Vol.2013-MPS-92 No.26
2013/2/28

実世界を直截に記述可能なOOJ記述環境の開発とその利用効果

池田 陽祐[†1]　三塚 恵嗣[†2]　上田 賀一[†3]　畠山 正行[†3]

概要：本論文では従来から数値計算で用いられている離散化の方法を拡張した離散構造化モデルを構築し，分析からプログラム開発までを一貫して行うための記述言語系 OOJ を開発した．OOJ はプログラムの信頼性を向上させるために分析，設計，実装，およびプログラムの 4 つの段階で構成する必要があった．それ故に OOJ を具体的に利用するためには特異な OOJ 記述環境が必要である．そのために特殊な OOJ エディタとトランスレータを開発した．OOJ 記述環境の特徴は離散単位をイメージさせる GUI と科学技術計算分野で広く用いられているエディタの特徴を組み合わせた点にある．これを用いてプログラムの信頼性を検証するための一貫した追跡性を持つ記述環境の実現を目標とした．その利用効果を評価するためにプログラム開発教育に適用した．その結果 OOJ 記述環境は 4 つのプログラム開発の段階を良好に支援して特長あるプログラム開発方式を実現すると共に，OOJ が目標としている分析からプログラムまでの離散単位毎の一貫した追跡性や記述間におけるある種の相似性を検証するために良好な記述環境を提供することを結論できた．またこの記述環境を利用した教育は良好な成果を生むことも結論された．

キーワード：プログラムの信頼性，プログラム開発支援環境，離散構造化モデル

A development of OOJ descriptive environments that straightly representable the real world and its applied usability

Yousuke Ikeda[†1]　Keishi Mitsuka[†2]　Yoshikazu Ueda[†3]　Masayuki Hatakeyama[†3]

Abstract: We have developed a simple discrete and structured model for the scientific and engineering calculations, and have designed a descriptive language system OOJ based on this discrete and structured model. OOJ contains four inner stages for developing the program, that is the analysis, the design, the implementation, and the program stages. The purpose of OOJ is to establish the reliability of the programs. Therefore, a special editor is essentially needed, and is called the OOJ description enviroments. This peculiar OOJ description enviroments has been developed by fitting together with the usual GUI systems for the scientific and the engineering purposes and the user needs of the assumed target users. The description enviroments have been developed with the translators. We have applied this description enviroments to the program development education to evaluate the usability for the assumed target users. As the results, it has been confirmed that the fundamentals for verifying the reliability of the programs have been established, and that the description enviroments have been confirmed to be sufficiently useful for the assumed target uses.

Keywords: reliability of program, program development support enviroments , discreted and structured model

1. はじめに

本研究においては科学技術計算のプログラムの開発を目

[†1] 現在，茨城大学大学院理工学研究科情報・システム科学専攻
Presently with Graduate School of Information and System Science, Ibaraki University
[†2] 現在，株式会社日立システムズ
Presently with Hitachi Systems, Ltd.
[†3] 現在，茨城大学工学部情報工学科
Presently with Department of Computer and Information Sciences, Ibaraki University

Vol.2013-MPS-92 No.26
2013/2/28

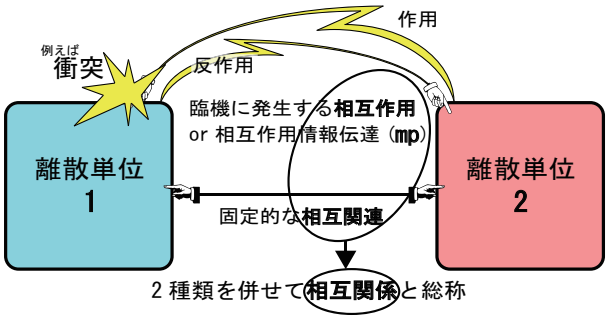

図 1 離散構造化モデル：離散化とその構造化の方法
Fig. 1 A concept expression of discrete and structured model.

的とした記述言語系 OOJ[*1][1] に基づく OOJ 記述環境について述べる．その狙いとしては，当初の分析記述がプログラムに正確に反映する仕組みを作ることである [2],[3]．それはプログラムの信頼性 [4]，[5]，[6] を検証できることである．そこで本論文では OOJ 記述環境に焦点を当て，その特長ある設計と実装，適用した結果としての有効性について述べる．

2. 記述言語 OOJ の概要と開発目的

2.1 離散構造化モデルとその記述方法

我々は，離散化の手法とオブジェクト指向パラダイムを組み上げて 離散構造化モデルを設計し，その概念図を**図 1** に示す．

離散構造化モデルは離散単位と相互関係から構成される簡潔なモデルであり，実世界で離散的に存在している“モノ，物や概念”を直截に離散単位とする．そして離散単位間の静的及び動的な関係[*2]を相互関係として付与する．

このように実世界の離散的な“モノ，物や概念”をそのまま離散単位とし，その離散単位間の構造をオブジェクト指向パラダイムで用いられる mp や集約などでモデリングを行う．このモデルを直截に記述するための記述言語 OONJ を開発した [8]，[9]．

2.2 記述言語系 OOJ の概要

科学技術計算の目的は計算結果を得ることであり，離散構造化モデルの記述に特化した記述言語 OONJ のみでは計算できない．そこで OONJ の記述から数値計算プログラムを得るための方法が必要である．

その一方で近年においては数値計算プログラムの信頼性 [4]，[5] も言及されており，OONJ の記述と等価のプログラムであることを検証でき，信頼性を個人規模で検証する機能も開発する必要がある．

以上のことを実現するために記述言語系 OOJ[1] の設計を行い，その全体の過程を**図 2** に示す．図 2 にあるよう開発過程を各段階に分け，仮に分析・設計・実装と名付けた．

2.3 OOJ の利用想定ユーザ

OOJ の想定ユーザは，自身の分野に近い記述ができる記述言語を望むと共に，自身の専門分野の情報を十分提供すれば計算結果が得られることを望む数値計算ユーザである．OOJ には教育の目的もあるため理工系の学部生，大学院生を対象とする．

3. OOJ 記述環境の概念設計

3.1 MATLAB

MATLAB[10] は，行列計算やベクトル計算などの数値計算を得意としており，常微分方程式や各種解析手法などの豊富なライブラリと記述・実行環境，グラフィカルな計算結果の表示を提供することでプログラミングを支援する．記述形式はコマンド形式，テキスト形式に加え，Simulink[11] によるブロックダイアグラム形式がある．

3.2 Modelica 言語と OpenModelica

様々な物理現象をモデリングし，評価することが可能な言語の 1 つに Modelica 言語 [12] がある．Modelica は MSL(Modelica Standard Library) と呼ばれる豊富なライブラリを組み合わせることでモデルを構築する．また，Modelica ソルバーを用いることで連立させた方程式を数式処理し C のプログラムを生成し，それを実行することで計算結果を得る．

ソフトウェアの 1 つに OpenModelica[13] がある．Open Modelica での記述形式は MATLAB と同様にコマンド形式とテキスト形式，ブロックダイアグラム形式の 3 種類を選択して行う．

3.3 OOJ 記述環境の概念設計

まず，想定ユーザの学習負担を考慮し，MATLAB や Modelica の画面構成や操作性を組み込む．記述形式は想定ユーザが容易に使えるよう，テキスト形式と離散構造化モデルを組み合わせて設計する．

想定ユーザに適したライブラリを当初の段階から抽出することが困難であるため想定ユーザと意見交換を行うことでライブラリを拡張できるよう設計する．それに加えてプログラミングを OOJ に基づいて形式化し，それを支援するための機構も実装する．

また，小規模な個人向けのプログラムであるので，分析者とプログラム開発者が同一人物である．OOJ で行う信頼性の作業は「分析記述からプログラムまでの作業過程の記述が同等内容であることを検証すること」であり，想定ユーザとの議論の結果，直感的に信頼性を理解しやすい性質として離散単位毎に追跡性 (Traceability)[14] を持つよう設計する．

[*1] Object Oriented Japanese の略
[*2] これを特別な用語として相互作用情報伝達（= message passing 略して **mp**）と呼ぶ [7]．この用語は以降省略形の mp を用いる．

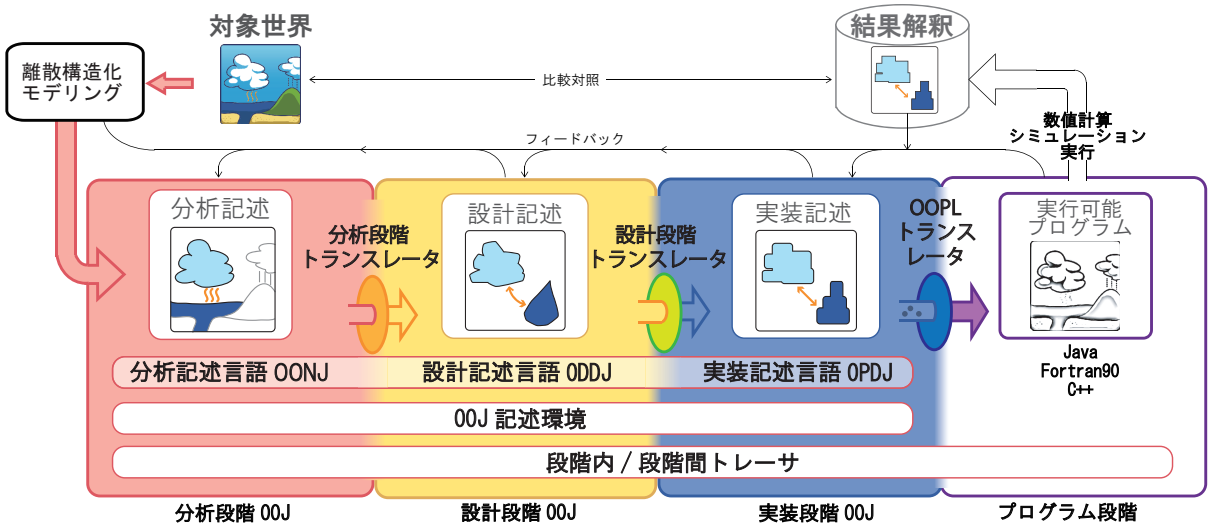

図 2 OOJ 記述環境を基盤とした開発過程の概念図

Fig. 2 A whole concept diagram of program development processes based on OOJ description enviroments.

4. OOJ 記述環境の実現と構成

OOJ を基盤言語とした記述環境開発の設計方針として 3 つを設定した.

(1) 離散構造化モデルの記述方法
(2) 分析からプログラムまでの一貫した記述支援機構
(3) ユーザがイメージ設計の当初から参画した設計

4.1 OOJ 記述環境の構成

図 2 のような 4 つの段階を記述して分析からプログラムまでの過程を順次詳細化していく方法をとる OOJ 記述環境の概要を**図 3** に示す. 構成は記述作業を行う OOJ エディタ (図 3 の (A)) と変換を行うトランスレータ (図 3 の (B), (C), (D)) からなり, これにより想定ユーザと OOJ 記述環境の作業を切り分けている.

OOJ エディタでは分析段階から実装段階までを一貫して行えるよう 1 つのアプリケーションとして実装したが, 段階毎に異なる記述作業を設定しているため支援する機能が異なり, 3 段階毎に記述画面を開発した.

分析段階から設計段階のトランスレータと設計段階から実装段階のトランスレータでは記述形式のみの変換を行う. 実装段階からプログラムへのトランスレータは, プログラム変換 [15] を用いて種類毎に実装している.

4.2 OOJ エディタの概要

分析段階の記述画面である**図 4** を用いて OOJ エディタの概要を示す. OOJ エディタは, 離散構造化モデルを基盤としたアプリケーションであり, 実世界の“モノや物”, 概念を直感的に記述できるよう設計してある.

実世界のモノや概念に相当する最大離散単位を中心とし, Open Modelica や MATLAB でも同様の画面構成を取り入れた (図 4 左上の (A)). 図 1 のイメージを用いて実世界を記述できるように離散単位を分かりやすく四角形の枠線で囲むようにし, “離散的”な操作感として全ての離散単位をセルエディタとして実装した.

図 4 中央の (B) が最大離散単位である. 最大離散単位の属性や振る舞いも離散単位であり, 図 4 中央の (C) に示した中間離散単位, より詳細に記述するために図 4 中央の (D) の最小離散単位を用いる. このように数式や日本語文を最小離散単位として記述し, これらを構造化することで中間離散単位, 最大離散単位の内容を表現し, 実世界の“モノ”を全て離散単位として直截に記述することができる.

構造化を行うための相互関係の記述例を図 4 右上の (E) に示す. 相互関係は「向き」と「種類」,「相互関係の相手先」の 3 項目から構成され,「相互関連」と「相互作用情報伝達」は同じ形式で表現される.

最小離散単位の構造化には日本語の箇条書き, プログラムの制御構造の一部を流用して行えるよう設計している (図 4 中央の (F)). 各離散単位を区別する識別子 (図 4 の (G)) や詳細な解説文を記述できるようにした (図 4 左下の (H)).

4.3 プログラムまでの OOJ エディタの機能

図 2 の分析・設計・実装の 3 段階毎の記述画面は, 図 4 左上の (I) のタブ形式で切り替えることが可能である. ただし, 一貫した記述作業を実現するために設計段階でも離散単位を中心とした画面構成と操作性を実現している. 設計段階ではデータ型やアクセス修飾子の設定を行い, この設定は分析段階のトランスレータ (図 3 の (B)) で付与され, 想定ユーザは選ぶだけである.

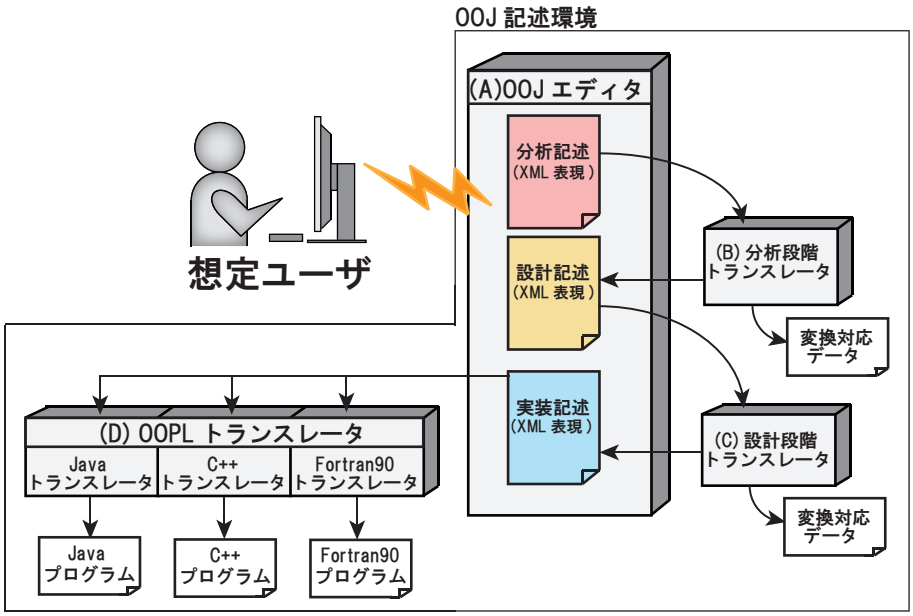

図 3　OOJ 記述環境のシステム構成

Fig. 3　System organization of OOJ description enviroments.

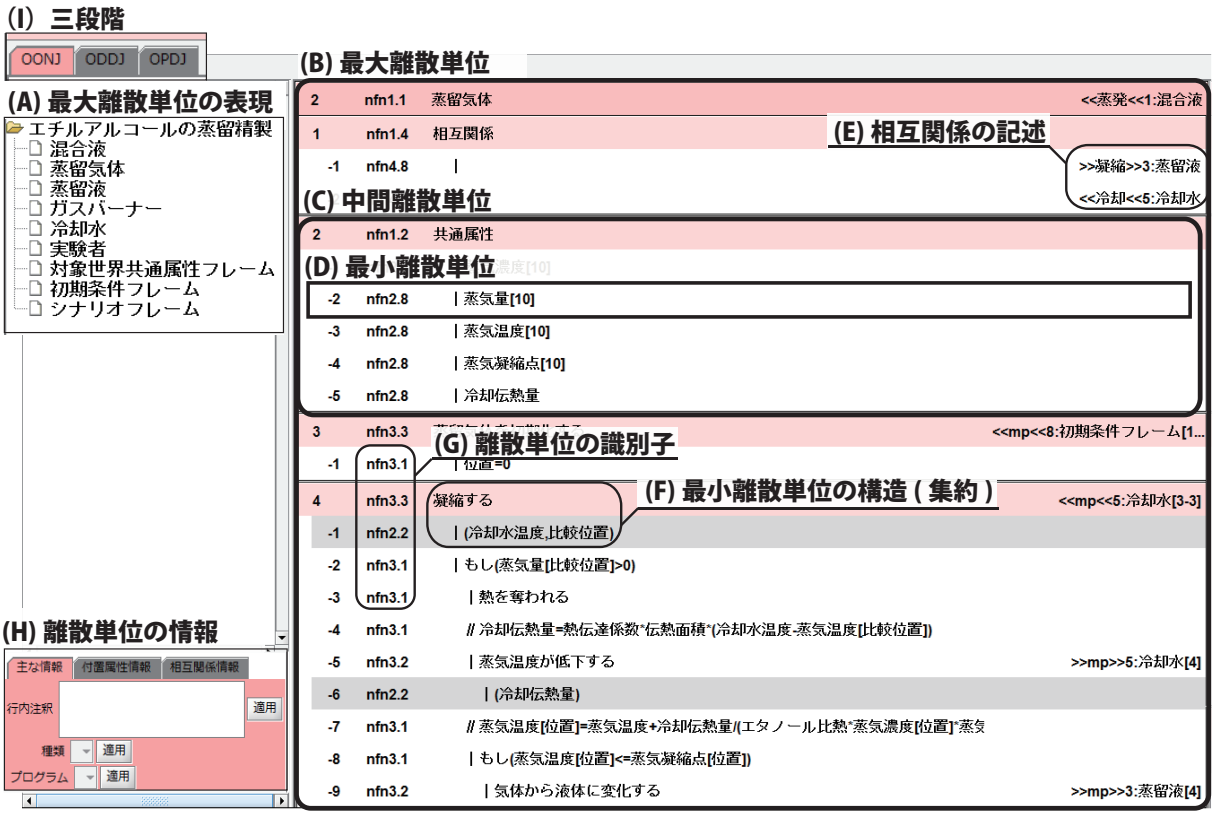

図 4　OOJ エディタの分析段階の画面例:エチルアルコールの蒸留を例として

Fig. 4　A screen example of description enviroments at the analysis stage.

実装段階ではプログラムへ自動変換するために論理的な矛盾を取り除いたりアルゴリズムの再確認や規定された文法に基づいた形式への変換を行う．また，設計段階と同様に離散単位を中心とした画面構成と操作性を実現している．

以上のように OOJ エディタでは分析段階から実装段階までを全て図 1 のイメージで記述できるよう実装し，実世界の記述を離散単位と構造化を用いて一貫して記述することができる．そのため実世界のモノや概念を直截に離散単位とする離散構造化モデルを適用した場合には，実世界を直截に記述することが可能である．また，離散単位を中心とした設計をしているためライブラリを最大離散単位群として記述することが可能となり，ライブラリの拡張を行う

ことが容易な設計になる．

4.4 トランスレータの概要

分析記述を設計記述へ変換するためのトランスレータ (図 3 の (B)) では，データ型やアクセス修飾子の枠組みを付与する．設計段階から実装段階のトランスレータ (図 3 の (C)) では，プログラムへ自動変換するためのデータが記述されているかを調査する．分析段階から実装段階までのトランスレータは各離散単位に記述形式のみを変換し．最小離散単位の内容は想定ユーザ自身が変換する．ただし，その大半は提供している関数やライブラリを選択して書き込むだけの作業となるよう支援を行なっているため想定ユーザの負担は重くない．

OOPL(オブジェクト指向プログラミング言語) トランスレータ (図 3 の (D)) では，Java，C++，Fortran90 のプログラムへ変換するために 3 つの言語仕様から抽象化した共通仕様を抽出し，それと実装段階の記述言語の規則間に対応関係を設定し，unfold/fold 方式 [15] を用いて実装している．

4.5 OOJ 記述環境の追跡機能

3.3 節で示したように OOJ では離散単位毎に追跡性を持たせるよう設計し，それを機能として組み込んでいる．

トランスレータによる変換前後の対応関係のファイルを変換対応データとして，変換完了と同時に別ファイルとして出力 (図 3) することで行なっている．以上の機能は．分析段階トランスレータと設計段階トランスレータに組み込んでいる．追跡機能を OOPL トランスレータに実装していない理由は，プログラム変換の技術を用いることで等価変換を行なっていることは明らかだからである．

5. OOJ 記述環境の利用効果の評価

5.1 ユーザへの記述実験

OOJ 記述環境の利用効果を確認するために情報工学科 3 年生 18 人，大学院生 5 人に対しても OOJ 記述環境を用いて Java プログラムの開発を行なってもらった．評価方法は各段階の記述と生成されたプログラムの分析とアンケート調査に依った．

5.2 各段階の記述とプログラムから推測される利用効果

3 年生と大学院生 5 人のプログラム変換の達成状況を調べたところ Java プログラムへ変換でき結果を出力したのは，大学院生は 5 人全員であり，3 年生は 18 人中 12 人であった．残りの 6 人全員は実装段階まで到達していた．これにより少なくとも実装段階までの操作は可能であったことが分かる．

プログラムへ変換できなかった理由を調査したところ，細かい文法事項 (例えば数式等の計算処理の文法) が規定通りに書けていないなどの記述ミスが 3 人であり，残りの 3 人は対象世界のモデリングに不備があることが分かった．以上のことから対象とした想定ユーザの大半が離散構造化モデルを用いてプログラムまで到達できたことは，OOJ 記述環境を用いることで一貫した記述作業が行えたといえる．

5.3 アンケート調査から推測される利用効果

3 年生と大学院生へ OOJ 記述環境に対してのアンケート調査を行った．

【OOJ 記述環境の使いやすさ】では，「他のアプリケーションと同程度である．」との回答が大半であった．コメントも含め他のアプリケーションと同程度の使いやすさで一貫した記述作業が行えたことは，OOJ を用いたプログラム開発過程を利用できるまでに容易化しているといえる．

【OOJ エディタの操作性】は「最終的には操作できるようになった．」との回答が大半であった．離散構造化モデルに慣れると OOJ エディタ上の操作性が良好になるとのコメントがあり，これにより OOJ エディタの操作イメージが離散構造化モデルであることを裏付けている．

追跡性を具体化した機能の利用アンケートは大学院生のみに行った[*3]．その結果，5 人全員が「最小離散単位に対しては，分析段階から実装段階まで追跡して変換が正しいことを確認した．」，「プログラムの信頼性が重要なことが分かった．」と回答した．

5.4 OOJ 記述環境の評価

以上の評価より，学部 3 年生や大学院生の経験と知識であれば OOJ 記述環境を用いて分析段階から実装段階までの記述が十分に行えると結論できる．

ただし，信頼性を検証する機構が不十分であったとしても大学院生 5 人は OOJ の目的である「プログラムの信頼性を認識・理解する．」を達成することはできた故に，想定した利用効果を得ることができたと判定した．

以上から離散構造化モデルのイメージで操作できる OOJ エディタの開発ができ，一貫した追跡性を持つプログラム開発を行える記述環境の開発ができたといえる．また，大学院生では想定した OOJ の信頼性に対する利用効果を得ることができた．

6. 関連研究との比較

6.1 MATLAB，Modelica との比較

3.3 節で既に取り上げたように MATLAB と Modelica は，豊富なライブラリとユーザに適したインターフェイスを用いたソフトウェアであり，OOJ とはプログラム開発手法や教育の目的が異なっている．さらに想定ユーザの多くが将来 MATLAB や Modelica を利用すると推測される．

[*3] 3 年生に対しては講義内で十分な説明を行わなかったためアンケートを行なっていない．

これらのことからMATLABやModelicaとOOJは用途や目的に応じて使い分けるべきであり，またMATLABとModelicaへのプログラム変換を実現することでOOJの有用性が高まると期待できる．

6.2 MDAとの比較

MDAはモデル駆動開発の手法の1つであり，OMGで正式に規格として認められている[16]．

使用される開発ツールとしてxUMLを基盤としたBridgePoint，iUMLiteがある．iUMLiteを用いた場合，システム全体をユースケース図などを用いてドメイン毎に分割して静的モデルと動的モデルを記述し，コードを自動生成する．

このようにモデルを複数の段階で記述し，ソースコードを自動生成する仕組みはOOJの開発プロセスと類似点が多い．ただし，MDAは多くの企業で行われる製品開発や組込みシステムの開発，いわゆる業務処理システムの開発で使用されており，iUMLiteをOOJが対象としている教育的な側面を含めた個人向けの科学技術計算プログラムの開発に利用する場合には，新たに多くの知識と技術が必要となる．このことからMDAで用いられているソフトウェアはOOJの目的に適していないといえる．

以上のことより，MDAのiUMLiteとOOJ記述環境は目的に応じて使い分けるべきであり，OOJ記述環境は個人向けの科学技術計算プログラムを開発・教育する目的に適しているといえる．

7. 結論と今後の課題

本論文ではOOJ記述環境の設計と開発，評価を行い，特徴的な4つのプログラム開発段階を十分に支援した開発方式を実現することができ，OOJの目的であるプログラムの信頼性を向上させ，教育するための基盤となる特徴を持つことが分かった．大学院生においてはOOJ記述環境が想定した利用効果を得られた確認でき，ユーザビリティの問題や信頼性評価機構の必要性が明らかになった．

今後の課題として，実用化の観点から記述例の蓄積，記述実験，詳細な記述ノウハウやライブラリ等の蓄積がある．

参考文献

[1] 畠山 正行，池田 陽祐，三塚 恵嗣，大木 幹生，加藤木 和夫，上田 賀一，離散構造化モデル記述言語系OOJの構築と効果的な利用法—分析からプログラムまでの一貫開発とV&V評価実現の検討—，第92回MPS研究会，2012-MPS-92-O (Feb. 2013).

[2] 畠山正行，高度なシミュレーションのためのオブジェクトベース一貫モデリング過程論とその駆動支援環境，第1回情 報処理学会 MPS 研究会研究報告，(May，1995).

[3] 畠山正行，一貫モデリング過程論に基づく物理世界のオブジェクトモデル，情報処理学会第20回MPS研究会 研究報告，pp.7-12，1998年7月24日．

[4] 電子情報通信学会，入手先〈http://www.ieice-hbkb.org/files/01/01gun_12hen_06.pdf〉，(accessed 2013-01-30).

[5] 日本計算工学会編，工学シミュレーションの標準手順，JSCESS-HQC002:2011 A Model Procedure for Engineering Simulation，日本計算工学会，2011年5月31日．

[6] American Society of Mechanical Engineers，Guide for Verification and Validation in Computational Solid Mechanics，2006.

[7] 青木淳，オブジェクト指向システム分析設計入門 第2章：広義のオブジェクト指向，(株) ソフトリサーチセンター，(1993).

[8] 池田陽祐，三塚恵嗣，加藤木和夫，大木幹生，上田賀一，畠山正行，UMLとの比較に基づくオブジェクト指向分析設計記述言語OONJの評価，情報処理学会論文誌：数理モデル化と応用，発行予定(2012年).

[9] 池田陽祐，三塚恵嗣，上田賀一，畠山正行，UMLとの比較評価に基づくオブジェクト指向分析設計記述言語OONJの記述技法の特徴，情報処理学会論文誌：数理モデル化と応用，発行予定(2012年).

[10] MathWorks MATLAB(online)，入手先〈http://www.mathworks.co.jp/products/matlab/〉，(accessed 2011-12-26).

[11] Math Works, Simulink, MathWorks(online)，入手先〈http://www.mathworks.co.jp/products/simulink/description1.html〉，(accessed 2012-04-09).

[12] Modelica，Documents-Modelica Version 3.2 Revision1 - Feb.2012(online)，入手先〈https://www.modelica.org/documents〉，(accessed 2013-01-30).

[13] OpenModelica，Introduction(online)，入手先〈https://www.openmodelica.org/〉，(accessed 2013-01-30).

[14] IEEE Std. 830-1998，IEEE Recommended Practice for Software Requirements Specification.

[15] 吉田紀彦，自動プログラミングハンドブック第4章「プログラム変換手法による自動プログラミング」，オーム社，pp. 109-120(1989).

[16] Object Management Group，MDA Specifications(online)，入手先〈http://www.omg.org/mda/specs.htm〉，(accessed 2013-01-30).