Vol.2011-SE-171 No.6
2011/3/14

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

沼崎隼一[†1]　池田陽祐[†1]
三塚恵嗣[†1]　畠山正行[†2]

我々の研究グループでは，以前より主としてシミュレーションプログラムの開発支援を目的としたオブジェクト指向一貫記述言語系 OOJ の研究を行って来た．従来のシミュレーションプログラムの開発では, 分析, 設計, 実装の各段階において，個人規模の開発ではプログラムの実行結果が分析と同等内容のプログラムに変換されていることを検証するのは難しい．そこで本論文では，分析段階の記述と実装段階の記述の相似性を検証するための仕組みとしてトレーサの実装を提案する．それにより実際の記述例を用いての一貫相似性の検証を行うことが出来るようなった．実際の具体例として「一次元衝撃波管の流れ」という記述例を用いて一貫相似性の検証を行った．その結果，対応関係を用いての記述の変換過程の追跡により，記述が正しく変換できているのかの確認に有用であることが確認できた．

A Development of Tracer for Object-oriented, Integrally Consistent Description Language OOJ

Toshikazu Numazaki,[†1] Yousuke Ikeda,[†1]
Keishi Mitsuka[†1] and Masayuki Hatakeyama[†2]

Our reseach group has been developed an Object-oriented description language OOJ for the purpose of developing a simulation program. The OOJ composed of the analysis stage , the design stage and the implementation stage. We have designed and implemented the mechanism of the Integrally Consistent and Similar Processes(ICSP) to verify the traceability and the similarity of the program using the OOJ. We have actualy verified the ICSP using the example description ”one-dimensional shock tube flow”. As the result, we have confirmed that the description example program has shown the traceability and the similarity by following the correspondent relationship. We could not demonstrate the effectiveness of the tracer because of the small scale example description. We will test and confirm the ICSP by using a large scale ones.

1. はじめに

OOJ(Object oriented Japanese)[1)–4)] とは我々の研究グループが行なっている個人規模のプログラミングの柱となる記述言語系である．OOJ はある対象世界を再現するシミュレーションを行うことを主たる目的として，OOJ を取り巻くプログラム開発支援環境が開発されてきた．OOJ は**図 1** に示すように，対象世界のモデリングと分析の段階に対応する OONJ(Object-oriented Natural Japanese)[5)]，設計段階に対応する ODDJ(Object oriented Design Description Japanese)[6)]，実装段階に対応する OPDJ(Object oriented Program Description Japanese)[7)] の 3 つの段階で構成されており，各段階に対応する 3 つの記述言語と記述支援のためのエディタや記述を次段階の記述に変換を行うトランスレータにより構成されている．

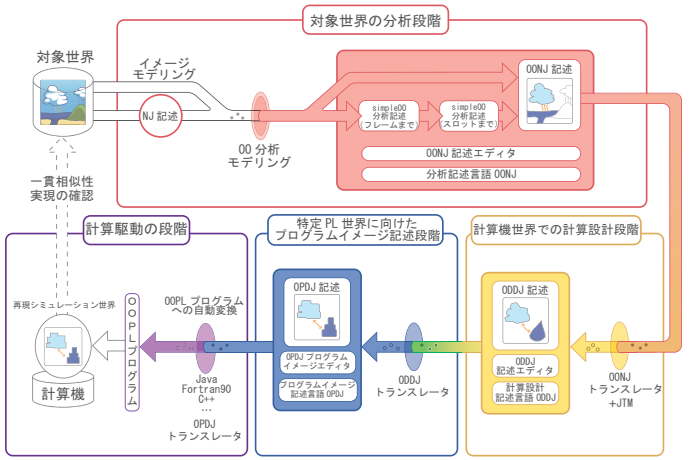

図 1　オブジェクト指向一貫相似性の方法

†1 茨城大学大学院理工学研究科
Graduate School of Science and Engineering, Ibaraki University
†2 茨城大学工学部情報工学科
Department of Computer and Information Sciences, Ibaraki University

この図 1 において，ユーザはまず分析段階の OONJ で自対象世界の分析を行いエディタを用いて OONJ 記述を作成する．次にトランスレータを用いて OONJ 記述を ODDJ 記述へ変換する．この時変換が不十分であったり，記述で足りないものがあればエディタを用いてユーザ自身の手で記述を行う．ODDJ ではデータ型情報の付与や日本語記述から計算機でも扱えるような式に変換する作業などを行う．次に ODDJ 記述をトランスレータを用いて OPDJ 記述へ変換を行う．ユーザは記述に間違いがないか確認し，必要があれば修正をして，最後にトランスレータを用いて OPDJ 記述をプログラムに変換する．プログラムはそのまま通常のコンパイラに掛けられて実行され，ユーザはその実行結果を得る．

しかし当然ながらここで一般的な疑問に突き当たる．それは，分析・設計・実装の各段階において，言語仕様としては前段階の記述と「同等内容の別表現」になるように記述や変換が行われている筈であるが，それらをどうやって確認/検証するのか？ という点である．OOJ では三つの記述言語の仕様として「同等内容の別表現」を狙って記述や変換を行う仕組みにしてある．それが「オブジェクト指向に基づく一貫相似性過程」[2),8)] の実現であるが，もう一方でその事実を客観的に確認/検証する必要性がある．

そこで，三段階各々の記述とその変換を追跡・比較・評価する仕組みを構築した．それが本発表のトレーサである．そういう経緯を持つトレーサであるので，通常のソフトウェア工学的な観点からのトレーサとは多少異なる．その点は関連研究との比較の章で考察する．

2. 一貫相似性過程とその必要性

2.1 再現シミュレーションの再現性

(1) 対象原世界との**高い相似性や再現精度**等を実現する再現シミュレーションのプログラム (ソフトウェア) の開発方法の実現/確立を目指す．この様な方法の実現は単に経験や知識のみに頼る従来のモデリングの方法や狭義のプログラミング技術とは異なる方法論/手法や対処方法の転換パラダイムが必要である．

(2) 前項の裏付けとして，開発されたシミュレーションプログラムと対象世界記述との追跡性を評価する方法の開発．すなわち，対象世界の分析 (記述モデル) から設計，実装，再現世界での駆動という一貫した過程を詳細に関連づけ/跡づけることで客観的に検証可能な技術を確立することを目指す．

上記の二項目が現状では困難或いは不可能であるという現実が，計算シミュレーション結果の妥当性や信頼性を著しく低下させ，処理の本質的な部分での非効率を生んでいることは昔から周知の事実である．

そこで本稿では対象世界との相似性の高い再現シミュレーションを実現する方法と表裏一体である客観的な裏付けのための基本ツールとしてのトレーサを開発した．

2.2 一貫相似性の定義と必要性

一貫相似性の定義：[2),8)] 一貫相似性とは，図 1 に示すように分析段階記述 (OONJ 記述) が設計段階 (ODDJ 記述)・実装段階 (OPDJ 記述) を経て駆動段階である OOPL プログラムに至った時，**OONJ 記述と OOPL プログラムが同等内容の別表現である**と断言できる様な性質を指す．したがって，一貫相似性があると分かれば，分析段階の記述が計算機の内部で忠実な再現計算あるいは処理を行っていると確信してあるいは保証されたものだと考えて間違いない．そういう意味で「一貫相似性」はプログラミングを行っている人にとって誰もが望むプログラムの性質であると言えよう．

一貫相似性の必要性：それは，一貫相似性が技術として確立されれば，例えば，設計段階あるいは実装段階のどこかを意図的に変えても，その変更部分が駆動段階にどう影響するかを十分に追跡できる．であれば駆動段階での影響予測が出来ることになり，必要ならばそれに対する処置を施せばよいことになろう．つまり，駆動段階における計算機の動きを完全に予測でき，コントロールも出来ることになる．ここまでならば，ソフトウェア工学の専門家も同じ様に主張するであろうが，一貫相似性が違う点はそれらを客観的に保証するという点にある．

3. OO に基づく一貫相似性の定義と判定および検証方法

3.1 一貫相似性設計のための基礎定義[2),8)]

変換の前後を考えて行くと，相似性は以下の二つの問題に帰着する．

(1) 一対一対応★脚注★[⋆1] の**縦の相互関連**★脚注★[⋆2]★脚注★[⋆3]が同定されること．**図 2** の A1

⋆1 ★**脚注**★本論文では議論の見通しを良くするため，一対多，多対一，多対多の関連は直接には扱わない．しかし本論文内部での議論と同様な議論がその様な三つの場合にも成立する．ただし，相似性の「高さ」の判断評価は難しくなる．

⋆2 ★**脚注**★用語定義：各モデル化単位での every 記述構成離散単位間の相互関連を「**横の相互関連**」と呼び，各記述構成離散単位毎のモデリング変換前後の一対一対応関連を「**縦の相互関連**」，本用語の場合には正確には**縦の相互相似関連**と呼ぶ．

⋆3 ★**脚注**★既に定義して使われているが再度述べると，相互関連とは複数離散単位間の静的な (時間の経過があっても変化しない固定的な) 関わり，相互作用とは複数離散単位間の動的な (時間の経過に従って変化する) 関わり，と定義する．複数の離散単位とそれ

に始まり C4 に至る対応関係を示す線がそれである．その同定の事実は相似性の『存在』を判断できる．高さ/低さは判断できない．

(2) その対応する離散単位の類似性 (相似性) の「高さ・良さ」の評価．
その判断はユーザの専決判断事項であるが，もしユーザにより客観的に定義されて，公開提示されていれば，定義した基準に従う客観的な判断事項でもあり得る．

上記の（1）については形式的・客観的な方法を見出せる．これをまとめたものが**表 1** の第一と第二項目である．一貫相似性については同じく表 1 の第三と第四項目のとおりである．この定義の議論は次の節で述べる．

（2）については多様な専門分野 (ドメイン) 全般に通用する客観的な方法はなく，各分野の専門家 (ドメインユーザ) の専決事項とするか，客観的な判断基準がユーザから提示される場合はそれで判断するかのどちらかである．

表 1 OOJ を前提とした一貫相似性の概念の定義

相似性の有無 変換前後で対応する記述離散単位の組が一対一対応の縦の相互関連を同定できること．
相似性の高さ (相似精度) 縦の相互関連が同定された記述離散単位間の記述が指す内容実体の差が無いか，小さければ相似性が高いと判断．ユーザの専決判断事項．
一貫相似性の有無 任意の複数の，したがって分析・設計・実装の全ての段階のモデル記述において，一対一対応すべき記述構成離散単位間の全ての縦の相互関連を同定できること．
一貫相似性の高さ (一貫相似精度) 縦の相互関連が同定された任意の全ての段階の記述構成離散単位間の記述の差が小さければ，一貫相似性は高いと判定．ユーザの専決判断事項．

3.2 OO 一貫相似性プログラミング過程

図 2 における対象世界の雲と分析モデリングしたオブジェクト雲との対応関係 A1，同じ

らの間の相互関係の組を構造と呼ぶ．相互関連と相互作用を相互関係と総称する．

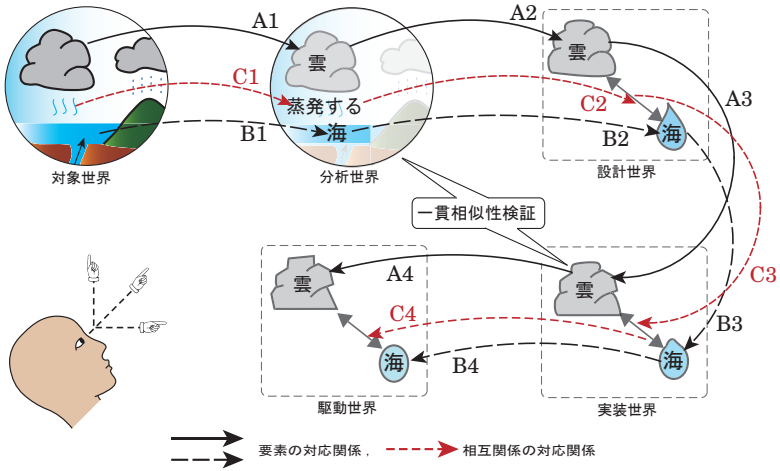

図 2 OO 一貫モデリング過程における一貫相似性の方法

く実装モデリングの対応関係 A2，同様に A3，A4 の対応離散単位の関係が設定でき，それらがすべて相似性を持つとき，A1〜A4 の四つの対応離散単位間の関係が一貫相似性を持つと言う．もちろん五つの対応する離散単位が一貫相似性を持つ，と言っても良い．海に関する変換モデリング B1〜B4 も同様である．

同じく対象世界では海と雲が蒸発という相互作用をすると分析モデリングした後の mp との対応離散単位間関係 C1，C2，C3，C4 も同じく一貫相似性を持つ．mp という相互作用や，集約などの相互関係が一貫相似性を持つと，(広義の) 構造の一貫相似性が成立したことになる．この雲や海，mp が回り回って駆動段階に至る対応離散単位間関係を示す矢印 A1〜A4，B1〜B4，C1〜C4 等が相似性を持つモデリングが一貫相似性を持つモデリングである．

最後に，以上の様な対応離散単位間での一貫相似性が対象世界から駆動世界の間でモデリングしたすべての対応離散単位に対して成立した場合，この過程を一貫相似性過程と言う．この過程が成立したときには，対象世界と再現シミュレーション全体との相似性が成立したことになり，各離散単位の相似性の高さを総合したものが対象世界と再現シミュレーション

(駆動) 世界の両世界の相似性の高さとなる．図 2 全体が両世界の一貫相似性とその成立過程を表現している．

なお，離散単位には相互関係を表す離散単位も含まれており，この相互関係が対象世界を構造化表現する役割をしており，この構造化離散単位が変換段階間で相似であるということは変換前後の表現対象世界の構造が相似であることの証しなのである．したがって，対応離散単位間全てで相似であれば対象世界全体の構造も即座かつ自動的に相似であることが検証できたことになる．

つまり各モデリング段階間でのすべての対応離散単位に対して個々に縦の相似性を持たせることが対象世界全体に対する相似性の高い再現シミュレーションの実現に繋がる，というのが核となる概念である．

3.3 OO に基づく相似性と一貫相似性の具体的な定義

【相似性】: 比較に基づく一対一対応関連の成立がモデル構築当時者 (ユーザ) によって同定 (Identify) されたという事実は実は，対応する構成離散単位間の相似性の「存在」を認めた事である．なぜなら，全く類似性が見出せない離散単位の対を，合理性のある理由や納得のいく根拠と共に縦の相互関連を同定することはできないからである．従って，モデル構築当事者の責任における縦の相互関連の同定により，相似性の高さは別議論として，相似性が何某 (なにがし) か「存在する」ことだけは検証できたことが分かる．

【一貫相似性】: 上記の個々の離散単位について縦の相互関連の同定を，任意の複数の (従って，最終的には全ての) モデリング段階間に適用することで一貫相似性の存在が同定される．言い換えると，各モデリング段階の各構成離散単位毎に上記の縦の相互関連を同定しながらモデリングを行い，その結果張られる全てのモデル記述離散単位毎の縦の一対一対応関連の相互関連の連鎖が明示的に確認されることで，対象世界から再現シミュレーションまでの両世界の一貫相似性の存在を検証できることになる．

従って対象世界の最初のモデル化・記述の各記述構成離散単位 (横の相互関連や ”モノ” 間の相互作用，振舞い等まで含めて) 全てに対して縦の相互関連を同定できれば，モデル化単位の ”モノ” や属性のみならず，構造や振る舞いの一貫相似性の検証が可能になる．

【一貫相似性の検証作業手順】

本節では各モデリング段階の記述作業手順を示す．

(1) 各々の記述に対して適切な離散単位 (離散モデリング単位 ≡ 離散記述単位) を設定し，モデリング段階の前後の対応する離散単位に対して縦の相互関連を張る．

(2) 張る際に縦の相互相似関連の存在をドメインユーザ自身が確認し，同時にその相似性の高さやその低下の原因を評価しておく．

(3) 概念モデル記述からプログラム記述に至るまでの縦の相互関連を辿って，一貫相似性の確認と検証を行う．必要に応じて改訂と再度の検証を行う．

(4) 以上の (1)〜(3) の作業を全ての最小離散単位間，四つの変換段階に対して行う．その作業を漏れなくかつ間違いなく行う実用的な作業には記述支援環境が必須になる．

3.4 従来のプログラム開発で一貫相似性の実現が困難な理由

図 3.4 の上の図は分析段階から駆動段階に至る各段階の離散単位 (相互関係の離散単位を含む) の対応関係が全て明示的になった状況で一貫相似性を確認している様子を表す．一方，下の図は設計及び実装段階の対応関係がブラックボックス化している．

途中の設計や実装の段階がブラックボックスになってしまって四段階の変換前後における対応関係 (縦の相互関連) が見えない場合には分析段階で記述した分析記述内部の全ての個々の離散単位と駆動段階の OOPL プログラムの記述要素間においては形式的な根拠を裏付けとした対応関係が同定できない．したがって，全ての個々の離散単位についての一貫相似性が同定されず検証もできない．したがって当然，対象世界全体についての構造の一貫相似性も同定も検証も確認できないという結論になる★脚注★[⋆1]．

一方 OOJ においては OOJ 内部の分析段階の OONJ，設計段階の ODDJ，実装段階の OPDJ，駆動段階の OOPL の各段階間の変換 (トランスレータ) 処理がその言語レベルで既に一貫相似性が検証されている．故に四つの段階全てで単純相似性が保障できると言えれば自動的に一貫相似性が保障される訳である．

⋆1 ★脚注★単言語方式の OOJ についても対応関係が直接見えない故に同じ様なことが言えそうであるが，それは違う．単言語方式の OOJ[4] では三言語方式の OOJ の構造記述規則レベルでの一貫相似性の裏付けが付されていること，必要ならば個々の記述レベルでの対応関係の同定が三言語方式の OOJ[9] での記述を支援しているシステムを使っていつでも可能な故に一貫相似性は保障されている．

Vol.2011-SE-171 No.6
2011/3/14

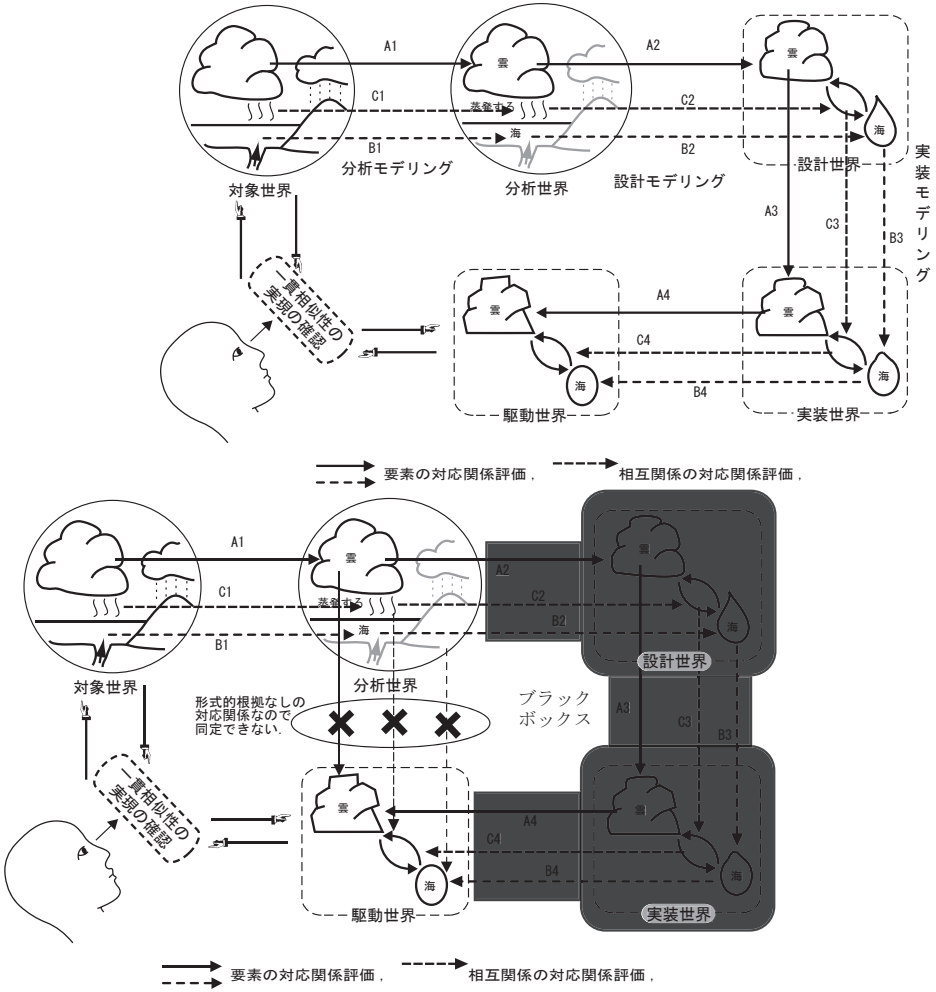

図 3 一貫相似性過程の方法：途中段階がブラックボックス化している場合との比較

Fig. 3 A Verification Scheme of Integarated Similarity for an Object-oriented, Integrally Consistent Modeling processes

4. 一貫相似性機構の設計

4.1 設計方針

一貫相似性機構は OOJ 記述エディタとは独立し，一貫相似性機構単体で実行し相似性の検証を行えるようにする．一貫相似性機構では三つの段階の記述を同時に表示し，各段階の記述を比較しながら，対応関係の付与や相似性の高さの評価などを行えるようにする． その対応関係や，相似性の高さや理由付けに関する情報 (以後 ICSP 情報と呼ぶ) の保存には，OOJ 記述と同様に XML を用いる．また対応関係や，相似性に関する情報の管理は OOJ の記述ファイルとは別のファイルで管理する．具体的には OONJ-ODDJ 間の ICSP 情報ファイル，ODDJ-OPDJ 間の ICSP 情報ファイルの二つのファイルを用いて管理する．

4.2 機能設計

4.2.1 基本設計要件

前章で説明したように一貫相似性の検証を行うには，要素間への対応関係の付与，対応関係に対する相似性の高さの評価，そしてそれに対する理由付けの記述が必要となる．そのようなことから一貫相似性機構にはこれら相似性の検証に必要な情報を追加/記述する機能が必要となる．具体的には以下の３つの機能が必要となる．

(1) 対応関係の操作
(2) 相似性の高さの評価
(3) 理由付けの記述

また上記の機能以外にも要素間の対応関係を確認する為の機能や，各要素間の相似性の高さやそれに対する理由付けを確認するための機能が必要となる．また一貫相似性機構の機能として OOJ ファイルや，ICSP 情報ファイルの読み込み/保存を行う機能も必要になる．まとめると以下の機能が必要であると言える．

- ファイルの入出力
- OOJ 記述の記述内容の表示
- 対応関係の表示
- 相似性に関する情報の表示

次節以下で各機能について詳細な設計を行う．

4.2.2 個別設計要件

【対応関係の操作】:対応関係の付与は記述表示部分で二つの段階間もしくは三つの段階間で相似性を付与したい要素を選択しボタン操作によって対応関係を保存する関数を呼び出し，対応関係の情報を ICSP 情報ファイルに追加することで対応関係の付与を行う．

【対応関係の表示】:各段階間の対応関係を表示する．対応関係の表示は OOJ 記述のツリー表示，テーブル表示それぞれにおいて，対応する要素同士をハイライト表示することで実現する．

【ファイルの入出力】:一貫相似性機構では相似性の検証を行うために以下の５つのファイルの読み込みを行う．

- 分析段階の記述ファイル
- 設計段階の記述ファイル
- 実装段階の記述ファイル
- OONJ-ODDJ 間の対応関係や相似性に関する情報の保存されたファイル
- ODDJ-OPDJ 間の対応関係や相似性に関する情報の保存されたファイル

また，相似性の検証後 ICSP 情報ファイルに変更があった場合は，以下の二つのファイルの保存を行う．

- OONJ-ODDJ 間の対応関係や相似性に関する情報の保存されたファイル
- ODDJ-OPDJ 間の対応関係や相似性に関する情報の保存されたファイル

4.2.3 相似性の高さに関係する機能設計

【相似性の高さの評価】:相似性の高さの評価は ICSP 情報専用のウインドウを用いて行う．ボタン操作によって ICSP 情報を表示するウインドウを起動/表示出来るようにし，そのウインドウ上でスライダーを用いて相似性の高さの値を変えることが出来るようにする．**【理由付けの記述】:**相似性の高さの評価は ICSP 情報専用のウインドウを用いて行う．ボタン操作によって ICSP 情報を表示するウインドウを起動/表示出来るようにし，そのウインドウ上でテキストエリアを用いて理由付けを記述できるようにする．

【OOJ 記述の記述内容の表示】:OONJ，ODDJ，OPDJ の三つの段階の記述の同時表示を行う．記述の表示にはツリー形式とテーブル形式の２種類を用いる．

【相似性に関する情報の表示】:相似性の高さや理由付けなどの ICSP 情報の表示は専用のウインドウ起動/表示することで行う．

4.3 ICSP 情報保存ファイルのデータ構造

一貫相似性機構で扱う主要な情報は以下の通りである．

- 対応関係
- 相似性の高さ
- 理由付け

対応関係では付与する要素同士を識別する情報が必要となる．ここでは OOJ の記述ファイル内で各要素を識別するために使われている ID を用いる．またその対応関係に付随する情報として相似性の高さ，それに対する理由付けに関する情報が必要となり，さらに評価したユーザの名前，評価した日付などについても情報を保存できるようにする．**List 1** に OONJ-ODDJ 間の ICSP 情報を保存する XML ファイルの DTD を示す．ODDJ-OPDJ 間も同様であるので，省略する．

List 1: OONJ-ODDJ 間の ICSP 情報保存ファイルの DTD

```
⟨?xml version=”1.0” encoding=”UTF-8” ?⟩
⟨!ELEMENT ICSP_ND (data)⟩
⟨!ATTLIST ICSPdata id ID #REQUIRED⟩
⟨!ELEMENT data (to_n+,to_d+,value,reasoning+)⟩
⟨!ELEMENT to_n (#PCDATA)⟩
⟨!ELEMENT to_d (#PCDATA)⟩
⟨!ELEMENT value (#PCDATA)⟩
⟨!ELEMENT reasoning (reason,reference*)⟩
⟨!ATTLIST date #CDATA #IMPLIED⟩
⟨!ATTLIST author #CDATA #IMPLIED⟩
⟨!ELEMENT reason (#PCDATA)⟩
⟨!ELEMENT reference (#PCDATA)⟩
```

4.4 GUI 設計

【メイン画面】:一貫相似性機構のメイン画面はメニュー，ツールバー，ツリービュー，テーブルビューで構成する．またツリービューとテーブルビューは OONJ，ODDJ，OPDJ のそれぞれに対応する 3 つを用意する．

【メニュー】:メニュー構成は各種ファイルの入出力メニューとエディタの終了メニューである．

【ツールバー】:一貫相似性機構上部に表示する．ツールバーには対応関係を操作するためのボタンと ICSP 情報を表示するウインドウを起動するためのボタンを表示する．

【ツリービュー】:ツリービューには各段階の記述のフレームレベル，スロットレベルの要素の一覧を表示する．ユーザはこのツリービューで記述の全体構成を把握し，詳細を見たい要素を選択する．

【テーブルビュー】:テーブルビューにはツリービューで選択されたフレーム要素やスロット要素の下階層の内容の詳細を表示する．ユーザはこのテーブルビューを用いて対応関係の付与に関する作業を行う．三段階それぞれに対応するテーブルビューを用意する．

【ICSP 情報表示ウインドウ】:相似性の高さや理由付けを編集する際に表示される．名前，日付，理由付けを記述するテキストエリアと，相似性の高さの評価を行うためのスライダーを配置する．

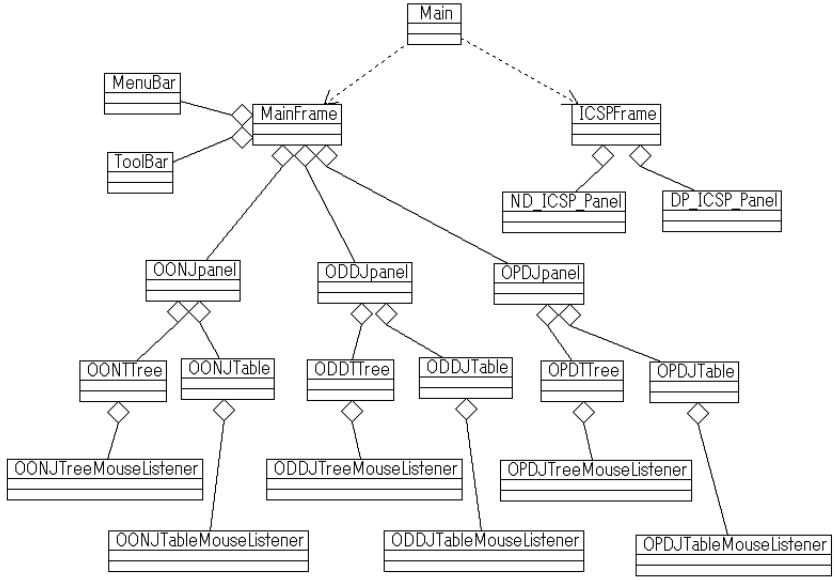

図 4 クラス構成図

5. 実装のクラス構成と ICSP エディタ利用

5.1 実装のクラス構成

前章までに示した設計にもとづき一貫相似性機構の実装を行った．実装に用いた環境は，OS が Microsoft WindowsXP，実装言語は Java であり，使用した開発環境は Eclipse 3.6.1 であった．実装した一貫相似性機構の代表的な静的なクラス構成を**図 4** に示す．

5.2 ICSP エディタを用いた記述実験

5.2.1 対応関係の確認

まずフレームとスロットの対応関係の確認方法を説明する．フレーム，スロットに付与されている対応関係を確認したい場合，ツリービューで確認したい要素を選択することで対応関係を確認することができる．もし選択した要素に対応関係が付与されていれば，**図 5** に示すように選択した要素に対応する他の段階の要素がハイライト表示される．図 5 は OONJ のスロット「衝撃波の予測速度を求める。」をクリックした場合である．またツリービューを選択した場合，ツリービューで選択された要素の詳細がテーブルビューに表示される．サブスロットの対応関係を確認したい場合は，テーブルビューで確認したい要素を選択することで対応関係を確認することができる．**図 6** はサブスロットを選択し対応関係が表示された状態である．

5.2.2 対応関係の追加と削除

対応関係の追加と削除はテーブルビューとツールバーのボタンを用いて行う．**図 7** にツールバーを示す．

【対応関係の追加】:対応関係の追加を行うにはまずテーブルビューで 2 つの段階間もしくは 3 つの段階間において対応関係を追加したい要素を選択する．そしてツールバーの追加ボタンを押下することで対応関係を付与することが出来る．

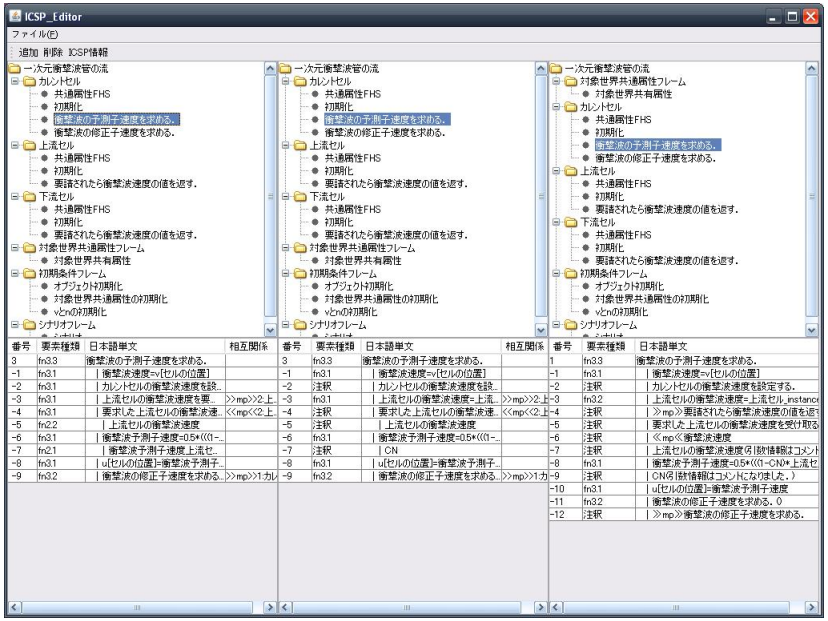

図 5　ツリービューでの対応関係の表示

【対応関係の削除】:対応関係の削除を行うにはまずテーブルビューを用いて削除したい対応関係を表示させる．そしてツールバーの削除ボタンを押下するとダイアログが表示される．このダイアログで削除したい段階間を選択し，ボタンを押下すると選択した段階間における対応関係が削除される．

5.2.3　相似性の高さの評価，理由付けの記述

相似性の高さの評価，理由付けの記述は**図 8** のツールバーにある ICSP 情報ボタンを押下することで表示される図 8 に示した編集ウインドウにより行う．

【相似性の高さの評価】:相似性の高さの評価は図 8 下部の囲みで示すスライダーまたはスピナーによりその値を変化させることで行う．同ウインドウ内の更新ボタンを押すことで変更内容が保存される．

【理由付けの記述】:理由付けの記述は図 8 の囲みで示すテキストエリアを直接編集することで行う．同ウインドウ内の更新ボタンを押すことで変更内容が保存される．

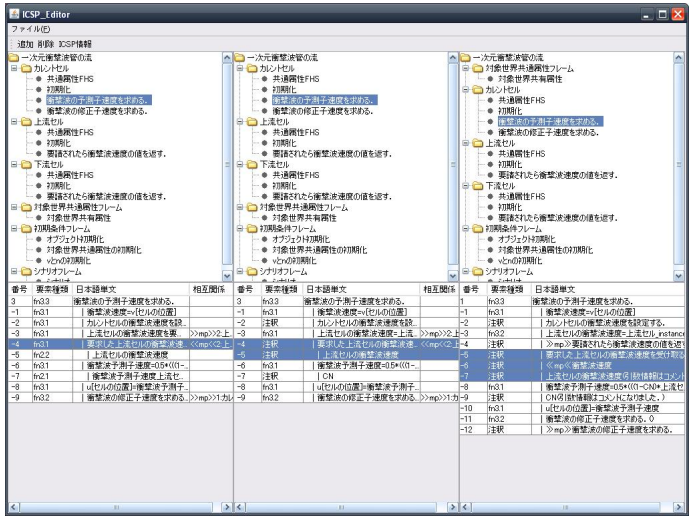

図 6　テーブルビューでの対応関係の表示

図 7　ツールバー

6. ICSP エディタ (トレーサ) の記述実験評価，関連研究との比較

6.1 ICSP エディタ (トレーサ) の記述実験とその評価

トレーサ機能の記述実験を行う記述例は「一次元衝撃波管の流れ」[10),11)] という世界の一

図 8 相似性の高さと理由付けの評価を行う編集ウインドウ

貫記述例で OONJ 記述の行数は 100 行程度の比較的小規模な記述例である．この記述例は ODDJ 記述，OPDJ 記述ともに，記述の離散要素すべてに対して変換機構が対応関係を付与して変換生成されたので，全ての記述構成要素に自動で対応関係が付与され，相似性の高さがフルマークで評価された．そのため主な検証作業としては対応関係の確認のみを行った．具体的には**図 6** に示すようにテーブルビューを用いて対応関係の確認を行い，記述の変換過程を辿っていくという作業を，全ての記述の構成要素に対して行った．これにより記述の構成要素の変換前後の比較を行うことができ，記述が正しく変換されていることを確認することができた．ユーザが自身で判断して評価し，理由付けを記入する記述例は未だ作成されていないので，記述実験は現時点ではここまでである．

6.2 関連研究との比較評価

我々自身が以前に行った同じ趣旨の研究[12] はあるが，直接詳細に比較すべき関連研究は見出されていない．そこで以下の製品と比較してみた．

【Simulink V&V との比較】 Simulink V&V[13),14)] (Simulink Verification and Validation) とは Simulink における確認、検証およびテストに関わるプロダクトであり，要件仕様のトレース、モデリング標準の準拠性チェックなどを行う．Simulink V&V では要求仕様，モデル要素，生成コードを関連付けることにより，要求仕様に対応するモデル要素，生成コードの比較を行うことが出来る．

Simulink V&V は ICSP エディタと比較すると，研究の方向性は類似のモノを持つが，OOJ や我々のトレーサは対象世界のそのものの分析から実装までの追跡を行い，Simulink V&V は要求分析からプログラムまでの追跡を行う．以上のことから，トレーサとして入力となる記述が実世界の分析とシステムの要件分析とで異なるので，具体的な機能にはかなりの隔たりがあることが分かる．また規模の大きいソフトウェア開発を当然狙った製品であり，その点では OOJ や ICSP エディタとは大きく異なる．

7. 纏めと今後の課題

本発表においては，一貫相似性の検証過程の提案と，一貫相似性の検証を行う為の仕組みの基盤であるトレーサの設計と実装を行った．トレーサには，一貫相似性の検証を行うための機能として，対応関係を付与する機能，相似性を評価する機能，対応関係を確認する機能，相似性の評価を確認する機能を実装した．またこれらの機能を用いて一貫記述例の相似性の検証実験を行い，実際の記述例を用いて対応関係の確認作業を行った．それにより対応関係を用いて記述の変換過程を辿っていくという作業が記述が正しく変換されているか確認を行うのに有用であることがわかった．これにより，理由付けや相似性の高さの有用性を確認することができた．

しかしながら本論文であつかった記述例は複雑な構造の対象世界ではなく記述自体も短く，その変換も問題なく行われるような記述例であったため一貫相似性機構の有効性を十分示したとは言えない．今後多くの記述例で一貫相似性機構を用いて一貫相似性の検証を行い有効性，問題点を検証していく必要がある．

参 考 文 献

1) 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).
2) 池田陽祐，大木幹生，片野克紀，加藤木和夫，涌井智寛，畠山正行，オブジェクト指向一貫相似性記述言語 OOJ の設計と検証，第 159 回 SE 研究会報告，2008-SE-159, pp.65-74, (2008).
3) 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, pp.49-56, (2009).
4) 池田陽祐，単言語方式 OOJ に向けた三言語方式 ICSP_OOJ の設計，平成 21 年度茨城大学大学院修士論文，平成 22 年 (2010 年)2 月 10 日.
5) 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).
6) 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，pp.65-74，(Nov. 29，2005).
7) 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).
8) 畠山正行，オブジェクト指向に基づく一貫相似性モデリング過程の実現とその検証方法の提案，第 62 回 MPS 研究会報告，2006-MPS-62, pp.45-48, (2006).
9) 大木幹生，三言語独立のオブジェクト指向記述言語 OOJ における記述規則と OOJ エディタの相似性検証，平成 21 年度茨城大学大学院修士論文，平成 22 年 (2010 年)2 月 10 日.
10) 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.
11) Schmidt, B., "Electron Beam Density Measurement in Shock Waves in Argon", J. Fluid Mech., Vol.39, part 2, pp.361-373, (1969).
12) 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).
13) http://www.mathworks.co.jp/products/simulink/
14) http://www.mathworks.co.jp/products/simverification/
15) http://gaea.cis.ibaraki.ac.jp/