Vol.2017-SE-195 No.6
2017/3/12

要求仕様の一貫性検証支援ツールを用いた要求仕様書のドキュメント品質の分析手法の提案と適用評価

位野木万里[†1]　大野昭徳[†1]　野村典文[†2]

概要：著者らは，要求仕様の品質特性である「一貫性」に着目し，ベテラン技術者が経験的に得た検証知識を形式知化し，それら知識に基づき，要求仕様の一貫性検証支援ツールを実現した．実システムの要求仕様書を用いて本ツールの適用評価を行い，本ツールは技術者の効率的な仕様検証を支援する点において一定の効果があることを明らかにした．著者らは，新たに，同一組織が作成した複数の要求仕様書の検証に本ツールを適用した．本ツールを用いて一貫性検証の観点で要求仕様書を分析した結果，ドキュメント品質の改善のノウハウが得られたので報告する．

キーワード：要求仕様書，一貫性検証，ドキュメント品質

Proposal and Its Evaluation of a Document Quality Analysis Method using Requirements Consistency Verification Support Tool for Requirements Specification

MARI INOKI[†1]　AKINORI OHNO[†1]　NORIFUMI NOMURA[†2]

Abstract: The authors developed a requirements consistency verification support tool, focusing on the consistency of the requirements quality. We have extracted tacit knowledge for verifying requirements and made it explicit as the verification knowledge which were incorporated into the tool. We have evaluated the tool by utilizing the actual requirements specifications; it was clarified that the tool was effective for an analyst to verify and improve the requirements specifications efficiently. In addition, the tool was applied to other actual requirements specifications. As a result of using the tool to analyze the requirements specification from a consistent verification standpoint, know-how for improving document quality was acquired and is therefore reported here.

Keywords: Requirements specification, Consistency verification, Document quality

1. はじめに

製品ソフトウェア開発において，要求定義工程は極めて重要であり，要求定義に関する標準[7]や知識体系REBOK[8] [9]が提供され，活用されている．要求定義は，開発対象となる領域，組織が直面する課題，利用する技術の条件に応じた試行錯誤が必要となり，初級の技術者が要求定義を行うことは，失敗のリスクが高く，要求定義はベテランの技術者のみが従事することとなり，効率的な要求定義の実施は困難な状況にある．

要求の分析者や要求の源泉となるステークホルダは，仕様記述ためのダイアグラムや形式言語の専門家ではないことが多く，要求仕様書は自然言語により記述されることが一般的である．また，アジャイル開発などの開発スタイルの需要が高まっているが[4] [21]，アジャイル開発ではユースケースシナリオ[1]や，ユーザストーリー[2] [23]などの自然言語により要求仕様を記述している．したがって，自然言語で記述された仕様を対象に，仕様の品質を一定に保つための取り組みは重要である．

木村らは，自然言語で記述された要求仕様の検証支援ツールを開発し，同ツールを用いた要求仕様の高品質化への取組み事例を報告している[12]．木村らによる検証技術は，設計要素の一つであるアクターにフォーカスし，ベテラン技術者による，アクターの表記ゆれを防止するための検証ノウハウをツール化した点が特徴である．しかし，事例[12]の成果は，アクター定義の観点で一貫した要求仕様を作成することに限定した，特定企業内での成功事例にとどまり，ノウハウやツールの一般化には至っていない．

著者らの研究グループでは，要求仕様の一貫性の検証の範囲に，アクターに加えて，機能への入出力となる「データ」，アクターとシステムとのインタフェースとなる「画面」，システムの働きに相当する「振る舞い」の設計要素を追加し，要求仕様の一貫性検証支援ツールを開発した[5]．本ツール開発にあたり，ベテラン技者にヒアリングを行い，仕様書内で設計要素間の矛盾を指摘するためのノウハウを形式知化した．本ツールの位置づけや実案件への適用評価については，既に資料[5]にて報告した．また，ツールはシナリオの一貫性検証支援ツールとして公開し[6]，本ツールは

†1 工学院大学
Kogakuin University
†2 伊藤忠テクノソリューションズ株式会社
ITOCHU Techno-Solutions Corporation

技術者の効率的な仕様検証を支援する点において一定の効果があることを明らかにした．

著者らは，新たに，同一組織が作成した複数の要求仕様書に本ツールを適用し，要求仕様書の検証を試みた．本ツールを用いて一貫性検証の観点で要求仕様書を分析した結果，ドキュメント品質の改善のノウハウが得られた．

以下，本稿は次のように構成する．2 章では，自然言語で記述された要求仕様の検証技術の現状を概観する．3 章では，2 章での技術動向を踏まえて，自然言語で記述された要求仕様書の検証技術に関する本研究のアプローチを示す．4 章では，実際の仕様書に対して，当該ツールを適用した結果を示す．5 章では，ツールの適用結果から有効性や妥当性について考察し，ドキュメント品質の改善に関して得られた教訓を整理する．6 章で本稿をまとめる．

2. 要求仕様書の検証技術

エディタやワードプロセッサを用いれば，特定の用語を検索したり，置換したりすることは容易である．各組織は，仕様書において使用することが不適切な用語のリストや，対象ドメインで統一して利用すべき用語のリストを作成しておき，組織の標準として共有している．これらのリストを用いて，エディタやワードプロセッサの検索や置換機能を用いて，用語の統一を図ることが一般的に行われている．また，Just Right!6 Pro [10]は，文書校正に特化したツールであり，文書全体に対して漏れなくかつ効率的に，NG ワードの検出や異音同義語リストに基づく用語統一を行うことが可能である．こうしたツールの活用には，NG ワードのリスト，異音同義語のリストが存在することが前提となり，これらリストの洗い出しが重要であるが，それらは利用者側が準備することが前提となっている．

樋口は自然言語で記述された文書の内容理解のためのツール KHCoder を開発し[3][11]，内容理解に加えて，用語の利用箇所の検索や用語間の関係性抽出など，文書の記述内容の可視化や，特定用語の周辺文章の抽出のための各種機能提供している．不整合のある用語の検出や，問題があると考えられる用語の周辺の文章を検出するには，それら用語自体を特定することが必要である．用語の特定ができれば，当該用語が利用されている周辺用語や，関連用語の抽出が行えるため，仕様書の改善に活用することができ，これらのツールの活用は有効である．

自然言語で記述された要求仕様を記述するために，Pohl らは機能要求仕様を記述するためのテンプレートを提案した[20]．テンプレートでは，英語で記述された一つの機能要求文は，条件，主語（システム名），助動詞，動詞，目的語，目的語の詳細により構成するとしている．日本語で記述された要求に対して，Pohl らの提案をそのまま適用することは妥当ではないが，テンプレートが示す構成要素にそって要求を記述することは，要求仕様の品質向上に貢献すると考えられる．Pohl らのテンプレートにおいて，目的語には，データや画面が相当するが，それら用語自体に表記ゆれがあれば，要求文自体の曖昧性は解消されない．よって，要求文を構成する設計要素に焦点をあてた，使用する用語の一貫性の検証を支援することが必要である．

Lucassen らはユーザストーリーの品質を安定させるための品質モデルと支援ツール AQUSA の開発および評価を報告している[18]．Lucassen らはユーザストーリーの品質評価のフレームワークとして，Syntactic，Semantic，Pragmatic の視点に分類される 14 個のメトリクスを定義した．AQUSA は，Atomic, Minimal, Explicit dependencies, Uniform, Unique の 5 つのメトリクスによる検証を自動化しているが，Semantics の観点の Unambiguous の検証はスコープ外である．自然言語で記述された要求仕様の品質向上には，Semantics の観点からの Unambiguous の検証，すなわち，表記ゆれなどの用語の一貫性検証が不可欠と考えられる．

日本語の自然言語で記述された仕様書の検証やレビューについても研究が行われている[13][17]．河野らは，ドキュメント内において，曖昧さや不備につながりやすいキーワードの洗い出しと，そのようなキーワードのチェックツールを考案した．キーワードとしては，「〜場合」などの条件を示す用語，「あれ」，「これ」などの指示代名詞が含まれる[13]．

久野らは，同じ語句でも，使われ方により曖昧の度合いが異なるとして，曖昧性の高い語句を選択的に検出することを目的に，語句の曖昧性を判定するツールを提案した[17]．久野らが提案するツールでは，曖昧性のレベルを，曖昧語，準曖昧語，非曖昧語に分けて検出する．提案されたツールを実際の仕様書にて評価したところ，複数の解釈が可能な曖昧語を高いレベルで網羅的に検出でき，仕様書の修正に対して有効であることを確認している．

先行研究[13][17]が着目した「曖昧さにつながりやすいキーワード」は重要なものの，要求仕様の品質向上には，対象ドキュメント内で定義された設計要素の曖昧さの解消がさらに重要である．例えば，ある設計要素を示す用語が，曖昧さにつながりやすいキーワードには非該当でも，その設計要素が，別の語句で定義されることや，明確な定義なく機能要求に使用される状況があれば，その機能要求の内容は，開発者や分析者を含む複数の読み手により解釈が異なる内容となるリスクが高い．自然言語で記述された要求仕様に対しての検証支援のためには，特定のキーワードに加えて，要求仕様の本質的な定義内容に踏み込んで，設計要素の曖昧さや不整合を指摘する検証を行うことが必要である．

木村らは，要求仕様の一貫性をツールにより検証することで要求仕様の安定化を図ることに取り組んだ事例を報告した[12]．これは，設計要素の一つであるアクターにフォーカスし，ベテラン技術者による，アクターの表記ゆれを

防止するための検証ノウハウを「用語不一致検証ルール」と「定義漏れ検証ルール」の機能を備えたツールとして実現した点が特徴である．事例[12]では，要求仕様書内のアクター定義とシナリオ記述を対象に，アクター定義表からの定義漏れを指摘するアクターの定義漏れ検証と，ユーザとユーザ（担当者）などの表記ゆれを指摘する用語不一致検証を自動化し，検証レポートを生成した．しかし，事例[12]における要求仕様の検証のノウハウは，アクター定義の一貫性にフォーカスした，特定企業の分析結果に基づいて定義された内容であり，様々な組織での活用を想定した一般化には至っていない．一貫性検証の対象となる，要求仕様を構成する設計要素の種類を拡張することや，様々な組織での活用を想定した柔軟性のあるツールのアーキテクチャの提供が必要と考えられる．

3. 要求仕様の一貫性検証支援ツールとドキュメント品質の分析手法

著者らは，2 章の内容を踏まえ，自然言語で記述された要求仕様書に対して，要求仕様の品質特性である「一貫性」に着目した，要求仕様の一貫性検証支援ツールを開発した[5][6]．以降，本ツールの概要を示し，本ツールを用いた要求仕様のドキュメント品質の分析手法を提案する．

3.1 要求仕様の一貫性検証知識の形式知化

事例[12]では，特定企業かつ，機能一覧表などの自然言語で記述された仕様説明内における「アクター」を対象として，「用語不一致検証ルール」と「定義漏れ検証ルール」に基づく二種類の検証の知識の形式知化と支援ツールによる自動検証にとどまっていた．本ツールでは，検証の対象を要求仕様書全体に一般化した上で，検証の観点を，アクターに加えてデータ，画面，振る舞いに拡張する．そして，複数企業の有識者インタビューを実施し，設計要素の一貫性に関する検証知識を抽出し，それらを共通部分と可変部分に仕分け，検証ルールおよび辞書として形式知化した．

図 1 にツールに組み込んだ検証知識の構造を示す．これは，要求仕様の品質特性である「一貫性」に着目し，アクター，データ，画面，振る舞いの設計要素の観点から，ベテラン技術者が経験的に得た検証知識を検証ルールと辞書として定義したものである．このツールを活用することで，初級者にありがちな，用語の表記ゆれや曖昧用語の使用箇所を自動的に検出することができる．各ルールの説明を表 1 に示す．

3.2 要求仕様の一貫性検証の支援ツール

3.1 で定義した検証ルールおよび辞書を組込んだ，要求仕様の一貫性検証を支援するツールを開発した．本ツールは，要求仕様書内で言及されている，アクター，データ，画面，振る舞いの記述が，要求仕様書中の記述と整合していることを検証する．検証の種類は，用語定義完全一致，NG ワード検証，NG ワードを妥当な用語に置換するシナリオ自動生成の機能を追加した．開発したツールの考え方と構成を図 2 に，各機能の説明を表 2 に示す．図 2 に示すように，本ツールは仕様書と設定ファイルを入力として，検証した結果を検証レポートに出力する．また，NG ワードを望ましい用語に置き換えた仕様書も自動生成する．

検証ルールや辞書は，製品ソフトウェアが対象とするドメインや利用する組織に依存する内容が含まれると考えられるため，検証ルールならびに辞書の共通部分を共有するとともに，各組織が容易にそれらの維持管理と拡張を実施可能にするため，検証のエンジン，検証機能，検証ルール，辞書を独立させたアーキテクチャとした．さらに，各社が独自に検証エンジンの拡張を可能とするために，形態素解析エンジンは OSS を用いた．

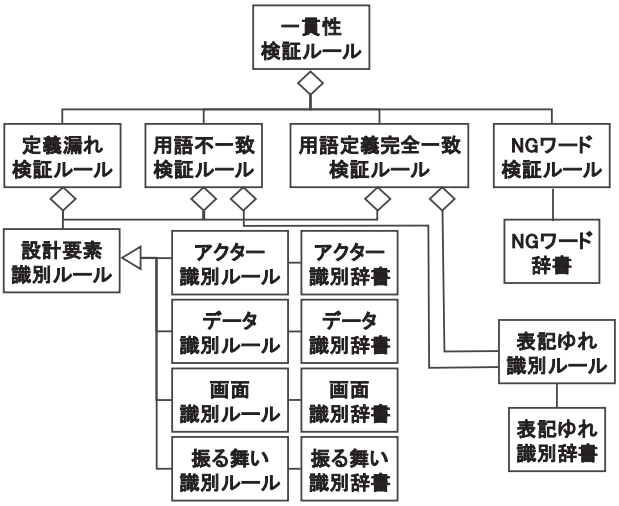

図 1　検証ルールと辞書の構造
Figure 1　Structure of Verification Rules and Dictionaries

表 1　検証ルール
Table 1 Consistency Verification Rules

No.	ルール	ルール説明
1	定義漏れ検証	要求仕様書中に定義されていないにもかかわらず，本文中で利用されている設計要素を検証する．
2	用語不一致検証	要求仕様書では同じ意味の語句に対して異なる文字表記がなされている設計要素を検証する．
3	用語完全一致検証	要求仕様書にて利用されている設計要素は，要求仕様書中に定義されており，かつ，定義された設計要素は，要求仕様書内で1回以上利用されている設計要素を検証する．
4	NGワード検証	要求仕様書に出現するNGワードを検証する．NGワードはNGワード辞書にて定義する．
5	アクター識別	要求仕様書からアクター用語を識別する．アクター用語として特徴づける単語はアクター識別辞書にて定義する．
6	データ識別	要求仕様書からデータ用語を識別する．データ用語として特徴づける単語はデータ識別辞書にて定義する．
7	画面識別	要求仕様書から画面用語を識別する．画面用語として特徴づける単語は画面識別辞書にて定義する．
8	振る舞い識別	要求仕様書から振る舞い用語を識別する．振る舞い用語として特徴づける単語は振る舞い識別辞書にて定義する．
9	表記ゆれ識別	要求仕様書の複数の用語から，表記ゆれ関係を識別する．表記ゆれ関係にある用語は，表記ゆれ識別辞書にて定義する．

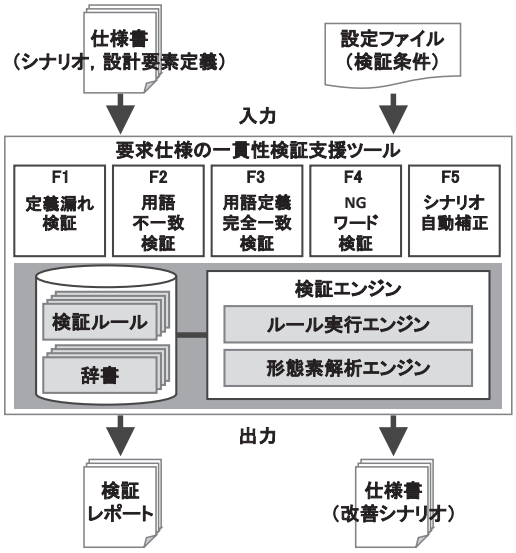

図 2　要求仕様の一貫性検証支援ツールの概要
Figure 2 Requirements Consistency Verification Support Tool

表 2　機能概要
Table 2　Functional Specification

ID	機能名	処理概要
F1	定義漏れ検証	検証対象の仕様（シナリオ）および，アクター，データ，画面，振る舞い等の定義表を入力として，シナリオに出現する，アクター，データ，画面，振る舞い等が，それぞれ，対応する定義表に定義されていることを確認し，未定義であれば指摘する．
F2	用語不一致検証	検証対象の仕様（シナリオ）および，アクター，データ，画面，振る舞い等の定義表を入力として，シナリオに出現する，アクター，データ，画面，振る舞い等が，それぞれ，対応する定義表と異なる文字表記がなされている場合に，その用語を指摘する．対応する定義表に定義された用語に対する，省略化，修飾化，表記ゆれ辞書一致のいずれかのパターンに該当する場合に，用語不一致とみなす．
F3	用語定義完全一致検証	検証対象の仕様（シナリオ）および，アクター，データ，画面，振る舞い等の定義表を入力として，シナリオに出現する，アクター，データ，画面，振る舞い等が，それぞれ，対応する定義表に定義されていること，かつ，アクター，データ，画面，振る舞い等の定義表に定義された各要素が，シナリオ中に1回以上出現していることを確認し，未定義または出現しない場合に指摘する．
F4	NGワード検証	検証対象の仕様（シナリオ）および，NGワード定義表を入力として，NGワードの定義表に定義された用語がシナリオ中に出現していれば，それを指摘する．
F5	シナリオ自動補正	検証対象の仕様（シナリオ）および，NGワード定義表を入力として，NGワードの定義表に定義された用語がシナリオ中に出現していれば，おなじくNGワード定義表に定義された置換候補用語でシナリオを置換し，改善シナリオを生成する．

3.3 要求仕様のドキュメント品質分析手法

要求仕様の一貫性検証支援ツールを用いて，要求仕様のドキュメント品質を分析する手法を次のように定義する．

① アクター，画面，データ，振る舞いの設計要素の定義が，要求仕様書中にあるかどうかを確認し，存在する場合には，その定義内容を確認する．これらを，設計要素定義表にまとめ，要求仕様の一貫性検証ツールへの入力とする．

② 要求仕様書中の機能一覧やシナリオなどの検証対象を抽出する．これらも要求仕様の一貫性検証ツールへの入力とする．本ツールにより，アクター，画面，データ，振る舞いなどの設計要素を特定し，上記の設計要素定義表と突合し，定義漏れや用語不一致の箇所を指摘し，結果を検証レポートに出力する．

③ 検証レポートを用いて，定義漏れや用語不一致となった用語に対して，機能一覧やシナリオなどの検証対象の本文を確認し，設計要素定義表を追加･拡張するか，本文の用語を置換するかなど改善を行う．

図 3 に，本ツールを用いた要求仕様の品質分析の事例を示す．図 3 の上部は，本ツールの F1 の定義漏れ検証を実施している過程を示す．これは，仕様書内にユーザやレジ係というアクター用語が未定義のまま使用されていると，定義漏れ検証によって，それらを指摘できることを示す．また，図 3 の下部では，F2 の用語不一致検証機能を活用している例を示す．これは，ユーザ（管理者）やユーザ（担当者）が定義されているにもかかわらず，本文中では「ユーザ」と一般化して使用している場合に，「ユーザ」は，「ユーザ（管理者）」，「ユーザ（担当者）」に対する表記ゆれであることを指摘している．

本ツールでは，検証結果を検証レポートで出力する．検証レポートを用いて，要求分析者は，要求仕様書本文の改善か，設計要素定義の拡張や追加を検討して，要求仕様書のドキュメント品質を高めていくことが可能である．

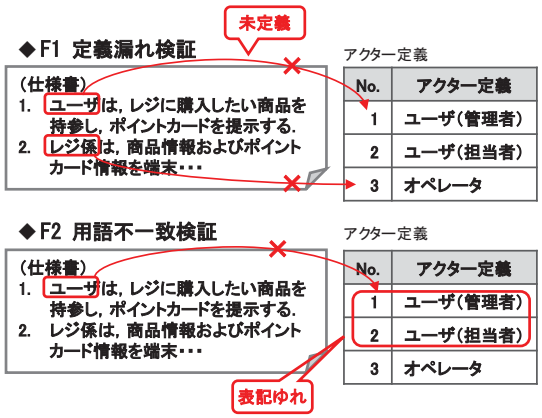

図 3　要求仕様の一貫性検証の例
Figure 3 Examples of Requirements Verification

3.4 適用実績

実システム開発で用いられた以下 2 件の要求仕様書に対して，要求仕様の一貫性検証支援ツールを用いて検証を実施した．これらは日本語で記述されたドキュメントである．それぞれの文字数は（ a）17,635，（b） 66,048 である．

（a）某ウェブサイトリニューアル事業要求仕様書
（b）某制度システム構築・運用等業務調達仕様書

上記仕様書へのツールの適用評価分析によれば，平均して，定義漏れ，用語不一致，用語完全一致は，再現率 90%以上，適合率 82%以上，F 値 0.84～1.00 の結果を得た．

NG ワード検証は，NG ワード辞書で定義した用語は 100%特定でき，あらかじめ用意した NG ワードに対する置換用語を用いてシナリオ自動生成が行えるなど，一定の効果を得た．なお，評価結果の詳細については，報告書[5]にて示している．

4. 実仕様書へのツール適用

3.4 で示したように，要求仕様の一貫性検証支援ツールは，既に実案件に対して一定の効果を示した．本稿では，さらに，別の実システムの仕様書に対しても本ツールを適用し，仕様書の要求品質の向上にどのように適用可能なのかを分析する．

4.1 評価の方法

対象とする仕様書は，厚生労働省による以下 3 案件の調達仕様書である．いずれの仕様書も一般に公開されている仕様書である．以降①~③をそれぞれ，仕様書①，仕様書②，仕様書③，と略すことにする．

①年金業務システム（個人番号管理サブシステム等（2 次開発情報連携分））に係る設計・開発等業務及びアプリケーションソフトウェア保守業務[14]

②労働保険適用徴収システムに係るシステム運用業務一式調達仕様書(案)[15]

③労働基準行政情報システムに係るアプリケーションプログラム改修等業務一式（平成 28 年度）調達仕様書（案）[16]

具体的には次の手順で調査した．

（1）準備

本ツールへの入力ファイルを作成する．公開されている仕様書は複数の種類の pdf, word, excel などファイルで構成されているので，これらからツールへの入力となるテキストファイルに変換した．

（2）検証対象の絞り込み

仕様書①~③は大規模システムであり仕様書全体の量も数百ページ以上になるため，検証の範囲を絞り込んだ．仕様書①は，情報登録管理が主体のシステムである．著者らは，アクターとエンティティの表記ゆれに絞って検証を行った．仕様書②および仕様書③については，アクターを中心に分析した．さらに，仕様書①~③を横断する特定アクターの表記ゆれについて分析することとした．

（3）要求仕様の一貫性検証支援ツールの実行

（1）整理したファイルを対象として，本ツールを実行しそれぞれの検証レポート出力した．本ツールへの入出力のイメージを図 4 に示す．

（4）検証レポートを用いた分析と指摘事項の評価

検証レポートで，定義漏れ，用語不一致となったアクター用語，エンティティ用語に対して，本文での使われ方を分析し，指摘結果が妥当な内容かどうかを検討した．また，上記分析に基づき，現状の仕様書の品質リスクと思われる事柄を洗い出した．

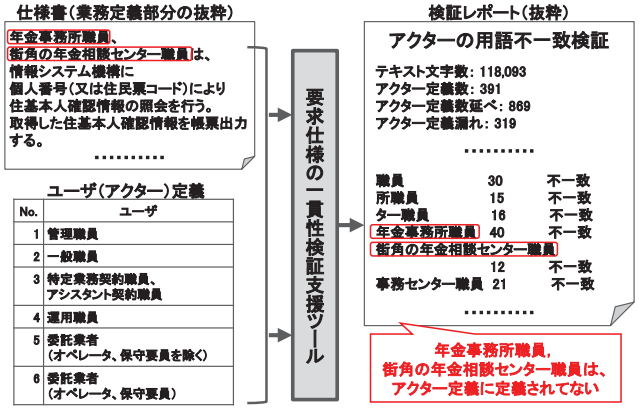

図 4 ツールの適用事例

Figure 4 Example of Application of the Tool

4.2 検証レポートによる指摘事項の分析結果

仕様書①~③の「アクターの用語不一致検証」に関する結果は以下の通りである．検証レポートの全体，用語不一致となった用語全体の一覧リストについては，紙面の都合により割愛する．

- 仕様書①：アクター用語 319 件（用語不一致：319 件）
- 仕様書②：アクター用語 354 件（用語不一致：353 件）
- 仕様書③：アクター用語 178 件（用語不一致：178 件）

以下，仕様書別に，検証レポートによる指摘事項に基づき，仕様書本文での使われ方の分析結果を示す．

4.2.1 仕様書①への指摘事項

- 用語不一致用語として，アクター用語の「被保険者・受給権者」,「職員」,「被保険者」が指摘された．また，データ用語の用語不一致用語として,「組織情報」,「組織」が指摘された．なお，仕様書のエンティティ定義では,「組織」はユーザではなく，エンティティとして定義されていることがわかった．
- 一般的にはアクター用語とみなされる用語（組織，被保険者・受給権者，職員）が，エンティティ用語として定義されていた．さらにこれらが，本文中で「組織情報」,「運用職員」のように，何等かの文言で修飾した用語として利用されていた．
- 用語不一致となる表現としては，アクター（エンティティ）名に「・」を用いることで曖昧な表現になる用語があった．例えば,「被保険者・受給者」と「被保険者」は，いずれもアクター用語として利用されているものの，明確な定義がないため，これらが同一なのかどうか曖昧な状態となっていた．
- 用語不一致の別の例としては,「街角の年金相談センター職員」,「年金事務所職員」,「事務センター職員」が仕様書内に複数回登場しているが，情報システムのユーザ定義には,「管理職員」,「一般職員」,「特定業務

契約職員」,「アシスタント契約職員」,「運用職員」,「委託業者」が定義されており，用語不一致の状態であった．その他,「職員」という用語が仕様書内で使用されており，一般化された用語が何を指すのか不明確になっていた．

- 「用語定義」で定義されている用語は，比較的一般的に使われている名称のものに対して，本案件での特別な意味を与えて利用している．また,「用語定義」内の用語名に「・・・等」という用語が用いられていることがわかった．

4.2.2 仕様書②への指摘事項

- 検証レポートによれば,「システム運用・保守統括者」は，用語不一致用語として，本文中で延べ 73 件が利用されているとの指摘があった.「システム運用・保守統括者」の出現箇所として，仕様書②の「用語の定義」の解説部分に複数箇所（少なくとも 2 箇所）あった．
- 検証レポートによれば,「運用オペレータ」は，71 件の未定義箇所があると指摘された.「運用オペレータ」の出現箇所として，仕様書②中の「用語の定義」の解説部分に複数箇所（少なくとも 2 箇所）あった．
- 仕様書②におけるアクター定義に相当する「適用徴収業務の実施体制概要(表 2-2)」と「利用者数（表 2-3)」には,「システム運用・保守統括者」,「運用オペレータ」の定義はなかった．したがって,「システム運用・保守統括者」および「運用オペレータ」は，本文中で頻繁に言及されているものの，当該システムのアクターなのかどうか不明確な状況になっていることがわかった.
- システム運用・保守統括者」という用語にも,「・」が用いられており，その結果,「システム運用」と「保守統括」なのか,「システム運用・保守」と「統括」なのか，業務の区切りが不明確であった．システム運用と保守を担当するアクターの業務範囲と，システム化の対象は，開発範囲に影響する内容であるため，アクター定義表において,「システム運用・保守統括者」というアクターの責務を明確に仕様化することが重要であると考えられる．

4.2.3 仕様書③への指摘事項

- 仕様書①でも指摘した「職員」用語は，検証レポートによれば 6 件の未定義箇所があると指摘された．
- 「職員」の出現箇所の一つに，仕様書中の「用語表」と呼ばれる用語の定義箇所のうち,「利用者」用語の説明部分があった．ここでは，利用者の定義として,「職員（本省、芝、上石神井、都道府県労働局、労働基準監督署等)」とあり，丸括弧の記号を用いて，補足説明がされている．
- 上記に関連して,仕様書③中の「拠点数一覧表(表 2-3)」には，拠点名と拠点数が定義されている．また，同「情報システム利用者（表 4-1)」では,「本省職員」,「地方局職員」,「監督署職員」,「事業受託者等」が「利用者区分」であるとして,「拠点数一覧表（表 2-3)」に示す具体的拠点名と対応づけて，各利用者区分がどのような職員に相当するのかが定義されていた．
- 「用語表」による「利用者」を理解するには，上記の 2 つの表（利用者区分の定義，拠点の定義）を相互に参照する必要があることがわかった．すなわち，仕様書③においては，3 つの表を相互参照しないと，本システムのアクターが理解できない構造になっていることがわかった．
- 拠点の定義において，地名に起因する拠点（例：芝庁舎，上石神井庁舎）と，組織の総称（労働基準監督署）に相当する拠点が混在していた．現状の仕様書では，クラスとインスタンスが混在して定義されている状態のため，拠点の理解が，第三者には困難であること，拠点（Where）と，利用者（Who）は異なる観点であるので，それぞれが独立して定義されていれば，上述したような複数の情報を相互参照する必要はなく，当該システムの重要なアクターである「職員」の定義と，システムの利用範囲の理解が容易になると考えられる．加えて，地名は変化する可能性も高く,「組織」や「職員」の定義には，組織区分と所在地を区別して定義することが，仕様の理解の加速に有用と考えられる．

4.2.4 仕様書①~③を横断した指摘事項

3 つの仕様書は，いずれも厚生労働省から提示された調達仕様書であるものの，共通して出現するアクター用語の定義の方法，本文での使用状況が，3 つの仕様書間で不統一であることがわかった．今回特に掘り下げたのは「職員」用語であるが，この他にも,「本部」,「外部」,「組織」,「運用者」などの用語不一致用語が存在している．

3 つの仕様書間で不統一である対象としては，用語定義，ユーザ定義の呼び方が異なっている．たとえは，用語定義については,「用語の定義」や「用語表」と呼ばれている．また,「ユーザ定義」の情報は,「ユーザ定義表」,「実施体制」,「利用者数」,「拠点数」,「情報システムの利用者」などとして異なる内容で異なる表現方法で示されている．

さらに，仕様書③でも示した通り，3 つの仕様書に共通して，用語定義，ユーザ定義が相互参照されている状況にある．つまり,「用語定義」の「説明」部分の方に，利用者に相当する用語が使用されており，その利用者自体の定義が，さらに別の記載部分を参照する状況にあり，複数の参照関係による定義方法がとられている．

今回はアクター用語を中心に調査したが，アクターとは，システムの利用者，組織，システムなどのシステム化の範囲に関わる重要な設計要素である．アクターの種類，名称，役割の定義は,「ユーザ定義」または「アクター定義」などと統一した名称の仕様にて，仕様書の冒頭に示すことが，仕様書の理解の円滑にすると考えられる．現状では,「定義

の定義」が複数存在する状態であり，加えて，仕様書が異なると，同一組織が作成した仕様書であるにも関わらず，説明の仕方が一貫していないため，属人性が高く，第三者による仕様の理解は困難な状況にあると考えられる．

5. 考察

5.1 要求仕様の検証支援ツールによる一貫性検証の効果

今回分析した 3 件の要求仕様書は，サイズ（ページ数）や分冊の種類が多く大規模なため，仕様書の全体を要求仕様の一貫性検証支援ツールに入力して検証をしていない．しかし，アクターやエンティティに限定した仕様書の検証の実行のみでも，システムの開発範囲に関する理解の齟齬を誘発するリスクのあるアクターの用語不一致や定義漏れを複数件洗い出すことができた．このことから，本ツールによる要求仕様の一貫性検証支援は有効と考えらえる．

第 2 章で示したとおり，エディタ，ワードプロセッサ，文書チェックツールを用いれば，特定の用語の文書全体の存在有無のチェックや，漏れなく全置換を行うことが，使いやすいユーザインタフェースにより実現できる状況にある．しかし，これらのツールの活用には，4 章で対象とした 3 つの仕様書から，「職員」，「組織」，「運用者」のような重要なアクター用語を特定できていることが前提である．このような用語の抽出をした上であれば，同エディタやツールの提供する全文検索や全文置換の機能は有用である．要求仕様の一貫性検証支援ツールは，仕様書の設計要素に特化した，アクター，データ，画面，振る舞いの用語を抽出するための辞書と識別ルールを備えており，これらによって，検索や置換の対象になる重要用語を洗い出せる点が特に有効である．

本ツールへの入力は，現状ではテキストファイルであることから，テキストファイルに変換する際に文章の途中で改行コードが混在し，本来の意図とは異なる部分で用語が改行されることがある．このことが要因となり，検証レポートにて抽出した設計要素に該当する用語には，誤指摘が含まれていた．今後，本ツールへの入力ファイルの種類の拡張や，不適切な用語抽出を回避する工夫が必要である．

実際に要求仕様書の検証を行う場合には，本ツールにて，重要な設計要素の用語の洗い出しを行い，件数が上位のものや，対象案件で重要と考えられる用語の優先度をあげて，用語の抽出状況を確認した上で，エディタ，ワードプロセッサ，文書チェックツールと併用して，全文検索や全文置換などを行うことで，さらに有効かつ効率的な要求仕様書の改善が行えると考えられる．

5.2 要求仕様書の改善に対して得られた教訓

要求仕様の一貫性検証支援ツールを用いて，要求品質のドキュメント品質を分析することを通して，要求仕様書の改善に関して得られた教訓を以下にまとめる．

- 大規模システムや複雑な業務を支援するシステムではとくに，用語定義，ユーザ定義を詳細に定義することが重要である．しかし，利用者定義，利用者数などのアクター定義を構成する要素が複数かつ相互参照の形式で定義されていると，仕様書全体の理解の妨げになる．少なくとも相互参照は解決すべきである．
- 複数の大規模システムを所有する組織では，複数の仕様書間においても，用語定義やユーザ定義などの定義表の名称，説明方法を統一すべきと考えられる．
- 一般的な用語であるにもかかわらず，対象とする仕様書では特定の意味を持つ用語「職員」，「組織」，「運用オペレータ」については，仕様書横断でも一貫性を保つべきである．これは，ステークホルダ定義表などを組織の資産として蓄積することが必要と考えられる
- 曖昧と思われる用語が文中でどのように使用されているかを確認する検索や，用語の置換を網羅的に行うために，文書チェックツールを用いることは効率的である．
- 曖昧と思われる用語を特定するには，アクター，データ，画面，振る舞いの辞書に基づく検索をする必要があるため，本ツールを通して，用語抽出をすることが妥当である．（併用しての利用がさらに妥当である）
- 仕様書が完成した後で，仕様書を改善するには，手戻りコストがかかる．新規開発に備えて，実際の案件を分析した結果から辞書を作ることや，アクター定義やエンティティ定義のひな形を作成することが重要と考えられる．
- 今回は既に開発が完了している案件の調達仕様書に対する一貫性検証の指摘事項と改善に関する提案をまとめた．仕様書の改善を実施することは困難でも，現状のシステムは，今後長期に渡り維持管理することが必要なものも多く含まれると考えられる．開発に直接関わっていなかった開発者や運用者が，今後，維持管理を行う場合にも，仕様書の理解は重要であり，そのために，現状の仕様書のアクターの考え方，仕様書上での表現の特性などを，仕様書と一緒に補足資料として用意し共有することで，第三者の理解の促進に効果的と考えられる．

6. まとめ

本稿では，同一組織が作成した複数の要求仕様書に，開発した要求仕様の一貫性支援ツールを適用し，仕様書の一貫性をチェックするとともに，本ツールが指摘した内容の有効性を確認した．さらに，本ツールを用いて一貫性検証の観点で要求仕様書を分析し，ドキュメント品質の改善に関するノウハウを整理した．具体的には，システムの開発範囲に大きく影響するアクターの仕様化方法について，理解の妨げのリスクを軽減するための，アクター定義の仕様化の方法や，アクター用語の定義方法を提案した．

今後も引き続き，同ツールによる要求仕様の一貫性検証適用を継続し，仕様書の高品質化に関するノウハウを明らかにし，要求定義の高品質化に貢献していく．

謝辞 本ツール開発は，独立行政法人情報処理推進機構技術本部ソフトウェア高信頼化センター（SEC: Software Reliability Enhancement Center）が実施した「2015 年度ソフトウェア工学分野の先導的研究支援事業」の支援を受けたものである．また，本研究開発の一部は，2016 年度科研費「要求定義の高品質化のためのシナリオの一貫性検証・シナリオ生成手法」JSPS 科研費 JP16K00105 の助成を受けて実施した．

参考文献

[1] Cockburn, A.. Writing Effective Use Case, Addison Wesley, 2000, ウルシステムズ株式会社 (監訳)，山岸耕二，矢崎博英，水谷雅宏，篠原明子(訳)，ユースケース実践ガイド―効果的なユースケースの書き方 (OOP Foundations)，翔泳社，2001.
[2] Cohn, M.. User Stories Applied: for Agile Software Development. Redwood City, CA, USA: Addison Wesley, 2004.
[3] 樋口耕一．社会調査のための計量テキスト分析―内容分析の継承と発展を目指して，ナカニシヤ出版，2014.
[4] 平鍋健児，野中郁次郎．アジャイル開発とスクラム～顧客・技術・経営をつなぐ協調的ソフトウェア開発マネジメント，翔泳社，2013.
[5] 位野木万里．要求定義の高品質化のための要求仕様の整合性の検証知識の形式知化と一貫性検証支援ツールの開発，独立行政法人情報処理推進機構，ソフトウェア工学分野の先導的研究支援事業, http://www.ipa.go.jp/sec/rise/#01-9, (参照 2016-06-01).
[6] 位野木万里．シナリオの一貫性検証支援ツールの紹介, http://www.ns.kogakuin.ac.jp/~wwa1076, (参照 2016-06-01).
[7] ISO/IEC/IEEE 29148:2011, Systems and software engineering -- Life cycle processes -- Requirements engineering, 2011.
[8] 一般社団法人 情報サービス産業協会 REBOK 企画 WG. 要求工学知識体系，近代科学社，2011.
[9] 一般社団法人 情報サービス産業協会 REBOK 企画 WG. 要求工学実践ガイド REBOK シリーズ 2，近代科学社，2014.
[10] 株式会社ジャストシステム．文章校正支援ツール Just Right!6 Pro, https://www.justsystems.com/jp/products/justright/?w=plst, (参照 2017-02-03).
[11] KHCoder, http://khc.sourceforge.net/, (参照 2017-01-03).
[12] 木村隼人，北川貴之，位野木万里. 要求仕様書の品質向上に向けた活動報告～ 一貫性検証の形式知化および自動化～, 情報サービス産業協会 技術シンポジウム SPES2014 SPES 事例研究（経験報告）2014 年 10 要求工学 S4a, 2014.
[13] 河野哲也，猪塚修，藤森麻紀子，本間周二，茂中義典. キーワードベースドレビュー，ードキュメントのあいまいさや不備に着目したレビュー手法ー，ソフトウェアテストシンポジウム JaSST 2010，http://jasst.jp/archives/jasst10e/pdf/C2-3.pdf, 2010.
[14] 厚生労働省．年金業務システム（個人番号管理サブシステム等（2 次開発情報連携分））に係る設計・開発等業務及びアプリケーションソフトウェア保守業務, http://www.mhlw.go.jp/sinsei/chotatu/chotatu/shiyousho-an/160428-1.html, (参照 2016-10-12).
[15] 厚生労働省．労働保険適用徴収システムに係るシステム運用業務一式調達仕様書(案), http://www.mhlw.go.jp/sinsei/chotatu/chotatu/shiyousho-an/160413-1.html, (参照 2016-10-12).
[16] 厚生労働省．労働基準行政情報システムに係るアプリケーションプログラム改修等業務一式（平成 28 年度）調達仕様書（案）, http://www.mhlw.go.jp/sinsei/chotatu/chotatu/shiyousho-an/151204-1.html, (参照 2016-10-12).
[17] 久野綾子，平尾英司，五藤智久．仕様書の曖昧性を検出するツールの試作と評価，一般社団法人電子情報通信学会，電子情報通信学会総合大会講演論文集 2012 年_情報・システム(1), 27, 2012-03-06，2012.
[18] Lucassen, G., Dalpiaz, F., Werf, J. and Brinkkemper, S.. Forging High-Quality User Stories: Towards a Discipline for Agile Requirements, Proc. of IEEE 23rd International Requirements Engineering Conference, pp. 126-135, 2015.
[19] MeCab, Yet Another Part-of-Speech and Morphological Analyzer, http://taku910.github.io/mecab/#download, (参照 2017-2-3).
[20] Pohl, K. and Rupp, C.. Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB compliant, 2nd Edition, Rocky Nook, 2015.
[21] Rasmusson, J.，西村直人(監訳)，角谷信太郎(監訳)，近藤修平(訳)，角掛拓未 (訳), アジャイルザムライ―達人開発者への道, オーム社，2011.
[22] Ruby, https://www.ruby-lang.org/ja/about/, (参照 2017-2-3).
[23] Wang, X., Zhao, L, Wang, Y. and Sun, J.. The Role of Requirements Engineering Practices in Agile Development: An Empirical Study, Proc. of the Asia Pacific Requirements Engineering Symposium, ser.CCIS. Springer, vol.432, pp. 195–209, 2014.