手作業での記述も加わっているという現状ながら，プログラムの自動生成にまで漕ぎ着けた経験/経緯があり，完全ではないにしろ，一度はフィードバックされた設計指針などがある．

以上の経緯を考慮に入れて，本論文では，これ等の一連の研究開発を行ってきたそもそも

†1 茨城大学工学部情報工学科
Department of Computer and Information Sciences, Ibaraki University
†2 茨城大学大学院理工学研究科情報・システム科学専攻
Graduate School of Science and Engineering, Ibaraki University
†3 茨城大学大学院理工学研究科情報工学専攻
Graduate School of Science and Engineering, Ibaraki University
†4 茨城県立産業技術短期大学
Ibaraki Prefectural Industrial Technology Junior College

Vol.2010-MPS-78 No.16
2010/5/21

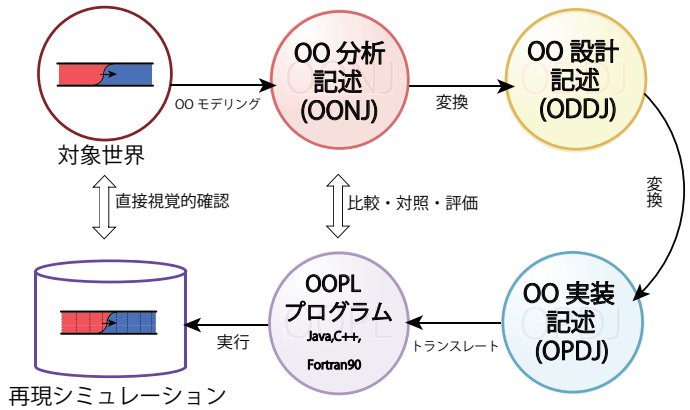

図 1 OO 分析・設計・実装・プログラム・シミュレーションの過程

の発端と理由，そして解決すべき問題，そしてその問題解決のための主題[⋆1]から説き起こすことから始めたい．

以上から本論文では，第 2 章では今までの経緯を考慮に入れた上での OONJ の設計主題を導くと共に，OO パラダイムの採用や記述言語の導入の経緯を述べる．第 3 章で OO の基礎から OO 記述を実現するための OOSF の仕組みを簡潔に述べ，OONJ の設計主題を具体的に導く．第 4 章では OONJ の主要な構成と記述法を述べ，第 5 章では OOSF の内部の最小記述要素の表記法である日本語文の記述法とその変換法についてその概略を述べる．第 6 章は OONJ の構築結果とその評価・考察である．第 7 章は結論と今後の展望である．

2. OONJ 設計主題

2.1 研究開発の発端と解決のアプローチ

そのそもそもの発端は著者の一人が専門とする自然現象の一つである流体の流れの問題において従来は知られていない現象がシミュレーション結果として出現し，そのような現象があり得るのか否か，そしてその現象があるものと仮定した条件下でシミュレーション結果がどの程度正確なのか否か，についての解決が迫られたことにある．

⋆1 問題解決のために何をどう実現すれば良いのかの考え方と系統的な実現方法のセットを指す．

このような問題に対しては，流体の流れという自然現象を対象世界として，計算機上で如何にして

【１】元の自然現象に近い相似性の高い再現シミュレーション

を実現するかの問題解決の必要に迫られる．この問題は計算機上で実現するには更にその奥にもう 1 つの問題を含んでいる．それは，

【２】その再現シミュレーションのプログラムがおよびそれに至る
までの作業過程が適切であることをどう客観的に検証するか？

という点である．例えば，上記の流体の流れの計算機での再現シミュレーションにおいてシミュレーション計算の結果が信頼するに足るものなのか否かは上記の【1】だけではなく，【2】の客観的な検証が必須なのである．しかしその実行あるいは実現は非常に困難であり，通常は再現実験あるいは観測等によって代替されてきた．しかし，常に代替する手法があり得るとは限らない．上記の問題においても宇宙空間での実験しか有り得ないので，技術上の問題以外のも費用や人手等の問題もあって極めて実験も困難であった．そこで【2】を計算機上での実現の範疇の中で直接に解決あるいは実現するアプローチを研究開発してきたのである．

そのアプローチのための基本的なアイデアは，対象世界[⋆2]の要素を特定し，その特定要素をプログラムに至るまでの過程の中でどう変換されて行くかを追跡することである．しかしこのアプローチは例えば Fortran 等の手続き型プログラム言語を用いるようであれば難しい．その主な理由を二つだけ述べると，対象世界の動きの分析記述とプログラム[⋆3]の構造や動きについて，その両者が複雑すぎてその分野の専門家だとしても客観的な比較対照分析と検証などは難しいのが現状であろう．

二つめは手続き型プログラムを最終表現とするような“手続分析”や“手続的な設計の方法論”が実質上，技術としては存在しないことである．

2.2 OO パラダイムの導入と記述言語系の設計

これ等二つの理由を解決する方法論 (パラダイム) として採用したのがオブジェクト指向 (OO) パラダイムである．OO では第一の点については，モデリングを離散単位あるいは離散化モデリングされた単位で行うので対象世界から分析されたその離散要素がどう変換さ

⋆2 対象とする自然現象の世界から少し一般的にみてこう呼ぶ．
⋆3 本論文でプログラムとは計算機用に開発されたプログラム言語の文法に従って作成されたモノを指す．OONJ の (OOJ においても) 作成されたモノは記述であって，プログラムとは呼ばない．

Vol.2010-MPS-78 No.16
2010/5/21

れて行くかのを追跡することは工夫することで十分に可能である．また第二の点については，分析から設計，実装を経てプログラムに至る過程のための OO 技術が (我々の問題解決に適しているとは限らないにしろ，) OO ソフトウェア工学あるいは OO プログラミング技術として存在しているからである．それを概略図として表したのが**図 1** である．

図 1 において対象世界を OO でモデリングし，各モデルの離散単位毎に全ての要素の OO 分析記述を作成し，それを変換して計算機の世界に持ち込んで OO 設計記述にし，その記述を特定の OOPL [*1]のプログラムに変換する準備として図 1 右下の OO 実装言語 OPDJ を用いた変換を行い，最後に OOPL プログラムに変換する．それを分析記述と比較対照して相似性の高さを評価する．並行して，離散単位毎の変換過程を経てどうプログラムに変換されたかの記述を追跡することで検証を実現する，という方針とする．

したがって，我々の主題としては OO パラダイムを用いてモデリングし記述し，OO ソフトウェア工学あるいは OO プログラミング技術に工夫を加え，対象世界について記述された全ての離散単位を追跡すれば,【2】が実現し，その結果として【1】の対象世界との相似性が判定できる道が開ける可能性が出てくることになる．これが我々の目指す研究全体を通しての一貫した主題であり，その作業過程の中でどう OO 分析モデリングを行って離散単位を記述するかという点が本論文の設計主題であり，具体的な提案としての記述法を提案してそれで実証する事が本論文の目的である．

3. OOSF の仕組みから OONJ 設計主題へ

3.1 オブジェクト指向 (OO) の原理と構造の定義

本論文で用いるオブジェクト指向の原理図を**図 2** に示す．OO の基礎概念はモデリングは全て離散的に行うことである．したがって，離散 (化) モデリングしたオブジェクトを図 2 のように楕円で描いてある．この離散化単位のことを以降，離散記述単位 (Diecrete Description Unit，(以降，**DDU** と略記) と呼ぶ．DDU とは，通常はオブジェクトが代表例であるが，離散的にモデリングされ，単位として記述可能な要素は全て DDU である．例えば，オブジェクトが内部に持つ部品も DDU であるし，属性の纏まりも単位として扱えれば DDU であり，単位として纏まった振舞い等も DDU になり得る．初心者は誤解を起こし易いので強調はしないが，相互関係も離散的なモデル単位として記述できれば DDU である．

⋆1 Object Oriented Programming Language を以降 **OOPL** と略す.

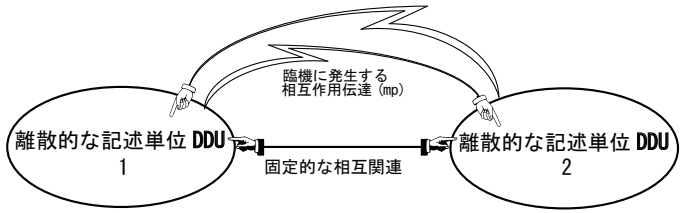

図 2 OO の原理，OO でモデル化された世界の見方

複数の DDU 間を繋いで付与されるのが相互関係である．相互関係には図 2 に示すように 2 種類ある．固定的な関わりである相互関連と，臨機的な (短い時間だけの関わりである) 相互作用伝達 (mp) とである．本論文では

DDU に相互関係 (二種のどちらかまたは両方) を付すことが構造化であり，
相互関係とそれを付与されている DDU 群を纏めて構造と呼ぶ.

と定義する[6]．通常は固定的な相互関連の場合だけが構造と呼ばれている事が多いが，本論文では論理的な簡潔さ/単純さを重視して，DDU 間に mp が行き来するだけでも構造 (化) と呼ぶので注意が必要である．

この様な定義から本論文での**オブジェクト指向**とは，対象世界から DDU 毎に着目・抽出・名前付けし，DDU 間を構造化 (＝相互関係を付与すること) することである．そして，この作業を対象世界の必要な要素の全てに対して行い，それによって対象世界全体を捉えて記述/表現するのが**オブジェクト指向の方法**である．

3.2 OOSF のを基本とした OONJ の設計原理

OOSF の概略を**図 3** に示している．ただし，既出版の論文[6] の OOSF ではなく現 OONJ における OOSF である．

【a】図では対象世界全体を表現している．OO でモデル化された対象世界には複数/多数のオブジェクトが存在する．オブジェクトを OOSF としてはフレームと呼ぶ．図中のオブジェクト 1〜3 は相互に固定的な構造（＝相互関連) を結び，あるいは，互いに相互作用をしながら動くことで対象世界全体の動きを形成してゆく．

【b】図では各フレームがスロットと呼ばれる記述単位を集めて纏めている様子を示している．各スロットは図中にもあるように，纏まった (=離散単位として扱える) 振舞いや属

Vol.2010-MPS-78 No.16
2010/5/21

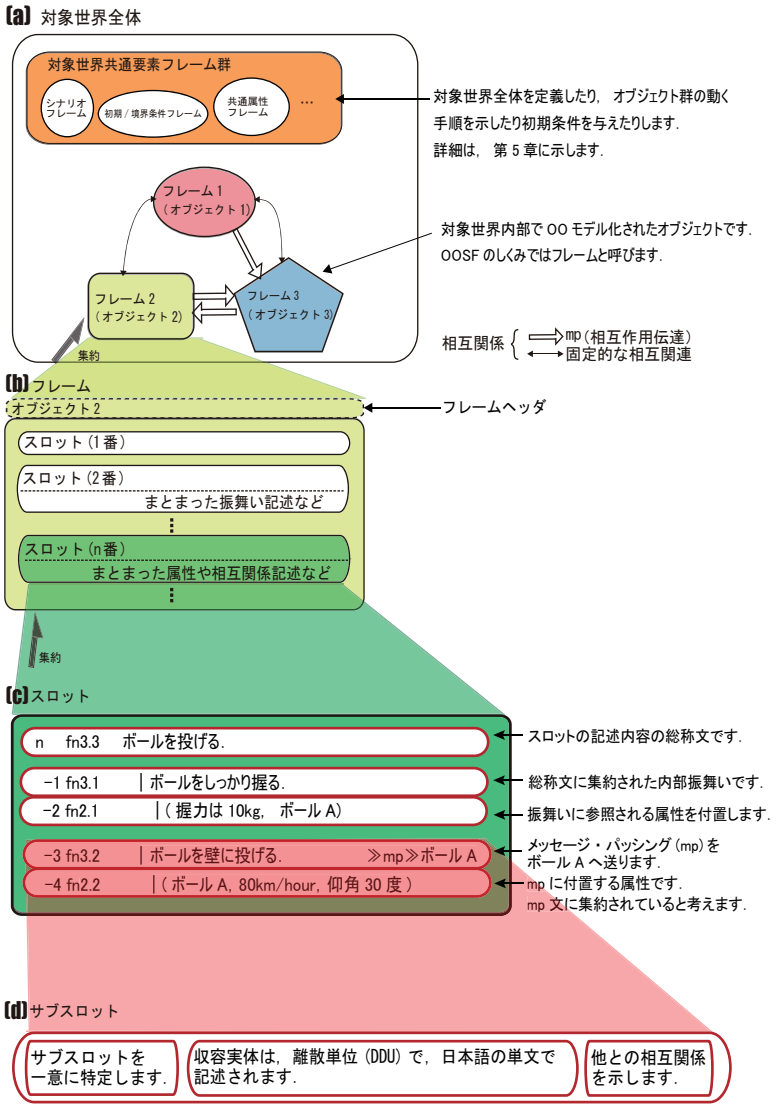

図 3 OOSF の仕組みと構造

性についての記述が格納されている．これ等の纏まった記述はオブジェクトの振舞いや共通の属性，相互関係等が記述される．フレームは言わばスロットのコンテナ (容器) であり，多数のスロットを束ねて 1 つの独立単位とし，それをオブジェクトと呼ぶモデル記述の形に形成している．

【c】図では各スロットが複数/多数のサブスロットを内包して集約している様子を表している．その典型的な例を【c】に示した．スロットの最初の行 (＝サブスロット) はスロット総称文と呼ばれ，スロットを纏めたり代表する内容を「総称する」文，この例では「ボールを投げる」，が格納されている．つまり，各スロットの内容はこの総称文代表され，以降の振舞いはこの総称文に集約されていることになる．[⋆1]左端にスロット番号を記す．以降の各サブスロットにはサブスロット番号をハイフォン付きで左端に記す．スロット番号は省略されている．

【c】図のサブスロットの多くは 2 行 (2 個のサブスロット) をセットとして記述される．図の例においてサブスロット 1，2 では「ボールをしっかり握る.」という振舞いに対して，握る対象のボール A を特定したり，握る力の値を記し，直上の振舞いに必要な属性値を付置している．この属性の値を付置するサブスロットを本論文では“振舞い参照属性”と呼ぶ．実際に振舞いをする際に必要な属性の値を与えるのが目的であり，振舞い本体が日本語で書かれるために別にして付置して置き，振舞いに「参照」される形にしたのである．

同様に，サブスロット 3，4 はその右端にボールという他のフレームとの相互作用伝達 (=mp) を示す記述に依って相互作用振舞いであることを表す．相互作用の内容は「ボールを壁に投げる.」であり，その振舞いに必要な属性を次行に付置する．これを本論文では「mp 付置属性」と呼び，相互作用相手に送るモノや情報，相互作用振舞いに必要な属性を纏めて付置しておき，その事を (｜) で表現することとしている．

【d】に示すように各サブスロットはその内部に 3 種類の記述を含んでいる．その左側にそのサブスロットフレーム内部において一意特定するための一連の番号 (の組) が書かれている．中央にはサブスロットに収容された離散的な記述実体[⋆2]を置く．サブスロットの右側には他のサブスロットやスロット，フレームとの相互関係を表す記述 (本論文では構造記述子と呼ぶ．) を書くことにする．

⋆1 総称文に集約されている事を示すサブスロットは字下げと (｜) で表現する．
⋆2 今後，この記述実体を DDU(Discrete Description Unit，離散記述単位) と呼ぶ．DDU の実体記述 (内容) は日本語で表現される．

以上の様に，サブスロットは最小 DDU を表す日本語文を収容するコンテナ (容器) である．コンテナはサブスロット，スロット，フレームの 3 種類であり，各々の内部に収容した記述をその閉じた枠線により視覚的イメージとしての離散記述単位 DDU に見せる効果を持つ．

コンテナ間の構造化 (＝相互関係の付与) はコンテナ右端に相互関係を記して表現し，スロットはサブスロットという容器を纏めて (集約して)1 つの纏まった振舞いや属性の塊を表し，フレームは複数の振舞いや属性の塊を纏めて収容したスロットを収めてフレームとする．見方を変えれば，OOSF はオブジェクトの定義の実体である最小 DDU を構造化するための枠組であるといって良い．こうした構造化によって 1 つのオブジェクトが秩序立って記述されていることが理解される仕組みに組み上げたのが OOSF である．OOSF のこの様な位置づけはちょうどオブジェクトのモデル表現に相当する自然な位置づけであろうと考えて設計した．

3.3 OONJ の設計主題

以下のように主題が決められた．

(1)OOSF の仕組みの精密化/詳細化および OOSF の全要素を一意特定した上での形式的構造化

(2) サブスロット内部の日本語単文の内部分析

結論としては，前節までのような OOSF を基礎にした記述言語 OONJ の実現には一足飛びに一つの記述言語として計画・実現するのではなく，図 1 のように，まず分析記述言語 OONJ から始めて ODDJ，OPDJ と設計と開発を進める方針をとることが挙げられる．つまり各段階の記述は各段階特有の記述を完成させれば良く，全体で最終に OOPL プログラムを得られる様に三言語を設計する，という方針とした．

4. 記述言語 OONJ の構成

4.1 OONJ で用いる要素種類

本章では，前章で決定した OONJ の設計主題の構成を具体的に述べる．まず OONJ において必要に応じて使われる要素種類を**表 1** に挙げる．

本論文における OONJ の特徴を要素種類という視点から見ると基本的には表 1 から明らかなように 4 種類のみであることである．勿論，各要素種類の多様な詳細要素は多いし，

表 1 対象世界の要素種類

離散記述単位 (DDU)	**fn3 振舞い**
fn1 オブジェクト (フレーム)	fn3.1 内部振舞い
fn1.1 モノ/概念 ／ fn1.2 属性	fn3.2 相互作用伝達 (message passing，**mp** と略)
fn1.3 振舞い ／ fn1.4 相互関係 (他は省略)	fn3.3 相互作用 (通常は振舞い総称記述として利用)
	(内部振舞いと mp の並び) (他は省略)
fn2 属性	**fn4 相互関係**
fn2.1 振舞い参照属性	fn4.1 相互作用伝達 (mp) ／ fn4.2 集約
fn2.2 mp 付置属性 (fn3.2 参照)	fn4.3 汎化/特化 ／ fn4.4 実値化
fn2.8 共有属性 (default は専有属性) (他は省略)	fn4.12 引用 (他は省略)

表 1 には省略されている種類もある．しかし，比較的に簡単なモデリングの段階でよく使われる要素種類は多くの記述例から見ても最大 15 種類程度で一応満たしている事が多かった．

さて，**fn1** はフレームである．フレームは主としてオブジェクト (fn1.1) を設定・格納するために使われるが，属性だけ，振舞いだけ，相互関係だけのフレームもあり得る．

fn2 は属性である．表から分かるように，属性は主に 2 つの目的で設定される．

(1) 振舞いは最終的には数式で計算されるが当初は日本語で書かれる．したがって数式の計算 (＝ オブジェクト指向の振舞い) に使われる“変量”を日本語の記述中でも予め設定しておくためにその変量を「振舞いに参照される属性」と設定し fn2.1 を充てる．これはプログラムで使う“変数”相当の量である．

(2) 次に，OO ではオブジェクト間の相互作用 (相互作用伝達) の情報モデルにメッセージパッシング (message passing，mp と略す) を用いるが，その mp はしばしば相手方に何か DDU を付置して (くっつけて) 送付することが多い．その場合に，mp の主文に付置される DDU を“mp 付置属性”として扱うという方法を我々は採用した．
これ等 2 種類の属性の設定は，サブスロットの主文が NJ 単文であり，NJ 単文の中に直接に属性 (変量) を設定できないことから採用した方法である．その他によく使われるのはフレーム内で共有する変量すなわち属性を定義する場合である．

fn3 は振舞いである．オブジェクトの振舞いはそれが内部か外部かを峻別するために，内部振舞い (fn3.1) と外部振舞いを設けた．内部振舞いは外部と関わらない振舞いであり，内部だけで使用可能な変量を参照する．外部振舞いは他のオブジェクト (DDU) と相互作用をする振舞い，すなわち相互作用の情報モデルである mp(fn3.2) であり，mp と一緒に伝達するために付置される属性を伴う．

オブジェクトは通常，mp を受け取り，それを解釈して内部振舞いを行い，自身の変化を

Vol.2010-MPS-78 No.16
2010/5/21

メッセージとして外部に mp を発する．そういう内部振舞いと mp の一連の複数の総計としてオブジェクトの纏まった振舞いが表されることが多い．そこで，これ等を複合的に (つまり，構造化して) 表現する振舞いが相互作用 (fn3.3) である．この相互作用文 (fn3.3) は通常は纏まった振舞いを表すスロットに現れるので，スロット内容の“総称文”として使われる．

fn4 は相互関係である．その中で図 2 や図 3 にもあるように臨機に発生する相互関係を表すのは，相互作用伝達 (mp) だけである．残り全ては相互関連である．それは，OO では相互作用を表す情報モデルがただ 1 つであることに由来する．

4.2 OOSF を用いた OONJ の具体的記述法

OONJ の記述の基本を決める OOSF は，以前の論文[6] で発表した OOSF と基本的には OONJ の構造記述規則 (SDR) から見ると，大幅に改良した．厳密な SDR そのものは拡張 BNF 形式で書かれているので，それは付録に収録して必要最小限の参照に留め，代表的なフレーム記述例を下敷きにした説明を**図 4** に示す．

4.3 日本語文の扱いとその変換過程

第 3 章で述べたように，サブスロットの内部に収容された DDU の実体内容の記述は日本語で表現される．しかし対象世界を分析するだけならば別として，最終的に計算機のプログラムに変換するのであれば，少なくとも分析段階での必須の準備をしておくことが必要である．そのために本節では日本語文の扱いとプログラムまでの変換過程の内，分析段階までに行う準備についての概略を**表 2** に述べる．本章の詳細については研究会の発表論文[7] に譲る．

ただし，記述ユーザの負担軽減のため，例えば日本語プログラム言語[8] のような制約あるいは変更された日本語文法は用いず，通常の自然日本語[⋆1]のままで用いる．また NJ 文はユーザ自身の書いたのであるから，それらは文法的にも意味的にも如何様にでも分析や書き直しが可能である，とする．

5. OONJ の設計結果とその評価

本章では第 3.3 節で述べた OONJ の設計主題の実現結果を述べ，評価・検証する．また，

⋆1 これを本論文では自然日本語 (Natural Japanese, 以降 NJ と略) と呼ぶ．

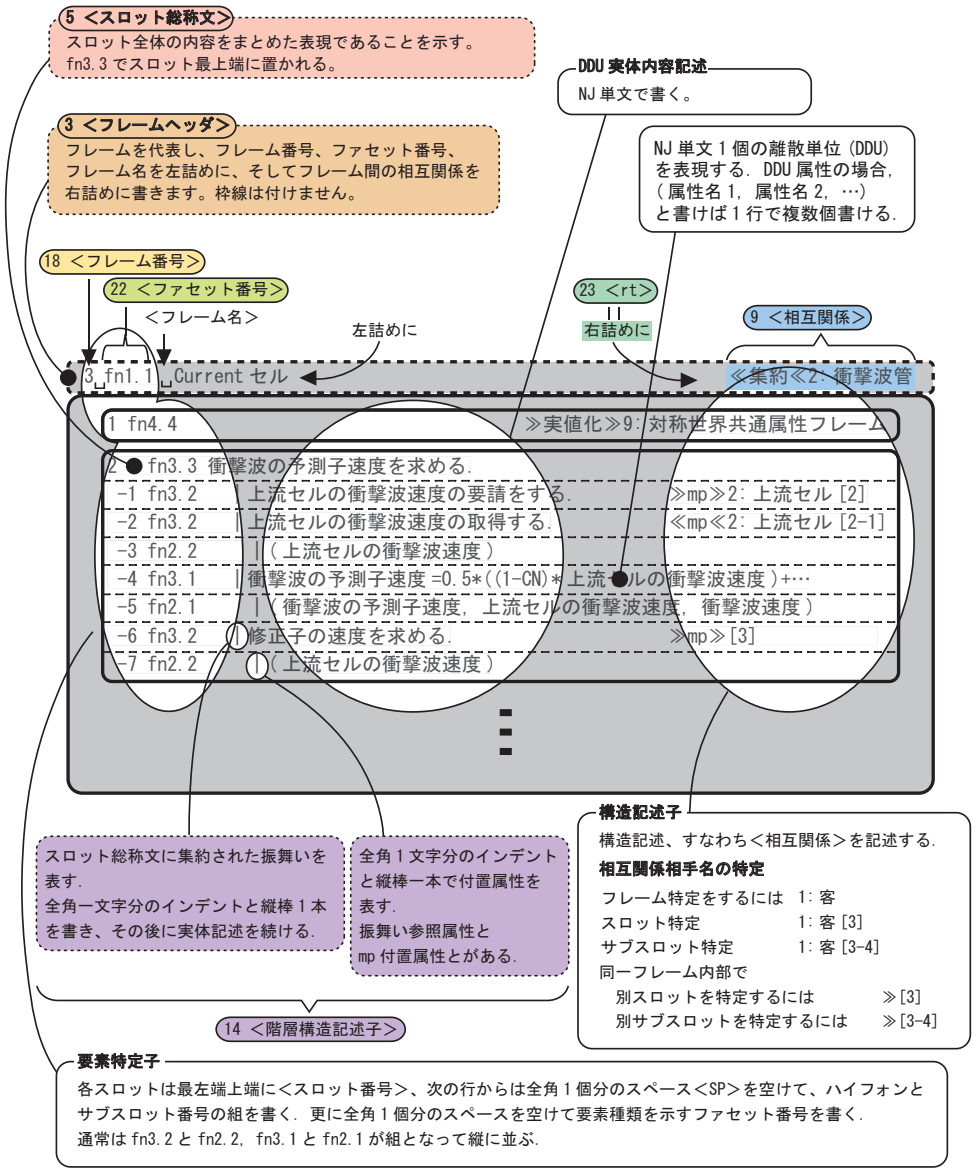

図 4 フレーム記述法：左から **DDU 特定子**，**DDU 実体記述**，**構造記述子の 3 つを離して置く．**

第 2.1 節で述べたような最終目標に一歩近づいたか否かも理論的に論証する．第 3.3 節で述べた設計主題を実現すれば，第 2.1 節で述べたような最終目標【1】と【2】の実現に対して OONJ が果たすべき機能と役割は実現できたことになり，この最終目標に一歩前進して近づいたことになり，本論文の主題は達成されたと考えられるからである．

つまり，まず OONJ の設計仕様での機能や記述力が十分にあるか否かに注目して論じたい．したがって，ユーザの記述し易さや理解しやすさ等は記述支援環境等が揃ってからの評価として別の論文に譲ることにしたい．

表 2 NJ 記述の分析と変換のユーザ作業

NJ 記述の分析と変換までのユーザ作業

第 1 段階：**全て NJ 文へ変換/整形/書き直す**
下準備として NJ 記述 [a] があれば分解して NJ 文 [b] の集まりに変換/書き直す [c]．
第 2 段階：**NJ 文を NJ 単文に，そして意味内容について論理的で一意性を持つ平叙 NJ 単文に書き直す**．
第 3 段階：**NJ 五文型に分類し，ガイドラインに従って OONJ 記述に変換**
平叙 NJ 単文を学校英語で馴染みの深い英語の五文型に当て填めて書き直す．NJ 五文型文はガイドラインに従った処置し OONJ 記述に変換する．
(1) 第 1 文型 (S+V) : 私 (S) は走る (V)．→内部振舞いの OONJ 記述に変換．
(2) 第 4 文型 (S+Od+Oi+V) : 私 (S) はボール (Od) を彼 (Oi) に投げる (V)．
→外部との相互作用振舞いに変換した後で OONJ 記述に変換．
(3) 第 2 文型 (S+C+V) : 私 (S) は学生 (C) です (V)．→例えばフレーム名に変換
(4) 第 3 文型 (S+Od+V) : 私 (S) はあなた (Od) を愛する (V)．
そのまま，または第 4 文型の mp 文に書き直し，OONJ 記述に変換．
(5) 第 5 文型 (S+Od+C+V) : 第 2 または第 4 文型に書き直す．
第 4 段階：**設計段階への引き渡し準備**
前段階までに行った NJ 五文型に分類された平叙 NJ 単文に関わる情報を，計算機に引き継ぐために，第 1 から第 5 文型までの文型情報と，S，Od，Oi，C，V の文要素情報を付加するディレクション付けを行う．

a 本論文で **NJ 記述**とは，NJ 文の集合や集約を指す．ただし稀には文脈や前後関係によっては，単に NJ で書かれた表現を指す場合もある．
b 本論文で **NJ 文**とは，形式的には 2 つの丸（。）かピリオド（.）間のひとまとまりの文を指す．形式上は次の段階で出てくる NJ 単文も含まれる．また，単に文と言えば，NJ 文のことを指すとする．
c 現実問題としてはユーザは記述例を参考にしながら作業することが多く，後で書き直しが必要となる NJ 記述や NJ 文を最初に書く場合は少なく，殆ど第 2 段階の**平述 NJ 単文**から出発すると考えられる．

5.1 OONJ における OOSF の拡張部分の評価

現仕様の OONJ と以前の論文[6] の OOSF との記述規則の差は，以下の諸点にある．

(1) フレームからサブスロット単位を経て，NJ 単文に至るまでの一意的な特定を含む構造化が確立した．その結果，全ての DDU が一意的に特定/指定ができるようになった．これは第 3.3 節の (1) に対応する．
(2) 対象世界内部に 1 つしか無いフレームや共通な情報を収容して設けられるべきフレームに関する扱いが確立した．これは第 3.3 節の (1) に対応する改良点である．
(3) ODDJ，OPDJ，OOJ に対する配慮が組み込まれた規則になったこと．あるいは，OONJ の記述規則が ODDJ 以降の記述言語でどう変換されるかの設計データが得られたこと．
(4) NJ 単文に対する付置属性の考え方，扱い方，記述法等が OOSF 内部の設計仕様として確立したこと．これは第 3.3 節の (1) と (2) に対応するこの点の詳細は次節で述べる．
(5) サブスロット内部に収容される NJ 単文自体の分析と変換の作業に関する方法が確立された．これは第 3.3 節の (2) に対応する．この点の詳細も次節で述べる．

5.2 NJ 単文の分析と変換の仕組みの評価

第 4.3 節で提案したサブスロット内部に記述された NJ 単文の扱いについては，仕組みとしては表 2 の分析と変換のユーザ作業によって設計段階に送れば設計段階の記述言語 ODDJ の記述に変換されることは，少なくとも我々自身の記述実験では変換の妥当性は検証された．勿論，十分多様な場合を想定した大規模の実験ではないが，我々自身の記述実験の結果はガイドラインや記述例・変換成功例としてガイドラインに組み込む予定である．ただし未だ，記述例や変換のガイドラインのマニュアル電子版が不十分である．

なお，この NJ 単文の分析と変換が仕組みとしては確立された理由は以下の三点であると考えられる．

(1) OOSF 側からはプログラムの内部構造と対象世界の記述枠組の全体構成が提供される．
(2) プログラム内部の空白部分，即ちサブスロット内部の NJ 単文の変換の仕組みは本論文の提案により実現した．ただし，詳しくは参考文献[7] を参照されたい．
(3) ユーザの NJ 文に対する十分な知識と分析及び処理能力が前提の基礎条件としてあった．

つまり，OOSF 側，NJ 単文の変換の仕組み側，ユーザ側の三者のニーズと提供技術と知識

の組み合わせが都合良くかみ合った事が，NJ 単文の分析と変換の実現に大きな貢献をしたと考えている．この結果は，OOPL プログラムの (半) 自動生成の実現に大きく貢献すると考えられる．

6. OONJ の設計結果と第三者評価・比較

6.1 記述例から見た OONJ の特徴と記述力

表 3 に代表的な記述例を示す．記述者は茨城大学の情報工学専攻の大学院生である．彼等は，OONJ の設計を含む内容の大学院の講義 (半期，2 単位) を受けた程度の知識を前提にし，ワープロソフトや一般のエディタ (Word，Excel，一太郎) 等を使って記述例を書いた．

表 3 OONJ の記述例データ

記述対象世界	フレーム数	スロット数	記述行数	記述量 (KB)	分析時間 (hour)	記述時間 (hour)	経過期間 (日)	行数 /hour
DNA 演算 (加算)	43	242	1188	234	10 以下	約 80	25 以上	14.9
プラント制御	21	143	712	368	10 以下	50 程度	40 以上	14.2
遠隔料理システム	15	470	2703	1321	12	59(記録)	18 以上	45.8
ヒトの免疫システム	28	136	563	89	20 程度	40 程度	14 以上	14.1
英日翻訳ソフト	31	159	824	278	約 73	94	約 25	8.8
視覚のメカニズム	23	126	523	76	20 程度	40 程度	14 以上	13.1
チャットシステム	24	119	637	187	10 程度	60 程度	10 以上	10.6
水の大気循環	36	251	904	147	約 20(*)	約 40(*)	約 6(*)	22.6

(注) すべて記述者の申告による大雑把な時間と期間である．単位は，記述量が KB，時間が hour，期間が日である．
(*)OONJ の仕様の改訂に沿って何度も書き直しているので，改めて書き直したときの推定の時間/期間である．

まずは記述例の半分以上が情報工学特有の世界以外であることに注目したい．つまり彼等の知識は専門家 (の卵) の知識で書いたのではなく趣味程度の知識で書いたと考えて良い．その程度の知識を持ってして，平均 1007 行程度を 57.5 時間の記述時間で書いたことになる．1 時間平均 20 行弱である．ただし,「遠隔料理システム」を書いた院生はちょっと例外的であるし，経過時間 (記述開始日から記述完成日までの期間) は相当に長いと考えられるので，割引して考える必要がある．

1. 1 時間平均 20 行弱で書けるならば，OONJ の記述法が結果として易しかったことが推定される．
2. 純粋な記述時間としてはどの対象世界も 40 時間〜80 時間である．彼等の記述力や対象世界の複雑さの違いが大きい点などを考えると，記述時間が桁違いにならないことから，OONJ の記述力と記述し易さが存在し，それが対象世界による記述時間の差の小ささに繋がった可能性が考えられる．この点を訊いたところ彼等からは OONJ を使って各自の対象世界を書くのに支障は無かった，あっても少なかったと例外なく答えている．
3. 「英日翻訳ソフト」の分析に使った院生は OONJ の分析が彼女の修士論文の準備に非常に役に立ったと感想に書いている．講義の当初から利用するつもりでいたとも答えている．OO 記述が対象世界の分析に役立つ，という典型例であるとも言えよう．これも OONJ の記述力の力であるといえるのではないか．
4. 対象世界を如何なる制限をしなくてもよいかについかについては，記述された範囲での実例評価と論理的な帰結から論証する．
5. 現時点のの OONJ の仕組みは簡単に言ってしまえば

   OOSF ＋ OONJ ライブラリ ＋ 記述ノウハウ

で構成されていると言える．OOSF の新規の規則はわずか 20 個程度でしかなく，図 4 からも分かるとおりの簡潔な仕組みであるし，OONJ ライブラリは記述例とガイドラインにサポートされれば，元々ユーザは記述に馴れている内容を OO 記述の形式にするだけであるので，これも容易に扱える．これ等以外は記述方法やノウハウに属する類が殆どである．

また，憶えなくてはならない規則が少ないことも事実である．したがってユーザの学習コストは低いことが予想されるし，現時点の記述例群 (記述例データと研究室の WebSite に掲載されているもの) の記述者の感想から言ってもその学習コストは低い事が分かっている．記述例データに掲載した記述例の記述者達の学習コストは 1.5 時間の講義 (講座)15 コマ以下であり，各々の対象世界の分析コストは (例外を除けば)20 時間以下である．

したがって，OONJ の記述力と記述しやすさは一般的には相関することを考え併せると，OONJ は記述力は (少なくとも記述例を書いた院生は，) 必要なだけは十分にあり，記述力に問題は無かろうと判断する．

6.2 関連研究との比較評価

【日本語プログラミング言語 (以下，JPL と略)[8] との比較評価】

JPL と我々の OONJ の最も基本的な違いは，プログラミング言語 (PL) か記述言語かという点にある．JPL のプログラムの構造がどうあるべきかを学習して理解し，使いこなせなければならないという点にある．その意味で，制約された日本語が使えるとは言え，新たなスタイルの PL を丸々一言語を学習する必要がある．その点，OONJ はユーザが理解している対象世界の OO 構造を理解していれば，OOPL のプログラム構造をユーザのドメインでの記述に非常に近い記述を純粋な NJ を使って書くことが出来る．ユーザの負担という点で，OONJ の方がメリットが大きいと言えよう．また，JPL が科学技術分野の複雑なシミュレーションに使われた実績があるとは聞いていない．

【DEQSOL[9] との比較評価】

DEQSOL は我々のと同じ方向性や目的を持つシステムである．ただし，微分方程式に限った数値計算プログラムを生成するという点でターゲットを絞り込んでいる点が異なる．日本語を主用するか否かという点で ICSP_OOJ に大きいメリットがある．すなわち守備範囲の広さや使い勝手の容易さという点では ICSP_OOJ からメリットを得る対象の人数が遙かに多い代わりに，生成するプログラムの規模や実用性という点で，DEQSOL の性能が遙かに勝っている．今後も DEQSOL の良い点を参考にしたい．

7. 結論と今後の展望

1.OOSF を基礎にした OONJ は記述言語の仕組みとしては一応の確立したものとなった．
2. サブスロット内部の NJ 文の NJ 単文への変換や設計段階記述への自動変換にためのディレクション付けは仕組みとしては確立した．

ただし，以上は実際のユーザにとっては OONJ 記述を支援する環境，OONJ エディタや同トランスレータの存在が必須である．本論文ではアピールすべき主題が異なったために，全て割愛したが，OONJ エディタや同トランスレータは既に稼動を開始しており，別の機会にそれらを発表する機会を持ちたいと考えている．

参 考 文 献

1) 松本賢人，畠山正行，安藤宣晶，オブジェクト指向分析記述言語 OONJ の設計原理構築と記述環境開発，第 150 回 SE 研究会報告,2005-SE-150，pp.57-64，(Nov. 29，2005)．
2) 川澄成章，野口和義，畠山正行，オブジェクト指向設計記述言語 ODDJ の設計とその記述環境の開発，第 150 回 SE 研究会報告, 2005-SE-150，pp.65-74，(Nov. 29，2005)．
3) 加藤木和夫，畠山正行，オブジェクト指向実装記述言語 OEDJ の記述環境およびトランスレータの開発，第 150 回 SE 研究会報告,2005-SE-150，pp.75-82，(Nov. 29，2005)．pp.49-56(2009)．
4) 畠山正行，オブジェクト指向一貫記述言語 OOJ の構成とその概念設計，第 150 回 SE 研究会報告, 2005-SE-150，pp.49-56，(Nov. 29，2005)．
5) 大木幹生，片野克紀，三塚恵嗣，沼崎隼一，涌井智寛，加藤木和夫，池田陽祐，畠山正行：三言語独立のオブジェクト指向記述言語 OOJ の実装と検証，第 163 回 SE 研究会報告，2009-SE-163．
6) 畠山正行，オブジェクト指向分析自然日本語構造化フレーム OOSF の設計と表現技法，日本シミュレーション学会誌，Vol.22，No.4，pp.195-209, Dec., (2004)．
7) 畠山正行，岩坂篤志，池田陽祐，三塚恵嗣，涌井智寛，オブジェクト指向記述言語 OOJ における日本語単文の自動変換機構，第 167 回 SE 研究報告会，2010 年 3 月 18 日．
8) クジラ飛行机，日本語プログラム言語なでしこ公式バイブル，ソシム (株)，2008 年 6 月．
9) 佐川暢俊，金野千里，梅谷征雄，数値シミュレーション言語 DEQSOL，情報処理学会論文誌，Vol.30, No.1, pp.36-45, (1989)．