2005－SE－150（7）
2005／11／29

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

—オブジェクトベース一貫相似性過程の方法論をベースとして，ドメインユーザにも使いこなせるオブジェクト指向技術の提案 —

畠　山　正　行†

要約: 本研究を含む五つの総括発表は，我々のグループがドメインユーザ (以降 DU) と呼ぶ技術者等々の人たちのために彼等自身の専門 (ドメイン) とする世界のシミュレーションプログラム開発環境として OO 技術に着目し研究してきたことに関する発表である．その目標を実現するために，自然日本語をベースにし，分析から実装に至るまでの基本のパラダイムと設計方針を共通にした四つの記述言語系，すなわち分析 OONJ，設計 ODDJ，実装 OEDJ と OBF を設計し，その記述支援環境を開発してきた．これらの記述は最後には Java 又は Fortran90 プログラムへとトランスレートされ，計算機上で実行される仕組みにしてある．現時点ではようやく分析から実装/駆動までを記述が一貫して流れようとしている段階である．　本論文の OOJ とは現時点ではこれらの記述言語の“ 単なる総称 ”である．しかし目標としては，分析から実装まで一貫して使える記述言語 OOJ を構築する計画である．そこで本発表では，四つの記述言語とその記述環境の設計と構築の現況を述べると共に,「一貫」記述言語として如何にそれらを統合あるいは新規に構成するか，そしてその概念設計を如何にすべきかについて議論し，提案する．併せて本発表の背景を形成しているオブジェクト工学なる研究開発プロジェクトを述べる．

Object-oriented, Integrally Consistent Description Language OOJ and Its Conceptual Design

—A Proposal of Object-oriented Simple Technology that All Domain Users can make full use of.—

Masayuki Hatakeyama†

Abstract: Our five blanket presentations concern the researches that simulation program development support environments based on the Object-oriented (hereafter, OO) technology. These researches aim at contributing to the engineers and researchers that are the experts at some specified domains called the Domain Users (hereafter, DU). To realize the aim, we have designed and developed the four description languages and their support environments based on the Natural Japanese. These four languages are, OONJ for OO analysis, ODDJ for OO design, OBF and OEDJ for programming. The descriptions out of the OBF or OEDJ are translated into finally the Java and Fortran90 programs, respectively. At the present moment, this architecture will finally get around to the stage that an integrally consistent descriptions flow from the analysis, design and finally up to the execution. At present, the OOJ in this presentation is only the generic name of these four description languages. But we aims at constituting an integrally consistent description language from the analysis stage up to the programming one. Then in this presentation, we discuss and propose how to integrate or newly design as the ”integrally consistent” description language, and how to make the conceptual design of the OOJ. We also present our research/development projects named the Object Engineering that have formed our background technology.

1. はじめに：想定ユーザと開発の狙い

–論文副題を実現する DU のための技術の提案–

最初に，我々の研究グループが構築しているシステムの使用想定のターゲットとしているユーザとそのユーザ向けシステムの狙いを述べる．まず想定ユーザは **表 1** に示す様な本研究の中でドメインユーザ (以降, DU と略) と呼ぶ何らかの専門分野 (Domain) を持ち，計算機を利用 (あくまで利用に留まる) している専門家 (プロ) である人たちである．

本システムは DU が自身の研究や技術開発，専門作

† 茨城大学工学部情報工学科
Department of Computer & Information Sciences, Faculty of Engineering, Ibaraki University

業を進めるために“ 充分に使いこなせる道具 ”として使えるようなオブジェクト指向 (以降，OO と略) に基づく分析/設計の技術，そしてプログラム開発技術を確立することを目指す．より具体的には，自身のドメインの対象世界の分析/設計/実装とその稼働と制御/データ取得までの全モデリング・記述 (・ means sequentially ”and”) の過程を計算機側として支援し，枠組み付ける技術であり，それは OO を基盤パラダイムとした技術であり，その実際に DU に使える技術として構築されている核になるのが，本研究で述べる OO に基づく記述言語系 OOJ とその記述環境である．

本論文では必要があって OOJ の記述に入る前に，OOJ とその記述環境の基盤であり，我々がオブジェクト工学と呼ぶ DU 向け (専用) の OO 技術を紹介する．オブジェクト工学とは (第 2 章で詳述するように)DU 向けに広く緩やかに定義された OO パラダイム (オブジェクトベース) を共通概念とし，上記 DU にために開発している記述言語系とその言語記述/プログラム開発をサポートする記述環境を扱う分野である．

ただしこれらは現在のオブジェクト指向ソフトウェア工学 (OOSE) とは幾つかの点でかなり異なる特徴を持つ．むしろソフトウェア開発の専門家を (敢えて) 除いた計算機“ 利用ユーザ ”，特に開発の進んでいない科学技術計算分野方面や，最終的には広い定義のすそ野の分野（エンドユーザコンピューティング（EUC））をも使える OO パラダイムの応用技術への注力を主眼としている．

本論文ではトップダウン的なスタイルで技術的な記述を行う．まずオブジェクト工学を概観し，その中核となる実用を目指す具体技術である記述言語 OOJ とその記述環境の構成とその概念設計を述べる．ただし本来は単一の記述言語 OOJ の構築が理想ではある．しかし一足飛びに分析/設計/実装/駆動までの四つの異なる各段階の要求や制約条件に十分に満たし，かつ DU の多くの要求に応えられる単一の記述言語を構築/実装することは現状では難しい．

そこでまずは，分析/設計/実装/駆動 (実行) の四つの段階毎に分割し，それぞれをその段階に合わせた仕様にして記述言語とその記述環境の設計/実装について述べる．次にその各段階間を連携させ，各段階の記述/変更作業や段階間の変換作業で得られた DU や設計者からの要求，構築された技術，OOJ の仕様との調整等を経て，再度統合された単一記述言語 OOJ とその記述環境を設計するという戦略を採った．

本発表に続く四つの発表は分析/設計/実装の各段階毎の記述言語の設計と開発を述べる．四つの記述言語，すなわち分析段階の OONJ[1),7)]，設計段階の ODDJ[2)]，実装段階の OBF[4)]，同じく OEDJ[3)] はそれぞれに適した記述環境 (エディタやトランスレータ等) を開発中である．次にそれらの技術を統合して，最終的には四つの記述言語とその記述環境の実装や運用，DU に依る多数の利用経験を蓄積した成果を反映させて，単一記述言語 OOJ として再設計/再実装する計画である．

表 1 想定した DU の特徴
Table 1 Assumed ”Domain Users: DU”
*

1. 情報工学・科学以外の技術/計算/研究開発等のドメイン (専門分野) の科学・工学・技術問題の分析・設計の専門家．流体や構造の解析, 画像分析, 化学合成等の多様な分野の解析, 設計, シミュレーション等を行う．理工系の殆どの研究者や開発技術者が含まれる silent majority である．
2. 業務支援（図書館，経理，銀行等の勘定系）システムや計算機特有 (通信，ウイルス対策，データベース）ソフトなどの擬似実世界ではなく，前項のような真性実世界を対象．
3. プログラム開発については非専門家で，実力・考え・通常の手法・特性，が千差万別．主たる関心は計算 (処理) 結果に在り，プログラムそのものの書き方等には関心は薄い (無い)．
4. 中小規模 (数千〜数万行) の試行錯誤や改訂の多い自家用プログラムを必要に迫られ個人か小人数で一貫自主開発．そのアルゴリズムは複雑なことが多く，別言語での書き換え等は嫌がる．
5. 常用 PL(注)(多くは Fortran) と常用自然言語 (母国語) は十分に駆使し，知識は前提に出来る．未知の新規 PL(特に異型の OOPL) は避ける．
6.CASE ツール，新しい開発環境や技法等は使いたがらず，面倒臭がって避ける傾向が強い．
7. 自身の専門の対象世界に関することならば，必要充分な知識があり，対象世界の詳細要素を識別でき，如何なるモデリングでも縦横に行うことが出来，表現や記述も的確/正確に出来る．
8.OO を概念としては理解し，専門用語を交えた自然言語 (母国語，NJ(注)) でならば，OO に基づく説明や記述も充分にできる，と仮定．

(注) **PL**：Programming Language，**NJ**：Natural Japanese

2. オブジェクト工学

図 1 に現時点 (2005 年 10 月) のオブジェクト工学の全体構成の状況を示してある．本章では基盤技術に当たる下の三段階の基礎部分の説明を行う．なお，図 1 において本総括発表の五つの内容は第 4 段階の実体技術に関するものである．

第 1 段階は主として DU からの要求と，DU でもあり設計者でもある畠山等のアイデアが書かれている欄である．まず出発は第 1 段階最下行の「相似性の高い計算機シミュレーション」を実現するのが DU の究極の望みであるという畠山のアイデアである．著者自身の経験から,「対象世界の計算機内部で自身のドメインの『相似性の高い』再現シミュレーション (広義の)」を実現するには，モデリング方法において分析/設計/実装/駆動の全段階を一貫した OO パラダイムで貫いて実現することがベストな方法論であると考えたことに発する．それは『一貫相似性モデリング過程』に集約され，それならば分析から実装までを一貫して DU 専用の (支援) 環境の構築が良いという自明で最適な結論 (第 1 段階最下行) となった．

次に第 1 段階の下から二行目〜四行目について述べる．勿論，殆どの DU はそうであるが，今使っていない PL を新たに習得しようという奇特なユーザは殆ど見かけない．したがって，自身が現に常用している PL と社会生活に必要な自然言語 (NJ) とだけで済ませたいという希望は強い．それを前提に考えると，NJ ベースの DU 向きの記述言語を作るか，プログラムの自動生成が望ましいことが結論される．

しかし，問題は例えば，(1) プログラムの自動生成機構や，(2)DU が習得する苦労の無い記述言語の実装が (簡単に) 実現できるか否か，ということである．勿論，DU という利用想定ユーザを考えれば通常技術/現用技術では簡単ではないことは常識である．本研究ではそれを DU の特性を巧く使うことで，あるいは DU が持つ特性を究極まで利用して実現可能な方法を捻り出すことで，かつ OO の特性も大いに活用しつつ，実現の方策に目途を付けたのである．その基盤技術が最上行の NJ ベースの一貫記述言語の構築，という結論になる．

第 2 段階は OO の基礎概念を再検討し，DU 向けの個別技術として適切な体系を作る基礎技術を開発してきた項目を示している[5),6)]．本来は時間順や成立順，相互関係，依存関係が存在するが，それらは図からは省かれている．基礎概念として重要な点は，第 2 段階の DU 用の OO 技術の基礎概念を再検討した結果，クラスや継承を用いない最も簡潔な OO パラダイム即ち，**オブジェクトベース (OB) パラダイム**が適切であると結論されたことである．

OB(オブジェクトベース) パラダイムとは，基本的には離散単位としてのオブジェクトとそのオブジェクト間のメッセージパッシング (以降，mp) だけで成り立つ OO パラダイムである．OB はプロトタイプベースの OO，コピーベースの OO と近く，DU は複雑なプログラミング技術を必要とするので使いたがらない継承/委譲の概念も使わずに済む OO パラダイムである．継承/委譲はプログラミング技術の一つに過ぎず，DU にとっては使わずに済むが，使ってはいけないものではなく，また，使うと有効/有用な場合もあり得る技術である．

DU 側の事情としては，手続き型言語を用いる DU にとって実装段階におけるシンプル (簡潔) な OO 技術が使えればそれに越したことはなく，また，それで満足なプログラム，なかんずく処理結果の数値データさえ得られれば「DU というユーザ」にとっては結果オーライであり，十分だからである．どうしてもクラスや継承という概念とその実装が必要な場合はアプリケーションシステムによる自動変換処理で対処するが，多分その必要は起こらない蓋然性が十分に大きい．

第 3 段階は体系的な理論構築とそれを現実のものとするための記述環境と記述言語の構築である．DU が分析/設計/実装/駆動までの全過程に沿って自身の対象世界に対して相似性の高いシミュレーシを実現させるためには,「モデリング」と「言語」の両面に対して根底的な基盤からして構築する必要がある．その核心は二本の柱から成立しており，

(　) モデリングの思考作業系の側面を

OB 一貫相似性モデリング過程の方法,

(　) 記述を担当する言語系の側面を

OB 一貫相似性記述言語系 OOJ,

である．これが第 3 段階の下の二つの基礎理論に相当する．両側面は後で詳しく述べる様に対照的/対称的な特性を持つが，常に相補的に作用して一貫相似性過程を創り上げ，構成する役割を果たす．OB 一貫相似性過程において，モデリングと言語の両側面とその相補性の仕組みは最も重要である．基礎理論の構成は以上である．それらが具体化した技術が最上段，第 4 段階の OOJ とその記述環境である．

ただ，この構成は確かに理想的ではあるが，DU が実際に使えるか否か，現実に DU に使ってもらえる

か否かは別問題として依然残る．プログラミング言語 (以降，PL) ではなく，NJ 記述をベースにしている記述言語であるから抵抗感は少なく習得は容易であるとは言っても，新しめの言語である部分が残ることは確かであるからである．その新しめの記述言語には勿論，記述を支援する環境は個別に DU が使い易いように構築/改良されるとしても，それだけでは不足であり，DU が使えるようにする具体技術が別途必要である，と我々は考えた．それが，

(　)DU に簡潔に使えるための OO 技術

simpleOO 技術，DUDJ と記述環境

であり，第 5 段階目の DU 専用の実用 OO 技術である．そしてその具体化技術が第 5 段階の「記述環境ベース」の simpleOO 技術である.「記述環境ベース」とは，新し目の記述言語の習得自体に記述環境を使うと共に，言語の記述規則の殆ど (9 割) を記述環境に託してしまって，DU は規則を憶えていなくても「これを書くという意図さえあれば,」書けるようにしてしまう記述環境と記述言語の統合化した技術のことである．

これら (　)(　)(　) を同時に満たす技術体系が我々が長年にわたって構築してきたオブジェクト工学の全体構成**図 1** である．DU から見える最終形のシステムは図 1 の第 5 段階の DU から直接に見える部分の simple 記述環境ベースの DUDJ がそれである．

本発表を含む五つの総括的な発表は，図 1 の第 4 段階に相当し，後の四編で分析/設計/実装の各々の段階の記述言語とその記述環境の設計と実装が述べられる．各発表は本発表を第 1 発表と呼び，以降，第 2〜第 5 発表と呼ぶ．各段階で作成され変換される一貫記述例は前後の発表で発表番号付きで引用される．

3. 一貫相似性記述言語 OOJ の段階的構築

3.1 OB 一貫相似性過程の記述言語

DU が要求するような相似性の高い計算機シミュレーションを実現するためには，OB 一貫相似性を持たせたモデリング/記述/変換の過程を実現させる必要がある．図 2 にその説明図を掲げる．図 2 のようにの OB 一貫相似性を持たせた過程を実際に実現するにはその過程の記述の側面をサポートする記述言語が必要である．それは「OB 一貫相似性を持たせた記述言語」☆である必要がある．

ただし，それを単一記述言語の設計と実装で一気に実現することは非常に難しいと考えられるので，幾つかの段階に分けた記述言語を設計し，それを再構成し，統合化することで最終的に単一記述言語という形に纏めるという方式を取る戦略の方が良策だと判断される．そこで，OB 一貫相似性単一記述言語に OOJ という名前を与え，その OOJ を最終の遠い目標として概念設計の視野に入れつつ，幾つかの段階に分けた各段階記述言語の要求仕様，構築方針を立てる．

3.2 OB 一貫相似性記述言語 OOJ

一貫相似性記述言語を一言で言えば前節の一貫相似性過程を実現するために適切な，即ち対象原世界に相似な記述を行える (実現する) 記述言語を指す．より具体的には，まずは

(1) **各段階の記述言語の記述が対象原世界と高い相似性を持たせた表現が可能なこと**

分析/設計/実装/駆動の各段階の全ての「記述」において，高い相似性を満たす記述を行える記述規則や記述方法，記述環境を十分に備えた記述言語を指す．

(2) **各段階間の記述変換が相互間に高い相似性を保った表現が可能なこと**

次には分析段階と設計段階での「記述変換」は相似な記述が出来ることを記述言語の仕様及び記述規則その他の記述方法において保証出来なくてはならない．それは両段階の各記述言語に対しての要求条件であると共に，両記述言語間の DU による変換作業あるいはトランスレータがそれを満たすように作られている必要があることを意味する．勿論，設計段階と実装段階，実装段階と駆動段階についても全く同様な条件が課せられる．更にいえば，

(3) **各段階および各段階間の記述言語記述の相似性の高さの形式的な検証機能/機構が必要．**

である．この (3) については，確立したものではないが，理論上は一貫相似性モデリング過程の論文中で示されている．それを簡単に言うと，各段階において記述されたオブジェクト，その最小の各内部構成要素，その相互関係，に至るまでの全ての要素に対して必要十分条件を満たした“ 騒動構造化 ”の対応関係にあること，とされている．したがって OOJ とはその条件を (記述環境の支援を受けながら) 実現することが可能な記述言語である．

☆ 一貫相似性を持たせた記述言語とは，記述言語内および各記述言語間でそれらの記述対応関係を形式的に明示することで，出来るだけ高い相似性を担保することを主眼として設計/実装された言語のことを指す．

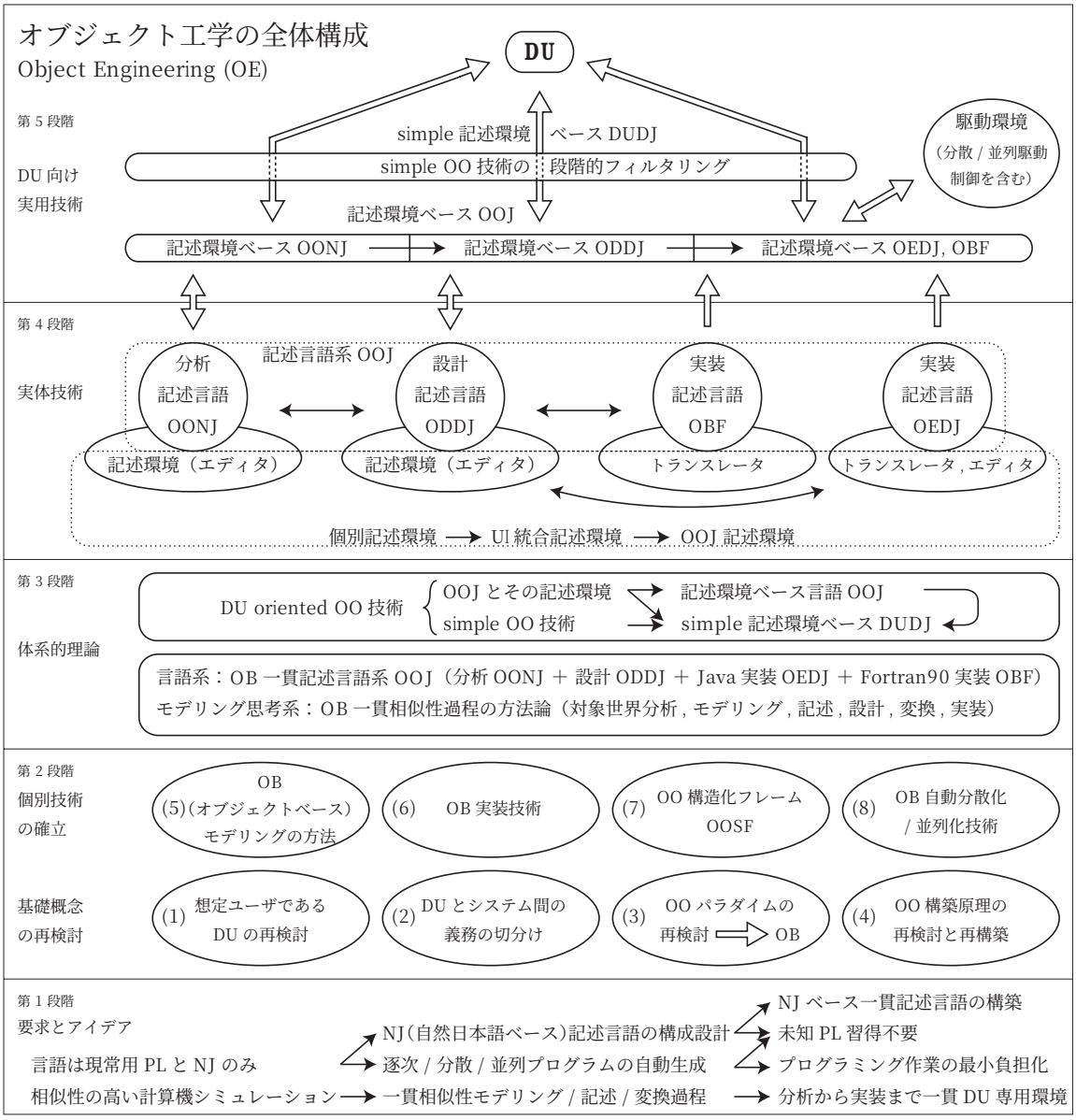

図 1　オブジェクト工学の全体構成
Fig. 1　Whole Constitutions of the Object Engineering

3.3　OOJ の三段階構築方針

OOJ には三つの完成度段階を設定している．

［1］次章で述べる四つの記述言語を ”束ねただけの総称 ”，つまり纏めただけで内実の四つの言語とその環境のパラダイムは一致していても，言語仕様としても，記述規則も，記述環境も異なる，すなわち独立した記述言語系の仮の形式的総称としての OOJ．現時点の段階がまさにそれであり，本研究がその総括発表に当たる．

［2］DU に相対したインタフェースから見れば単一記述言語に見える様な GUI を構築/提供することで実現される段階．記述規則とその記述環境の実体はまだ四つ別々である．これは図 1 で言えば，第 4 段階から第 5 段階への移行段階にあたる．

［3］前二段階の運用によって蓄積されたノウハウや設計データを活用して，OB パラダイム，DU 自身の想定実力等を基礎として，設計原理を策定し直して

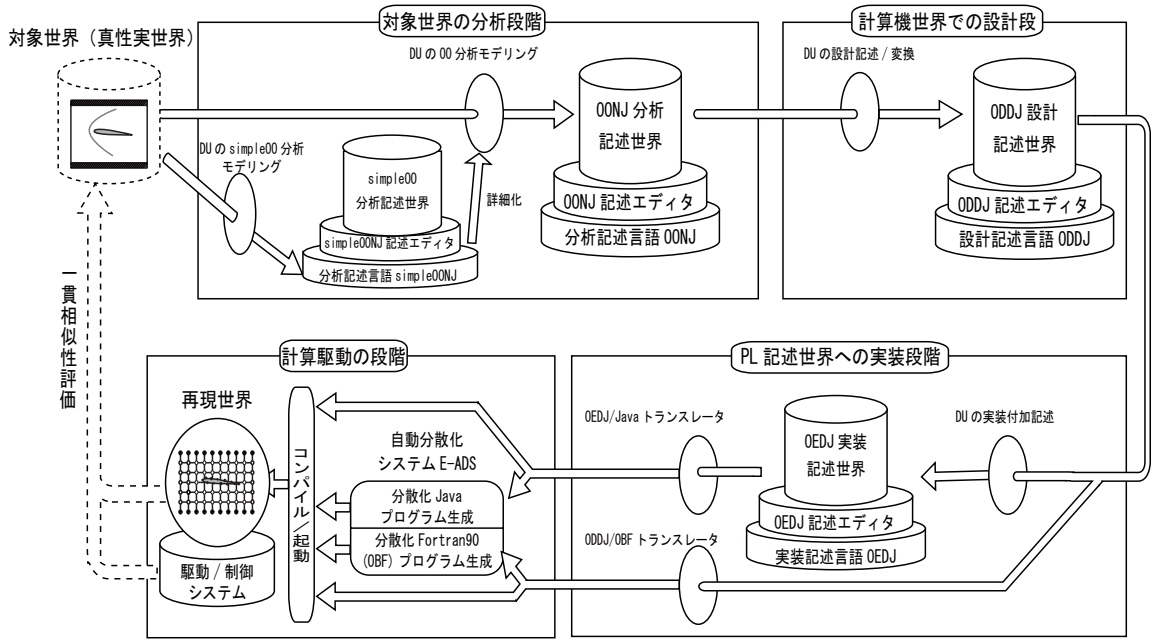

図 2 オブジェクトベース (OB) 一貫相似性モデリング/記述/変換過程の方法
Fig. 2 Object-based, Integrally Consistent and Similar Modeling/descripting/transforming processes

改めて設計する予定の単一記述言語 OOJ で，これが最終形である．記述環境は勿論，言語設計上から見ても OB 一貫単一記述言語 OOJ である．

完成形の OOJ は単一記述言語でありながら，対象世界のみを記述し計算機世界に依存しない分析記述と，計算機世界のみを記述し特定の PL に依存しない故に DU に負担の少ない設計記述と実装記述を可能にし，PL 記述への変換は OOJ トランスレータでほぼ完全な自動化を実現，という記述言語である．これは「DU 向けの究極の記述言語」となろう．

しかしこの様な完成形の単一記述言語 OOJ は未だ実装化されておらず，実際にどの様な言語として可能であろうかという疑問は起きるかも知れない．我々の考えでもこの様な記述言語の設計と構築は，記述言語の記述規則の設計だけでは難しいであろうと考える．その解決策とはつまり，記述環境と一体化した記述言語という形を取って設計/実装されることで実現可能になる道筋があり得るのではないかと考えている．その考えを表現したのが第 5 段階の**記述環境ベースの記述言語**という構想である．

この構想においては記述言語の実体は DU からのイメージとしては記述環境に埋もれて見え，少ししか記述言語に沿っているらしい記述はおこなわないで済む．記述に必要な殆どの規則や記法は記述環境がサポートしてくれるからである．OOJ の中，この記述環境を通して DU から見える部分のみを指した記述言語の名称が DUDJ(Domain User oriented Description Japanese) である．OOJ は DUDJ だけが記述環境内で見えるようになったこの段階において始めて，PL よりは習得および利用が遥かに容易な言語となり，DU の負荷は解放される，と考えている．

4. 四つの記述言語の構築

本章では図 1 の第 4 段階において，“ 総称として ” OOJ と呼ばれる四つの独立記述言語 OONJ，ODDJ，OBF，OEDJ について述べる．実際の設計や構築の報告は各発表に任せ，ここでは OOJ から見た四つの記述言語への要求仕様について述べる．

4.1 四つの記述言語の概念設計仕様

まずモデリング/記述/変換の過程を幾つかの段階に分け，各段階毎に記述や変換する言語とその記述/変換を支援する環境を開発し，各段階間で (半) 自動変換で実現することで，DU にとって有用なプログラム開発環境を作るという戦略を構想した．その戦略の下，以下のような方針を定めた．

1. 段階分けとしては，分析，設計，実装，駆動 (実

行) に分ける．この段階分けは次節の記述言語の任務境界の切り分けの議論からして妥当なものと考えられる．ただし，駆動 (実行) 段階については，実装段階の出力が既存 PL 記述であるため，特別なシステム構築は考えていない．

2. 当初の開発段階では上記各開発段階において独立的に記述言語とその記述開発支援環境を開発する．

3. 各段階間の記述の変換は DU 自身又は各段階の言語仕様を考慮したトランスレータを用いて (半) 自動変換を行う．

4. 記述環境とは主として構造化エディタを主ツールとする．記述するための言語間の表現変換はトランスレータを開発して自動変換させる．

5. その他に計算機内部表現言語として，また，各段階の記述言語記述間の変換を行う共通データ表現形式として XML を用いる．

6. 言語は NJ(自然日本語) を主用する“ 記述言語 ”という形式とする．

7. 記述言語は具体的には分析，設計，実装の各段階に対応させるものとして分析段階には OONJ[1),7)]，設計段階には ODDJ[2)]，実装段階には二種類の記述言語 OEDJ[3)] と OBF[4)] を設計構築する．

4.2 四つの記述言語の要求と持つべき特性

分析記述言語 OONJ[1),7)]

1.OONJ の目的は，DU が再現シミュレーションを実現したい対象世界を正確かつ詳細に記述 (写像) することだけである．したがって OONJ に基づく記述には計算機世界に関する用語や記述は一切使わない．つまり，対象世界の記述力の強さが最重要必須条件である．

2. 記述する事項，使用用語等は対象世界，即ち各ドメインで用いられているものをそのまま使用可でなくては行けない．

3. 記述は各ドメインの用語を使って全くの NJ として読めなくてはならない．

4. 次の設計段階での「計算機世界」の事項や用語を使わない．

設計記述言語 ODDJ[2)]

1. 設計段階の ODDJ は，計算機システム内部においてシミュレーション駆動させることを想定した“ 計算機世界 ”特有の記述を行うと共に，OONJ から受け取った記述を計算機世界内部の仕様に変換すること (だけ) を目的とする．

2. 計算機世界での設計記述言語として，一般的/汎用的な用語や事項は使うが，特定 PL に依存した事項，概念，用語を使わない．

3.NJ で書かれた記述については，計算機世界に通用する記述に変換するために DU はその変換作業を全て担当しなければならない．これは DU しかできない DU 必須の作業である．

4. データの型の概念，処理 (又はメソッド.“ 振舞い ”ではなくなる)，関数の概念等々が導入される必要がある．

4. ただし計算機上での実行 (駆動) には必ず用いられる PL(Programming Language) に関わる用語や記述/表現は原則として用いない．汎用の計算機世界システムを想定した汎用の概念と用語で記述できる範囲内に仕様が限定される．

実装記述言語 OEDJ[3)]

1.ODDJ から受け取った設計記述 (XML 表記) は自動的に OEDJ 記述に変換されて OEDJ エディタ画面に表示される．OEDJ 独自の実装表現形式 (言語表現) で画面への表示しこれの修正機能を持たせる．

2.OEDJ の記述規則は特定の PL の構文規則から独立でなければならない．OEDJ の記述規則は多くの PL の汎用的な，即ち共通に使われている，構文規則を採用することは構わない．したがって，制約としては Java 出力には DU は触れる実力も知識もその気も無いという前提である．

3.OEDJ 記述に対して DU が OK を出せば，(DU の入力支援等無しに) 自動的に特定 PL 記述が出力される．

4. 出力 PL は当面 Java とする．これは OO では最も多く使われている PL だと考えられるからである．トランスレータを変えることで他の PL にも変換可能とする．

5. 最終的には自動分散化機能を持たせる．

実装記述言語 OBF[4)]

1.OBF は OB に準拠した Fortran90 拡張記述言語である．出力 PL は Fortran90 である．Fortran90 が DU の利用言語として最も多く (多分大多数に) 使われている主要な PL であろうというのが理由である．

2.OBF では OEDJ と異なり，ODDJ 記述が実装記述言語に近いと考え，ODDJ 記述から OBF 記述への変換には特定 PL から独立した (非依存の) 汎用の実装記述言語を中間に設けて DU が入力や変更 (編集) するべきデータは殆ど無いものと考え，直接に OBF トランスレータに掛ける方式を取る．

3.OBF に関する僅かな知識を習得するだけで，Fortran90 プログラムと同じ扱いが出来る．DU は自身が常用するエディタやコンパイラを用いて，DU 自身の通常行ってきた作業の範囲内で直接に自身でプログラムを修正/完成させる．

4.OBF 記述は更に自動分散化記述のプログラムに変換する E-ADS に掛けることで分散計算用のプログラム“ にも ”自動変換/出力される．

5. 一貫単一記述言語 OOJ の構成の試み

四つの記述言語とその記述環境の設計/実装/駆動の状況から，一貫単一記述言語 OOJ の構成の提案を試みる．まず記述言語の側面の構成であるが，四つの記述言語の現設計状況からは，

1.OOJ の仕様は DU に向けた「表の言語仕様」の部分と，四つの言語記述間における「自動変換の仕様」の部分とに分ける．

2.OONJ の言語仕様は「表の言語仕様」の部分としてそのまま OOJ に残す．理由は OONJ は DU に向けた言語であることもあるし，DU に必要な対象世界内部の要素は全て DU が記述しなくてはならないからである．

3.ODDJ 以降の言語仕様において，DU に新たに記述を要求する要素の新規な記述部分や，データを提供する必要のある部分は記述環境の該当部分と共に DU 向きの言語仕様に残す．

4. それ以外については原則的に言語仕様の表面には出さず，自動変換仕様の部分に編入する．

5. 記述環境については，3.3 節で OOJ 構築上の三つのレベル分けで述べた様に，まず四つの記述言語の存在を裏側の前提とした上で，統合されて如何にも OOJ が実現されたように，DU からは見える/扱える/動く記述環境を設計する必要がある．

6. まとめと今後の課題

我々の研究段階は図 1 でいえば，第 4 段階の個々の記述言語とその記述環境が稼働し始めたところにある．従って，まず当面の目標はこれらを十分な稼働状態にするべく，完成度を上げることに注力する．その間に

(1)DU に多くの記述例を書いて貰いその意見や記述結果などを考察して個々の記述言語とその記述環境と DU との親和性や，使い勝手を高める．これは 3.3 節の［2］や［3］段階の OOJ の設計データを得るためでもある．

(2)UI レベルで統合された OOJ とその記述環境という第 4 段階を構築する．

第 4 段階が完成すれば，この段階においても，DU にとってかなりのモデリングや記述の容易さ，が期待できると推定している．更には

今後の課題は勿論，図 1 の第 5 段階に出来るだけ早期に進むことである．この段階は第 4 段階までの基礎研究を基にした実用開発，という性格が強く，学外からのいわゆる競争的研究資金の獲得によるプロジェクト実施が必要であり，現在その計画の詳細策定と申請を行っているところである．

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, Nov.29，(2005).
2) 川澄成章，畠山正行，野口和義，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, 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, Nov.29，(2005).
4) 齋藤正樹，畠山正行，オブジェクトベース Fortran の設計とそのトランスレータの開発，第 150 回 SE 研究会報告, 2005-SE-150，Nov.29，(2005).
5) 畠山正行，オブジェクト指向分析モデリングの明示形式化・詳細化・手順化，シミュレーション学会誌，**21**-4，295/309, Dec., (2003).
6) 畠山正行，オブジェクト指向自然日本語構造化フレーム OOSF の設計と表現技法，シミュレーション学会誌，**22**-4，195/209, Dec., (2004).
7) 畠山正行, 松本賢人，オブジェクト指向自然日本語記述言語 OONJ の設計とその記述環境, 情報処理学会第 102 回 HPC 研究会報告, 2005-HPC-102, 13/22，(2005).