情報処理学会論文誌 Vol.57 No.2 694–706 (Feb. 2016)

ReqQA：ソフトウェア要求仕様書品質解析ツールの提案と評価

青山 幹雄[1,a)] 中根 拓也[2]

受付日 2015年5月18日, 採録日 2015年11月5日

概要：ソフトウェア要求仕様書（SRS）の品質はソフトウェア開発とその成果物の品質を左右する．しかし，SRS の品質に関する研究は限定的である．本稿では，SRS のインスペクション設計方法論の成果に基づき，SRS の品質解析方法とその自動化ツール ReqQA（Requirements Quality Analyzer）を提案する．多様な SRS を統一的に解析するために，RDF で表現された中間言語 SRS-CIL（Common Intermediate Language）を提案する．Word 文書の SRS を SRS-CIL に変換し，パースペクティブに基づくプラグマティック品質特性を解析するツールのアーキテクチャを提案し，プロトタイプを REST サービスとして実装した．プロトタイプを実際の RFP（提案依頼書）に適用し，提案方法とツールアーキテクチャの妥当性，有効性を示す．

キーワード：要求工学，ソフトウェア要求仕様書，要求品質，インスペクション，解析ツール

ReqQA: A Software Requirements Specifications Quality Analyzer and Its Application

Mikio Aoyama[1,a)] Takuya Nakane[2]

Received: May 18, 2015, Accepted: November 5, 2015

Abstract: The quality of SRS (Software Requirements Specification) influences the quality of software development and its product. However, few works have been done on the quality of SRS. This article proposes a method of quality analysis of SRS and its automated tool ReqQA (Requirements Quality Analyzer) based on the work on SRS inspection design methodology. To accommodate diverse forms of SRS, the authors propose SRS-CIL (Common Intermediate Language) represented by RDF. The authors propose the architecture of ReqQA consisting of the converter from SRS in Word, and plug-able tool sets, analyzing the SRS-CIL based on the pragmatic quality model from the reader's perspective. The prototype of ReqQA is implemented and applied to a real RFP, and proves the concept and demonstrates the effectiveness of the ReqQA.

Keywords: Requirements Engineering, Software Requirements Specifications, Requirements Quality, inspection, analysis tool

1. はじめに

ソフトウェア要求仕様書（SRS: Software Requirements Specifications）の品質がソフトウェア開発の成否とソフトウェアシステムの品質を左右することが指摘されている [5]．そのため，SRS の品質確保がきわめて重要となっている．

一般に，作成された SRS の品質を確認するためにレビューやインスペクションが広く実践されている [8], [9]．しかし，これらは人手によっていることから，コストを要し，かつ，SRS の大部分が自然言語で記述されていることとあいまって誤りやすく，個人のスキルにも依存する．さらに，大規模ソフトウェア開発では SRS の規模が数百ペー

[1] 南山大学理工学部ソフトウェア工学科
Department of Software Engineering, Nanzan University, Nagoya, Aichi 466–8673, Japan
[2] 南山大学大学院理工学研究科ソフトウェア工学専攻
Graduate School of Sciences and Engineering, Nanzan University, Nagoya, Aichi 466–8673, Japan
[a)] mikio.aoyama@nifty.com

ジにもなり，作業に要するコストに加え，その一貫性などの問題もある．

このような現状に対して，SRS の品質とその解析方法に関する研究は限られている [17]．たとえば，自然言語処理を用いた SRS の解析方法が提案されているが [31]，全文検索に頼っていることから，その構造や表現の多様性により解析の自動化が困難である．

近年，SRS の表現や解析の新たなアプローチが提案されている．ALM（Application Lifecycle Management）[11] ツール間の連携基盤として開発された OSLC（Open Services for Lifecycle Collaboration）では，ツール間で SRS などの構造化文書を連携するために RDF（Resource Description Framework）[28] を用いて表現する方法が提案されている [21]．また，SRS を異なる表現に変換して扱う方法も提案されている [1], [10], [27]．しかし，これらの方法は表現や解析などの特定の機能にとどまっている．

本稿では，SRS の品質を解析するために，SRS を RDF で定義された中間言語 SRS-CIL（Common Intermediate Language）に変換し，解析する方法とその自動化ツールとして SRS 品質解析システム ReqQA（Requirements Quality Analyzer）を提案する．ReqQA のプロトタイプを実装して実際の RFP [17] に適用し，提案方法の妥当性や効果を示す．

2. 研究課題

本稿では，現在広く実践されている自然言語で記述された SRS を解析の対象とする．このような SRS の品質解析の研究課題を，SRS の品質に関する課題と SRS の表現多様性に関する課題としてとらえる．

2.1 SRS の品質と本稿における解析対象

良い SRS が満たすべき品質特性として IEEE 830 により次の 8 つの品質特性が定義されている [12]．なお，本稿では，下記の[ ]内に記述した特性名を同じ意味で用いることとする．

(1) 正確である（Correct）[正確性]
(2) 無曖昧である（Unambiguous）[無曖昧性]
(3) 完全である（Complete）[完全性]
(4) 一貫している（Consistent）[一貫性]
(5) 重要度/安定度がランク付けされている（Ranked for Importance and/or Stability）[ランク付け可能]
(6) 検証可能である（Verifiable）[検証可能性]
(7) 変更可能である（Modifiable）[変更可能性]
(8) 追跡可能である（Traceable）[追跡可能性]

これらの特性には，SRS の記述の表現に関する品質特性と内容に関する品質特性が混在している．これを，本稿では，表現品質，内容品質と呼ぶこととする．正確性などの内容品質の解析には，内容の理解と要求に関する知識が必要となることから，機械処理は本質的に困難である．そのため，本稿では表現品質を解析の対象とする．これは，内容品質が表現品質に依存し，まず，表現品質の保証が必要となることが理由である．たとえば，正確性を保証するためには，完全性と一貫性の保証が必要である [31]．

さらに，自然言語で記述された SRS の表現品質は，一般に，次の 3 層の言語学的品質特性でモデルできる [7]．

a) 文法的品質（Syntactic Quality）：記述言語の文法的正しさに関する品質．
b) 意味的品質（Semantic Quality）：記述言語による表現の良さに関する品質．
c) プラグマティック品質（Pragmatic Quality）：SRS を読む読者の読解の良さに関する品質．

一般に，SRS は多様な関心事が織り込まれていることから，人手による読解に人間的要因が影響し，適切な評価が困難であることが指摘されている [27], [31]．そのため，本稿では，読者の視点を定義し，それによって，読解の多様性を抑制できる機械処理可能な新たな解析方法の提案を課題とする．したがって，本稿における解析対象は自然言語で記述された SRS の表現品質のプラグマティック品質である．

2.2 SRS の多様性がもたらす SRS 解析問題

自然言語で記述された SRS の機械処理を困難にする主因に SRS の表現の多様性がある．この多様性は，文書構造の多様性と記述の多様性がある．

一般に SRS は木構造，あるいは，ハイパーテキスト構造をとる [31]．IEEE 830 は良い SRS の記述内容を木構造のプロトタイプ SRS と呼ぶ SRS テンプレートを提示している．REBOK では，IEEE 830 を発展させた SRS テンプレートを提示している．また，開発現場でも，固有の SRS のテンプレートの利用が推奨され，実践されている．このような事実に基づき，本稿では，標準 SRS と呼ぶ SRS の参照モデルを前提とする．さらに，プロジェクト個別の SRS（以下，プロジェクト SRS と呼ぶ）の要素は，標準 SRS の要素と対応づけ可能であると仮定する．ここで，標準 SRS の要素が多様なプロジェクト SRS の要素と対応づけ可能であるために，標準 SRS の構成が課題となるが，これは，SRS インスペクションで導入された標準 SRS として提供されていている [24]．この結果，プロジェクト SRS の解析は標準 SRS の解析に帰着できる．しかし，標準 SRS の解析を機械処理可能とするためには，その構造の意味，すなわち，要素とその間の関係を機械処理可能な表現にできる必要がある．このための表現を提示することが本稿の課題となる．

2.3 本稿の研究課題

SRS 品質解析ツールを実現するための上記の 2 つの課題

の整理から，本稿の研究課題は以下の 2 つとする．

(1) SRS の多様性に対応できる品質解析の自動化の実現可能性

SRS の統一的な品質解析の実現を阻害している主因は，企業やプロジェクトごとの SRS の構造や表現の多様性にある．そのため，SRS の構造を表現する共通中間言語 SRS-CIL を，RDF を用いて定義する．さらに，一般に広く利用されている Word で記述された SRS から SRS-CIL への変換の実現可能性を明らかにする．これらのアプローチの有効性を示す．

(2) 指定された SRS の表現に対して指定されたパースペクティブからのプラグマティック品質特性の解析の自動化の実現可能性

SRS-CIL 上で，指定されたプラグマティック品質特性に対する品質解析の自動化の可能性を明らかにする．具体的な方法として SRS インスペクション設計方法論 RISDM [23] に基づき，SRS 品質解析システム ReqQA を構成し，具体例を用いてその妥当性，有効性を評価する．

3. 関連研究

本研究の関連研究として，次の 3 つがある．

3.1 SRS の品質とその解析ツール

ソフトウェアシステムの品質に関する研究は数多いが SRS の品質に関する研究は限定的である [5]．

IEEE 830 では，SRS の表現と内容に関する 8 つの品質特性を規定している [12]．さらに，Davis らは，SRS の表現に関する 24 の品質特性を特定している [4]．これらの成果に基づき，要求工学知識体系（REBOK: Requirements Engineering Body Of Knowledge）では 11 の要求の特性を定義している [16]．

このような特性に対し，SRS の品質を解析するツールの必要性が指摘されている [17]．しかし，SRS の内容が企業やプロジェクトごとに異なることから解析ツールの実現は困難である．da Silva らは，要求仕様の形式的中間言語 RSL-IL を提案しているが [1]，その文法は固有であり，解析については具体的な提案に至っていない [27]．

また，SRS の記述に関する文法的，意味的な品質に対して，特定の読者を想定し，その読者にとっての品質をプラグマティック品質として定義されている [7], [19]．プラグマティック品質は，設計者や要求者などの特定の読者を想定することから，不特定な読者を対象とする従来の記述品質に比べ，より明確な品質基準を定義できるという利点がある．しかし，それを用いた SRS 解析ツールや適用は報告されていない．

3.2 SRS の表現とその支援環境

SRS を処理するために，適切な形式で表現する必要性が指摘されている．たとえば，SRS を含む開発成果物を管理するためのリポジトリが IDE や ALM などのコンポーネントとして提供されている．しかし，これらはベンダ固有であり，特定のツールに利用が限定されている．

これに対し，成果物の管理情報を連携するためのオープンな技術として OSLC（Open Services for Lifecycle Collaboration）が提案され，標準化が進められている [21]．OSLC では，異なるツール間でデータを連携するために Web 上での意味定義言語である RDF（Resource Description Framework）[28] を用いてデータを表現する．OSLC 上で SRS を管理する仕様が公開されているが，対象は管理のための SRS のメタデータに限定されている．

OSLC 上で SRS の内容を表現するために，IEEE 830 と要求工学知識体系（REBOK）に基づく SRS のデータモデルが提案され，Word で作成された SRS を変換するツールのプロトタイプも提案されている [2]．しかし，SRS の解析にまでは至っていない．

表 1 インスペクションポイントセット
Table 1 Inspection points set.

PQC		標準 SRS の目次			
ID	名称	2.1 システム化の目的	2.2 業務概要とシステム化の範囲	2.3 制約事項	2.4 用語定義
C1	合目的性	X			
C2	記述項目網羅性	X	X	X	X
C3	テンプレート使用	X	X	X	
C4	標準記法使用		X		
C5	用語定義				X
C6	識別子の付与	X	X	X	X
C7	一意識別性	X	X	X	X

3.3 SRS のインスペクションとその設計方法論

インスペクションとは，Fagan により提案されたソフトウェア開発の成果物に対するレビューを厳格，かつ，組織的に行う手順と体制などを定めた方法である [8]．Saito らは SRS の正しさの確認や品質評価のためのインスペクションの体系的な設計方法論を提案している [24]．ここでは，SRS の満たすべき品質としてプラグマティク品質特性（PQC: Pragmatic Quality Characteristic）[7], [19] の概念に基づき，IEEE 830 から SRS の PQC を定義している．SRS の PQC に対応する SRS のインスペクションを PBR（Perspective Based Reading）[26] により設計する．ここで，PQC は，SRS の読者のパースペクティブと参照 SRS に対する品質特性から各パースペクティブに基づいて SRS が備えるべき特性として定義される．たとえば，文献 [24] では，**表 1** の左側の列に示す C1 から C7 の 7 つの PQC が定められている．

PQC に基づくインスペクションを SRS の適切な箇所で行うため，標準 SRS の目次項目ごと，かつ，PQC ごとにインスペクション可能な箇所をインスペクションポイント

表 2 質問セット
Table 2 Questions set.

PQC		質問セット
ID	名称	
C1	合目的性	SRS のビジネス・システム要求はプロジェクトの目的に対応付けて記載されているか?
C2	記述項目網羅性	SRS の要素は標準 SRS に対応しているか?
C3	テンプレート使用	SRS の成果物は標準 SRS の中で定められているテンプレートを用いて記載されているか?
C4	標準記法仕用	SRS の成果物は標準 SRS の中で定められている標準(記述)記法で記載されているか?
C5	用語定義	SRS の用語集は作成されているか?
C6	識別子の付与	SRS の成果物や特定の要素は識別子が付与されて表で一覧化されているか?
C7	一意識別性	SRS の成果物や特定の要素は識別子を用いて一意に特定できるか?

（表 1 の X で示される箇所）として定義し，その集合をインスペクションポイントセットとしている（表 1）[24]．

インスペクションポイントで確認すべき内容が Yes/No で判断できる形式で質問を定義し，その集まりを質問セットとする（**表 2**）[24]．インスペクションでは，すべてのインスペクションポイントに対して質問を実行し，その結果に基づき SRS の品質を評価し，グラフなどで表現できる．しかし，インスペクションは人手で行われ，自動化ツールは提供されていない．

4. アプローチ

4.1 SRS の品質モデル

本稿の対象とする主として自然言語で記述された SRS は，研究課題で議論したように，その表現（Syntax）と意味（Semantic）ともに多様性が避けられない．このような SRS の読解において，読む視点であるパースペクティブが多様性の抑制に効果があることが指摘されている [26]．これは，SRS の解析におけるパースペクティブの有効性を示唆するといえる．

一方，自然言語で記述された SRS の言語学的な記述品質として，表現（syntax）品質，意味（Semantics）品質に加え，特定の読者の読解の品質がプラグマティック品質として提案されている [9], [19]．

これらをふまえ，本稿では，SRS の品質を次のように定義する．

定義［SRS 品質］：SRS の品質は，その想定読者のパースペクティブに対するプラグマティック品質である．

図 1 に SRS 品質のモデルを示す．SRS 参照品質とは，たとえば，IEEE830 [12] で規定されている 8 つの品質特性や REBOK [16] の 11 の品質特性である．SRS 品質は，これらの SRS 参照品質から読者のパースペクティブに適した品質特性へ具体化したものである．

なお，この定義は，文献 [24] において SRS のインスペクションのための品質特性として提案され，PBR と組み合わされて利用されているプラグマティック品質の概念を，SRS の品質を解析するために一般化したものである．

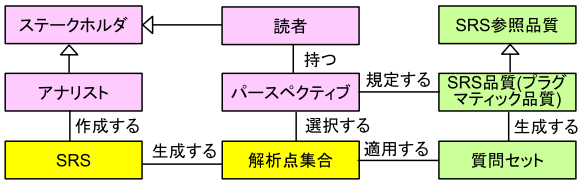

図 1 SRS の品質モデル
Fig. 1 A Quality model of SRS.

ここで，インスペクションポイントセットに相当する SRS の要素で解析対象となる要素の集合を解析点集合と呼ぶ．

4.2 SRS 品質アナライザ

本稿で提案する SRS の品質解析は，上記の定義に基づき，文献 [24] で提案された SRS のインスペクション設計方法論を，品質解析へ一般化し，自動化を図るものである．この品質解析モデルに基づき，SRS の品質解析を自動化するツールとして SRS 品質アナライザ ReqQA（Requirements Quality Analyzer）を提案する．

5. SRS 品質アナライザ ReqQA のアーキテクチャ

5.1 ReqQA アーキテクチャ

ソフトウェアによる SRS 品質解析を実現する SRS 品質アナライザ ReqQA のアーキテクチャを**図 2** に示す．

ここで，前提条件として，解析対象の SRS は企業やプロジェクトごとに定められたテンプレートに則っており，内部が Open XML [14], [23] で定義されている Word（Word 2003 以降）文書で記述されているものとする．また，このテンプレートの目次項目と IEEE 830 で定義されている標準 SRS の目次項目が対応付けられているものとする．

5.2 SRS-CIL：SRS の共通中間表現

SRS は企業やプロジェクトごとの構造の多様性がシステムによる統一的な解析の妨げとなってきた．そこで，Word や Excel で記述された SRS を前提として，その内部表現である Open XML [14] を意味構造に基づいて分解し，その意味構造を表現できる言語による表現へ変換する．この表現を SRS-CIL（Common Intermediate Language）と呼ぶ．

SRS-CIL は意味表現可能でオープンな標準言語である RDF [27] を用いて表現する．これにより，Web 上で特定ベンダによらず多様なツールによる解析が可能となる．

5.3 SRS 品質モデルに基づく SRS 品質アナライザの構成

SRS-CIL に変換された SRS を解析するための SRS 品質アナライザを構成する．SRS アナライザは，次の 2 つのコ

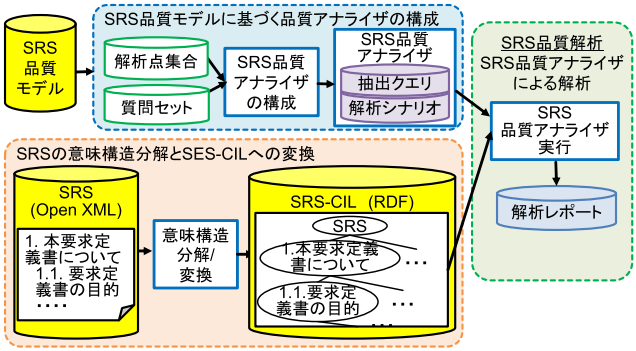

図 2 ReqQA のアーキテクチャ
Fig. 2 Architecture of ReqQA.

ンポーネントから構成される．

(1) 抽出クエリ：SRS 品質解析方法に基づく解析を行うため，SRS-CIL から解析すべき要素の集合である解析点集合に対して，質問セットの質問に応じた要素を抽出する SPARQL クエリの集合を生成．

(2) 解析シナリオ：抽出した要素に対し解析目的に合わせて解析を行うために解析手順を記述したシナリオの集合を生成．

5.4 SRS 品質アナライザの実行による解析レポート生成

SRS 品質アナライザのシナリオを実行し，その結果を SRS の要素ごと，あるいは，品質特性ごとに適切な表現に変換し，レポートとして提示する．

6. ReqQA の実現

6.1 プロジェクト SRS の多様性のモデル化

企業やプロジェクトごとに多様な表現をとる SRS を統一的に扱うために，**図 3** に示すように，プロジェクト SRS を内容と RDF による表現の 2 つの階層構造を通して，統一的に参照 SRS-CIL へマッピング可能とする．図において，左側の階層構造は SRS の内容の階層構造を表し，右側は，その RDF による表現である SRS-CIL の階層構造を表す．

(1) プロジェクト SRS の参照 SRS への内容のマッピング

左側の階層構造を通して，解析対象のプロジェクト SRS の内容は，プロジェクトによらない SRS の共通モデルである参照 SRS への特殊化であるので，その共通要素はプロジェクト SRS の要素へマッピング可能となる．ここで，プロジェクト SRS から参照 SRS へのマッピング可能範囲が解析のために必要な要素を網羅できるためには，参照 SRS の定義が多様なプロジェクト SRS を網羅できている必要がある．そのため，本稿で定義した参照 SRS は，IEEE 830 [12] の要素に，REBOK [16] で定義されている SRS の要素を付加している．さらに，要求管理ツール間で SRS の情報を交換するためのインタフェース仕様である OSLC [21] から筆者らが抽出したモデルを付加している．

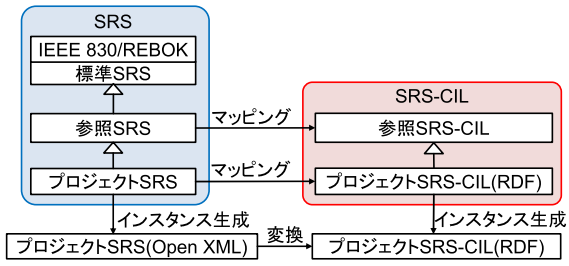

図 3 SRS-CIL のメタモデル
Fig. 3 Metamodel of SRS-CIL.

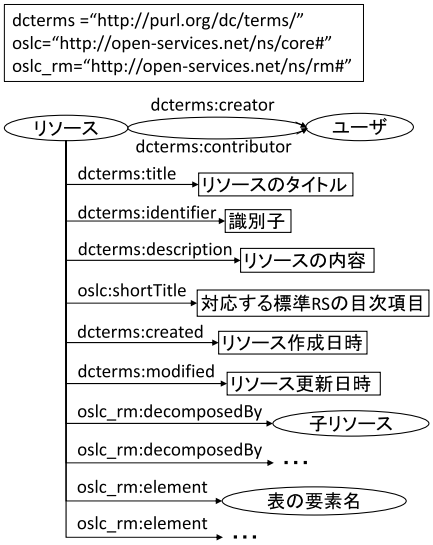

図 4 SRS-CIL のリソース定義
Fig. 4 Resource definition of SRS-CIL.

このモデル化の詳細については，文献 [2] を参照願いたい．

(2) 参照 SRS の SRS-CIL への表現のマッピング

参照 SRS の構造を RDF の語彙と名前空間で表現したものが参照 SRS-CIL である．参照 SRS-CIL は要素とその間の関係を主語，述語，目的語の三つ組みで表現できる RDF を用いて定義している．したがって，SRS の内容とその意味を表現する対として RDF で表現される．語彙は，参照 SRS の目次項目とその関係を定義する．

6.2 SRS リソースモデル

ReqQA では SRS を意味構造に基づき分割し，RDF で記述される SRS-CIL のリソースとして定義する．RDF ではすべて要素をリソースと呼ぶことから，本稿では以下，リソースと呼ぶこととする．標準 SRS-CIL のリソース定義では OSLC Core と OSLC RM (Requirements Management) のリソースを基礎としていることから名前空間に Dublin core の dcterms に加え，oslc，oslc_rm を利用する．

標準 SRS-CIL のリソース定義の一部を**図 4** に示す．リソースはプロパティとして識別子，タイトル，内容，リソース作成日時などを持つ．

子リソースを持つ場合は oslc_rm:decomposedBy プ

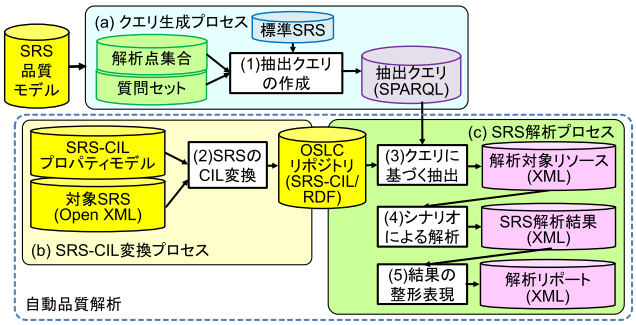

図 5 SRS 品質解析プロセス

Fig. 5 SRS quality analysis process.

ロパティの値として子リソースの URI を持ち，dcterms:description プロパティの値として，子要素となる章，節の目次項目を記述する．子リソースを持たない場合は，dcterms:description プロパティの値として SRS の本文を記述する．また，SRS 内の表から生成されたリソースの場合，oslc:element プロパティの値として表の要素名の URI を持つ．

6.3 SRS 品質解析プロセスと自動化の範囲

SRS-CIL を用いた SRS 品質解析プロセスを図 **5** に示す．

次の 3 つのプロセスから構成される．クエリ生成プロセスは解析に先立って 1 度実行すればよい．生成されたクエリを用いて，対象 SRS に対して SRS-CIL 変換プロセスと SRS 解析プロセスを自動実行することにより対象 SRS の品質解析を自動化できる．

(a) クエリ生成プロセス

SRS 品質モデルと標準 SRS に基づき定義された解析点集合と質問セットに対し，品質解析を行う SPARQL クエリを生成する．

(b) SRS-CIL 変換プロセス

SRS-CIL プロパティモデルに基づき，対象 SRS を SRS-CIL へ変換する．

(c) SRS 解析プロセス

生成された SPARQL クエリを SRS-CIL へ適用し，品質解析を実行する．

以下，各プロセスのアクティビティ (1)〜(5) を説明する．

(1) 抽出クエリの生成

解析点集合と質問セットを基に，解析に必要なリソースを抽出するためのクエリを生成する．クエリは RDF クエリ言語 SPARQL [29] を用いて記述する．

(2) SRS 文書の SRS-CIL への変換

Word 文書で記述された SRS を RDF で記述された SRS-CIF へ変換する．変換した SRS-CIF は RDF リポジトリで管理する．

(3) クエリに基づく SRS のリソース抽出

上記 (1) のクエリを用いて OSLC リポジトリ上の SRS-CIL から検証に必要なリソースを抽出する．

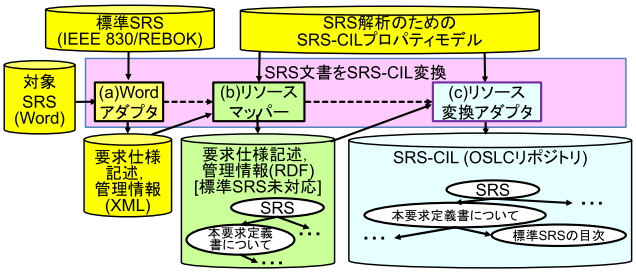

図 6 SRS 文書の SRS-CIL への変換プロセス

Fig. 6 Process of converting from SRS to SRS-CIL.

(4) シナリオに沿った解析の実行

上記 (3) の結果に対し，質問セットの質問に基づいて品質を解析するためのシナリオに沿ってクエリを実行する．

(5) 解析レポートの生成

上記 (4) の結果を人に理解しやすい形式に変換してレポートとして提示する．

6.4 SRS 文書の SRS-CIL への変換

SRS 文書の SRS-CIL への変換プロセスを図 **6** に示す．

構造の異なる SRS を文献 [2] に基づき定義された SRS プロパティを SRS-CIL へ変換する．さらに，企業ごとの SRS テンプレートを基に，RDF 化した SRS の各リソースに対して標準 SRS の目次項目を対応付けることで意味付けを行う．変換した SRS は OSLC リポジトリ上で管理する．

これにより企業ごとに構造が異なる SRS を参照 SRS に基づく共通表現に変換でき，SRS に対する統一的な品質解析が可能となる．以下に各コンポーネントの詳細を示す．

(a) Word アダプタ

入力された Word 形式の SRS を任意の XML 形式に変換する．

(b) リソースマッパ

Word アダプタにより任意の XML 形式に変換された SRS を SRS 分析のためのプロパティモデルに基づき RDF で表現された SRS-CIL へ変換する．

(c) リソース変換アダプタ

企業ごとの SRS テンプレートに対し対応付けられた標準 SRS の目次項目に基づいて，SRS-CIL の各リソースに対して標準 SRS の目次項目をプロパティとして付与する．作成したリソースは OSLC リポジトリ上に配置する．

6.5 リソース抽出クエリの生成

解析に必要なリソースを抽出するため，解析点集合と質問セットを基に，リソース抽出クエリを作成し，SPARQL で記述する．リソース抽出クエリは PQC の項目と品質解析を行う標準 SRS の目次項目ごとに作成する（図 **7**）．

リソース抽出クエリの導出過程を記述項目網羅性の例を用いて示す（図 7）．SRS-CIL に対して，記述項目網羅性は検証対象となる目次項目の内容を表すリソースの有無に

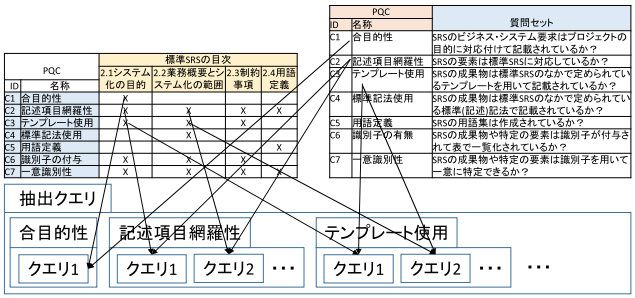

図 7 リソース抽出クエリの生成

Fig. 7 Generation of resource selection query.

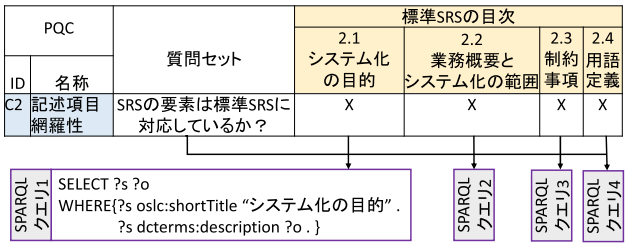

図 8 記述項目網羅性に関するリソース抽出クエリの導出

Fig. 8 An example of resource selection query.

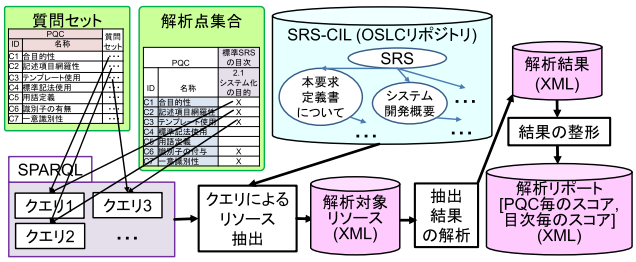

図 9 SRS 解析プロセス

Fig. 9 Process of analyzing SRS-CIL.

より判別可能である．そのため，解析対象の目次項目に対応するリソースの内容を抽出するクエリを作成する．

図 8 のクエリ 1 はシステム化の目的に関する記述項目網羅性の解析に用いるクエリの例である．

対応する標準 SRS の目次を表す oslc:shortTitle プロパティの値が「システム化の目的」であるリソースの内容を示す dcterms:description プロパティの値とそのリソースの URI を取得するクエリとなっている．また，これに対応するシナリオは「dcterms:description プロパティの値の有無のチェック」となる．

6.6 シナリオに基づく SRS の解析

OSLC リポジトリ上の SRS-CIL に対して SPARQL の抽出クエリを用いて解析に必要なリソースを抽出し，シナリオに沿った SRS の解析を行う（図 9）．

まず，OSLC リポジトリ上の SRS に対して SPARQL の抽出クエリを用いて解析に必要なリソースを抽出する．次に，抽出したリソースに対し，対応する質問セットを基にしたシナリオに沿った解析を行う．解析では，リソースの

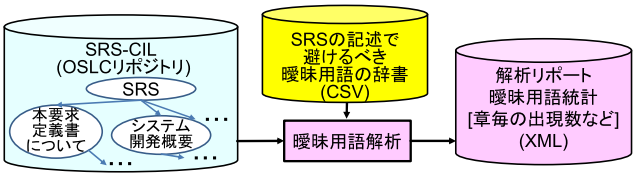

図 10 曖昧用語使用の解析

Fig. 10 Analysis of the use of ambiguous terms.

記述の有無やリソースの構造，リソース間の対応の可否により，質問を満たしているかの判定を行う．

解析結果を整形し，解析リポートを作成する．解析リポートは人に理解しやすい形式にするため，解析結果を基に，PQC ごとの品質スコアと標準 SRS の目次項目ごとの品質スコアを算出する．品質スコアは未解析の質問を除いた総質問数に対し回答が Yes の質問の比率を示す．

6.7 品質アナライザの拡張

品質アナライザの機能を拡張し，SRS 内で用いられている曖昧用語を検出する機能を追加した（図 10）．

曖昧用語は文献 [29] の「要求で避けるべき曖昧な用語」を用いた．ここでは，要求の曖昧さを生み出す共通の原因が説明されており，その中で検証不能な要求につながる曖昧用語と曖昧さを除去する方法が示されている．それを基に，曖昧用語のリストを曖昧用語辞書として作成し，CSV 形式で記述する．SRS-CIL の各リソースの内容を抽出し，その内容に対し曖昧用語辞書の用語が用いられているかチェックすることで SRS 内の曖昧用語を検出する．

検出結果は解析リポートとして提示する．解析リポートには，曖昧用語ごとの出現回数と対象 SRS の章ごとの曖昧用語の出現回数を算出し，記載する．解析リポートは XML 形式で出力している．

7. ReqQA プロトタイプの開発

提案する SRS 品質アナライザ ReqQA の妥当性と有効性を確認するために，プロトタイプを開発し，例題に適用して評価した．

ReqQA のプロトタイプを Eclipse 上での OSLC の参照実装である Eclipse Lyo [6] を拡張して RESTful Web サービスとして実現した．プロトタイプのシステム構成を図 11 に，その実装環境を表 3 に示す．

Word アダプタ，SRS 解析器，曖昧用語解析器は Java，JSP を用いて実装した．リソースマッパ，リソース変換アダプタ，解析リポートジェネレータは Java を用いて実装した．実装規模は Java が 879（LOC），JSP が 68（LOC）である．以下，各コンポーネントの概要を示す．

(1) Word アダプタ

Word 形式の SRS を章や節ごとのまとまりに分割し，XML 形式に変換する．

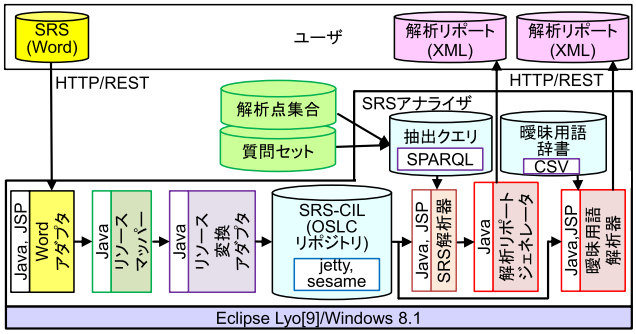

図 11 SRS 品質アナライザのプロトタイプの構成
Fig. 11 Configuration of the prototype of SRS analyzer.

表 3 プロトタイプ実装環境
Table 3 Platform for the prototype of SRS analyzer.

システム	バージョン
OS	Windows 8.1 64bit
JDK	JDK 1.6.0_45
Eclipse	Eclipse 4.4.1
Web Server	Jetty 7.3.0
RDF Store	Sesame 2.6.10

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:oslc="http://open-services.net/ns/core#" xmlns:oslc_rm="http://open-services.net/ns/rm#"
xmlns:rio="http://open-services.net/ri/" xml:base="http://localhost:8080/rio-rm/requirement/11">
  <oslc_rm:Requirement rdf:about="http://localhost:8080/rio-rm/requirement/11">
    <rdf:type rdf:resource="http://open-services.net/ns/rm#Requirement"/>
    <dcterms:title>1. 現行ホームページの概要</dcterms:title>
    <oslc:shortTitle>section2_1</oslc:shortTitle>
    <dcterms:description>現行システムは、平成 18 年 3 月に導入したもので、オープンソース CMS（Joomla!）を用いてコンテンツを作成。FTP によるアップロードは行わず、ホームページアクセスがあった場 合、CMS がリクエストを受け、直接データベースからコンテンツを返す仕組みとなっている。現行 Web コンテンツ容量等（平成 22 年 10 月 31 日現在）
現行 Web サイト訪問者数（平成 21 年 11 月?平成 22 年 10 月のページビュー）
現行関連サーバの構成概要
現行ネットワーク環境
現行 IDC 内ネットワークシステムの機能概要
現行システム構成
</dcterms:description>
    <dcterms:identifier>11</dcterms:identifier>
    <dcterms:contributor rdf:resource="http://localhost:8080/rio-rm/_UNKNOWN_USER_"/>
    <dcterms:modified rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2015-01-05T16:25:27.160+09:00</dcterms:modified>
    <dcterms:created rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2015-01-05T16:25:27.160+09:00</dcterms:created>
    <dcterms:creator rdf:resource="http://localhost:8080/rio-rm/_UNKNOWN_USER_"/>
    <oslc_rm:decomposedBy rdf:resource="http://localhost:8080/rio-rm/requirement/12"/>
    <oslc_rm:decomposedBy rdf:resource="http://localhost:8080/rio-rm/requirement/18"/>
    <oslc_rm:decomposedBy rdf:resource="http://localhost:8080/rio-rm/requirement/22"/>
    <oslc_rm:decomposedBy rdf:resource="http://localhost:8080/rio-rm/requirement/35"/>
    <oslc_rm:decomposedBy rdf:resource="http://localhost:8080/rio-rm/requirement/38"/>
    <oslc_rm:decomposedBy rdf:resource="http://localhost:8080/rio-rm/requirement/47"/>
    <oslc_rm:stdTitle>調達内容・業務の詳細</oslc_rm:stdTitle>
    <oslc_rm:stdTitle>現行システムとの関連</oslc_rm:stdTitle>
  </oslc_rm:Requirement>
</rdf:RDF>
```

図 12 例題の SRS-CIL 表現（部分）
Fig. 12 SRS-CIL of the RFP (Part).

(2) リソースマッパ

Word アダプタにより処理された SRS を RDF 形式に変換する．XML 形式の SRS を章，節などの要素ごとに分割し，URI を付与することで要素ごとに RDF に変換する．

(3) リソース変換アダプタ

企業ごとのテンプレートに対する標準 SRS の目次項目の対応付けを基に，リソースマッパにより RDF 形式に変換された SRS の各リソースに対し，対応する標準 SRS の目次項目をプロパティとして付与する．これにより，SRS が SRS-CIL のリソースへ変換される．後述する RFP の例題で生成された SRS-CIL のリソースの一部を**図 12** に示す．このリソースは RDF リポジトリ Sesame [25] で管理している．

本プロトタイプでは，変換アダプタを Java で実装しているが，XSLT を用いた非手続き的な変換もある [3]．

(4) SRS 解析器

SPARQL を用いた抽出クエリによって OSLC 上の SRS-CIL から解析に必要なリソースを抽出する．あらかじめ定義された抽出クエリセットの親ディレクトリを入力として指定することにより各クエリに対する結果を一括して生成可能である．抽出したリソースは XML 形式で出力する．

図 13 (a) は**表 4** に示す PQC の U3-5「一意に特定可能」を解析するための SPARQL の抽出クエリの例を示す．同様に，図 13 (b) は U3-1「標準 RFP との整合性」を解析するための SPARQL 抽出クエリの例である．

(5) 解析結果アナライザ

SRS 解析器により抽出された SRS の各リソースに対してシナリオに沿った解析を行う．すべての解析対象リソースに対する解析結果を XML 形式で出力する．解析結果は，質問に対する回答が「Yes」の場合は「○」，「No」の場合は「×」，未解析の質問は「？」を用いて表現する．

図 14 に後述する例題の解析結果の出力例を示す．

この解析結果から人に分かりやすい表現に整形し，解析リポートを生成する．解析リポートは，標準 SRS の目次項目ごとの品質スコアと PQC ごとの品質スコアを算出し，提示する．

(6) 曖昧用語検出器

SRS-CIL に対し，曖昧用語辞書を基に各リソースで使用されている曖昧用語の利用状況を解析する．曖昧用語は，文献 [29] で定義されている「要求で避けるべき曖昧な用語」

```
PREFIX dc:<http://purl.org/dc/terms/>
PREFIX oslc_rm:<http://open-services.net/ns/rm#>
SELECT ?table ?elmt
WHERE {
  ?uri oslc_rm:stdTitle "提案の範囲" ;
       oslc_rm:decomposedBy ?table .
  ?table dc:shortTitle "table" ;
         oslc_rm:decomposedBy ?tableItem .
  ?tableItem oslc_rm:element ?elmt .
}
```

(a) 「一意に特定可能」を解析するための SPARQL 抽出クエリの例

```
PREFIX dc:<http://purl.org/dc/terms/>
PREFIX oslc_rm:<http://open-services.net/ns/rm#>
SELECT ?uri ?description
WHERE {
?uri oslc_rm:stdTitle "解決したい課題" ;
    dc:description ?description .
}
```

(b) 「標準 RFP との整合性」を解析するための SPARQL 抽出クエリの例

図 13 SPARQL 抽出クエリの例
Fig. 13 Examples of SPARQL query.

表 4 RFP のプラグマティック品質特性
Table 4 PQC (Pragmatic quality characteristics) of RFP.

ID	品質特性	ID	副品質特性
U1	合目的性	U1-1	システム目的の独立性
		U1-2	業務要求のシステム目的への適合
U2	生産効率性	U2-1	定量的具体性の有無
		U2-2	要求の独立性
U3	堅実性	U3-1	標準 RFP との整合性
		U3-2	文書の参照関係の明示
		U3-3	例外要求の網羅
		U3-4	変更可能性の明記
		U3-5	一意に特定可能
		U3-6	用語集の存在
U4	充足性	U4-1	ランク付けの有無
		U4-2	用語の整合
		U4-3	動作の整合
		U4-4	制約条件と要求の整合

```
<?xml version="1.0" encoding="Shift_JIS" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="result_style.xsl"?>
<result>
        <R id="R1-1">
                <U id="U1-1">×</U>
                <U id="U1-2">?</U>
                <U id="U2-2">×</U>
                <U id="U3-1">○</U>
                <U id="U4-2">?</U>
        </R>
        <R id="R1-2">
                <U id="U1-1">○</U>
                <U id="U1-2">?</U>
                <U id="U3-1">○</U>
                <U id="U4-1">×</U>
        </R>
        <R id="R1-3">
                <U id="U1-2">?</U>
                <U id="U2-2">×</U>
                <U id="U3-1">×</U>
        </R>
        ⋮
```

図 14 解析結果の例
Fig. 14 An example of analysis results.

を用いた．曖昧用語辞書は CSV 形式で定義した．OSLC リポジトリの SRS-CIL からすべてのリソースの内容を取得し，取得した内容に対して曖昧用語辞書に記載されている用語が用いられているかチェックした．

解析結果として，曖昧用語ごとの出現回数と対象 SRS の章ごとの曖昧用語の出現回数を算出し，解析リポートとして提示する．

8. 例題への適用

本稿では SRS に対する品質解析ツールを提案しているため適用対象として SRS を想定している．しかし，実際に開発で用いられている SRS は一般に公開されていない．そのため，適用する例題を公開情報の中から選び，地方自治体のホームページリニューアルに関する RFP [18] とした．この RFP は A4 版で 25 ページある．

一般に RFP の記述の標準化はされていない [22]．このため，参照 SRS に代わり IT コーディネータ協会が公開し

ID	プラグマティック品質	有効質問数
U1-1	システム目的の独立性	3
U1-2	業務要求のシステム目的への適合	5
U2-1	定量的具体性の有無	4
U2-2	要求の独立性	3
U3-1	標準SRSとの整合性	42
U3-2	文書の参照関係の明示	2
U3-3	例外要求の網羅	0
U3-4	変更可能性の明記	0
U3-5	一意に特定可能	1
U3-6	用語集の存在	0
U4-1	ランク付けの有無	1
U4-2	用語の整合	4
U4-3	動作の整合	0
U4-4	制約条件と要求の整合	3
合計		68

解析可能なプラグマティック品質	質問数
システム目的の独立性	3
要求の独立性	3
標準SRSとの整合性	42
文書の参照関係の明示	2
一意に特定可能	1
ランク付けの有無	1
合計	52

解析不可能なプラグマティック品質	質問数
業務要求のシステム目的への適合	5
定量的具体性の有無	4
用語の整合	4
制約条件と要求の整合	3
合計	16

図 15 解析結果
Fig. 15 Results of analysis.

ている「RFP 見本」[15] を参照 RFP として利用した．このため，解析点集合と質問セットとして文献 [20] で提案されている RFP に対するインスペクションマトリクスと RFP の品質を解析するための質問セットを利用した．この結果，RFP の品質特性も文献 [20] で定義されている，表 4 に示すプラグマティック品質特性を用いた．表 4 に示す品質特性はシステムのユーザのパースペクティブから ISO/IEC25010 で定義されたシステムの利用品質 [13] を含むプラグマティック品質特性となっている．一方，表 1，表 2 に示すインスペクションのための 7 つのプラグマティック品質特性は SRS に基づき開発する開発者のパースペクティブから定義されている．この点で，表 4 に示すプラグマティック品質特性は表 1，表 2 に示すプラグマティック品質特性とは異なるものとなっている．

本稿の提案方法では検証対象の RFP に対する標準 RFP の目次項目の対応付けが必要であるが．適用対象 RFP は標準 RFP との対応付けがされていない．そのため，適用対象 RFP の目次項目と標準 RFP の目次項目の対応付けを行ってから例題へ提案方法を適用した．

RFP の品質を解析するために文献 [20] で定義した質問セットには質問ごとにペルソナの視点による差分がある．これは，パースペクティブとして定義されるものである．本稿では，ペルソナごとの詳細な品質解析は必要ではないと判断し，この複数のペルソナの視点をまとめてユーザパースペクティブとして定義している．また，RFP 内の図に関する質問も解析の対象とはしていない．そのため，図に関する質問とペルソナの視点による差分を除いた 68 の質問を有効質問とし，解析を実行した．解析結果のまとめを**図 15** に示す．

この図に示すように，68 個の質問セット中，52 個の質問について解析が可能であった．ここで，「例外要求の網羅」，「変更可能性の明記」，「用語集の存在」については RFP に対する質問項目がなく，「動作の整合」については図に対する解析が必要なため，これらのプラグマティック品質特性に関する解析は対象外とした．

表 5 対象 RFP の章ごとの曖昧用語検出結果
Table 5 Chapter-by-chapter usage distribution of ambiguous terms in the example of RFP.

章	文字数	種類数	出現数
1	790	2	2
2	1,032	0	0
3	916	0	0
4	1,114	7	8
5	3,101	7	8
6	3,960	4	7
7	4,732	8	9
8	2,879	2	3
9	617	0	0
合計	19,141	37	

表 6 対象 RFP の頻出曖昧用語
Table 6 The most frequent use of ambiguous terms.

用語	出現数
その他	6
～しない	5
i.e.	5
など	4
～から～まで	3
含めて	3

注：種類数：重複なし
出現数：重複あり

さらに，文献 [30] に基づく 75 種類の曖昧用語の利用解析の結果を**表 5** と**表 6** に示す．曖昧用語解析の結果，14 種類の曖昧用語を検出した．RFP の入力から解析結果の出力までの実行時間は約 5 分であった．

9. 評価と考察

9.1 評価の目的

2.3 節で述べた研究課題 (1)，(2) に対して，提案した ReqQA の例題への適用結果を評価することにより，その実現性と実用性を評価する．

9.2 評価の方法

ReqQA を評価するため，次の 3 つの評価尺度を定義した．
(1) 網羅率：SRS の解析項目中で解析できた項目の比率．
(2) 有効率：SRS の解析項目中で正しい結果を得た比率．
(3) 品質スコア：プラグマティック品質特性に対する SRS 全体の包括的な品質として，式 (3) で定義する．

$$\text{網羅率} = \frac{\text{解析可能質問数}}{\text{質問総数}} \quad (1)$$

$$\text{有効率} = \frac{\text{想定した結果を得た質問数}}{\text{解析可能質問数}} \quad (2)$$

$$\text{品質スコア} = \frac{\text{Yes の数}}{\text{解析可能質問数}} \quad (3)$$

9.3 解析可能性の評価

ReqQA による解析の網羅率は 76.4%であった．自動解析可能となったプラグマティック品質特性は以下の 6 項目である．
(1) システム目的の独立性（U1-1）
(2) 要求の独立性（U2-2）
(3) 標準 RFP との整合性（U3-1）
(4) 文書の参照関係の明示（U3-2）
(5) 一意に特定可能（U3-5）
(6) ランク付けの有無（U4-1）

これらのプラグマティック品質特性は提案方法による解析が可能であるといえる．しかし，独立性に関する品質の解析は，個々の目的，要求が分割されているかという限定的な解析にとどまった．これは質問をより詳細化し，分割することで改善可能と考えられる．

また，以下のプラグマティック品質特性に関しては，現在の実装においては解析が不可能であった．
(1) 業務要求のシステム目的への適合（U1-2）

システム目的が 1 つの文章で記述されており，RFP を RDF 形式に変換する際，単一リソースとして変換されたため，業務要求のリソースとシステム目的のリソース間での対応がとれず，解析が不可能となった．
(2) 定量的具体性の有無（U2-1）

解析対象となるリソースが細分化できず，対象リソースの内容の具体的な値や表現がされているか判別できず，解析不可となった．
(3) 用語の整合（U4-1）

文章中の用語がその用語の意味にそって正しく用いられているかにどうかの解析が困難である．また，RFP では用語集との関連が確認できないため，用語集に関する解析も不可能となった．
(4) 制約条件と要求の整合（U4-4）

要求および制約条件が複数のリソースではなく，単一のリソースとして変換されたため，制約条件のリソースと要求のリソースでの対応の確認ができず，解析が不可能となった．

これらのプラグマティック品質特性の解析では，解析対象のリソースが分割されず単一のリソースとして変換されたため，リソース間の対応による確認が不可能となる共通の問題が明らかとなった．これに対しては，細粒度のリソース定義を行い，複数のリソースに分割することで，解析可能になると考えられる．

9.4 プラグマティック品質特性ごとの解析結果の評価と考察

プラグマティック品質特性ごとの品質スコアを**表 7** と**図 16** に示す．

総数は質問の総数を，解析数は総数のうち解析可能で

表 7 プラグマティック品質特性ごとの品質スコア
Table 7 Pragmatic quality score.

ID	総数	解析数	網羅率[%]	Yes 数	品質スコア[%]
U1-1	3	3	100.0	1	33.3
U2-2	3	3	100.0	0	0.0
U3-1	42	42	100.0	27	64.3
U3-2	2	2	100.0	1	50.0
U3-5	1	1	100.0	0	0.0
U4-1	1	1	100.0	0	0.0
合計	52	52	100.0	29	55.8

表 8 章ごとのプラグマティック品質特性のスコア
Table 8 Pragmatic quality score by chapter.

章 ID	総数	解析数	Yes 数	未解析数	品質スコア[%]
R1	21	14	5	7	35.7
R2	28	20	15	8	75.0
R3	5	5	1	0	20.0
R4	4	4	4	0	100.0
R5	3	2	2	1	100.0
R6	7	7	2	0	28.6
合計	68	52	29	16	55.8

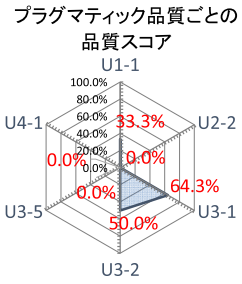

図 16 プラグマティック品質特性ごとの品質スコアのグラフ
Fig. 16 Distribution of pragmatic quality score.

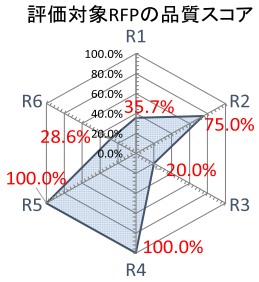

図 17 章ごとのプラグマティック品質特性のスコアのグラフ
Fig. 17 Distribution of pragmatic quality score by chapter.

あった質問数である．さらに，Yes 数は解析可能であった質問数のうち，Yes となった質問数である．網羅率，品質スコアは，それぞれ，式 (1)，式 (3) で定義したものである．

解析を行った 6 つのプラグマティック品質特性は網羅率が 100%であり，漏れなく解析できている．このことから，解析対象の目次項目によらず，リソースの構造やリソース間の対応による解析が可能であるといえる．そのため，これらのプラグマティック品質特性に関しては，システムによる自動解析が可能であるといえる．

また，「要求の独立性」，「一意に特定可能」，「ランク付けの有無」の品質スコアは 0%となっている．特に，「一意に特定可能」と「ランク付けの有無」は解析数が 1 のため真偽の結果となっており，品質スコアの精度が低いと考えられる．これに対しては，リソース定義の細粒度化を行うことにより解析の細粒度化を行い，品質スコアの精度を向上させる必要がある．

9.5 目次項目ごとの解析結果の評価と考察

標準 RFP の章ごとの品質スコアを**表 8** と**図 17** に示す．各列の数値の定義は前節と同一である．

6 つの章のうち，3 つの章，「R3：提案手続き」，「R4：開発に関する条件」，「R6：契約事項」では未解析数が 0 である．このことから，これらの章に対する ReqQA による解析が可能であるといえる．しかし，「R1：システム概要」，「R2：提案依頼事項」，「R5：保証要件」は未解析項目があり，質問総数に対して未解析の割合が高い．そのため，これらの章に関する品質スコアは正確性が低いと考えられる．

品質スコアの正確性を向上させるため，未解析項目の低減が必要である．

9.6 SRS-CIL の有用性に関する考察

Word 形式で記述された RFP を意味構造に基づき分解し，RDF 形式に変換することで複数のリソースに分割し，要素間の意味構造を表現できる SRS-CIL を定義した．そのため，リソース間の対応やリソースの構造に関する品質の解析が可能となった．これにより，質問セットに基づく解析が，記述の有無だけでなく，RFP の構造や意味に基づく解析が可能となった．また，構造が異なる RFP を共通表現に変換したことにより，構造が異なる文書に対し統一的な解析が可能となった．

SRS-CIL はオープンな標準に基づいており，RDF を用いた意味表現が可能であることから，関連研究 [1], [26] に比べ，多様なステークホルダが参画する SRS の中間表現として適していると考えられる．

提案方法による解析では，網羅率が 100%とならず，限定的な解析と未解析にとどまったプラグマティック品質特性があった．これらのプラグマティック品質特性に対する解析は，リソースを解析可能な意味要素とその関係となる適切な粒度へ分割し，リソースへ対応付けることで解析可能になると考えられる．このためには，SRS-CIL とその RDF リソース定義を見直す必要があると考えている．

提案した品質解析方法により，ソフトウェアによる RFP に対する構造，表現の解析が可能となった．システムによる自動化によって，個人差の低減が可能となる．このことから，提案方法を用いることで SRS の品質解析のコストの削減と品質の向上が期待できる．

9.7 曖昧用語の解析結果に関する考察

表 5 に示す結果から章により曖昧用語の出現に偏りがあることが明らかになった．これは，章の内容，著者の個人差などに起因すると考えられる．また，表 6 に示す結果は特定の曖昧用語が頻用される傾向があることを示唆している．これは，曖昧用語解析によって特定の章の執筆，あるいは，著者ごとの曖昧用語の利用を抑制するための効果的なアドバイスを提供できることを示唆している．

9.8 提案解析システムの拡張性に関する考察

提案解析システムを拡張し，曖昧用語の検出機能を追加したが，SPARQL のクエリとシナリオの追記のみで実現している．これは，オープンな標準に準拠した共通の中間言語である SRS-CIL の効果である．対象の SRS や RFP の構造によらず，多様な解析機能が容易に追加可能である．

10. 今後の課題

10.1 実際の SRS への適用と実践

本稿では SRS に代わり RFP に提案方法を適用した．そのため，実際の SRS に提案方法を適用した場合に品質改善の効果を確認する必要がある．あわせて，提案した品質解析システム ReqQA を実開発へ適用するためのツールの改善も必要である．

10.2 網羅率の向上

例題への適用では，4 つのプラグマティック品質特性が解析不可能であった．これらのプラグマティック品質特性に関する質問の具体化，SRS 要素の粒度の細分化とそのリソース定義を行い，当該プラグマティック品質特性に対する解析を可能とし，網羅率を向上させる必要がある．

10.3 SRS 規模の増大に対するスケーラビリティの向上

SRS の規模増大にともない，変換した RDF のリソース数も増大し，探索型のクエリである SPARQL の処理時間が増加すると考えられる．例題では約 5 分を要していたが，処理時間のスケーラビリティの評価と処理の効率化を検討する必要がある．

11. まとめ

これまで，実践における重要性は指摘されていながら研究が進んでいなかった SRS の品質解析について，SRS インスペクション設計方法論と OSLC の成果を発展させ，品質の自動解析の方法と解析システム ReqQA を提案し，プロトタイプとして実装した．

本提案の意義は，多様な SRS に対して，その多様性を扱える RDF を用いた意味構造を表現できる中間言語 SRS-CIL とその上での解析方法の自動化にある．これにより，研究課題を解決できる実現性を具体的に示した点で，実用的でもある．

提案方法により SRS の品質向上が期待できる．

今後，ReqQA を実開発の SRS へ適用し，実用性を評価する予定である．

謝辞 本研究は，著者と（株）NTT データの斎藤忍氏らとの共同研究の成果に基づき，発展させたものである．斎藤氏と関係各位に感謝する．

参考文献

[1] de Almeida Ferreira, D. and da Silva, A.R.: RSL-IL: An Interlingua for Formally Documenting Requirements, *Proc. MoDRE 2013*, IEEE, pp.40–49 (July 2013).
[2] 青山幹雄，壁谷孝洋：OSLC に基づく要求管理方法と支援環境の提案と評価，ソフトウェア工学の基礎 XX（FOSE2013 論文集），pp.263–272, 近代科学社 (Dec. 2013).
[3] Breitling, F.: A Standard Transformation from XML to RDF via XSLT, *Astronomische Nachrichten*, Vol.330, No.7, pp.755–760 (Aug. 2009).
[4] Davis, A. et al.: Identifying and Measuring Quality in a Software Requirements Specification, Proc. 1st Int'l Software Metrics Symposium, pp.141–152, IEEE (May 1993).
[5] Denger, C. et al.: Quality Assurance in Requirements Engineering, *Engineering and Managing Software Requirements*, Aurum, A. et al. (eds.), pp.163–185, Springer (2005).
[6] Eclipse Lyo, available from ⟨http://eclipse.org/lyo/⟩.
[7] Fabbrini, F. et al.: Achieving Quality in Natural Language Requirements, *Proc. 11th Int'l Software Quality Week*, 17 pages (May 1998).
[8] Fagan, M.E.: Advances in Software Inspections, *IEEE Trans. Softw. Eng.*, Vol.SE-12, No.7, pp.744–751 (July 1986).
[9] Fanmuy, G. et al.: Requirements Verification in the Industry, *Proc. CSDM 2011*, pp.145–160, Springer (Dec. 2011).
[10] Garg, P.K. et al.: A Hypertext System to Manage Software Life-Cycle Documents, *IEEE Software*, Vol.7, No.3, pp.90–99 (May 1990).
[11] Hüttermann, M.: *Agile ALM*, Manning (2012).
[12] IEEE Std. 830-1998, IEEE Recommended Practice for Software Requirements Specifications, IEEE (1998).
[13] ISO/IEC 25010:2011, Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE), Systems and Software Quality Models (2011).
[14] ISO/IEC 29500-1:2012, Information Technology – Document Description and Processing Languages – Office Open XML File Formats – Part 1: Fundamentals and Markup Language Reference (2012).
[15] IT コーディネータ協会：RFP/SLA 見本 (2004)，入手先 ⟨http://www.itc.or.jp/foritc/useful/rfpsla/rfpsla_doui.html⟩.
[16] JISA REBOK 企画 WG（編）：要求工学知識体系，第 1 版，近代科学社 (2011).
[17] Katasanov, A. et al.: Requirements Quality Control: A Unified Framework, *Requirements Eng. J.*, Vol.11, No.1, pp.42–57 (Mar. 2006).
[18] 甲府市：ホームページリニューアル業務に関わる受託事業者選考の事業公告，2012, 入手先 ⟨http://www.city.kofu.yamanashi.jp/koho/shise/koho/hp/renewal.html⟩（参照

2014-12）.
[19] Krogstie, J. et al.: Towards a Deeper Understanding of Quality in Requirements Engineering, *Proc. CAiSE '95*, LNCS, Vol.932, pp.82–95, Springer (1995).
[20] 森下月菜，青山幹雄：ペルソナ観点からの利用品質に着目したソフトウェア要求仕様書のインスペクション方法の提案と評価，情報処理学会 第 183 回ソフトウェア工学研究会，Vol.2014-SE-183, No.7, pp.1–8 (Mar. 2014).
[21] OSLC (Open Services for Lifecycle Collaboration), available from ⟨http://open-services.net/⟩.
[22] Porter-Roth, B.: *Request for Proposal: A Guide to Effective RFP Development*, Addison-Wesley (2001).
[23] Rice, F.: Introducing the Office (2007) Open XML File Formats, available from ⟨https://msdn.microsoft.com/en-us/library/aa338205(v=office.12).aspx⟩.
[24] Saito, S. et al.: RISDM: A Requirements Inspection Systems Design Methodology, *Proc. RE 2014*, pp.223–232, IEEE (Aug. 2014).
[25] Sesame, available from ⟨http://rdf4j.org/⟩.
[26] Shull, F. et al.: How Perspective-Based Reading Can Improve Requirements Inspections, *IEEE Computer*, Vol.33, No.7, pp.73–79 (July 2000).
[27] da Silva, A.R.: Quality of Requirements Specifications: A Preliminary Overview of an Automatic Validation Approach, *Proc. SAC 2014*, pp.1021–1022, ACM (Mar. 2014).
[28] W3C, RDF 1.1 Primer, available from ⟨http://www.w3.org/TR/rdf11-primer/⟩.
[29] W3C, SPARQL 1.1 Query Language, available from ⟨http://www.w3.org/TR/sparql11-query/⟩.
[30] Wiegers, K.: *Software Requirements*, *3rd Ed.*, Microsoft Press (2013).
[31] Wilson, W.M. et al.: Automated Analysis of Requirement Specifications, *Proc. ICSE 97*, pp.161–171, ACM (May 1997).

青山 幹雄 （正会員）

1980 年岡山大学大学院工学研究科修士課程修了．同年富士通株式会社入社．大規模ソフトウェア開発とプロジェクト管理，ソフトウェア工学の実践に従事．1986～1988 年米国イリノイ大学客員研究員．1995 年 4 月～2001 年 3 月新潟工科大学情報電子工学科教授．2001 年より南山大学数理情報学部情報通信学科教授，2009 年より同大学情報理工学部ソフトウェア工学科教授．博士（工学）．クラウドコンピューティング，サービス指向アーキテクチャ，自動車組込みソフトウェア等を対象として，要求工学，ソフトウェアアーキテクチャ技術，ソフトウェア進化の研究・開発と教育・人材育成に取り組む．著書『要求工学知識体系』（2011 年刊：共著）ほか多数．IEEE Software，IEEE Transactions on Services Computing 等の編集委員，本会理事を歴任．1993 年情報処理学会研究賞受賞．ソフトウェア科学会，自動車技術会，IEEE，ACM 各会員．

中根 拓也

2015 年 3 月南山大学大学院理工学研究科ソフトウェア工学専攻修士課程修了．現在，株式会社インテックにて開発業務に従事．在学中，要求工学の研究に従事．