TOC 思考プロセスを用いたテスト工程の課題分析と改善
－テストにおけるトレーサビリティ確保の落とし穴－

八木 将計† 小川 秀人†

†株式会社 日立製作所 横浜研究所 〒244-0817 神奈川県横浜市戸塚区吉田町 292 番地
E-mail: †{masakazu.yagi.zd, hideto.ogawa.cp}@hitachi.com

あらまし トレーサビリティの確保は，ソフトウェアの品質保証において欠かせない要素の一つである．ソフト品質向上のためトレーサビリティ確保を内部規定としている，ある組込みソフト開発部署では，テストによる不具合の見逃しが多いという問題があった．本報告では，この組織のテストプロセスの課題分析と改善施策の導出に TOC 思考プロセスを用いた．課題分析の結果，トレーサビリティ確保がテストによる不具合摘出を阻害していることがわかった．この問題の背景に，トレーザビリティに対する開発者の誤った思い込みが含まれていることを明確にし，その思い込みを解消する施策として，不具合摘出のためのテストの定義とテスト観点レビューを提案する．

キーワード TOC 思考プロセス，トレーサビリティ，テストエビデンス，客観的証拠，テスト観点レビュー，マインドマップ

Problem Solving in Test Process Using TOC Thinking Processes
－Pit Trap of Test Traceability for Quality Assurance－

Masakazu YAGI† and Hideto OGAWA†

† Hitachi, Ltd., Yokohama Research Laboratory 292, Yoshida-cho, Totsuka, Yokohama, Kanagawa, 244-0817 Japan
E-mail: †{masakazu.yagi.zd, hideto.ogawa.cp}@hitachi.com

Abstract Traceability is a key aspect of software development for quality assurance. An embedded software development organization provides the traceability. Software testing of the organization was not enough. In this paper, we solve a problem in test process of the organization using TOC thinking processes. We clarified that test traceability constrains finding defects by testing. Therefore, in order to improve problem of the test process, this paper proposes new test definition and its design review by using mind map.

Keyword TOC thinking processes, Traceability, Evidence, Test design review, Mind map

1. はじめに

近年の製品に対する品質要求の高まりに伴い，品質保証において，トレーサビリティが注目されている．トレーサビリティは，ISO9000:2005（JISQ9000:2006）で「考慮の対象となっているものの履歴，適用又は所在を追跡できること」と定義されており[1]，ソフト開発においては，要求仕様書，設計仕様書，変更来歴，テスト，不具合の記録などを相互に参照・確認できるように紐付けることである．トレーサビリティの確保は，車載機器，産業用電子機器，医療機器の国際規格(ISO26262[2]，IEC61508[3]，IEC62304[4])や CMMI[5]でも求められており，ソフト品質保証において欠かせない要素となっている．

日立グループのある組込み製品のソフト開発を行っている組織においても，ソフトの品質を向上する目的のため，トレーサビリティの確保を内部規定としている．しかし，そのような取組みにも関わらず，設計による不具合見逃しが多いという問題が発生していた．

報告者は，この組織におけるテストプロセス改善に従事し，課題解決のために TOC(Theory of Constraints)の思考プロセス[6][7]を用いた. TOC 思考プロセスは，目的達成を阻害する本質的な問題を発見，解決していく,体系的な問題解決アプローチである.本報告では，この TOC 思考プロセスにより,テストにおけるトレーサビリティの一つである，テストケースとそのテスト結果の客観的証拠を紐付ける作業が心理的，物理的な制約となり，テストによる不具合摘出を阻害していることを明確にした．その課題解決のため，不具合摘出のためのテストを定義し，そのテストのためにテスト観点レビューを提案する．また，本提案手法を試行・評価し，多くの不具合を摘出できることを確認した．

第 2 章では，テストプロセス改善の対象組織と目的

について述べる．第 3 章では，TOC 思考プロセスによるテストプロセスの課題分析について述べる．第 4 章では，第 3 章で明確にした課題の解決策の提案を行う．第 5 章では，提案手法の試行結果を定性的，定量的に評価する．最後に第 6 章でまとめを述べる．

2. テストプロセス改善の対象と目的

2.1. 対象組織

本報告のテストプロセス改善の対象とする組織は，ある組込みソフトを開発しており，設計部と別に製品品質の責任部署として，品質保証部(以下，QA)を置いている．対象組織における「テスト」には，設計部が担当する設計テストと QA が担当する認定試験(製品の妥当性確認)の二つがある．

2.2. トレーサビリティ

対象組織では，ソフトの品質保証を目的にソフト開発におけるトレーサビリティの確保を内部規定としており，テストにおいては，要求からテストケース[1]までの紐付けと，テストケースとテスト結果の客観的証拠（以下，テストエビデンス）の紐付けを定めている[2]．

具体的には，テスト仕様書のテストケースに，対応する要求仕様書や設計仕様書などの仕様項目を明確に示す．また，テスト結果画面のスクリーンショット画像や出力データログなどといったテストエビデンスをテストケースに対応する形で資料に纏めることとしている．この資料をテストエビデンス資料と呼ぶ．

2.3. テストプロセス改善の目的

対象組織では，上述のようにトレーサビリティ確保を規定しているが，設計テストで不具合を見逃し，QA の認定試験で摘出される不具合が減少しないという状況にあった．このように設計部の不具合見逃しが多い状況では，QA における不具合摘出の負荷が高まり，充分な試験ができず，出荷後不具合発生のリスクを増大してしまっていることになる．

よって，テストプロセス改善の目的は，設計部内のテストによる不具合の抜け漏れを防ぐこと，つまり，「設計内の摘出不具合数を増加すること」とする．また，納期遵守が顧客や市場から求められているため，「納期を守ること」を必要条件とする．

[1] 対象組織では，入力値，前提条件，期待結果，事後条件に実行手順を加えたものをテストケースとしている．

[2] テストエビデンスは，ISO9000:2005(JISQ9000:2006)の「検証」の定義の「客観的証拠」に基づいている(表 1 参照)．

3. テストプロセスの課題分析と改善方針

3.1. TOC 思考プロセスを用いたテストプロセスの課題分析と改善の概要

前章に示すテストプロセス改善の目的達成のため，改善すべき課題の分析に，TOC 思考プロセスを用いた．一般的な問題解決手法は，問題を複数の細かな要素に分割し，それら一つ一つの改善を考えるのに対し，TOC 思考プロセスは，全体を考慮してごく少数の要素を改善することで，全体の成果を大きく向上する手法である[6][7]．特に，人が介在し，複数の事象が相互に関係するシステムにおいて効果が高いといわれているため，本報告では，テストプロセス改善にこの方法を適用した．具体的には，TOC 思考プロセスのツールである「現状構造ツリー」と「対立解消図」を用いた．

現状構造ツリーは，目的や必要条件を阻害する様々な「好ましくない結果（UDE: UnDesirable Effect)」の因果関係をたどっていくことで，システムが抱えている解決すべき「中核問題」を特定する．TOC 思考プロセスでは，全ての問題は対立構造を内包していると考え，「中核問題」を持続する対立を対立解消図で表現する．この構造を明確化することで問題解決方法を導く．この TOC 思考プロセスの課題分析方法を纏めると図 1のように，現状構造ツリーが「何を変えるか？」を示し，対立解消図が「何に変えるか？」を示す．

なお，本報告では，TOC 思考プロセスにおける「3 クラウド法」を用いている[7]．この方法では，まず，いくつかの UDE から中核問題の対立構造図を導出し，それが全ての UDE を発生していることを現状構造ツリーで確認する．そのため，現状構造ツリーに対立解消図の構造が含まれる形となる．

ただし，本報告では，流れを明確にするため，現状構造ツリー→対立解消図という流れで説明する．

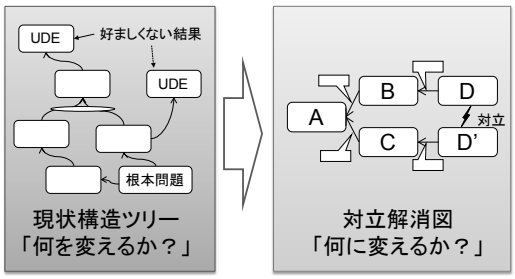

図 1　TOC 思考プロセスによる問題解決

3.2.「現状構造ツリー」によるテストプロセスの課題分析

3.2.1. 現状構造ツリーの作成

前章に示した目的と必要条件を阻害している UDE を以下の方法で抽出した．

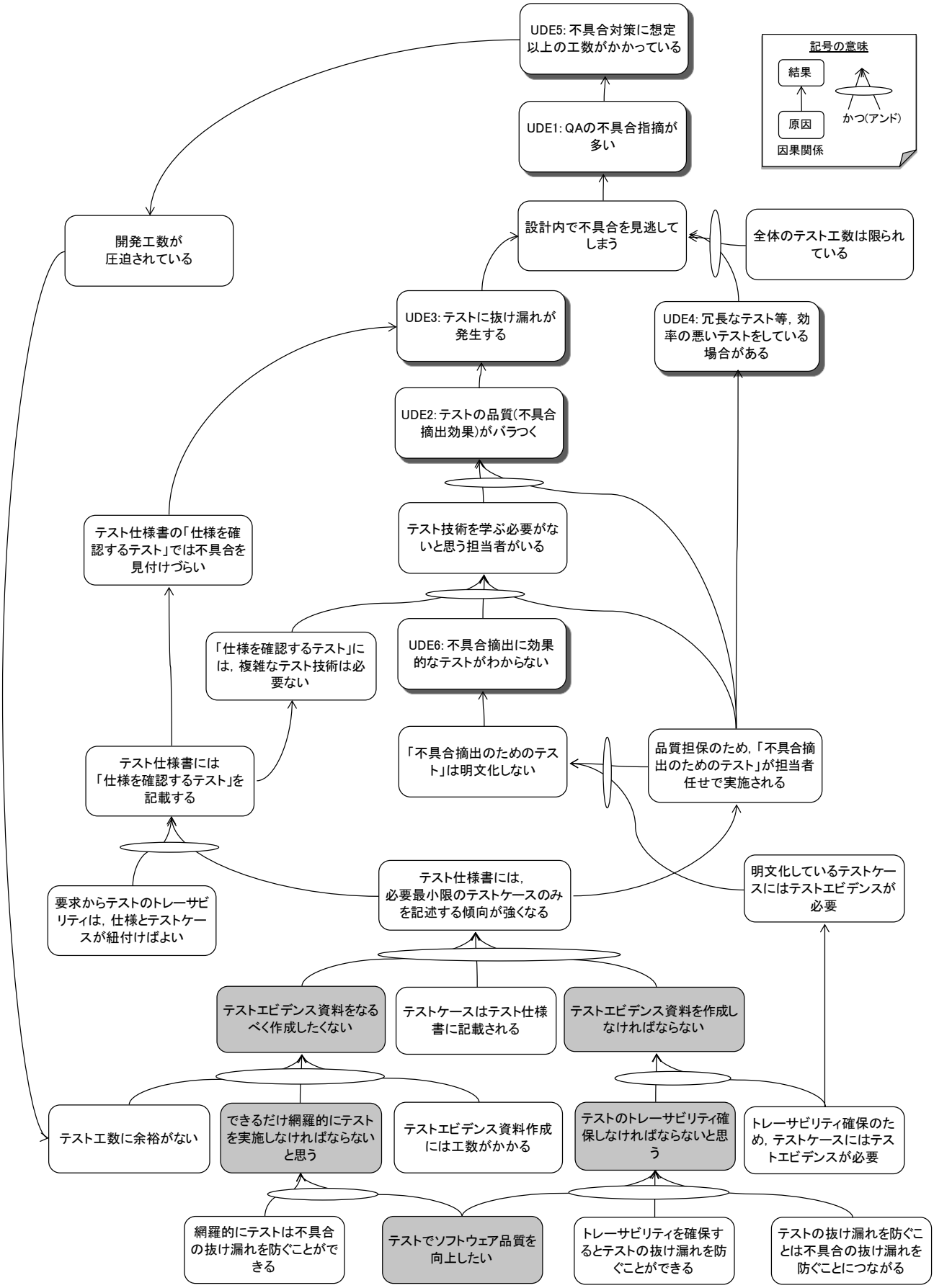

図 2 テストにおける現状構造ツリー

- 開発者へのヒアリングや開発者との議論内容
- ドキュメント等の調査

これらの方法で抽出した UDE は，以下である．

UDE1：QA の不具合指摘が多い
UDE2：テストの品質(不具合摘出効果)がバラつく
UDE3：テストに抜け漏れが発生している
UDE4：冗長なテスト等，効率の悪いテストをしている場合がある
UDE5：不具合対策に想定以上の工数がかかっている
UDE6：不具合摘出に効果的なテストがわからない

この UDE より，因果関係を整理して現状構造ツリーを作成した(図 2)．なお，図中のボックスは事象を示しており，矢印が因果関係を示す．つまり，矢印元のボックスが「原因」を表し，矢印先がその「結果」を表し，「もし○○(原因)ならば，○○(結果)である」と読む．明確なルールではないが，主に，上方に「結果」を下方に「原因」を配置する．また，複数の矢印を束ねる楕円は，アンド条件を表し，「もし○○(原因 1)かつ○○(原因 2)ならば，○○(結果)である」と読む．

3.2.2. 現状構造ツリーによる課題分析結果

図 2より，テストプロセスにおける課題を分析する．

対象組織では，ソフトウェア品質を向上するという目的があり，前章に示したとおり，トレーサビリティ確保を内部規定としている．テストにおけるトレーサビリティの一つであるテストエビデンスとテストケースに紐付けを確保するためには，テストエビデンス資料をまとめなければならない．

一方，ソフト品質向上のため，不具合の抜け漏れを防止するには，網羅的なテストを実施しなければならない．しかし，テスト工程は開発工程上の終盤にあり，想定外の開発工数の増加がテスト工数を圧迫しているため，テストエビデンス資料作成は，なるべく工数をかけたくないという心理が働く．

このような要因に基づき，要求仕様書や設計仕様書などの仕様を確認するテストケースのみを必要最小限としてテスト仕様書に記載する傾向が強くなる．しかし，必要最小限のテストケースでは，ソフトの品質を担保できない．そこで，テスト仕様書に記述するテストとは別に，内部規定には明示されていない「不具合摘出のためのテスト(動作確認など)」を開発担当者に水面下で委任して実施することとなる．また，明文化しているテストケースにはテストエビデンスが必要であることから，このような不具合摘出のためのテストは明文化されない．よって，不具合摘出は開発担当者の力量に依存してしまい，テスト品質にバラつきを生じる．結果として，設計内で不具合を見逃してしまう．さらに，不具合見逃しによる QA 指摘不具合の対応工数の増加が，想定外に開発工数を増加し，テスト工数の圧迫を招く．

つまり，ソフト品質の向上をめざして，テストのトレーサビリティ(テストエビデンス)を確保することが，テスト工数の圧迫などにより，心理的，物理的な負荷となり，不具合摘出のためのテストを水面下で担当者任せにし，不具合の抜け漏れにつながっている．

3.3.「対立解消図」によるテストプロセスの課題改善方法の導出

前節でテストプロセス上にテストエビデンスが関わる中核問題を示した．本節では，この中核問題について対立構造図を作成し，この問題の解消方法を導く．

3.3.1. 対立解消図の作成

対立解消図のフレームワークは，図 3である．(A)が目的であり，(B)と(C)が(A)の達成に必要な条件．(B)達成に必要となる行動(D)と(C)達成に必要となる行動(D')が対立関係となる構造である．さらに各矢印には理由（アサンプション）があり，破線四角で記載する．

現状構造ツリーで明らかになった中核問題の対立構造図は図 4となる．図 4より，(A)テストでソフトウェア品質を向上するためには，(B)できるだけ網羅的なテストを実施しなければならない．(B)できるだけ網羅的なテストを実施するためには，(D)テストエビデンス資料を作成しない．一方，(A)テストでソフトウェア品

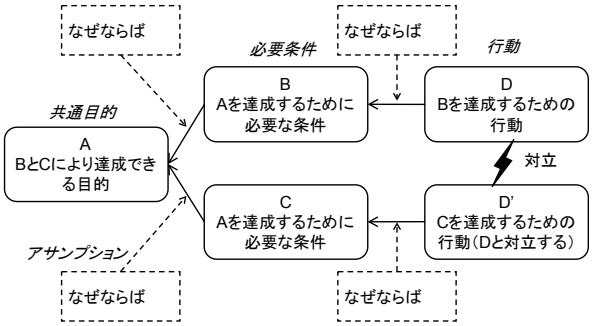

図 3　対立解消図のフレームワーク

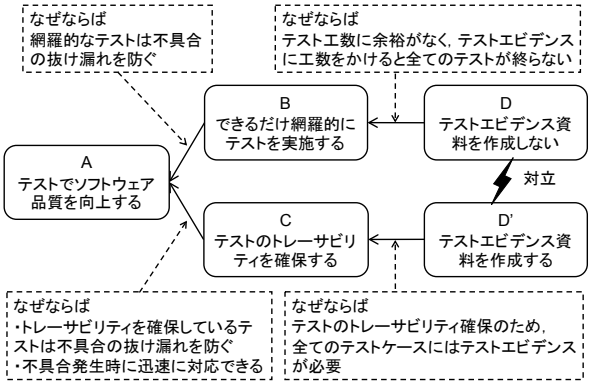

図 4　テストの課題における対立解消図

質を向上するためには, (C)テストのトレーサビリティを確保しなければならない. (C)テストのトレーサビリティを確保するためには，(D')テストエビデンス資料を作成しなければならない．(D)と(D')は対立する．

3.3.2. 対立解消図による改善策

対立解消図では，図 4のような対立関係の解消を検討することで,中核問題の改善策を導く.具体的には，破線の四角で示しているアサンプションの正当性を確認し，(B)と(C)の両方を満たす解決策（インジェクション）を検討する．検討の結果，図 4の(C)と(D')の間にあるアサンプション「全てのテストケースにはテストエビデンスが必要」に開発者の思い込みが含まれていることを明かにした．

テストエビデンスは，ISO9000 に定義される「検証」の客観的証拠に相当するが，前節に示した不具合摘出のためのテストは，ソフト品質向上活動であり，ISO9000 で必要に応じて実施すると定義される「検査」に対応する(表 1)．この「検査」では，ISO の規格上も明に客観的証拠(テストエビデンス)は求めていない．つまり，不具合摘出のために実施するテストには，テストエビデンスは必須要件ではないといえる．

したがって，不具合摘出のためのテストについては，テストエビデンス資料を作成せずにテストを明文化できる．そこで，本報告では，不具合摘出のためのテストの明文化・品質のバラつき排除をテストプロセス改善のためのインジェクションとして提案する．

表 1　ISO9000:2005(JISQ9000:2006)における「検証」と「検査」の定義[1]

	定義
検証 (Verification)	客観的証拠を提示することによって，規定要求事項が満たされていることを確認すること.
検査 (Inspection)	必要に応じて測定，試験又はゲージ合せを伴う，観察及び判定による適合性評価.

4. テストプロセス改善

4.1. テストプロセス改善概要

前章に示したテストエビデンスの持つ問題点を解消するため，以下の施策で，不具合摘出のためのテストを明文化・品質バラつきを排除する．

(1) 従来のテストと不具合摘出のためのテストを明確に区別するため，それらの違いを定義する．
(2) 不具合摘出のためのテストを明文化し，レビューを実施することで，テスト品質の開発担当者依存を排除する．

これら施策の詳細を次節以降で説明する．

4.2. テストの定義

仕様確認のためのテストと不具合摘出のためのテストを以下に示すように明確に定義する．なお，これらテストはともに設計テストである．

- 仕様確認テスト：ソフトが仕様を満足していることの確認を目的とするテスト. ISO9000 の「検証(Verification)」に対応する．
- 不具合摘出テスト：不具合の摘出を目的とするテスト．ISO9000 の「検査(Inspection)」に対応する．

本定義は，ソフトウェアテストが持っている「仕様を確認する」と「不具合を摘出する」の二つの目的を明確に分離するものである．

仕様確認テストは，ISO9000 の「検証」に対応するため，テストケースは内部規定準拠のテスト仕様書に記載し，テストエビデンス資料も作成する．

不具合摘出テストは，ISO9000 の「検査」に対応するため，テストケースの記述フォーマットは特に定めず,テストエビデンス資料も特に作成する必要はない．

対象組織では，仕様確認テストのテストケースと不具合摘出テストのテストケースの関係を図 5に示すとおりとした.仕様確認テストのテストケースの多くは，不具合摘出テストの一部とし，それ以外は性能テストやメモリリーク対策確認などを補完として追加するものとした．

仕様確認テストと不具合摘出テストについて，その差異を表 2に纏める．

本定義により，対象組織では，設計テストとして，不具合摘出テスト(Inspection)で不具合を摘出して品質を向上し，仕様確認テスト(Verification)で仕様を満たしていることを確認する．その後，QA による認定試験(Validation)で製品の妥当性確認を行うこととした．

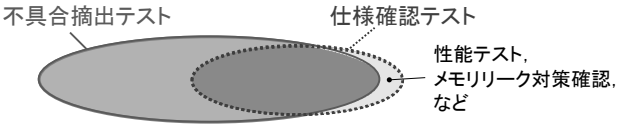

図 5　不具合摘出テストと仕様確認テストの関係

表 2　仕様確認テストと不具合摘出テストの差異

	仕様確認テスト	不具合摘出テスト
テストの目的	仕様を満足していることの確認	不具合の摘出
ISO9000対応	「検証」に対応	「検査」に対応
テストケースのフォーマット	内部規定のテスト仕様書	フリーフォーマット
テストエビデンスの必要性	有	無
テストケースの相互関係	（仕様確認テスト－性能テスト等）⊆ 不具合摘出テスト	

4.3. テスト観点レビュー

不具合摘出テストの開発者依存性を排除するため，テストレビューを導入する．レビューのためには，不具合摘出テストを明文化する必要があるが，本報告では，マインドマップ[8]を用いることとした．その理由は，以下である．

- 対象組織では，「テストケースにはテストエビデンスが必要」という文化が根強く，テスト範囲を無意識的に狭めてしまう傾向がある．マインドマップは発散的に発想を広げるため，その障害を取り除くことを期待した．
- 内部規定のテスト仕様書(従来の仕様確認テストのフォーマット)からテストの全体像を把握するには，複数のテストケースからテストの設計意図をくみとる必要があり，テストの抜け漏れを発見しにくい．マインドマップで，テスト設計者の意図をノードに記載したものをレビュー対象とすることで，レビュー効果を得やすいと考えた[9][10].

対象組織では，従来の仕様確認テストであるテスト仕様書のレビューを「テストレビュー」として既に行なっていたこともあり，上述のテスト設計者の意図を「テスト観点」と呼び，マインドマップによる不具合摘出テストのレビューを「テスト観点レビュー」と名付けた．また，テスト観点レビューは，あくまでテスト観点のレビューなので，その後，それに基づいて不具合摘出テストケースを作成するが,効率の観点から，それらはレビューする必要はないものとした．

テスト観点レビューを実適用にあたり，以下の三点について工夫を行なった．

(1) マインドマップツールの使用

マインドマップはその構造そのものに意味を持つため，レビュー指摘により，マインドマップの構造の変更が必要になる可能性もある．手書きのマインドマップでは，このような変更に柔軟に対応することは困難である．

そこで，テスト観点レビューでは，PC 上でマインドマップを作成できるツール XMind[11]を用いることとした．また，レビュー時に，指摘箇所をできる限りその場で即座にマインドマップに反映することとした．

(2) テストケース作成の省略

不具合摘出テストは，ソフト品質向上活動のために実施するものであるので，効率的に多くのテストを実施することが望ましい．そこで，マインドマップそのものを用いれば，開発担当者がテストを実施できる場合に限り，テストケースを作成する必要はないものとし，テスト実施に注力できるようにした．

(3) テスト実施範囲の決定の分離

テスト観点レビューでは，テスト観点の抜け漏れを指摘しやすい反面，実時間以内に実施不可能なほど多くのテストを洗い出せてしまうことがある．しかし，テスト観点レビュー中にテスト実施を意識すると，発想を阻害され，テスト観点の抜け漏れを発生する可能性がある．

そこで，テスト観点レビューと別に「テスト実施範囲の決定」というアクティビティを設けた．テストの想定漏れをなくすため，テスト空間の想定範囲を一度発散的に拡張し，その後実際にテストする範囲を収束させる．この「発散」が「テスト観点レビュー」であり，「収束」が「テスト実施範囲の決定」に対応する．

5. テストプロセス改善の効果評価

5.1. パイロットプロジェクトによる試行・評価

本提案手法によって，テストエビデンスの持つ問題点を解消し，設計内での摘出不具合が増加することを確認するため，複数のパイロットプロジェクトについて提案手法を試行し，評価を実施した．

評価は以下の二つの点で実施する．

(1) 定性的評価：パイロットプロジェクトメンバーへのヒアリングとアンケート

(2) 定量的評価：パイロットプロジェクトでの不具合摘出テストのテスト観点レビューによる追加テストケース数と不具合数の分析

5.2. 定性的評価

パイロットプロジェクトメンバーへのヒアリングで得られた意見で主要なものを以下に示す．

(1) 効果の実感

不具合摘出テストについて

- テストの定義で,不具合を摘出しやすくなった．
- テスト観点を挙げやすくなった．
- テストで不具合を摘出する意識が高まった．

マインドマップについて

- テスト観点の整理・洗い出しに良い
- テスト観点を残せるのが良い

テスト観点レビューについて

- テスト仕様書よりレビューしやすい
- 重大な抜け漏れを発見できた
- 例外処理などを多く叩き出すことができる
- 設計の詰めが甘いところを見付けられた
- 若手への教育的効果もありそう
- 協力会社のテストレビューに使うと良さそう

(2) 改善検討事項

マインドマップについて

- 性能系など適していないテストもある
- 大きな機能の場合，表示等に工夫が必要
- 単語のみで作成すると意図がわからなくなることがある

テスト観点レビューについて

- レビューアにはドメイン知識が必要
- レビューの人数・時間の目安がわからない
- レビュー後のマインドマップの整理が大変

その他

- マインドマップからテストケースが生成できるとよい

また，ヒアリングで得られた意見を元にパイロットプロジェクトメンバー52 名にアンケートを実施した．アンケートは，各項目に対し「強く同意する/同意する/どちらとも言えない/同意しない/強く同意しない」の 5 段階で評価してもらった．結果を図 6に示す．

アンケート結果より，仕様確認テストと不具合摘出テストを分けて定義することについて，「品質向上に効果がある」に肯定的な回答(「強く同意する」「同意する」)が 75%,「定義することは良い」に肯定的な回答が約 75%であった．また，同様にテスト観点レビューについても「品質向上に効果が高い」に肯定的な回答が 90%以上,「全体的に良い方法だ」に肯定的な回答が約 80%であった．

よって，本提案手法は，開発者に対し，テストエビデンスの課題を解消し，品質向上に効果があるという認識を与えているといえる．定量的評価

パイロットプロジェクトにおいて，テスト観点レビューで追加になったテスト観点から作成されたテストケース数とそのテストケースから摘出された不具合数を集計した．結果を表 3に示す．ただし，( ) 内に追加分も含む全体数を示す．また，各プロジェクトにおける開発規模と認定試験における QA の指摘件数も併せて示す．

本節では，集計したデータの合計値を用いてテスト観点レビューの定量的評価を実施する．

表 3　テスト観点レビューで追加となったテストケース数とそのテストによる摘出不具合数

PJ	テストケース数[件]	摘出不具合数[件]	開発規模[KLOC]	QA 指摘[件]
A	22(144)	1(1)	3.5	1
B	36(160)	2(3)	1.0	1
C	31(48)	2(4)	1.5	0
D	29(300)	0(5)	4.1	0
合計	118(652)	5(13)	10.1	2

ただし，( )内は全体での数

(1)　QA 指摘密度

対象組織では，開発規模に対する QA 指摘の件数である「QA 指摘密度」を用いて開発プロジェクトの品質を評価している．

パイロットプロジェクトにおいて，QA 指摘密度を算出すると，パイロットプロジェクト以外の QA 指摘密度の約 1/3 となり，パイロットプロジェクトの認定試験時の品質が高いという結果を得た．

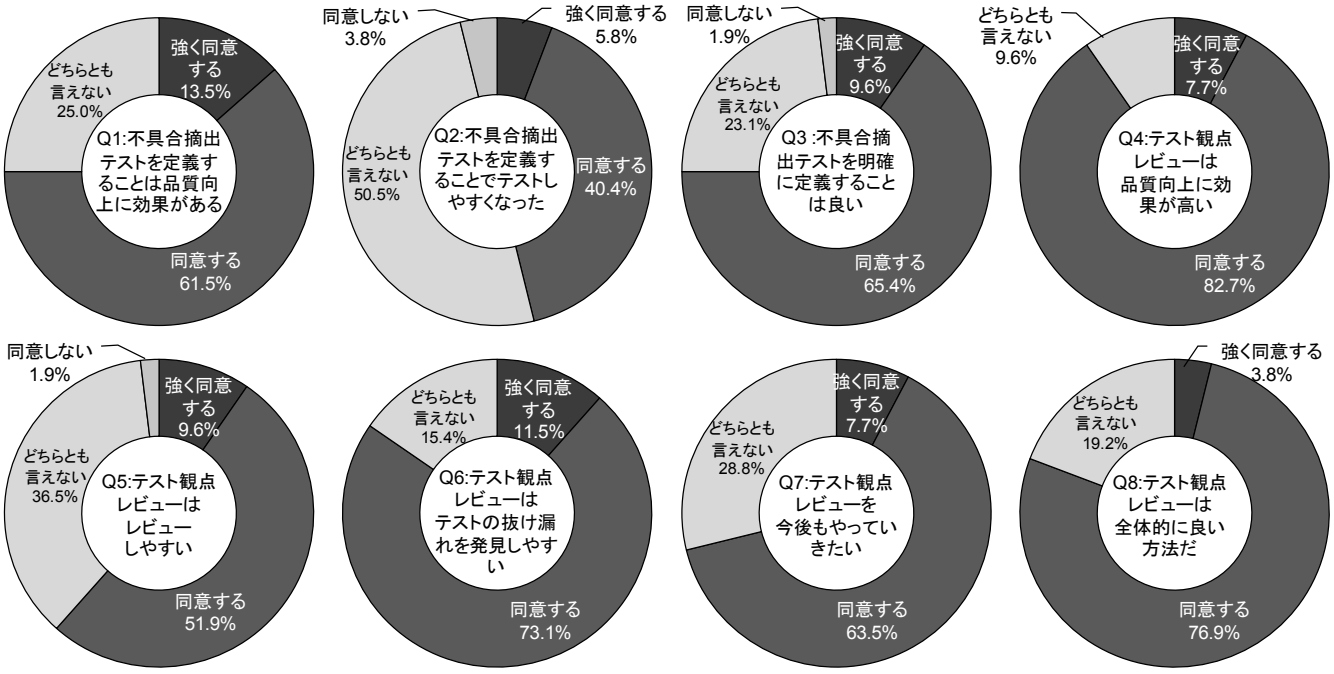

図 6　パイロットプロジェクトへのアンケート結果

(2) 不具合見逃し率

パイロットプロジェクト全体で，設計で摘出した不具合は 13 件，QA 指摘は 2 件であった．よって，設計部での不具合見逃し率は，以下で算出できる．

テスト観点レビュー実施時の不具合見逃し率

= 2/(13+2) = 13.3%

また，パイロットプロジェクト全体の設計摘出不具合 13 件のうち,テスト観点レビューにより追加となったテスト観点で摘出された不具合は 5 件である．この不具合は，テスト観点レビューを実施しなければ，設計では摘出できなかったと考えられる．つまり，この不具合が認定試験で QA に指摘されると仮定すると，テスト観点レビューを実施しなかった場合の不具合見逃し率は以下で見積ることができる．

テスト観点レビュー未実施時の不具合見逃し率(見積)

= (2+5)/(13+2) = 46.7%

よって，テスト観点レビューにより，設計部での不具合見逃し率が 46.7%から 13.3%に低減したと考えることができる．

(3) 不具合摘出効率

テストケース当たりの摘出不具合の数である不具合摘出効率を用いて，テスト観点レビュー時の指摘で追加となったテスト観点とそれ以外のテスト観点の不具合摘出能力を比較する．

テスト観点レビュー指摘追加のテスト観点から作成されたテストケースの数は 118 であり，同テストケースで摘出された不具合は 5 件であった．よって，テスト観点レビューにより追加となったテスト観点に関する不具合摘出効率は，

レビュー追加分の不具合摘出効率 = 5/118

= 0.042 件/テストケース

である．一方，レビュー指摘以外のテスト観点から作成されたテストケースの数は 652-118=534 であり，同テストケースで摘出された不具合は 13-5=8 件であった．よって，テスト観点レビューの指摘以外のテスト観点に関する不具合摘出効率は，

レビュー指摘以外の不具合摘出効率 = 8/534

= 0.015 件/テストケース

となる．つまり，テスト観点レビューで追加となったテスト観点では，不具合摘出効率が約 3 倍高いということがいえる．

上述(1)，(2)，(3)に示すとおり，不具合摘出テストのテスト観点レビューは，テストの抜け漏れを防ぎ，ソフトの品質の向上に効果があると考えられる．

6. まとめ

本報告では，ある組込みソフトにおけるテストプロセス改善として，TOC 思考プロセスを用いて中核問題を特定し，改善施策を提案した．課題分析の結果，対象組織でソフト品質向上のために内部規定としているトレーサビリティの一つであるテストエビデンスがテストによる不具合摘出を阻害していることを示した．本報告では，その問題の背景に「全てのテストにテストエビデンスが必要である」という開発者の誤った思い込みが含まれていることを明確にし，その思い込みを解消する施策として，不具合摘出テストの定義と不具合摘出テストのテスト観点レビューを提案した．

本提案手法を評価するため，複数のパイロットプロジェクトに対して提案手法を試行した．効果が出ないプロジェクトもあったが，全体で平均すると，今回のケースでは，テスト観点レビュー試行により不良見逃し率が 46.7%から 13.3%へ低減し，QA 不良密度が約 1/3 になるという結果を得た．また，レビューで指摘されたテストケースの方が指摘以外のテストケースと比較して，不良摘出効率が約 3 倍高いという結果を得た．さらに，本提案手法を試行したパイロットプロジェクトメンバーに対し,アンケートを実施したところ,提案手法が品質向上に効果が高いと思うという意見が全体の 90%を超えるという結果を得た．

よって，不具合摘出テストの定義とテスト観点レビューは， TOC 思考プロセスを用いて明確にした，テストエビデンスの持つ問題点を解消し，設計内での摘出不具合の増加に効果的であると考えられる．

文　献

[1] JIS Q 9000 :2006, 品質マネジメントシステム―基本及び用語.

[2] ISO26262, Road vehicles - Functional safety.

[3] IEC61508, Functional Safety of Electrical / Electronic / Programmable Electronic Safety-related Systems.

[4] IEC62304, Medical device software - Software life cycle processes.

[5] Mary Beth Chrissis, Mike Konrad, Sandy Shrum, JASPIC CMMI V1.1, “CMMI 標準教本 ~プロセス統合と成果物改善の指針,” 日経 BP 社，2005.

[6] エリヤフ ゴールドラット， ザ・ゴール 2 - 思考プロセス，ダイヤモンド社, 2002.

[7] 村上悟，問題解決を「見える化」する本，中経出版，2008.

[8] Tony Buzan, Barry Buzan, ザ・マインドマップ，ダイアモンド社, 2005.

[9] 池田暁，鈴木三紀夫，マインドマップから始めるソフトウェアテスト，技術評論社, 2007.

[10] 西康晴, “テスト観点に着目したテスト開発プロセス(VSTeP)の概要,” JaSST2009 東京, 2009.

[11] XMind, http://www.xmind.net/