Vol.2013-MPS-95 No.9
2013/9/26

OOJ上に構築するソフトウェアV&Vのためのテスト環境の提案と実装

大木 幹生[†1]　池田 陽祐[†2]　上田 賀一[†3]　畠山 正行[†4]

概要：　個人規模のプログラム開発を目的として記述言語系 OOJ が構築されてきた．この OOJ は 4 つの内部サブ言語に分けて任意に詳細な追跡性を備えている．OOJ は一つにはプログラムがユーザの狙いや目的を正確に反映していること、および計算結果の妥当性を検証できる仕組みの実現を狙っているからである．本研究では OOJ 上にソフトウェア V&V のためのテスト環境を構築した事を報告する。具体的には客観的な検証のために IEEE Std 1012，829 等に準拠するテストケースを生成する仕組みを提案すると共にそのテスト環境の一部の実装を試みた．その結果，構造テストのテストケースとして利用可能なドキュメントを生成することが出来た．

キーワード：　V&V，IEEE_Std 1012，OOJ，ソフトウェアテスト，テストケース，IEEE_Std 829

A proposal and its implementation of test environment for software V&V based on OOJ

Mikio Ohki[†1]　Yousuke Ikeda[†2]　Yoshikazu Ueda[†3]　Masayuki Hatakeyama[†4]

Abstract: A descriptive language system OOJ and its descriptive environment for personal use has been developed. OOJ contains 4 inner sub-languages and the function for any detailed traceability. OOJ has been developed to attain the aims that the user-made program always reflect the user objectives and to confirm the validity of the calculated results. In the present paper, we will report on the test environments for the software V&V tests that we have developed. To attain this aim, we have proposed the mechanism to generate the test cases based on the IEEE Std 1012, 829, and a part of these test environment have been tried to implement. As the results, we have succeeded in generating some usable documents as the test cases of the structure test.

Keywords: V & V, IEEE_Std 1012, OOJ, software test, test case, IEEE_Std 829

1. はじめに

現代のプログラム開発[*1]に対するニーズは，例えば未解明の物理現象解析のための再現シミュレーションや工学製品の安全性の確保などのプログラムの発展などに伴って，より高度となり複雑化し続けている．さらにはプログラム開発効率の向上や信頼性の検証などへの要求も増えつつある．一方プログラム開発を最小単位である個人で行うユーザはずっと以前から以下のような問題を抱えている。

1．体系的・形式的なプログラム開発技術の不足

†1 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
†2 現在，茨城大学工学部
Presently with Faculty of Engineering, Ibaraki University
†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
†4 現在，茨城大学工学部
Presently with Faculty of Engineering, Ibaraki University

*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．よく使われるソフトウェア開発という用語は，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．この点についての詳細な比較は参考文献 [1] のいずれも表 2 にある．

２．プログラムの妥当性や検証の欠如

本論文ではこの様なプログラム開発のニーズに対応して，分析から設計・実装を通してプログラムに至るまでを一貫して作業できる記述言語系 OOJ[2] と実際にプログラム開発で運用するために OOJ 記述環境 [3] も開発した．これは上述の１．にはある程度の解決に貢献した。

そこで本研究においてはもう一つのテーマである上記２．を取り上げる。それは OOJ 記述環境の上にプログラムの信頼性を向上させるツールあるいはプログラムの検証を行うためのテスト環境の構築である．ただし、対象とする想定ユーザの特徴からしてテスト環境などは使ったこともなく、深く学習する気が無いユーザのために負担の少ない簡易なテスト環境を構築する。

2. OOJ とその記述環境，ソフトウェア V&V

2.1 記述言語系 OOJ とその記述環境の概要

OOJ は個人が作る小規模の科学技術計算用のプログラムを開発することを主な目的にしており，離散・構造化モデルを用いて対象世界の分析モデリングに始まり，その分析記述を基にして設計や実装の段階の記述に変換し，最後に Java，C++，Fortran90 等の言語のプログラムに自動変換する記述言語系であり，3 つの記述言語から成り立っており，記述言語間はトランスレータによって結ばれている．この言語系専用に開発されたのが OOJ 記述環境である．その概要を**図 1** に示す．図中の離散・構造化モデルは数値計算に常用されている離散化モデルであるが，同時に特化したオブジェクト指向モデルとしての特徴も持っている．

2.2 プログラムの検証、V&V，テスト環境

前章で挙げた問題のうち、１．は OOJ 記述環境の開発によって，２．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）を実現した特殊なトレーサ [4] によって多少は解決されている．

一方，２．は現在盛んに検討されている V&V の概念 [5] 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）と，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．

そこで２．解決するために，ソフトウェア V&V の検証という観点から IEEE_Std1012、同 829[6] 等に準拠し、それらの中から適切に抜粋したテスト項目を選んで簡易なテスト環境を構築する。これ等の実現手法の提案と実装の一部が本発表のテーマでありその報告である．

3. OOJ への std1012 と std829 の適用

本章では，IEEE Std 1012 [5] と IEEE_Std829 [6] の規格から OOJ の必要な項目の取捨選択について述べる．

3.1 IEEE Std 1012 に対する V&V task の抽出

IEEE Std 1012 [5] はソフトウェア V&V の標準のガイドラインとして規格化されている．規格上ではソフトウェアの life cycle processes を主体に V&V activity や V&V task が規定されている．しかしながら，OOJ は個人規模のプログラム開発を対象としているため，これら全てに対応する必要はない．そこで，IEEE Std 1012 の規格に対して前節で示した OOJ-テスト環境の項目を基に OOJ-テスト環境で実現する V&V task の抽出を行う．

OOJ-テスト環境は個人規模であることから，IEEE Std 1012[*2]の life cycle processes 内の Development process のみを対象とする。同様の理由から対象となる V&V activity は，要求 V&V activity，設計 V&V activity，実装 V&V activity，テスト V&V activity の 4 つのアクティビティである．また，IEEE Std 1012 では，Software integrity levels が 4 段階に分かれており，social loss 等を調査し段階を決定する．OOJ では，管理者と開発者と利用者が同一人物である個人利用のプログラムを対象としているため level 1 を対象とする．以上を前提とした V&V task を**表 1** に示す．

対応しない V&V task は，Criticality analysis，Management review of the V&V effort，Software design evaluation，Software requirements evaluation であり，これらは OOJ が個人規模のプログラム開発であるため対応する必要が無い．以上から OOJ-テスト環境で実施する V&V task は，各種 Integration V&V test と各種 System V&V test の 2 つである．

なお、以上の V&V task に必要なドキュメントである SRS [*3], IRS [*4], SDD [*5], IDD [*6], Source code, Executable code 等は OOJ 記述環境から供給されるので本論文では紙幅の関係で割愛する。

3.2 IEEE Std 829 を対象とした OOJ-テスト環境のドキュメントの設計

前節までに示した V&V task では，required input document としての Test plan が IEEE Std 829 [6] に準拠することが示されている．IEEE Std 829 [6] は，ソフトウェアテストに対するドキュメンテーションの規格であり，Test plan と Test design，Test case 等の仕様が規定されてい

[*2] 詳細は，IEEE Std 1012 の Table2 を参照.
[*3] **S**oftware **R**equirements **S**pecification の略記 [7].
[*4] **I**nterface **R**equirements **S**pecification の略記 [8].
[*5] **S**oftware **D**esign **D**escription の略記 [9].
[*6] **I**nterface **D**esign **D**ocument の略記 [10].

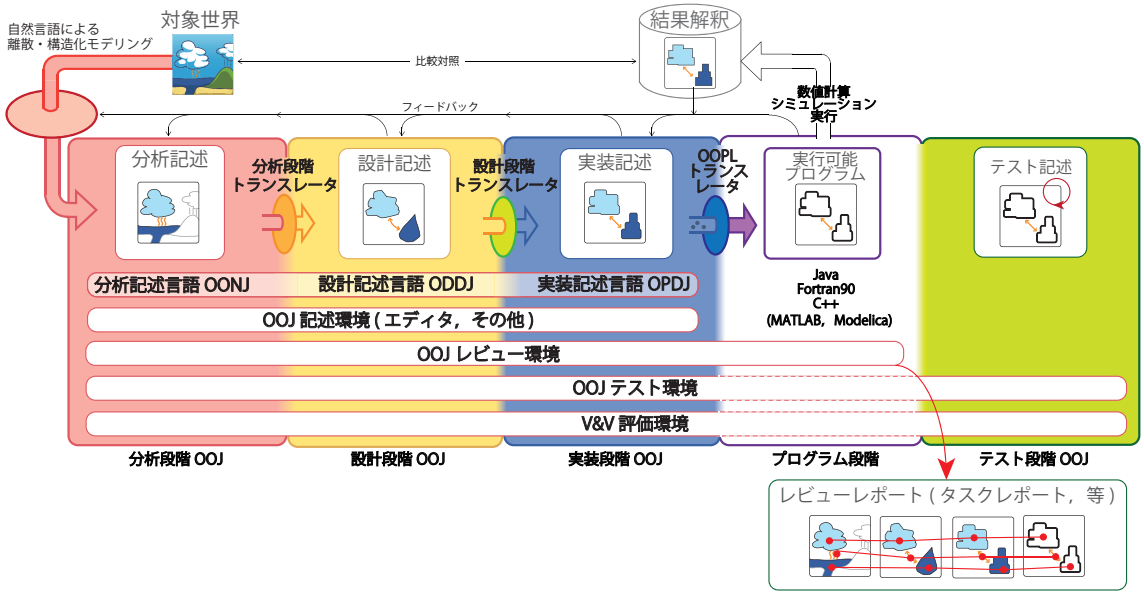

図 1 OOJ を基盤とした記述環境，レビュー環境，テスト環境，V&V 評価環境の全体概念図
Fig. 1 A whole concept diagram of various support environments based on OOJ

表 1 OOJ-テスト環境で実施する V&V task
Table 1 V&V task to be executed on OOJ test environment.

Life cycle processes	Development			
V&V activity V&V task	要求 V&V activity	設計 V&V activity	実装 V&V activity	テスト V&V activity
Criticality analysis	×	×	×	
Identify improvement opportunities in the conduct of V&V	○*1	○*1	○*1	○*1
Integration V&V test case generation			○	
Integration V&V test design generation		○		
Integration V&V test execution				○
Integration V&V test plan generation		○		
Integration V&V test procedure generation		○		
Management review of the V&V effort	×	×	×	×
Software design evaluation		×		
Software requirements evaluation	×			
System V&V test case generation			○	
System V&V test design generation		○		
System V&V test execution				○
System V&V test plan generation	○			
System V&V test procedure generation			○	

○印は OOJ-テスト環境で行う V&V task である.
×印は個人規模のプログラム開発であるため対応しない V&V task である.
*1:ユーザサポートとして別の方法で支援を行う V&V task である.

る．そこで V&V task の実施手法に従い，IEEE Std 829 で示されている Test plan，Test case specification，Test procedure specification の Outline を基に OOJ-テスト環境で使用する Test plan，Test case，Test procedure の設計を行った．

一例として，Test case specification について示す．IEEE Std 829 の Test case specification の outline として 7 つの section が示されており，これらから OOJ-テスト環境の Test case で利用すべき項目を抽出した．その一覧を**表 2** に示す．これらを Test plan，Test procedure に対しても行い，Test plan，Test case，Test procedure を設計した．

表 2 Test case outline から抽出される Test case 項目の一覧
Table 2 List of Test case items derived out of Test case outline

Test case outline sections	利用の有無
Test case specification identifer	○
Test items	○
Input Sepcification	○
Output Sepecification	○
Environmental needs	△
Special procedural requirements	○
Intercase dependencies	○

○印は OOJ-テスト環境の Test case で用いる section.
△印は OOJ-テスト環境の Test case で一部を用いる section.

4. OOJ テスト環境の概要

4.1 OOJ-テストとそのドキュメントの構成

OOJ では図 1 で示した分析段階，設計段階，実装段階，プログラム段階へと段階を経る毎に段階的詳細化を行うことで実行可能プログラムへと変換されていく．段階的詳細化の作業は，離散・構造化モデルの 3 つの集約階層 (最大離散単位[*7]，中間離散単位，最小離散単位) に基いて形式化されている．これらの作業は対象世界をよく知る想定ユーザならば十分に行うことが可能である．そこで OOJ の開発手順に合わせたテスト作業を行うこと，OOJ テストのドキュメントと各段階の記述を関連付けすることにより管理することの 2 点を目標として OOJ-テスト作業を段階的詳細化の開発過程に準拠するよう設計した．その概要を**図 2** に示す.

まず始点となる分析段階では離散・構造化モデルに従ってモデル化されるので、存在するオブジェクトや必要な機能などを離散単位として洗い出し，これらの離散単位がどのように相互関係を持っているか（構造化されているか）についても記述する．その結果、モデル化する対象世界の目的が明確になる．そのため対象世界に対するテスト「対象世界テスト」(図 2-(a)) [*8]の設計を行えば良い事が分かる．

次の設計段階では洗い出された離散単位オブジェクトや構造に対して，計算機特有の情報を追加していく．各離散単位の中身に関しては次の段階でも修正が可能であるため，ここでは離散単位同士の構造化について先に確定されていることがことが望ましい．そこで離散単位同士の構造（静的な構造と動的な構造とがある）に対するテスト「構造テスト」(図 2-(b)) [*9]の設計を行えば良い事が分かる．

実装段階は全ての離散単位がプログラムに変換されるのに適切な記述がされているかの確認等の微調整を行う．したがって，各離散単位の処理内容が明らかになるため離散単位に対するテスト「離散単位テスト」(図 2-(c)) [*10]の設計を行えば良い事が分かる．

以上から、図 2 に示す各テストで作成されるドキュメントは、前章で示した IEEE Std 1012 に準拠した構成となっていることが分かった[*11].

4.2 IEEE Std 1012 と OOJ-テストとの関係

OOJ は前節で示したように分析段階，設計段階，実装段階，そしてプログラム段階へと 4 段階に分けたプログラム開発過程である．一方，IEEE Std 1012 の Development も要求 V&V activity，設計 V&V activity，実装 V&V activity，テスト V&V activity の 4 段階に分かれている．

OOJ の分析段階は対象世界の目的を明確にする段階であるため要求 V&V activity に対応付けられる．同様に設計段階も設計 V&V activity と実装段階も実装 V&V activity と対応付けられる [3] が，プログラム段階に対応する activity が無く[*12]，test V&V activity に対応する OOJ の段階も無い．そこで IEEE Std 1012 に準拠した評価過程とするため test V&V activity に対応する「テスト段階」(図 2-(d)) を設定した．また，前節で示した V&V 評価に関わるドキュメントは図 2 の英文表記部分のように位置づけられ，OOJ 記述環境からも SRS 等のドキュメントが (図 2-(e)) のように得られることで OOJ-テストに対して Std 1012 に準拠した V&V 評価が可能であることが分かった．

5. OOJ テスト環境の試実装と結果

本論文では 4.2 節で述べた 3 種類のテスト項目のうち「構造テスト」に限定して実際に構造テストケース作成ツールの試実装を行った．なお、テスト環境を構築するに際しては想定ユーザ [1] が容易に使えるようにするためには、作業負担が少なくてテストドキュメントを可能な限り自動生成することが求められる。

5.1 構造テストケース作成ツールの設計

構造テストは，離散単位の動的構造についてテストを行う．この動的な構造は相互作用情報伝達を行うメッセージパッシング（以降、mp と略す）の数に比例してパターンが増加するため全てのパターンについてテストを実施するというのは現実的ではない．そこで，ユーザにテストを実施する動的な構造を指定してもらうように設計した．具体的には，ユーザは動的な構造として抽出する始めと終わりの離散単位を指定することで，その間の離散単位を自動的に構造化して抽出される．動的な構造の記述には三つの相互関係の中の 1 つを用いる．

[*7] 代表例としてはオブジェクトが相当する.
[*8] ソフトウェアテストのシステムテストと対応する.
[*9] ソフトウェアテストの結合テストと対応する.
[*10] ソフトウェアテストの単体テストと対応する.
[*11] 対応する IEEE Std 1012 のドキュメントを図 2 に英文で示した.
[*12] Installation and checkout V&V activity に対応付けられるが，OOJ は個人利用のプログラムであるため Installation and checkout V&V activity とは目的や多くの内容が異なり，対応しないとした.

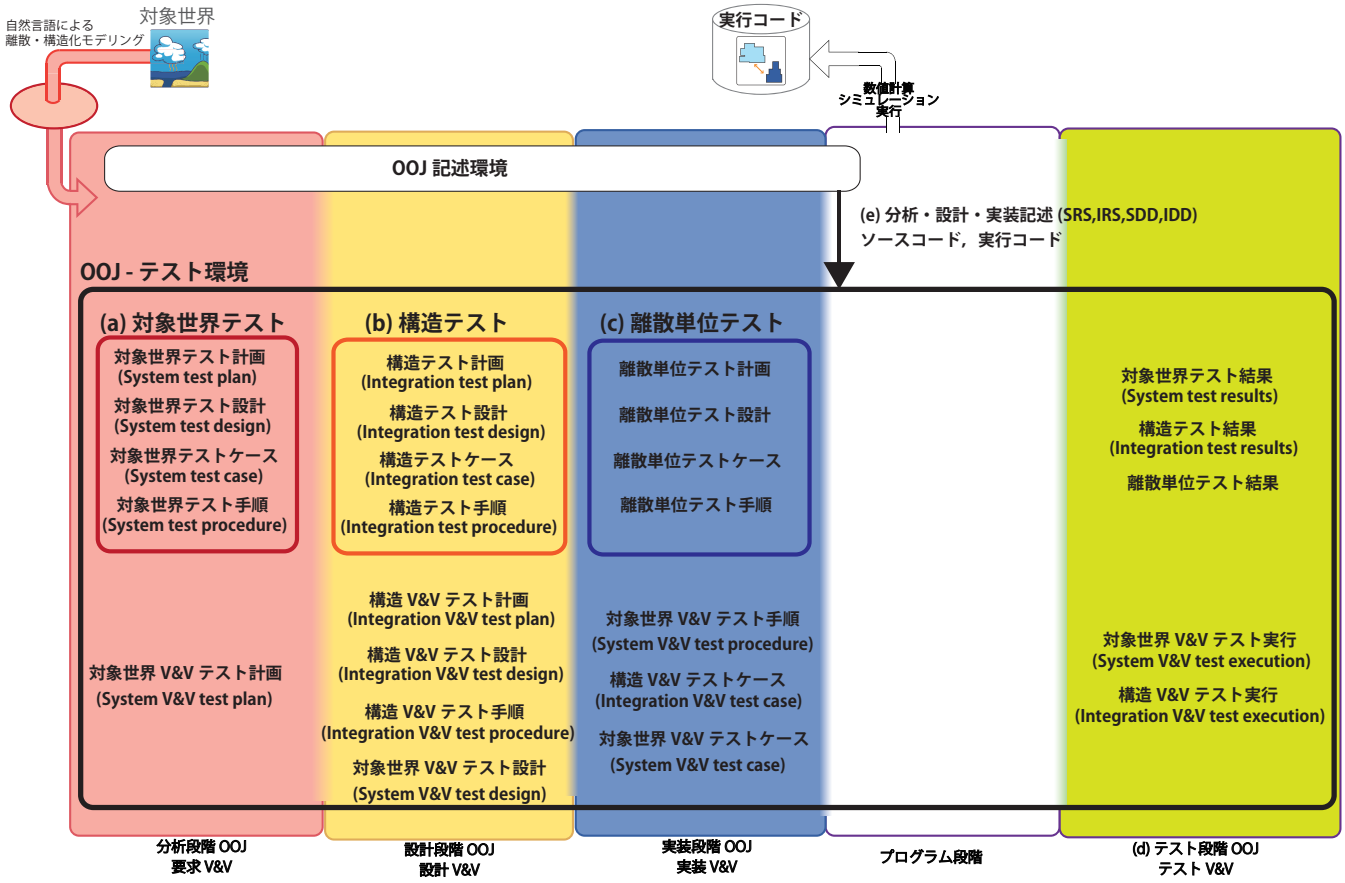

図 2　std 1012 に準拠した OOJ テスト環境の全体構成

Fig. 2　Whole constitutions of OOJ test environment based on the std 1012

(A) 離散単位の入れ子構造による相互関係.
(B) 離散単位のシーケンシャルな相互関係.
(C) 相互作用情報伝達による相互関係.

これらは，B，A，C の順番でユーザによる意図性が増すため，一つの離散単位に対して複数の相互関係が示されていた場合，C，A，B の順で抽出される．この様な優先順位に従ってユーザの指定した離散単位間を走査し複数の離散単位を一つの構造として切り出す．この機能を離散単位群抽出機構として実装した．

また構造テストでは mp を実際に評価しながら行うため，抽出された構造の中に含まれる全ての mp に対して入力するデータとそれに対して期待される出力データを指定する必要がある．これらの情報には，それぞれについて属性と値を指定する必要がある．この情報の入力を支援する機能として mp 設計機構を実装した．

5.2　構造テストのテストケース生成

テストケース生成ツールでは，テストケース設計ツールで作成されたテストケース仕様書の各項目を Std829 の項目に対応付けてテストケースに変換される．具体的には，抽出された動的な構造はテスト範囲，各 mp への入力データは入力値，各 mp の出力データは出力値に対応付けられる．これらの情報は，XML を用いて構造化されたドキュメントとして出力される．

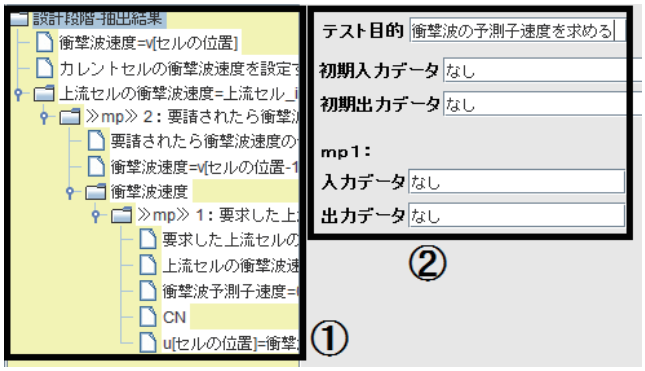

図 3　テストケース設計ツールの実装画面

Fig. 3　An implementation screen for test case design tool

5.3　テストケース設計の実装

テストケース設計ツールの実装画面を**図 3** に示す．ユーザインターフェイスは二つの部分で構成されており，左側には離散単位群抽出機構によるユーザ指定範囲を走査した結果のツリー表示 (図 3 の丸 1)，右側には抽出された動的構造に含まれる mp の入出力情報を設計 (mp 設計機構) の

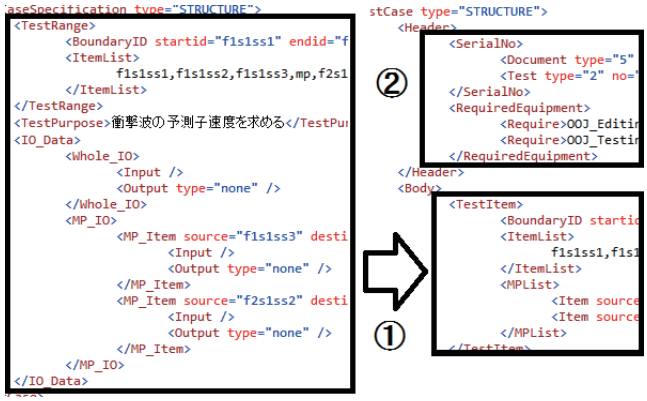

図 4 テストケース生成の過程：基情報とテストケース情報

Fig. 4 Testcase generation processes : base information and test case information

部分と抽出された動的な構造がユーザにとってどのような意味があるのかといった，テストの目的に関する情報を自由入力するテキストボックスが表示されている (図 3 の丸 2)．ユーザはこれらのインターフェイスを通して作業を行うことで構造テストのテストケース仕様書が作成される．

5.4 テストケース生成の実装

変換の過程をテストケース設計ツールで生成された基情報とテストケース生成ツールで生成されたテストケース情報を**図 4** に示す．図 4 の左側が基情報（一部）で右側がテストケース情報（一部）である．基情報では，このように，Std829 に対応する情報が抽出されており，テストケース内で Std829 の項目として変換され記録される（図 4 の丸 1)．また，基情報にはユーザの入力によらない情報 (主にドキュメントの管理情報) は含まれていないため生成ツールで新たに自動生成され付加される（図 4 の丸 2)．この様に，テストケース変換ツールを通して，Std829 に対応したテストケースが生成されることが確認できた．

6. 結論と今後の課題

提案し実装した一部のテスト環境試作版は稼動しており，当初の設計目的はある程度満たしていることが確認された．しかし未だ試実装であって十分ではない．したがって今後は必要なテスト環境を充実すると共に、ソフトウェア V&V 環境の構築を進展させる．

参考文献

[1] 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, 63-78 (Sep. 2012)．

[2] 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 の構築と科学技術計算教育への応用—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—，情報処理学会論文誌：数理モデル化と応用，Vol.6，No.3，(Oct. 2013)．

[3] 池田陽祐，大木 幹生，三塚恵嗣，上田賀一，畠山正行，V&V 評価の仕組みを組み込んだ OOJ 記述環境の開発と評価，情報処理学会 95 回 MPS 研究会報告，2013-MPS-95-　(Sep. 2013)．

[4] 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，情報処理学会第 92 回 MPS 研究会，2012-MPS-92-　(Feb. 2013)．

[5] IEEE Std 1012-2004-IEEE Standard for System and Software Verification and Validation, IEEE Standard Association，2005.

[6] IEEE Std 829-1998-IEEE Standard for Software Test Documentation，IEEE Standard Association，1998.

[7] IEEE Std 830-1998-IEEE Recommended Practice for Software Requirements Specifications，IEEE Standard Association，1998.

[8] IEEE Std 100-2000-IEEE 100 The Authoritative Dictionary of IEEE Standards Terms Seventh Edition，IEEE Standard Association，2000.

[9] IEEE Std 1016-2009-IEEE Standard for Information Technology–Systems Design–Software Design Descriptions，IEEE Standard Association，2009.

[10] IEEE Std 24765-2010-Systems and software engineering – Vocabulary，ISO/IEC，2010.