Vol.2013-MPS-92 No.25
2013/2/28

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—

畠山 正行[†1]　池田 陽祐[†2]　三塚 恵嗣[†3]　大木 幹生[†4]　加藤木 和夫[†5]　上田 賀一[†1]

概要：　本論文では分析・設計・実装・プログラムの4つの段階を順次追って一貫したプログラム開発ができる記述言語系OOJの開発を報告する．OOJの適用分野は科学技術計算分野であり，この分野に適した離散構造化モデルを開発した．そして4つの段階間の記述が必ず「同等内容の別表現」となる特性，一貫相似性，を実現する設計とした．この特性により分析記述は忠実にプログラムに反映され，開発過程の正しさや成果の妥当性が確保される．この特性はV&Vの実現でもあり，プログラムの信頼性向上に貢献する故にプログラム開発に有効な道具になる．OOJは開発過程における信頼性の向上やV&Vの評価の実現，および理解と実記述の容易性の特長の故に，効果的な利用法を持つ言語系であると結論できた．

キーワード：離散構造化モデル，記述言語，一貫相似性，プログラム開発，プログラムの信頼性

A design of discretized and structured model descriptive language system OOJ and its effective utilization method

Masayuki Hatakeyama[†1]　Yousuke Ikeda[†2]　Keishi Mitsuka[†3]　Mikio Ohki[†4]　Kazuo Katougi[†5]
Yoshikazu Ueda[†1]

Abstract: In the present paper, we will report a descriptive language system OOJ that can be applied from the analysis stage up to the program stage throughout the design and the implementation stage. OOJ is designed based on the discrete and structured model and applied in the fields of science and engineering calculations. In OOJ, the corresponding descriptions among four stages are integrally similar. That is, the descriptions in these four stages have some different phrases but the equivalent contents. These characteristics realize the concept of the V&V, and contribute to the upgrade of the reliability of the program. We got the conclusion with high feasibility that OOJ contributes to the upgrade of the program reliability, to the realization of the V&V concept, and finally to the easiness and usability.

Keywords: discrete and structured model, descriptive language, integral consistency and similarity, program development, reliability of program

1. はじめに

自然現象の解明や工学的問題の解決のために，現象の再現精度の高いシミュレーションは従来にも増して活発に行われている [1]．それに伴ってその処理プログラムもますます高度化・複雑化しており，プログラム開発[*1]技術の大幅

†1 現在，茨城大学工学部情報工学科
Presently with Department of Computer and Information Sciences, Ibaraki University
†2 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
†3 現在，株式会社日立システムズ
Presently with Hitachi Systems, Ltd.
†4 現在，群馬工業高等専門学校教育研究支援センター
Presently with Technical Support Center for Education and Research, Gunma National College of Technology
†5 現在，茨城県立産業技術短期大学
Presently with Ibaraki Prefectural Industrial Technology Junior College

*1 本論文で多く用いる「プログラム開発」という用語は個人で開発する小規模なものを指す用語として用いる．「ソフトウェア開発」

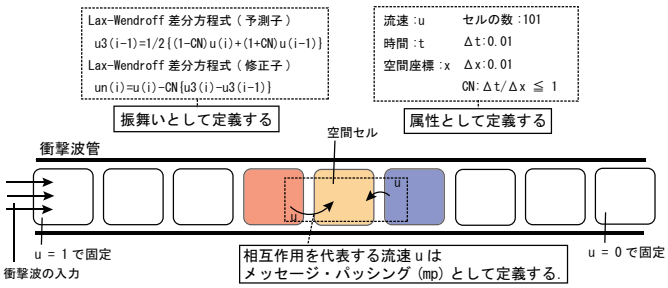

図 1　一次元衝撃波流れの離散化されたモデルの概念図
Fig. 1　A conceptual illustration of discrete model for one-dimensionnal shock wave flow

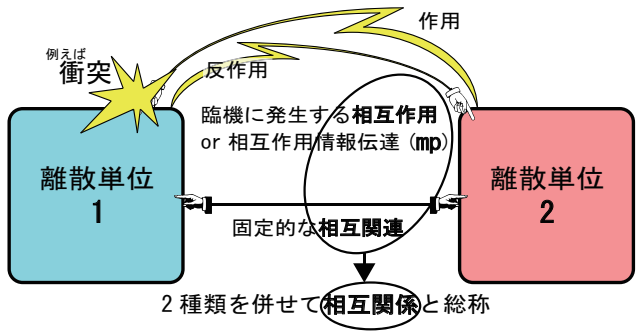

図 2　離散構造化モデルの概念図
Fig. 2　A concept expression of discrete and structured model

な向上も社会的な要請となっている．それらの要請の一半に応えるべく，主として科学技術計算のプログラムの開発を目的とした記述言語系 OOJ [*2]の構築を行ってきた [4]．その狙い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を目的とした [5]．それはプログラムの信頼性 [6], [7] の確保と言い換えられる．本論文ではその点に関する解決を OOJ を利用して試みる。

2. 離散構造化モデルの構築とその背景・狙い

2.1 想定対象世界と想定ユーザ [2], [3]

想定対象世界は主として科学技術分野の計算や再現シミュレーションである．そのような対象世界の典型例として，衝撃波管内を伝播する衝撃波流れ [8] の再現シミュレーションである**図 1** の「一次元衝撃波流れ」を取り上げた．OOJ の利用想定ユーザはいわゆる数値計算ユーザである．

2.2 離散構造化モデル

図 1 の例では連続媒体の空気の満ちた一次元空間を“セル”と呼ぶ微小な空間 (厚さ Δ x) に区切り，セルの振舞いとしての数式（差分方程式）と属性を持つ (図 1 上部に示す)．この作業は離散化と呼ぶモデリングであり，**離散 (化) モデル**といわれる．本論文では，この離散化モデルと最も基本的なオブジェクト指向[*3]モデルとの共通部分で構成したモデルを構築する．

まず図 1 のセルを改めて離散単位として着目し，その離散単位間を**図 2** の相互関係で結ぶことにより構造化し，対象世界を近似モデル表現に記述する．この相互関係を付与することを**構造化**と呼ぶ．このような近似表現を施したモデルを**離散構造化モデル** [9] と呼ぶ．

この離散構造化モデルは OO モデルでもある．そこで両モデルの比較を**表 1** に示した．離散構造化モデルは図 1 の差分方程式から離散単位間の相互作用を行う物理量を抽出し，メッセージパッシング[*4]という情報モデルを使って流速 u を他のセルに送ることで，手続き型の計算を OO 計算に変換した [9].

表 1　離散構造化モデルとオブジェクト指向モデル
Table 1　Discreted and structured model and OO model

事項	離散構造化モデル	オブジェクト指向モデル
必須事項	離散単位	クラス，インスタンス
	属性	属性
	振る舞い	メソッド，ステートメント
	相互関係 (図 2 参照)	関係
	×	カプセル化（情報隠蔽）
	×	継承
特徴事項	集約	集約
	相互作用	メッセージ/シグナル
	×	ポリモルフィズム
	×	動的結合

2.3 一貫相似性という概念と用語の採用

本論文で用いる**相似性**はある段階の離散単位と次の段階の離散単位が『表現形式は異なっても，同等/同値の記述や計算結果になる特性』を指す用語であると定義する[*5]．近似的に同等/同値であることを許す場合もある．

次に，**一貫相似性**とは 4 つの段階の対応する離散単位 (群) において『表現形式は異なっても，同等/同値の記述や計算結果になる特性』を指す，と定義する．この特性を強調するために敢えて「一貫」という言葉を使う．この一貫相似性が OOJ 設計の必須目標であり，離散構造化モデルに対してはもちろん最小の離散単位の実体表現である日本語文等の全てにわたって適用される．

2.4 一貫相似性を持たせた記述言語系 OOJ の構築

想定ユーザは OONJ 記述の内容がそのまま計算機での忠実な再現（シミュレーション）になることを前提に記述する．しかし記述の一貫相似性を実現するには semantics が関わる部分はユーザ自身が OOJ エディタ [12] を使って

という用語は，特に，ソフトウェア工学に基づく大規模な開発を意味する用語として用いる [2], [3].
[*2] Object Oriented Japanese の略
[*3] 以降，必要に応じて **OO** と略す.
[*4] message passing．略して **mp** と記す．図 2 参照.
[*5] 本論文の「相似性」という用語の概念は，分野は異なるが類似性，同値性，相同性，双模倣性，あるいは近似双模倣性などの用語の概念にも比較的近い．しかし用法や意味が多少異なる．そこで誤解を避けるため敢えて特別な用語を採用した.

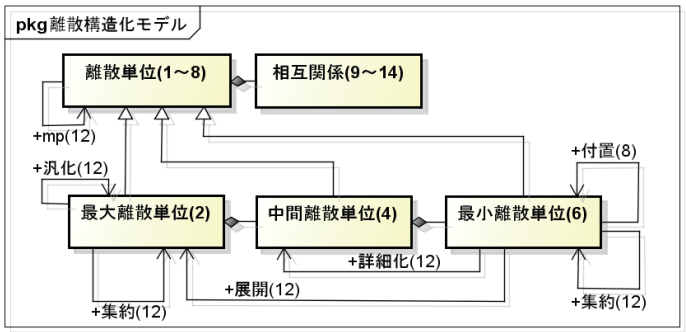

図 4 離散構造化モデルに基づく OONJ の全体構造
Fig. 4 Whole structures of OONJ based on the discrete and structured model

作業する．さらには，ユーザによる変換作業をチェックし検証する仕組みが必要である。

すなわち，一貫相似性を持たせるためには，離散構造化モデルと分析・設計・実装段階の記述規則，semantics を踏まえたユーザの記述を支援する記述環境，そして途中経過を含めて記述の一貫相似性をチェック・検証する仕組み，の 3 つを総合的に構築・運用することが必要である．

3. OOJ の構築方針と概念設計

OOJ 構築の根本方針は，まず先に OOJ 全体を設計し，その後サブ言語の設計，というトップダウン設計にある．OOJ 内部の 3 つのサブ言語構成の設計は**表 2** の【I】に示す．(1) が分析段階に，(2) が設計段階に，(3) が実装段階に相当する．(4) は一貫相似性の検証に必要な機能である。各段階のサブ言語の機能分担と特徴を表 2 の【II】に，OOJ の概念設計を表 2 の【III】に示す。

4. 記述言語 OOJ の詳細設計

実現した OOJ 全体の概念図を**図 3** に示す.

4.1 OONJ の概要 [2], [3]

OONJ は離散構造化モデルを直截に記述できる仕様の設計にした．それを**図 4** で示した．OONJ は図左上の離散単位を中心とし，相互関係を用いて離散単位間を構造化する仕組みを持っている．その仕組みは図下部の全ての離散単位に引き継がれて (継承されて) いる．この設計は OONJ が離散構造化モデルの仕組みを直裁に表現していることが分かる．これらを視覚的に確認するために，空間セルの記述例を**図 5** に示す [13].

4.2 ODDJ や OPDJ への変換の考え方と変換作業

全体の設計方針は，表現形式を変更すること以外の変換を避けることである．まず離散構造化モデルはそのままの形で実装段階まで引き継ぐ.

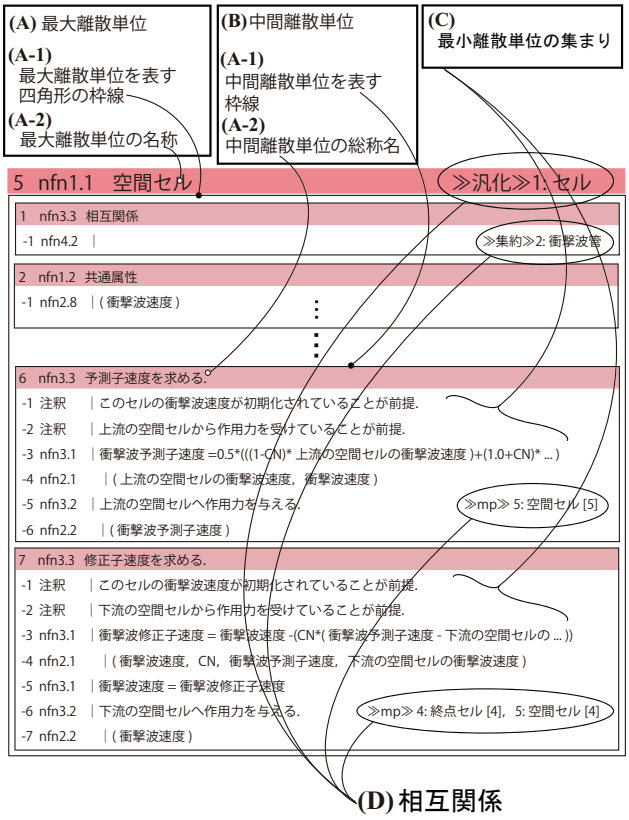

図 5 最大離散単位の空間セルの記述例（記述画面のみ切り出し）
Fig. 5 Description example of the maximum discrete unit " Space Cell"

設計段階以降では，分析段階の最小離散単位の日本語文を変換する．その概略を**図 6** に示す。図 6 の左端の最小離散単位は OONJ の図 4 最右の最小離散単位に相当する．設計と実装の段階において図 4 の他の部分は内容的にはほぼ変化が無く，最小離散単位だけが，式，mp，ライブラリ，OOPL（OO プログラミング言語）起源の規則に特化されたことが分かる．

実装段階の追加と変換の作業は上述の考え方とほぼ同じ方式で行われる．ただしその作業内容に関する方針は前節と異なり，相似性を持たせたプログラム表現に一意に変換できる記述規則に変更する必要がある．OPDJ は複数種 OOPL の文法を抽象化した設計を取り入れる.

4.3 OOPL プログラムへの自動変換

OPDJ 記述はプログラム段階の OOPL トランスレータ（図 3 の右側）に依って 3 種の OOPL（Java，C++，Fortran90）のプログラムに変換される．このトランスレータは 3 言語間の対応関係を基に unfold/fold 方式 [15], [16] を実際に適用したプログラム変換を行う．本トランスレータにおいても OPDJ 記述と OOPL プログラム間の相似性を保つ設計と実装とした．ただし両段階間のトレーサ検証機能は未提供であるが同値変換は確保・保証されている．図 5 の記述例の場合は，人手を全く加えることなく 3 種類

表 2　OOJ の設計方針，各サブ段階言語の役割分担，OOJ 概念設計
Table 2　Design principles, role sharing of each sub languages, concept design of OOJ

【I】OOJ の全体的な設計方針（設計者の立場から）

(1) 分析記述はユーザ自身の分野の言葉で書けることを実現する．：これは当該分野の教科書や論文の記述に，また対象世界のイメージに可能な限り近い記述を許すことである．ユーザは自身の知識や情報の殆どを分析記述にするので，多様な分野で使える記述法や言語が必要である．そこで分析段階の主用言語は教科書や論文の多くがそうであるようにユーザの分野の技術用語を含む日本語とする．
(2) 設計や実装の段階ではユーザでなくては書けない情報のみの提供を求める．：これは勿論，想定ユーザの心理的な負担感を含めてプログラミングの負荷・負担を軽くするためである．それ以外は分析段階の情報を自動変換する方針とする．
(3) 最後には実行可能なプログラムへ自動変換して出力する．：これも勿論，想定ユーザの負担を少なくするためである．想定ユーザはプログラミングの文法の中，細かい記法に神経を消耗させられるのを避けたい気持ちが強いので，必須である．
(4) 途中段階を含めた記述に対する自在な分析とトレースを行う機能を付与する．：これは OOJ の設計において各 3 つの段階毎の記述間の変換を追跡（トレース）して比較評価したり，出力されたプログラムと分析記述との一貫相似性を評価するのに必要である．

【II】内部の各サブ言語の機能分担と特徴

(A) 分析段階記術言語 OONJ	対象世界に関する情報で，プログラム変換に至るまでに必要となる情報 (離散単位や構造についての情報) を全て記述すること．最小離散単位の日本語文は単文であることが推奨される．そしてその離散単位の構造化を要求される．数式は各記述者の考えや通常使う記号と方式で記述して良い．この段階では計算機世界に関わる情報は記述しない．
(B) 設計段階記述言語 ODDJ	計算機世界特有な要素の追加や表現変換を行う．例えば，データ型やアクセス属性を加え，属性を変数と呼び変え，配列やリスト構造を導入したりする．この段階では特定の OO プログラミング言語に関わる事項は記述しない．
(C) 実装段階記述言語 OPDJ	各特定プログラム言語 ((OO)PL) 向けのプログラムに自動変換するための記述を作る．現時点では，Java，C++，Fortran90 に変換可能である．
(D) 統合記述言語 OOJ	上記 3 つの記述言語を概念上で統合した言語であるが，3 つの記述言語の設計主題と解決目標の枠組を示す言語でもある．ただし本来は一貫した記述言語 OOJ がまず設計されたが，必要があって個別のサブ言語が先に発表された．

【III】OOJ の概念設計

1．サブ言語 3 つ全てで**離散構造化モデル**（図 2）を用いる．設計段階以降ではこのモデルを特化した OO モデル（表 1）として使う．
2．この離散構造化モデルを正確に表現できる分析記述専用の記述言語を設計する．（上記【I】の (1) の実現）
3．この離散構造化モデルまたは特化した OO モデルを表現する設計及び実装段階の記述言語を設計する．
　3.1　設計と実装の段階では，分析記述と等価な離散構造化モデルとプログラムに変換し表現できる言語を設計する．
　3.2　設計段階の記述言語では，計算機世界の共通な事項に関する変換を行う設計にする．
　3.3　実装段階の記述言語では，3 つのターゲット言語の共通な事項の変換を行う設計にする．
　3.4　各段階での記述を完成すればトランスレータに掛けて次の段階の記述に変換する．（【I】の (2) の実現）
　3.5　実装段階のトランスレータにおいては実装記述を実行可能なプログラムに自動変換を行う設計にする．（【I】の (3) の実現）
4．以上の記述言語系，記述エディタ，トランスレータ，トレーサ等は全て共通なデータ形式 (XML) で定義しておき，改めて各段階“内”および各段階“間”の記述の任意な分析と変換前後の記述追跡を実行可能にして，将来ニーズに備える．（【I】の (4) の実現）

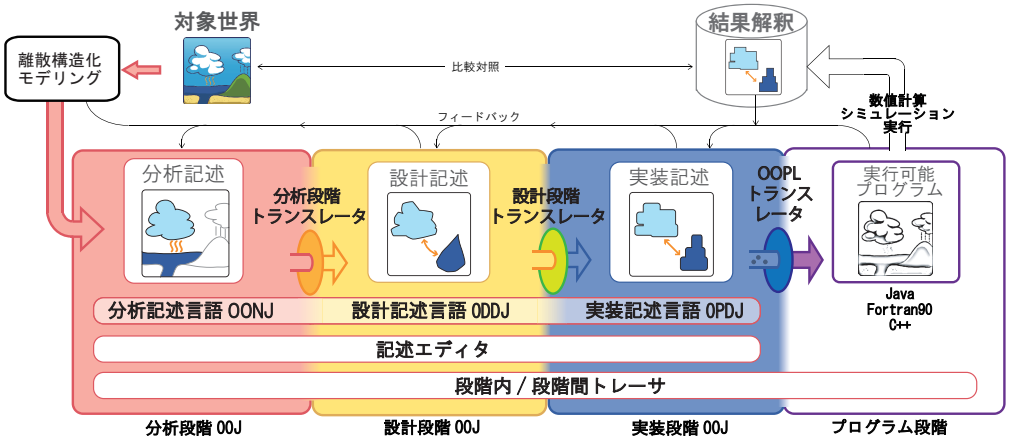

図 3　記述言語系 OOJ を基盤としたプログラム開発過程の全体概念図
Fig. 3　A whole concept diagram of program development processes based on OOJ

の OOPL のプログラムに自動変換された．

4.4　OOJ に持たせた一貫相似性の仕組みの検証

(1) **離散構造化モデルの一貫相似性**は OOJ エディタとトランスレータを使えば OOPL まで保証される．

(2) **最小離散単位である日本語文の一貫相似性**はエディタとトランスレータの支援を受けて，ユーザが式や制御構文に変換することで相似性を保証される．

(3) **ユーザ判断による保証**：ユーザ自身がエディタの支援の下で日本語文を変換すると、間違える可能性がある

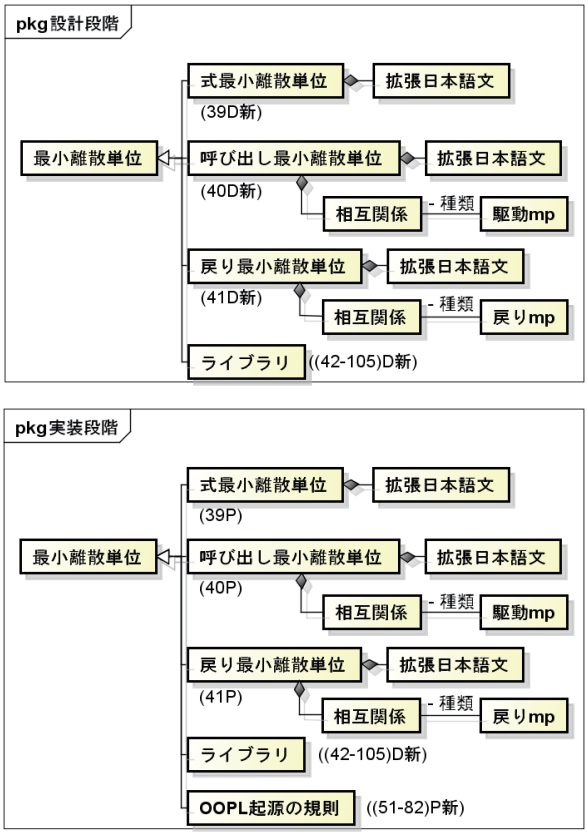

図 6 設計と実装の段階の最小離散単位の段階的詳細化
Fig. 6 Stepwise refinement of the minimum discrete unit in the design and implement stage

ので，特殊なトレーサ [17], [18], [19] を開発して検証・保証する仕組みを構築した．

最小離散単位の日本語文の一貫相似性を実現している 2 つの変換記述例を**図 7** の A) と B) に示す．A) の式計算は表現形式が変わるだけで内容は変わらず一貫相似性が実現している．B) の日本語文は相互作用情報伝達 (mp) に順次変換されて Java プログラムに至る．一貫相似性が十分に保たれていることが分かる．

5. 分析記述の再現性の検証と信頼性の向上
—OOJ が持つ一貫相似性の効果的な利用 (1)—

5.1 プログラムとそれに至る過程の 2 つの信頼性

科学技術計算のプログラム開発で本質的に重要な点は，

（1）プログラムがユーザの分析記述（OOJ で言えば OONJ 記述）を正しく反映したプロセス（開発手順）を経ているか[*6]．という**開発過程の信頼性**．

（2）計算結果の信用性つまり**分析記述の再現性**[*7]

という 2 点にある．この 2 点を合わせた概念をここで仮に**プログラムの信頼性**と呼ぶことにする．

[*6] 通常の業務システムであれば顧客の要求仕様等が使われるが，科学技術計算においては分析記述が出発点となる．
[*7] 分析記述の再現性が十分ならば目的を達成したことになるので，プログラムの信頼性は（どの程度かを別問題とすれば近似的には）分析記述の信頼性に置き換え可能である．

5.2 段階間トレーサによる開発過程の一貫相似性の検証

図 8 に 4 つの段階間の離散単位の相互関連リンクを模式的に示す．各段階の丸印は離散単位，各段階内部の丸印と同色の線は各段階記述内部の（図 2 の意味での）相互関係線である．この相互関係線は各段階の記述における通常のトレーサである**段階内トレーサ** [19] の対象でもある．一方，黒の線で代表される 4 つの段階の離散単位間を結ぶ線は離散単位の変換前後の対応関係を示す．この線は特異なトレーサである**段階間トレーサ** [19] が扱う．これらの問題の一部は分析記述の個々の離散単位に関する詳細な追跡性（traceability）[20] の問題と言って良い．

具体的には変換前後の 4 つの離散単位 n1，d1，p1，pl1 間の相互関連を示す黒のリンク l(n1,d1) の変換前後を比較して 3 つの相似性の高さを検討・評価すれば良い．それら 3 つの相似性が同時に高ければ一貫相似性が高いあるいは成立しているという．同じように図 8 の N1，D1，P1，PL1 は構造化された離散単位群であるが同じく一貫相似性評価が可能である．これらの手順を実行するツール [19] を用いれば短時間で効率的に評価可能である．

5.3 V&V 評価システムへの適用の検討

OOJ の一貫相似性の別表現でもある 5.1 節の（1）と (2) の信頼性は，現在盛んに検討されている V&V の概念 [21], [22] の一部とほぼ同じ概念である．V&V とは当初の要求仕様や設計が正確な手順と過程（プロセス）を踏んでプログラムに正しく反映されていることの検証（Verification）であり，開発の狙い・要求と意図が達成されたプロダクトとしてのプログラムであることの妥当性の確認（Validation）を指す概念である．

OOJ における V&V は設計方針を決め検証や確認の評価基準を決める．その際 OOJ エディタは常により強い制約である一貫相似性を確保・保証しようとするが，ユーザ側での変更は容易である．勿論，段階間トレーサによる検証機構は利用する．V&V の“一貫整合性”を検証するのは OOJ にとっては十分に形式的であり容易であることは明白である．つまり OOJ は V&V 評価のための基礎準備は既に出来ているので，V&V の評価システムに転用することは十分に容易である．

6. 結論

一貫相似性がプログラムの信頼性の向上に貢献すること，および日本語文の変換に関するミス等を 4 つの段階間で詳細にトレースすることによって信頼性を確保すると共に，OOJ をプログラム開発の有効なツールにすることもできた．また、OOJ を V&V の評価システムとしての転用が容易なことを論証した．以上から離散構造化モデルを基幹

OONJ 記述 A)
```
-3      nfn3.1      | 衝撃波予測子速度 = 0.5*(((1-CN)*上流の空間セルの衝撃波速度)+(1.0+CN)*衝撃波速度)
-4      nfn2.1        | (衝撃波予測子速度,上流の空間セルの衝撃波速度,衝撃波速度,)
```
B)
```
-5      nfn3.2      | 上流の空間セルへ作用力を与える.                    》mp》 4:空間セル[4]
-6      nfn2.2        | (衝撃波予測子速度,)
```

ODDJ 記述 A)
```
-3      dfn3.1      | 衝撃波予測子速度 = 0.5*(((1-CN)*上流の空間セルの衝撃波速度)+(1.0+CN)*衝撃波速度)
```
B)
```
-4      dfn3.2    | 上流の空間セル=>下流からの作用力を受ける.（衝撃波予測子速度）》mp》 4:空間セル[4]
-5      dfn2.2      | 衝撃波予測子速度                          実数型      中間離散単位内共有
```

OPDJ 記述 A)
```
3       pfn3.1      | 衝撃波予測子速度 = 0.5*(((1-CN)*上流の空間セルの衝撃波速度)+(1.0+CN)*衝撃波速度)
```
B)
```
4       pfn3.2      | 上流の空間セル=>下流からの作用力を受ける.（衝撃波予測子速度）
5       コメント    | //≫mp≫下流からの作用力を受ける.
6       コメント    | //衝撃波予測子速度(引数情報はコメントになりました.)
```

Java プログラム
```
A) FJvar01=0.5*(((1-Main.WorldVar.WJvar03)*FJvar03)+(1.0+Main.WorldVar.WJvar03)*FJvar00);
   //OPDJ記述上では数式:id:
B) //上流の空間セルへ作用力を与える.
   Main.WorldVar.instance02.method05(FJVar05);
```

図 7 最小離散単位毎の日本語文の一貫相似性を持った記述例

Fig. 7 An integrally consistent and bisimilar description examples of the Japanese sentences

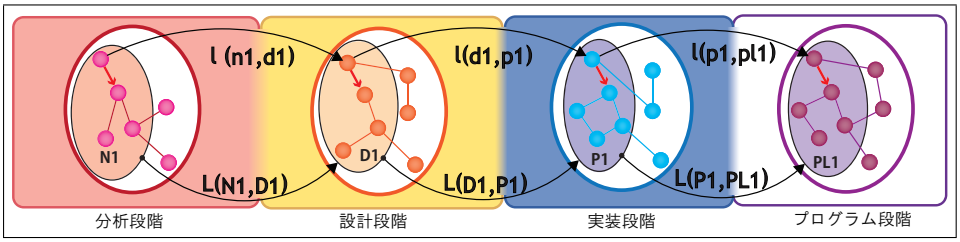

図 8 離散単位間の対応関係リンクの模式図

Fig. 8 Correspondence links among discrete units

とし一貫相似性を持たせた OOJ は，プログラムの信頼性と開発の容易さの点で有効であることを結論とした．

参考文献

[1] 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，日本計算工学会，(May, 2011).
[2] 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，Vol.5，No.3，pp.63-78，(Sep. 2012).
[3] 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，(2013 年 2 月発行予定).
[4] 池田陽祐，大木幹生，三塚恵嗣，畠山正行，単言語方式のオブジェクト指向プロセス記述言語 OOJ の設計，第 163 回 SE 研究会報告，2009-SE-163, pp.57-64, (2009).
[5] 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.
[6] 電子情報通信学会，http://www.ieice-hbkb.org /files/01/01gun_12hen_06.pdf (accessed 2013-01-20).
[7] WikiPedia, http://ja.wikipedia.org/wiki/ソフトウェア品質 (accessed 2013-01-20).
[8] 数値流体力学編集委員会編，数値流体力学シリーズ 2，圧縮性流体解析，東京大学出版会，1995 年.
[9] 畠山正行，池田陽祐，三塚恵嗣，加藤木和夫，メッセージパッシングモデルに基づく差分方程式の計算方式とその実行例，第 86 回 MPS 研究会，2011-MPS-86-, (Oct. 2011).
[10] 峯村吉泰，流体・熱流動の数値シミュレーション，森北出版株式会社，ISBN4-627-91751-1.
[11] メイヤー B. 著，酒匂寛 訳，オブジェクト指向入門　原則・コンセプト　第 2 版，(株) 翔泳社，(Jan. 2007).
[12] 池田 陽祐，三塚 恵嗣，上田 賀一，畠山正行，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，第 92 回 MPS 研究会報告，2012-MPS-92-E　(Feb. 2013).
[13] 畠山研究室，〈http://gaea.cis.ibaraki.ac.jp/〉，(accessed 2013-01-25).
[14] Michael M. Tiller，Modelica による物理モデリング入門，オーム社，平成 15(2003 年) 年 11 月.
[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] Burstall,R. M.,Darlington,J.:”A Transformation System for Developing Recursive Programs”,J. ACM, 24, 1, pp. 44-67(1977).
[17] 永松泰成，畠山正行，オブジェクト指向日本語一貫記述環境 OOJDE の開発，情処研報，01-SE-136，pp.25-32，(2002).
[18] 沼崎隼一，池田陽祐，三塚恵嗣，畠山正行，オブジェクト指向一貫記述言語系 OOJ におけるトレーサの開発，第 171 回 SE 研究会報告，2011-SE-171-6, (Mar. 2011).
[19] 大木 幹生，池田 陽祐，三塚 恵嗣，畠山 正行，記述言語系 OOJ 上で実現する個人向け V&V 環境の提案，第 92 回 MPS 研究会，2012-MPS-92-　(Feb. 2013).
[20] IEEE Std. 830-1998，IEEE Recommended Practice for Software Requirements Specification.
[21] ASME，“Standard for Verification and Validation in Computational Fluid Dynamics and Heat Transfer ”，V&V20-2009.
[22] What is Verification and Validation. , 2009-July-24, http://www.nafems.org/publications/brows_buy/qa/vandv/access2013/01/10.