社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－MPS－58 (15)
2006／3／17

オブジェクトベース一貫日本語記述言語OOJに基づくプログラムの自動生成

畠山正行[†] 松本賢人[††] 川澄成章[††]
齋藤正樹[††] 加藤木和夫[†††]
野口和義[†] 安藤宣晶[†]

要約: 我々は計算機応用 (非情報系) の分野において科学技術計算や流体解析，分子の合成設計等々を行う技術者や研究者たち (ドメインユーザ (DU) と呼ぶ) のために DU の専門分野 (ドメイン) のプログラム開発向けの記述言語 OOJ とその開発支援環境の研究を行ってきた．OOJ は自然日本語をベースにし，分析から実装に至るまで一貫してオブジェクト指向 (OO) に基づいて記述できる (本来は) 単一の言語である．完成した OOJ 記述は 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を構築している．ただし現時点では分析/設計/実装までを四つの記述言語に分割し，その記述支援環境と共に開発中である．現時点では 1000 行程度の分析記述が設計，実装の各段階を経た後にプログラムに自動生成されている． 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，自動生成され始めた四つの記述言語の仕様を検討して，「一貫」記述言語として如何にそれらを統合化するかについて提案する．

An Automatic Program Generations based on Object-based, Integrally Consistent Description Japanese OOJ

Masayuki Hatakeyama,[†] Yoshito Matsumoto,[††]
Shigeaki Kawasumi,[††] Masaki Saito,[††] Kazuo Katougi,[†††]
Kazuyoshi Noguchi[†] and Noriaki Ando[†]

Abstract: The aim of the present paper is to design and provide a new Object-Oriented (OO) description Japanese OOJ and its description support environments for the Domain Users (DUs) who analyze and describe the target phenomena in their own expert domain. To attain the aim, four description languages, that is the OONJ for OO analysis stage, the ODDJ for OO design stage, the OBF and the OEDJ for programming stage, have been designed and developed. The descriptions out of the OBF and OEDJ are translated into the Java and Fortran90 programs, respectively. At the present stage, we have succeeded in generating the Java, Fortran90, and C++ programs out of the ODDJ and OEDJ descriptions for the description example the "Atmospheric circulations of the water". These programs have compiled and executed succesfully. In the next stage, we aims at constituting an integrally consistent description language OOJ from the analysis stage up to the programming one.

1. はじめに：想定ユーザと開発の狙い

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ユーザは，ドメインユーザ (以降，DU と略) と呼ぶ．彼等は自身の専門分野 (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．情報分野以外の技術/計算/研究開発等のドメイン (専門分野) で，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人たちである．

そこで本研究の目的は，自身のドメインの対象世界の分析/設計/実装の過程を支援する技術であり，その核になるのが OO に基づく記述言語系 OOJ とその記述環境である．その様子を図 1 に示す．本発表の目的は DU が OO に基づいて分析記述を行い，それを設計記述に変換し，プログラミング言語に共通なデータを記述すればその記述がプログラムに自動変換されるシステムである．

問題は，(1) プログラムの自動生成機構，(2)DU が習得する苦労の無い (少ない) 記述言語の実装が実現で

† 茨城大学工学部情報工学科
Department of Computer & Information Sciences, Faculty of Engineering, Ibaraki University
†† 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
††† 加藤木技術士事務所, Katougi Tecnical Office

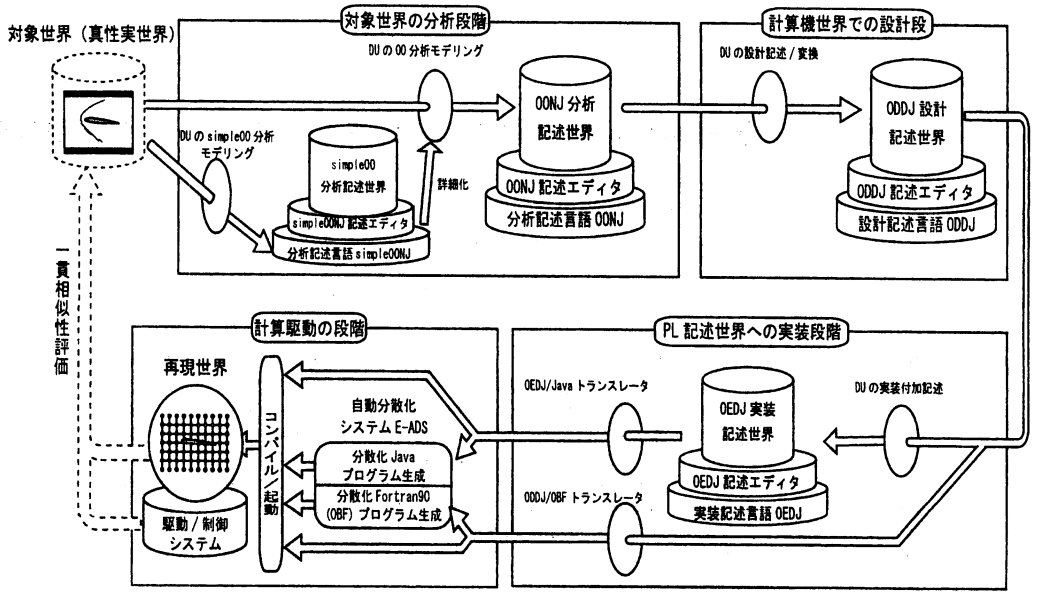

図 1 オブジェクトベース (OB) 一貫相似性モデリング/記述/変換過程の方法
Fig. 1 Object-based, integrally consistent and similar description and transformation processes

きるか否か，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，OO の特性も大いに活用して DU の特性に巧く適用させることで，あるいは DU が持つ特性を究極まで利用してすることで，実現の可能性がある点に着目した．

この様な方式が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) 除き，計算機の“応用利用ユーザ”を目標として，開発の進んでいない科学技術計算分野方面でも使える OO パラダイムの応用技術への適用を主眼としている．本発表ではこの様な条件も踏まえつつ議論して行く．

2. OO 一貫相似性記述言語 OOJ

OOJ の目標は DU 記述からプログラムを自動生成することである．その様な目標の発想の原点は，「DU の本来の義務を十分に果たしさえすれば，計算機 specific な事項については計算機の方の責任で全て処理すべき，それを実現すればプログラムの自動生成☆は可能である.」ということが基本的な考えとしてある．

したがって完成形の 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を排除した分析記述と，計算機世界のみを記述し特定の PL(Programming Language) 記述を排除した設計記述と，自動変換を実現する実装記述を行うことで，PL 記述への変換を実現する記述言語である．

しかしこの様な記述言語の開発は一足飛びには無理である．そこで記述言語 OOJ を当初は四つの記述言語として分割した．それは分析記述言語 OONJ，設計記述言語 ODDJ，実装記述言語は直接トランスレータに掛けるタイプの実装記述言語 OBF と，実装記述言語表現に書き直す OEDJ の四つである．

3. 四つの OO 記述言語の構築

3.1 四つの記述言語の概念設計仕様

まずモデリング/記述/変換の過程を段階分けにし，各段階毎に記述言語とその記述/変換の支援環境を開発し，各段階間で (半) 自動変換することで，DU にとって有用なプログラム開発環境を作る．具体的には以下のような方針を定めた．

1. 分析，設計，実装の三つに分ける．
2. 当初の開発では上記三つの開発段階において独

☆ (注) 本研究における“自動”とは，計算機 specific な事項について変換が自動で実行可能であることのみを指す．DU specific な事項については勿論，自動化は原理的にもできない．

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. 記述環境とは記述支援エディタを主とする.

5. 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のために共通の記述データ交換方式として 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. 分析段階には OONJ[1),4)], 設計段階には ODDJ[5),8)], 実装段階には二種類の記述言語 OEDJ[6)] と OBF[7),9)] を設計構築する.

各言語の記述任務境界の切り分け

(1)OONJ では対象世界の知識だけで構成するので計算機世界の事項や用語を入れない.

(2)ODDJ では計算機世界の一般的/汎用的な用語や事項入れるが PL 世界の事項や用語を入れない.

(3)OEDJ は汎用 PL 仕様の記述言語とする.

(4)OBF は Fortran90 のトランスレータを作る.

3.2 四つの記述言語の開発仕様

分析記述言語 OONJ[1),4)]

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) することである. 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. 記述事項や用語等は各ドメインのものをそのまま使用出来ることが必要である.

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. 次の設計段階の「計算機世界」の事項や用語を使わない.

設計記述言語 ODDJ[5)]

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), 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 に関わる用語や記述/表現は用いない. 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[6)]

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, 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. 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.

実装記述言語 OBF[7)]

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, 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.

3.3 プログラムへの自動変換/自動生成

現時点において, 同時に発表する分析記述言語 OONJ で取り上げた「水の大気循環現象」の分析記述を ODDJ 記述環境を用いた設計記述への変換, OBF への自動変換, OEDJ を用いた実装記述への変換を通って, Java, Fortran90, C++プログラムが自動生成できた.

プログラムは各 PL のコンパイラを用いて変換され, 実行 (駆動) も実際に行われている. ただし, 熱収支か

質量収支の少なくともどちらかに問題を抱えており，数値的には妥当な値ではない．

分析段階の水の大気循環現象の記述は 904 行，147KB のいわばソースコードであり，十分に実用性を検証した記述例とは言えないまでも，小さすぎる記述例ではない．同程度の規模の他の記述例の変換を現在検討中である．

4. OB 一貫単一記述言語 OOJ の構成の試み

前章の実装状況から OOJ を以下のように作る．

1.OONJ，ODDJ，OEDJ，OBF の記述規則は，ひとまず原則そのまま採用し，DU 向けの言語仕様と内部の自動変換の仕様の部分とに分ける．

2.OONJ の言語仕様は「DU 向けの言語仕様」としてそのまま OOJ に残る．分析記述は全て DU が記述しなくてはならないからである．

3. 記述環境 (エディタ) を用いる DU の作業は，

1．DU の手変換を要求する部分，

2．データ入力をする必要のある部分

3．新規記述を要求する部分，

が生じる．新規記述のその主たる部分は NJ 記述を式やライブラリ名等に変換する作業である．データ入力とは例えば，データ型等がある．

4. 上記 1．～3．の DU 負担の部分を除けば，記述環境が DU の記述を全面的に支援して記述させるので，自動変換部分の記述規則は記述言語 OOJ の記述規則としては外す．つまり，DU は記憶しなくとも馴れさえすればよい規則となる．OONJ，ODDJ，OEDJ 各記述環境はまずは GUI だけを統合化する．次に内部処理も含めて統合化する．

5. 計算機内部の自動変換仕様も参考規則としては DU からも参照可能にしておく．

6.OOJ の記述規則の内訳は，OONJ が 70 %くらい，ODDJ が 20 %くらい，OEDJ が 10 %くらいではないかと予想している．OBF は OOJ 記述環境にボタン 1 個だけ顔を出すだけであろう．

7. 記述環境の GUI は統合されて等，当面は 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．

8. 各記述言語間の垣根は本質的に不要になるまで，少なくとも自動変換率がこれ以上は上がらないことが判明するまで残しておく．

5. まとめと今後の課題

分析段階で 1000 行規模の記述例が実行可能なプログラムとして出力され始めたことで，一段落ではある．現在，DU の負担の重さや各段階での変換過程の妥当性等を詳細に検討中であり，それが済めばより高い変換機能や DU 負担の軽減を狙って改訂を進める．同時並行して DU に多くの記述例を書いて貰ってそれらをプログラムに変換する実績を積み，DU との親和性や使い勝手を高めて OOJ へ順次移行して行く．

参 考 文 献

1) 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).

2) 畠山正行, 池田武徳, 生井沢和也, 松本賢人, ドメインユーザにもやさしいオブジェクト指向自然日本語記述言語 simpleOONJ 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, pp.23-26, (2005).

3) 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).

4) 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).

5) 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).

6) 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-OOSE-150, pp.75-82, (Nov. 29, 2005).

7) 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-OOSE-150, pp.83-92, (Nov. 29, 2005)

8) 川澄成章, 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, 平成 17 年度茨城大学大学院情報工学専攻修士論文, 2006 年 3 月.

9) 齋藤正樹, オブジェクト指向設計記述からのプログラム自動生成システム, 平成 17 年度茨城大学大学院情報工学専攻修士論文, 2006 年 3 月.