第 5分科会（テスト分析グループ）

要求仕様書におけるテストエンジニアの視点を活かした欠陥検出方法の提案

A proposal for the defects detection method on software requirements specification with a point of view of software testing engineer

主査　　秋山浩一（富士ゼロックス（株)）、副主査　奥村有紀子（(有）デバッグ工学研究所）
研究員（リーダ）　近江久美子（ＴＩＳ（株)）、研究員（サブリーダ）　永田敦（ソニー（株））
研究員　阿部修久（キヤノンＩＴソリューションズ（株)）、天野佑太（(株）日立メディコ）
伊藤慈朗（東芝デジタルメディアエンジニアリング（株)）、
上田克則（(株）大和コンピューター)、朝永糸子（東京海上日動システムズ株式会社）
森崎一邦（東芝電波システムエンジニアリング（株））

概要

プロジェクトにおいて、欠陥の検出は早期、特に発生工程内で行われることが望ましい。上流工程に起因する問題が後工程であるテスト工程で見つかる場合について言及している先行研究もある。しかし現実には、早期検出できないプロジェクトも多いと考えられる。

そこで本研究では、開発の初期工程で作成されることが多い要求仕様書に対して、欠陥検出する手法を検討した。テストエンジニアの視点に基づいたレビューにより、要求仕様書の欠陥検出を容易にすることを狙いとしている。当該工程でより多く欠陥を検出することで、後工程での欠陥検出や修正の工数を抑えられる。また、早期に要求仕様書の欠陥の状況を踏まえることで、漏れのないテスト分析を可能とする。

本論文では、提案する手法の詳細と、実際に適用した結果について報告する。

Abstract

Software defects should be detected as early as possible. The best way is to detect them at same development stage in software development project. An earlier study describes the defects that were inserted at upper level of process were found at test process. But software defects could not be found in early stage in many software development projects.

We study the detection method for software defects in Software Requirement Specifications that shall be made at earliest stage of development. We attempt to find defects in Software Requirement Specification easily by review in a point of view of testing engineer. We could find more defects by this method at the stage where they were inserted and reduce the cost of finding and fixing defects at the next stage. Also we could find the defects of the Software Requirement Specification earlier, we could issue test analysis without oversight.

We report the proposed method and result of review with it.

1. 現状と問題

1.1 現状

先行研究（ [1]）において、次の課題に対して改善策を研究している。

- 上流工程での品質問題は目に見えないため見過ごされがちであり、後工程になってモジュールを組み合わせたときのテストで不具合として発覚することが多い
- テスト工程で発見された不具合が、上流工程である分析や設計工程のミスに起因している場合、手戻り作業に多大な時間とコストを費やすこととなる

本課題は、80年代から次のとおり提唱されている。

・手戻りは、全開発費の 30～50%を消費する （[2]）

・要求の誤りは 手戻りコストの 70～85%を占める（ [3]、 [4]）

一方、本研究グループのメンバからも次の問題があがった。

(1) 要求仕様の決定が遅い場合があり、十分なテストが実施できない

要求仕様に未決定項目があると上流工程でテストのアプローチや戦略が計画できず、テスト分析・設計の開始も遅れる。そのため十分なテスト分析・設計ができず、設計網羅やテスト網羅の十分性が説明できない。一方、要求仕様の未決定項目はあとの設計で決めるから問題ないという風土がある場合があり、テストエンジニアが仕様を確認し質問をしても真摯に対応しない（できない）ことがある。

(2)要求仕様書の内容が粗く行間を読み取れないため、テスト項目の抜けが発生する

(1)と関係するが、特に、ユースケーステストのシナリオ不足が発生する。

1.2 問題点

先行研究（ [1]）では、改善策として上流工程の品質の可視化技術をあげており、対象項目としてはソフトウェアの見積もり（規模・工数）・品質メトリック計算・要求分析フェーズにおけるレビューが重要と捉えている。しかし、具体的な施策には要求分析の品質メトリック計算にスコープを当てており、レビュー方法について言及されてはいない。

また、世の中では、要求の曖昧さを取り除くために要求工学の普及をすすめているが、明確な要求を定義する具体的な方法には、至っていない状況である。

2. 目的

そこで、本研究では、静的テスト技法のひとつであるレビューに着目した。また、[1]より上流工程に起因する欠陥がテストで検出される場合があるということは、上流工程でもテストエンジニアの視点による欠陥検出が可能ではないかと考えた。ここから、テストエンジニアの視点を活かして要求仕様書をレビューする方法、すなわち「要求の欠陥」の検出方法について、具体的な分析方法を提案することを本研究の目的とした。

なお、本提案により、上流工程における要求の欠陥を検出することで、早期に要求の是正が可能となり、結果として早期に質の高いシステムテストにつながるテスト分析作業が可能になると考えている。

質の高いシステムテストとは、以下を想定している。

・要求の誤解から発生するテストミスがない

・テスト項目の漏れや重複がない

・上流工程における適切なテストのアプローチや戦略が作成できる

など

3. 活動の経緯

要求仕様書に対しての具体的なレビュー方法を検討するにあたり、我々は、以下 (1)～ (6)の手順で活動を進めた。

(1)上流工程におけるテストエンジニアの貢献度の検討

まず、要求の欠陥検出に対してテストエンジニアができることとして、次の 3点をリストアップした。

1)要求仕様書のレビューにテストエンジニアが参画する

現状では、上流工程におけるテストエンジニアとしてのレビューの貢献度が低いという認識である。

2)当該レビューにおいて有効な指摘を行う

現状では、欠陥を指摘しても、あとで設計するから大丈夫と真摯な対応がとられないことが多い。また、質問し難い雰囲気、すなわち「そのぐらいも判らないの？」という空気が漂っている。

3)要求仕様書の記述内容の理解を深める

日本語のねじれやドメイン知識不足によりテスト項目（ユースケーステストのシナリオ）の抜けが発生すると考えた。

ここで、ユースケースを取り上げた理由は、次のとおり。

A) システムテストフェーズ以降で、要求の妥当性を確認するためユースケーステストを実施する。

B) ユースケースドキュメントは、作業依頼者（発注者など）と開発者間でコミュニケーションをとりやすいため。ここでのユースケースドキュメントは、ユースケース図やユースケースシナリオ（付録 1）など、様々なダイヤグラムやドキュメントを含んでいる。例えば、ユースケースシナリオはアクタが活用する場面を想定して、テストケースを駆動させる条件やフローを記述する。そのため作業依頼者と開発者間での認識が合いやすい。なお、本研究でのシナリオの記述粒度は、ソフトウェア技術者が設計をイメージできるレベルと想定している。

(2)開発現場の開発プロセスの整理を行い、要求の欠陥を検出するフェーズを決める

開発プロセスは、本研究グループのメンバ間でも多少齟齬があるため、開発の共通物差しになっている共通フレーム（Software Life Cycle Process）を元に検討した。

本研究では、ソフトウェア要求分析フェーズでのレビューを対象とした。ソフトウェア要求分析はソフトウェア方式設計の入力資料を作成するため、ソフトウェアの要求の欠陥を検出するフェーズとして適正だと判断したためである。参考に本研究グループのメンバの要求分析状況を箇条書きに記述する。

・顧客要求仕様書をもとにソフトウェア要求分析を実施し、ユースケース作成から始めることが多い

・要求仕様書は、A4用紙 1枚程度から機能要求一覧を記述しているレベルまで多岐にわたっている

当該分析の成果物としては、ソフトウェア方式設計の入力資料（本論文では、「ユースケースドキュメント」（UCDoc）とする）を作成する。UCDocの選定理由は、上述の B) を参照。

(3)先行研究の調査

先行研究調査の結果、 [5] にて「要求の曖昧さ」と要求レビューについての解説を確認できた。 [5] は要求の欠陥のひとつである曖昧さに焦点を当てているため、検討のベースとした。

(4)レビュー視点の検討

ソフトウェアのレビューを実施する上でレビュー視点は、大きく３つに分類できる。「ユーザ」・「開発エンジニア」・「テストエンジニア」である。そして、それぞれの立場でレビュー視点を検討した（ [6]）。

1)「ユーザ」の視点

1.要求仕様書に記述されたすべての要求は、ソフトウェアに期待する内容か（Validation）

2.システムを利用する全ての関係者は識別されているか（Validation）

3.すべての状況における入力に対して振る舞いが定義されているか（Verification）

2)「開発エンジニア」の視点

1.設計に必要な情報は全て定義されているか（Verification）

2.すべての外部インタフェースの定義は明確か（Verification）

3.要求仕様書の内部で、他の記述と矛盾している部分はないか（Verification）

3)「テストエンジニア」の視点

1.要求仕様書に記載された、値について、単位や要求される正確性および精度が定義されていることを確認できる方法はあるか（Verification）

2.すべての機能が要求を満たしていることを確認できる方法はあるか（Verification）

3.すべての機能が正しい結果を出力していることを確認できる方法はあるか（Verification）

4.どのくらいのテストをするべきか（テストの量と質の妥当性）判断し、確認できる方法はあるか（Validation）

(5)要求分析表の検討

(1)から (4)をもとに、システムテストで実施することが多いユースケーステストに着目した。シナリオ不足を防止するという観点から、ユースケーステストを行う「テストエンジニア」の視点、すなわち (4)の 3)に記載した視点から要求工学の情報 [5]をベースにして「要求仕様書分析表」を考案した。

この表を利用することで、要求仕様書に対する質問事項や欠陥が簡単に検出できるのではと考えた。

ここで注目したのは、 (4)の (3)「テストエンジニア」の視点にある「確認できる方法はあるか」（アンダーライン部分）である。「確認できるか」は、言い換えると「以下のようなフレームを埋めていくことができるか」ということであると考えた（図 1）。このフレームは [5] の一部を引用したものである。

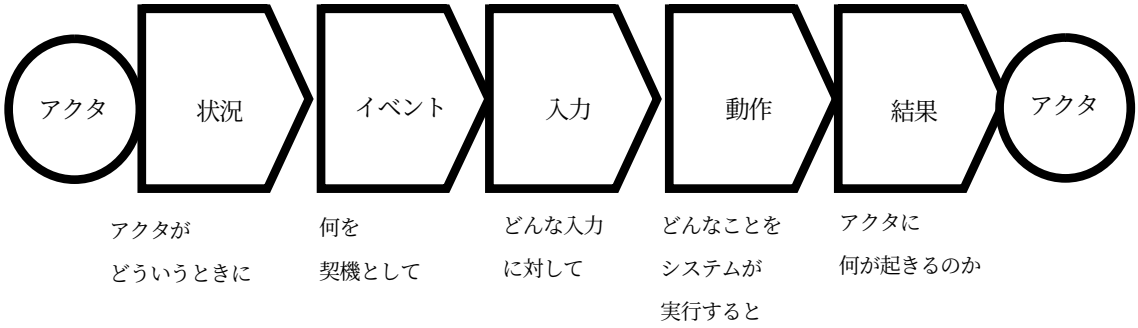

図 1　要求記述の構成要素のフレーム

我々は、上記のフレームを表にした「要求仕様書分析表」を用いて、テストエンジニアの視点（正常系、異常系の識別、境界値分析、同値分割など）で要求仕様書を分析することにより、問題を見つけることができるのではないかと考えた。

各項目の説明と想定している利用手順は、次章を参照のこと。

(6)要求とユースケースの網羅性（要求 UC（ユースケース）マトリックス）の検討

UCDoc は、要求仕様書分析表と同じく要求仕様書をもとに作成しているため、要求分析表で要求の欠陥を検出した場合、UCDoc に影響を与えることが想定できる。そのため、本表に記述している要求と UCDoc（とくに、ユースケースシナリオ）との網羅性を確保する必要があるため「要求 UCマトリックス」を考案した。

上記フレームを活用するプロセスは、 5章に記載する

4. 本手法に登場する表の利用方法

4.1 要求仕様書分析表（付録 2）

(1)要求仕様書分析表の列項目

・機能要件：ソフトウェアに求められる機能面の要求

・非機能要件：ソフトウェアに求められる機能面以外の要求

・目的：ソフトウェアを実行する目的

・アクタ：ソフトウェアを利用する関係者（入力側）

・状況：機能が動作する状況

・イベント：機能が動作する契機。ソフトウェアの内部から発生するものと外部からのものがある

・入力：機能に対する入力
・動作：状況やイベントおよび入力にもとづく動作内容
・結果：期待する出力や結果
・アクタ：ソフトウェアソフトウェアを利用する関係者（出力側）
・重要度：不具合発生時の影響から判断した分類

(2)重要度の判断基準

以下の 3段階で判断する。

A) 重大な不具合を引き起こすか、または、アーキテクチャ等システムの根幹にかかわる部分において手戻りを発生させる可能性がある

B) 不具合、手戻りを発生させる、または、テストが漏れるおそれがある

C) 質問事項等

(3)利用手順

1)要求仕様書を読みながら、表を埋めていく。読み取れないところは空白のままとする。正常系と異常系の識別、境界値分析などテスト技法を活用してパターンを増やし、行を挿入して追記する。

2)出来上がった表の見直しも兼ねて、問題の内容を該当するセルに記入する。その際、目立つように「？」やセルと文字に「色」をつける。

4.2 UCDoc

アクタの目的を達成するために、システムが実行する（すべき）振る舞いを記述したものである。ユースケース図、ユースケースシナリオだけでなく、状態遷移図やデータフローダイアグラムも含める。開発側で作成している場合がありそれを使用することを想定している（ [7]）。

4.3 要求 UCマトリックス（付録 3）

要求 UCマトリックスは、「UCDoc」と「要求仕様書分析表」との関係を表わすマトリックス表である。縦軸に「ユースケース」横軸に「要求」を記述し、関連するところに「●印」をつけていく。要求に対してユースケースに漏れが無いことを俯瞰的に表し、漏れによる大きな手戻りの発生を防ぐ効果がある。縦方向の粒度は要求仕様書分析表の機能要件、横方向はユースケースシナリオが望ましいと考えている。しかしながら、細かくしすぎると要求の変更や追加が発生したときのメンテナンスが大変になるので、トレーサビリティを確保したうえで、できるだけ粗くした方が使いやすい。

5. 3つの表、ドキュメントの使用方法と位置づけ

5.1 使用方法

各表の使用方法を次に説明する。

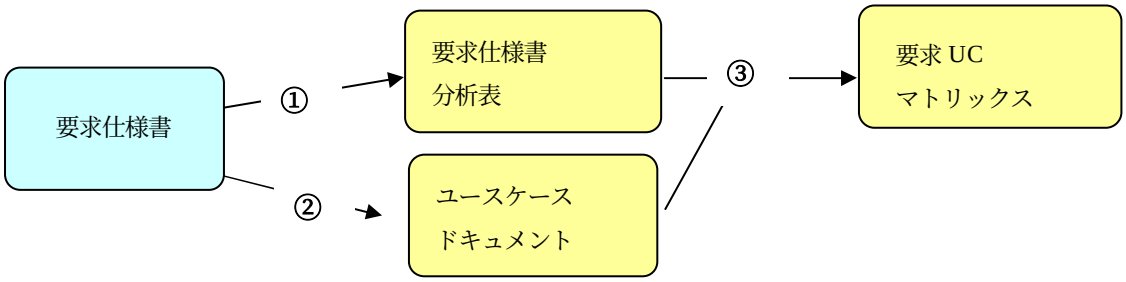

図 2　各表の使用方法

手順① .テストエンジニアが要求仕様書から「要求仕様書分析表」を作成して、要求仕様書の欠陥を検出する

手順② .開発エンジニアが要求仕様書から、UCDocを作成して、システムの振る舞いを記述する。

手順③ .テストエンジニアが「要求仕様書分析表」「UCDoc」から、「要求 UCマトリクス」を作成する。

5.2 開発全体における位置づけ

開発全体では、付録 4のようなイメージとなる。

6. 効果測定

6.1 効果比較実験の目的

本研究にて考案した要求仕様書分析表を使用し、要求仕様書のレビューを行い、その効果及び、有効性を検証する。

6.2 レビュー対象の要求仕様書

レビュー対象の要求仕様書は、効果実験の実施、および結果の分析を効率よく行うことを考慮し、 [6]の「FPMSシステム ソフトウェア要求仕様書」を使用した。

この要求仕様書は、仕様の簡潔化、記述項目の整理などが行われ、比較的短時間でレビュー可能な状態に編集されている。また、意図的に 21個の欠陥が事前に埋め込まれており、それらの欠陥は SRS品質特性と一意に対応付けが行われ分類可能な状態となっている。

品質特性	件数
完全性	14
無矛盾性	4
非曖昧性	2
妥当性	1

表 1　埋め込まれている欠陥の SRS 品質特性別件数

6.3 被験者

本研究グループのメンバにより、前述の要求仕様書のレビューを実施した。被験者は業務分野、職種、及び経験年数などがそれぞれ異なっているため、それらを分類した結果を表 2と表 3にまとめた。

業務分野	職種	人数
エンタープライズ系	SE	3
	マネージャー	1
	QA	1
組込み系	QA	1

表 2 業務分野／職種の分類

経験年数	人数
5年以上 ～ 10年未満	2
10年以上 ～ 15年未満	2
15年以上 ～ 20年未満	2

表 3 経験年数の分類

6.4 実験方法

手順は、以下の通り。

(1)要求仕様書を配布し、被験者は任意の時間にその要求仕様書の評価を実施する

(2)実施後、各被験者はより欠陥及び所要時間を報告する

また、評価方法による違いを確認するため、２チームに分かれて要求仕様書の評価を実施した。

(A)従来型のレビュー形式

レビュー観点やチェック項目などを指示しないアドホック形式。検出した欠陥を別途、一覧に記入。

(B)要求仕様書分析表を使用したレビュー形式

担当者には事前に本研究にて考案した要求仕様書分析表のフォーマットを配布し、要求仕様書の要求記述（文章）を構成要素単位に分解し記述した。要求記述を構成要素に分解する際に検出した欠陥を、同じく別途一覧に記入した。

6.5 実験結果

欠陥の検出結果をレビュー方法別に表 4 表 5にまとめた。

	完全性	無矛盾性	非曖昧性	妥当性
埋め込まれている欠陥の総数	14(100%)	4(100%)	2(100%)	1(100%)
従来型レビューによる検出数の平均	7(50%)	1(25%)	0(0%)	0(0%)
要求仕様書分析表を使用したレビューでの検出数の平均	6(43%)	1(25%)	2(100%)	0(0%)

表 4　埋め込まれた欠陥の SRS 品質特性別検出結果

レビュー方法	欠陥検出数	レビュー時間(h)	欠陥数／時間(h)
従来型レビュー	9.5	3.3	2.9
要求仕様書分析表を使用したレビュー	15.8	2.7	5.9

表５　全欠陥（事前に埋め込まれた欠陥と、それ以外で検出された欠陥）に対する検出結果

今回の実験より、本手法の効果について、以下の３点を確認することができた。

(1)非曖昧性に関する欠陥の検出に有効である

表 4より、埋め込まれた欠陥の検出数の合計には大きな差はないようにみえる。しかし、非曖昧性に関する欠陥の検出率は、「従来型のレビュー形式」が 0%に対して、「要求仕様書分析表を使用したレビュー形式」が 100%と、大きな差があることに着目したい。事前に埋め込まれたもの以外に検出された欠陥を分類したところ、同様に非曖昧性に関するものがより多く検出されていることがわかった。

要求仕様書分析表に記述することは、文書を構成要素単位に分解し記述するために、目的、及び具体的な事象などを検討することになる。それにより、レビューアーは文書に含まれる日本語の曖昧な表現、及び不完全な説明などの欠陥の検出が容易になっていることが分かった。

(2)単位時間当たりに検出した欠陥の数が多い

表 5は、事前に埋め込まれた欠陥以外に検出された欠陥数、および時間との関係を表している。「欠陥数／時間 (h)」より、要求仕様書分析表を使用したレビューの方が、より効率的に欠陥を検出したことが判明した。文書を要求仕様書分析表の構成要素へ機械的に分解し記述することにより、その作業に掛かる時間が短縮した。また、構成要素の項目が埋まらない、構成要素の項目に明確な内容を記述できない等の事象が視覚的に分かるため、欠陥の見逃しを抑制する働きもあると考えられる。

(3)異なる文書に記述された矛盾または、不完全な要求記述の欠陥には有効ではない

１つの文書を要求仕様書分析表の構成要素単位に分解して記述するため、他の文書で記述した内容とのチェックは困難となる。但し、この事象は従来型のレビュー形式でも同様の状況が発生するため、欠陥の検出結果に大きな差異は発生しないと考える。

7．考察と結論

要求仕様書分析表により要求仕様書を分析すると、この表を使わず、精読だけで分析した場合よりも、より多くの欠陥を発見することがわかった。これは、次の 2つの作用によると考えられる。要求仕様書分

析表の第 1行にある分類項目が、それぞれ適切な質問、そして理解する際の分析の軸となっていることである。それにより分析者は、分析される文章を表の分類項目ごとに仕分けし表のセルに記録する。その際、分析者は、分析内容を確認する方法が明確に理解できているか、つまり、テストをどのように行うかを説明できるかというテストエンジニアの視点をルールとしたレビューを行っているのである。それにより、分析者は仕分けられたセルの内容から明確性、非曖昧性、無矛盾性に問題があるリスクを認識し、欠陥を発見している(付録 5に例を示す)。一方、この方法では、文書間にまたがる完全性、無矛盾性については、その書かれている位置の距離が遠い場合、見つけにくい傾向にあることがわかった。これは、表にある分類項目の問いに対する答えを埋めていくという形式なので、分析スコープが対象文章に集中してしまうためと考えられる。まとめると、要求仕様書分析表を用いて要求仕様書を分析するポイントは、表によって分類された項目を、分析者の視点、ここではテストエンジニアの視点により分析を行っているということである。これにより、要求仕様書の欠陥をより多く発見することができた。

8. 今後の課題

研究会で検討した要求仕様書分析表については、検証結果より一定程度の有効性の検証を行うことができた。ただし、今回使用した要求仕様書分析表は非機能テストや文書間にまたがる完全性・無矛盾性については効果を発揮しきれていない。これを補うためには、抽象度を上げて UCDocが利用できないか検討する必要がある。今後の課題としては、UCDocが有効であることを検証すること、要求 UCマトリクスについて漏れのないテストのために有効であると検証することが挙げられる。

また、要求仕様書分析表、UCDoc、要求 UCマトリクスすべてを使いレビューを実施し、要求仕様書作成者にフィードバックする場合、最も有効活用できるタイミング・方法についても検討する必要がある。

9. 参考文献

[1]森俊樹 ,櫻庭紀子 ,中野隆司 :ソフトウェア品質技術の開発と適用 ,東芝レビュー 2006Vol.61 NO.1,2006
[2]B. W. Boehm , P. N. Papaccio:Understanding and Controlling Software Costs, IEEE Transactions on Software Engineering, v.14 n.10, p.1462-1477, October 1988
[3]Leffingwell 1997: Leffingwell, Dean. 1997. “ Calculating the Return on Investment from More Effective Requirements Management.” American Programmer 10(4):13-16.
[4] Software Requirements(http://www1.ocn.ne.jp/~kobakan/contents/software_requirements.html)
[5] 山本修一郎 ,連載 要求工学 第 38回 要求の曖昧さ ,エンタープライズ ICT総合誌 月刊ビジネスコミュニケーション 2007年 12月号 ,2007(http://www.bcm.co.jp/site/youkyu/youkyu38.html)
[6] 岡本博幸 ,内山敬太 ,鈴木彩子 ,高橋一仁 ,西山茂 ,野中 誠 ,要求仕様書の特性に着目した個人レビュー手法の実験的評価 , ソフトウェア品質管理研究会 第 20年度 (2004年度) 分科会成果報告 ,日本科学技術連盟 ,2005(http://www.juse.or.jp/software/study_data2004_6.html)
[7] @IT情報マネジメント(http://www.atmarkit.co.jp/aig/04biz/usecasedescription.html)
[8]ドロシー・グラハム ,エリック・ファン・フェーネンダール ,イザベル・エバンス ,レックス・ブラック ,秋山浩一 (翻訳),池田 暁 (翻訳),後藤和之 (翻訳),永田敦 (翻訳),本田和幸 (翻訳),湯本剛 (翻訳):ISTQB シラバス準拠 ソフトウェアテストの基礎 ,センゲージラーニング ,2008
[9]経済産業省 商務情報政策局 情報政策ユニット 情報処理振興課 組込みソフトウェア開発力強化推進委員会(監修) :2004年版組込みソフトウェア産業実態調査報告書〔1部(概要)〕〔2部(資料)〕 ,平成 16年(http://www.meti.go.jp/policy/it_policy/technology/softjittaityousa-gaiyou.pdf, http://www.meti.go.jp/policy/it_policy/technology/softjittaityousa-siryou.pdf)