reqchecker：IEEE 830の品質特性に基づく日本語要求仕様書の問題点検出ツール※

林　晋平[†a)]　有賀　顕[†*]　佐伯　元司[†]

reqchecker: A Tool for Detecting Problems in Japanese Requirements Specification Documents Based on IEEE 830 Quality Characteristics※

Shinpei HAYASHI[†a)], Ken ARUGA[†*], and Motoshi SAEKI[†]

あらまし　要求仕様書は主に自然言語で記述されているため，文意のあいまい性などの問題がある．高品質な要求仕様書を得るためには，要求分析者がこれらの問題点を認識し発見することが重要である．本論文ではIEEE 830で定義された品質特性をもとに，日本語で記述された要求仕様書の問題点を検出する手法及びその自動化ツール reqchecker を提案する．提案手法では，要求仕様書全体と要求文の解析，更に要求文間の関係解析を行い要求仕様書中の問題点を検出する．reqchecker では非あいまい性など六つの品質特性に関する問題点を検出しマーキングを行うことで，要求分析者による問題点の発見を支援する．例題への適用及び被験者実験により reqchecker の有用性に関する予備評価を行ったところ，良好な結果を得た．

キーワード　要求仕様書，構文解析，品質特性

1. まえがき

要求分析者に対して要求仕様書の品質向上支援を行うことで，ソフトウェア開発の失敗を防ぐことが重要である．ソフトウェア開発の失敗の一因に要求定義における失敗があると言われている[1], [2]．そのため要求定義，要求分析の成果物である要求仕様書の品質が重要である[3]．要求分析者に対して要求仕様書の品質向上支援を行い，作成される要求仕様書の品質を高めることは，ソフトウェア開発の成功に貢献する．

要求分析者に対する品質向上支援の方針には，要求分析者のスキルアップ支援[4], [5]，高品質な要求仕様書の作成のための前工程の導入[6]，要求仕様書の問題点検出等がある．本論文では要求仕様書の問題点検出として，自然言語で記述された要求仕様書の様々な種類の問題点を発見する，人手と計算機を用いたインタラクティブな支援を扱う．ここでインタラクティブな支援とは，計算機により自動検出した問題点を分析者が吟味することで問題点の発見を支援することを指す．要求分析工程では，要求獲得，要求記述，要求検証，要求管理を繰り返すことにより要求仕様を開発していく[7]が，本論文における支援はこのうち要求記述工程に対するものである．

IEEE 830 [8]では要求仕様書に対して八つの品質特性が定義されており，更に要求工学ワーキンググループがまとめている成果報告書[7]（以降，報告書）では各品質特性に対して計測可能な手段とその項目（以降，品質特性項目）を示している．また，Davisの教科書においても，同様の観点により品質特性を満たさない要求文の具体例が示されている[9]．しかし，これらの報告書や書籍では，品質特性項目の定性的な特徴や例文が述べられるのみで，それらを計算機によりどのように検出すればよいかについて十分に述べられていない．本論文で扱う問題点とは，こういった品質特性項目に関する問題点を指し，これらを計算機により自動的に検出することを目指す．

本論文では，要求仕様書の文章構造と要求文の構文構造を用いて，報告書[7]が整理する品質特性項目

† 東京工業大学，東京都
Tokyo Institute of Technology, Meguro-ku, Tokyo, 152–8552 Japan
*（株）ユー・エス・イー
a) E-mail: hayashi@c.titech.ac.jp
※ 本論文はシステム開発論文である．
DOI:10.14923/transinfj.2017SKP0036

に関する問題点を検出する手法及びその自動化ツール reqchecker を提案する．提案手法においては，要求仕様書全体と要求文の分析，更に要求文間の関係分析を行い，仕様書中の問題点を検出する．自動化ツール reqchecker は，非あいまい性など六つの品質特性に関する問題点を検出し，その結果をマーキングしたHTML を出力する．例題への適用及び被験者実験により reqchecker の有用性に関する予備評価を行ったところ，良好な結果を得た．本論文の貢献は，以下の 3 点である．

- 報告書 [7] が整理している品質特性項目について，それらの検出可能性及び利用する分析方法を，仕様書の語彙・構文的な分析に限定して整理したこと．
- 整理した分析方法に基づき，6 品質特性，14 品質特性項目に対して自動検出方法を与えたこと．
- 計算機による実装を行い，ツール reqchecker として広く利用可能としたこと．

reqchecker の特徴の一つに，IEEE 830 の品質特性の広い範囲の問題点を検出し，品質特性ごとに分類して出力する点がある．この品質特性は IEEE が定めた標準規格であり，その内容を解説する教科書も出版されている [3] ため，これらを向上させることの重要性は広く認知されている．実際に要求仕様書の品質検査を行う際には，仕様記述者は各品質特性が十分に満たされているかを精査する必要があり，その際には，検出された問題点がどの品質特性のどういった特徴に基づくものかの対応関係が明確である必要がある．reqchecker は問題点を品質特性ごとに分類して出力するため，こういった精査が行いやすい．また，よく知られた標準に従って問題点が分類されるため，問題の内容も把握しやすくなることが期待できる．このような形で構築されたツールは，我々の知る限り存在しない．

本論文の構成は以下のとおりである．まず **2.** では本論文で扱う要求仕様書の問題点とその例を説明する．**3.** で提案手法を，**4.** でその実装 reqchecker について述べる．**5.** ではツール reqchecker に対する予備評価の結果について述べる．**6.** で関連研究を紹介し，最後に **7.** で本論文をまとめ，今後の課題を示す．

2. 要求仕様書の品質特性

IEEE 830 [8] では要求仕様書に対して非あいまい性，完全性など全八つの品質特性を定義している．更に報告書 [7] では，各品質特性に対して計測可能な手段と品質特性項目を示している．例えば，語句の係り受け関係が複数あるような文は非あいまい性の低下につながるとしている．報告書は品質を低下させるような要求仕様書，要求文の具体例も含んでおり，例えば「返答していない PC メンバー及び論文リストを出力できること.」という文が前述の品質特性項目に該当するあいまいな文であるとしている．この文には「返答していない」という句が PC メンバーと論文リストの両方を修飾する解釈と，PC メンバーだけを修飾する解釈の 2 通りがある．この解釈の違いは文中での論文リストが返答していないもののみを指すか，返答のいかにかかわらず全てを指すかの違いとして表れる．報告書はこうした品質特性項目を 8 品質特性に対して 31 項目定義している(注1)．本論文ではここで示された品質特性項目に関する問題点を要求仕様書の問題点として扱う．

示された品質特性項目に関する問題点の要求仕様書からの検出において，我々が考慮した要件は以下のとおりである．

自動検出. 報告書には計測手段と品質特性項目が示されているが，検出の自動化やそのツールについては論じていない．要求分析者が計測手段に従って，手動で問題点を発見するコストは高い．要求分析者の分析コスト削減のため，本論文では検出の自動化を目指す．

品質特性項目の広い範囲のカバー. 既存の問題点発見手法は非あいまい性などの特定の品質特性や特定の品質特性項目に閉じた深い解析を行うもの [10]～[12] のみで，要求仕様書の問題点を広く扱う手法は著者らの知る限り提案されていない．要求分析者が複数の手法を組み合わせて利用することは現実的ではない．また，要求仕様書の品質向上を意識しやすくするため，検出された項目がどういった品質特性に基づいて導出されたかの関係性も合わせて出力することが望ましい．

自然言語で記述された仕様書を扱える. 自然言語で記述された要求仕様書を形式仕様に変換することで複数の品質特性項目の範囲を検出する手法もある [13] が，そのためには記述に制限を加える必要がある．一方，実際の要求仕様書の多くは制限のない自然言語で書かれており，制限に従わない記述を含む要求仕様書に対してはそういった手法は適用できない．

当然ながら全ての品質特性項目の自動化は難しいため，reqchecker は表層的に検出可能な項目をできるだ

(注1)：実際に報告書には 21 項目が示されている．ただし，その幾つかは複数の対象を併記しているため，これらを分けて扱い，31 項目としている．

表 1　検出対象の品質特性項目と利用する検出手法
Table 1　Problem types of each quality attribute and how they are detected.

品質特性	項目	品質低下要因	検出手法
非あいまい性	a	語句の係り受け関係が複数ある	構文の分析
	b-1	主語が省略されている	構文の分析
	b-2	目的語が省略されている	構文の分析
	c	指示語が含まれている	語句の分析
	d	判断基準があいまいな語句を含む	語句の分析
	e	範囲や境界を表す語を含む	語句の分析
完全性	c-2	用語の定義が記載されている	構成の分析
無矛盾性	b	定義が矛盾している	構文の分析，文間関係の分析
検証可能性	a	定性的な表現を含む	語句の分析
変更可能性	a-2	索引が付けられていない	構成の分析
	b	同じ要求が 2 箇所以上に現れる	文間関係の分析
	c	要求が互いに依存している	文間関係の分析
追跡可能性	a	章節番号が付与されていない	構成の分析
	c	各要求にユニークな番号が振られていない	構成の分析

け網羅することを目指す．

3. 提 案 手 法

3.1 概　　要

提案手法の概要を図 1 に示す．提案手法の入力は自然言語である日本語で書かれた要求仕様書である．ここで要求仕様書は章節構造になっており，各章節は要求文の集合で構成されていることを想定している．出力は入力された要求仕様書から検出された品質特性項目に関する問題点のリストとなる．

提案手法では大きく語句の分析，構文の分析，文間関係の分析，構成の分析の四つにより問題点の検出を行う．語句の分析と構文の分析は単一の要求文を分析するもので，入力となる要求仕様書から抽出された各要求文に対して実行される．一方，文間関係の分析と構成の分析は要求仕様書全体を分析するもので，仕様書そのものが入力となる．ただし文間関係の分析では，各要求文の分析結果として構文の分析で行うパターンマッチの結果と形態素解析の結果を利用する．

表 1 に提案手法でカバーする品質特性項目とその検出手法の対応関係を示す．各品質特性は複数の品質特性項目をもっているため，以降では便宜上，品質特性項目に「非あいまい性 a」のようなラベルを用いる(注2)．提案手法では前章で述べた計測可能な品質特性項目全 31 項目のうち 14 項目（45.2%）を，品質特性数としては全 8 のうち 6（75%）をカバーしている．「妥当性」と「重要度と安定度のランク付け」の 2 特性は，対応する品質特性項目に要求仕様書単体から検出できるものがなかったため，カバー範囲から外れた．妥当性の検出のためには，問題領域で正しいことが何かを仕様書外より与え，それに基づき各文の正しさを推論する必要がある．また，重要度と安定度のランク付けに関しても，どの要求項目にランクが付けられるべきかを仕様書外より与える必要がある．いずれも，要求仕様書単体の自然言語記述のみからは特定が難しい．

(注2)：表においてラベルが連番となっていないのは，提案手法で検出しない品質特性項目を省いているためである．*-1，*-2 などの枝番は，複数の対象を併記した項目を分けたことを表す．

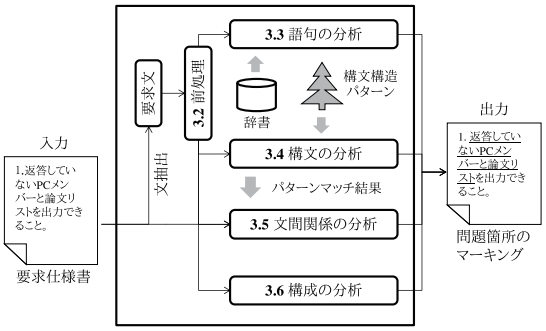


図 1　提案手法の概要
Fig. 1　Overview of our technique.

提案手法で利用している各種の分析は，日本語に特化したものではなく，英語をはじめとした多くの他言語の仕様書にも適用可能と考える．提案するシステムでは，類義語語彙情報，係り受け関係の解析及び解析結果に対するパターン，索引や章番号といったラベル情報のみを使用しているため，対象としている言語においてこれらを活用できる自然言語解析ツール基盤があれば，提案手法を適用可能と考える．

3.2 前　処　理

前処理として各要求文に付与されている識別番号を特定する．識別番号は数列として文頭に記述されてい

るとみなし，正規表現を用いて検出する．識別番号が検出できた場合，要求文本体からは取り除く．

3.3 語句の分析

語句の分析では，要求文中の語句から問題となり得るものを検出する．この検出は，語幹処理，類義語の展開，辞書との比較の 3 ステップからなる．

まず，要求文を形態素解析して得た各形態素に対して語幹処理を行い，原形を得る．例えば，要求文「システムはユーザがストレスを感じない範囲において反応すること.」は「システム/は/ユーザー/が/ストレス/を/感じ/ない/範囲/において/反応/する/こと/.」と形態素に分割される．また，その中の語句「感じ」はその原形「感じる」に変換される．

次に，語のゆれの問題を解消するため，既存の類義語辞書 [14], [15] を用いて語句をその類義語の集合に変換する．例えば，前ステップで得られた語「感じる」からは { 感じる，思う，···} が得られる．

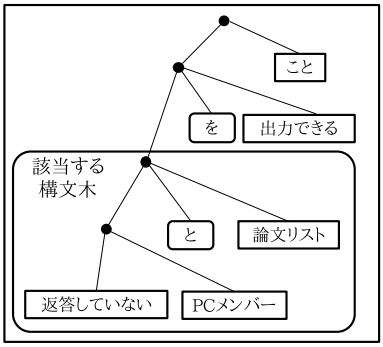
最後に，得られた類義語の集合と，あらかじめ辞書として作成しておいた品質特性の問題となり得る語句を比較する．辞書には，非あいまい性 c，非あいまい性 d，非あいまい性 e，検証可能性 a に対応するものとして，指示語，判断基準があいまいな語，範囲や境界を表す語，定性的な表現に基づく語句を収録している．辞書を構築する際，我々はまず，報告書 [7] にある例に用いられている語彙を候補とした．例えば，判断基準があいまいな語の例として「見栄え良く」が，範囲や境界を表す語を含む文の例として「扱える PC メンバーの数は 200 人までとする」が報告書に記載されており，これらに従い「見栄え」を非あいまい性 d の，「まで」を非あいまい性 e の辞書に含めた．更に，著者らの経験等に基づき語彙を拡充し，得られた語句の類義語も辞書に追加した．得られた類義語の集合の中に辞書中の語句があれば，もととなった形態素に問題があると判断し，これを検出する．例えば，前ステップで得た類義語の集合中の語「思う」は判断基準があいまいな語句の辞書に含まれるため，この文をあいまいと判断する．

3.4 構文の分析

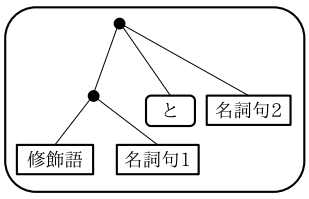
構文の分析では各要求文を構文解析し，得られた構文解析木に対してあらかじめ用意した構文構造パターンとマッチングを行うことで問題点を検出する．

まず，要求文に対して，文節特定と係り受け解析を行い，構文木を生成する．ここで，文節特定と係り受け解析は，既存の自然言語処理系を用いて行うことを想定している．例として，要求文「返答していない PC メンバーと論文リストを出力できること.」から得られた構文木を図 2 (a) に示す．この構文木は要求文を形態素解析したのち，一つ以上の形態素からなる文節を特定し，更にそれらの係り受け関係を解析して得たものである．この図で各ノードは句または助詞（連体化助詞，副詞化助詞を除く）を表現している．兄弟関係にあるノードを合わせて，意味のまとまりである文節，または文が構成される．

次に，あらかじめ用意しておいたパターンデータベース中の全ての構文構造パターンに対して，得た構文木がマッチするかどうかを調べる．構文構造パターンの例を図 2 (b) に示す．このパターンは複数解釈パターン 1 と命名されており，ある修飾語によって修飾された名詞句 1 と他の名詞句 2 が並立助詞「と」で結合されている文の断片を表しており，修飾語が構文構造のとおりに名詞句 1 のみを修飾しているのか，あるいは名詞句 2 も両方修飾しているのかがあいまいな場合を表現している．このように，パターンは構文木における具体的なノードそのものを指し示すこともできるし（「と」），あるノードの型のみを抽象的に指定することもできる（「修飾語」,「名詞句」）．また，マッチングの際には語幹処理も行っており，語幹の揺れにも対応している．構文解析木がパターンを部分木とし



(a) 構文解析結果



(b) 複数解釈パターン 1

図 2 パターンマッチの例
Fig. 2 Example of the patterm matching.

て含んでいた場合，該当パターンは構文木にマッチするという．図 2 (a) に示した構文木は，修飾語として「返答していない」，名詞句 1 として「PC メンバー」，名詞句 2 として「論文リスト」を対応付ければ，該当パターンを含んでいるとみなせるため，該当パターンにマッチしている．該当パターンは要求文に複数の解釈の可能性があることを示唆するものであり，この文は非あいまい性 a に問題点があると検出する．

パターンデータベースには，構造から要求文の種類を把握するためのパターンも含まれている．データベース中のパターンの概要を表 2 に示す．例えば，図 3 は入力の文が定義文かどうかを判定するパターンの一つである．これを用いると，要求文「30ヶ月目に出荷されるものはクリアンサと呼ばれる.」は,「30ヶ月目に出荷されるもの」が「名詞句 1」,「クリアンサ」が「名詞句 2」,「呼ばれる」が「動詞（呼ぶ）」に対応し，該当パターンにマッチするため，定義文とみなす．ここで得られた要求文の種類の情報は，次節で説明する文間関係の分析で用いる．

構文の分析による問題点の特定の精度は，利用する自然言語処理系の構文解析精度に依存している．例えば複文や重文などの複雑な文が与えられた場合，構文解析結果の係り受け関係に誤りが生じる可能性があり，誤った係り受け関係に基づいて特定した問題点は誤検出となり得る．一方，そういった複雑な文においても，構文解析によって係り受け関係が正しく特定できた範囲においては，通常通り構文の分析による問題点が検出される．

表 2　定義されたパターン
Table 2　List of defined patterns.

品質特性項目	名称	概要
非あいまい性 a	複数解釈 1	“A の B と C” 形
	複数解釈 2	“A の B・C” 形
	複数解釈 3	“A 及び B、C” 形
非あいまい性 b-1	主語省略	動詞に対する主語の欠落
非あいまい性 b-2	目的語省略	能動詞に対する目的語の欠落
無矛盾性 b	定義文 1	“A は B と呼ぶ” 形
	定義文 2	“A が B である” 形
	定義文 3	“A とは B である” 形
	定義文 4	“A とは B をさす” 形

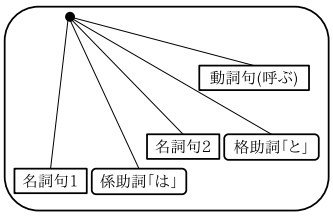


図 3　定義文パターン 1
Fig. 3　Pattern 1 of definition statements.

3.5　文間関係の分析

文間関係の分析では要求文と要求文の間にある関係から問題点を検出する．形態素解析により得られた形態素情報と，前節で得られたパターンマッチングの結果を用いて要求文間にある関係を推定し，問題となり得る関係があればそれを検出する．

無矛盾性 b「定義が矛盾している」は以下のプロセスにより検出する．まず，定義文の構文構造パターンとのマッチングにより定義文を発見する．次に，発見した定義文における定義語を抽出し，同一の定義語であれば二重定義とみなし，矛盾の可能性があると考え問題点として検出する．ここで定義語とは，定義文で定義される語を指し，構文構造パターンの特定の節にマッチした名詞句として特定する．図 3 に示した定義文パターン 1 では，名詞句 2 にマッチする語句を定義語とみなす．例えば,「例えば，熟成開始から 24ヶ月までの間に出荷する場合はクリアンサと呼ぶ.」「30ヶ月目に出荷されるものはクリアンサと呼ばれる.」の 2 文はいずれも「クリアンサ」の定義を行っているため二重定義であるが，これは前述のプロセスにより検出できる．

変更可能性 b「同じ要求が 2 カ所以上に現れる」は，類似する要求文に対して警告を行うもので，要求文を構成する語の集合の類似度が一定のしきい値 T_{sim} を超えたときに，同様の意味をもつ可能性があるとして指摘する．この検出のため，全ての要求文を形態素に分割し，それらの対に対して語の集合の類似性を計算する．

変更可能性 c「要求が互いに依存しない」では，要求の参照関係を特定し，相互依存の可能性として警告する．ここでは，要求文の意味により相互依存を引き起こしている場合も考え，一方向でも参照関係があれば，相互依存の可能性を検証すべき要求文として指摘する．要求の参照関係においては，一方の要求文中で，特定の要求文の番号を記して参照する場合と，指示語により他の要求文を参照する場合を考える．他の要求文の番号に相応する語あるいは辞書中の指示語が，要求文中の語の集合に含まれている場合に，他の要求文に関係があるとして検出する．

3.6　構成の分析

構成の分析では入力された要求仕様書中の章，節，

要求文の構成を把握し，品質特性に関わる問題点がその構成に存在するかを検出する．

完全性 c-2「用語の定義が記載されていない」では，「定義」などの特定の名称のついた章節見出しが存在するかを調べる．同様に，変更可能性 a-2「索引が付けられていない」でも，「索引」と命名された章節見出しが存在するかを調べる．

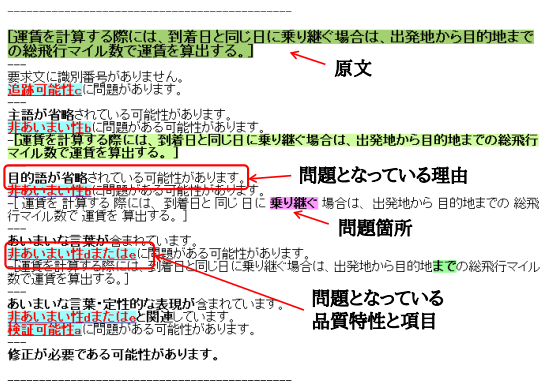
追跡可能性 c「各要求にユニークな番号が振られていない」は，得られた要求文のうち識別番号のないものを特定する．要求文「返答していない PC メンバーと論文リストを出力できること.」が入力された場合，この文の冒頭には識別番号が振られていないため，警告される．同様に，章節番号が付与されていない章節については，追跡可能性 a「章節番号が付与されていない」に概当すると考え，問題点として指摘する．

4. reqchecker の実装

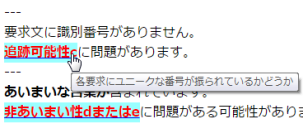
提案手法をツール reqchecker として実装した．入力された要求仕様書に対して問題点の検出を行い，検出された問題点をマーキングして出力を行う．reqchecker のソースコードは，GitHub にて，Apache License v2.0 のもと公開されている(注3)．

前工程で抽出した要求を理解している要求分析者が仕様書を修正することが適当と考え，reqchecker は問題点を示すまでに留め，提示された問題点を要求分析者が吟味し，必要に応じて仕様書を修正する．reqchecker の基本的な利用方法は以下のとおりである．まず，要求分析者は用意した要求仕様書を reqchecker に入力として与え，検出された問題点の一覧を出力として受け取る．次に，要求分析者は得られた問題点を検討する．問題点は，品質特性によって分類された形で，理由とともにまとめられており，要求分析者はこれらの情報をもとに個々の問題点を検討し，解消すべきと判断した問題点に対処すべく仕様書を修正する．要求分析者は修正後の仕様書を reqchecker に再度適用することにより，修正により問題点が正しく解消されたかを検討できる．このように，問題点の検出及び修正のプロセスを繰り返すことにより，要求仕様書の品質を向上していく．

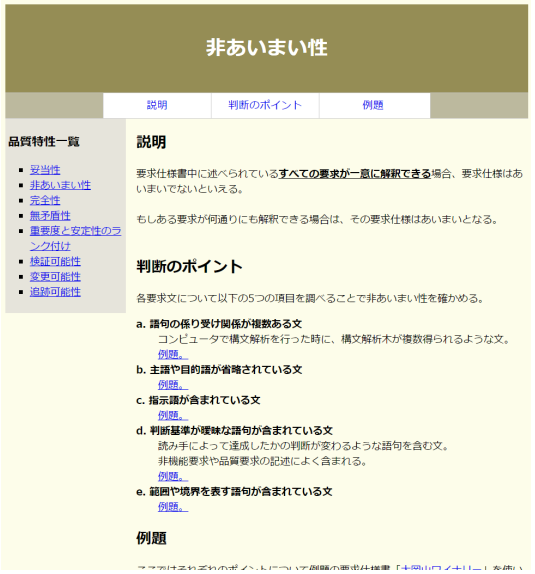
reqchecker は Java 言語により実装されている．入力となっている要求仕様書のテキストファイルを読み込み要求仕様書の構成を把握する．ここで，入力のテキストファイルは，章節の見出しや各文が行によって区切られているものを想定している．要求仕様が記述された章に対して，要求文と要求文間の関係を解析し，各品質特性項目に関する問題点を検出する．形態素解析，文節特定，係り受け解析には CaboCha [16], [17] を，類義語辞書には日本語 WordNet [14], [15] を用いた．語句の分析で用いている辞書は，現在のところ，4



(a) 問題点の一覧



(b) 品質特性項目の説明



(c) 品質特性の詳説

図 4 reqchecker の出力例
Fig. 4 Examples of reqchecker output.

(注3)：https://github.com/salab/reqchecker

品質特性項目合わせて計 360 語を収録している．変更可能性 b で用いる類似性のしきい値 T_{sim} には 0.5 を用いた．それぞれの検出方式が個別のクラスとして実装され，それらの組合せにより全体的な検出結果を構築しており，今後新しい検出方式を追加することも容易なよう構成されている．

reqchecker では要求仕様書に存在している品質特性に関する問題点をマーキングして出力する．マーキングには問題点を視覚的にわかりやすく表現するために HTML を用いた．問題点一覧の出力の例を図 4 (a) に示す．仕様書中の問題箇所は，背景色が変更されて表示されている．更に問題点の解説として，問題と判断した理由とどの品質特性のどの項目に対して問題があったのかを出力している．項目にカーソルを合わせると，図 4 (b) に示すようにその項目の解説が表示され，クリックすることでより詳しい品質特性の解説ページにジャンプできる．図 4 (c) は解説ページの例で，ここでは非あいまい性についての解説を表示している．

5. 予備評価

以下の観点で reqchecker の予備評価を行った．

Q1. reqchecker の問題点検出精度はどの程度か?
提案手法は構文に基づく自然言語解析によるものであり，辞書は利用しているものの文の意味等には踏み込んでいない．このような，比較的軽量な解析ではあるものの，要求仕様書中の問題を適度に発見できれば有用と考える．

Q2. 要求分析者は reqchecker を便利と感じるか?
ツールが正しく指摘を行っても，その結果を確認する要求分析者が結果を有効に使いこなせなければ要求仕様書の品質は上がらない．この問いでは，実際に reqchecker を被験者に利用させ，検出が分析に役立ったと感じたかをアンケート形式で問うた．

5.1 例　　題

用意した例題を表 3 に示す(注4)．これらは，工学部情報工学科におけるソフトウェア開発の演習で用いられている例題から集めたもので，14–15 の要求文（見出しを含む）から構成されている．この演習では，オブジェクト指向設計やプログラミングを理解した情報工学科の学生が，与えられた仕様書をもとに 2ヶ月程度で設計・実装を行うもので，学生に仕様書の問題点を指摘させるために，意味的な問題や文章の表層的な問題が意図的に混入されている．例えば,「直ちに」「次第」「簡単」などのあいまいな語句や係り受けが含まれている．こういった特徴が提案手法の予備評価の例題に適していると判断し，利用した．

表 3　評価に用いた例題
Table 3　Examples used for the evaluation.

記号	名称	要求文数
R_{winery}	ワイン予約システム	14
R_{airline}	航空チケット予約システム	15
R_{meeting}	会議室予約システム	14

(注4)：https://github.com/salab/reqchecker/tree/master/sample

実験に先立ち，例題それぞれに存在する問題点の集合（正解）を著者らのうち 2 名が分析して用意した．全ての品質特性項目に関する問題点を多く含んでいるわけではないが，様々な項目にわたる問題点を含んでいる．なお，完全性 c-2，変更可能性 a-2，追跡可能性 a は仕様書全体に対する問題点を検出するため，検出数も高々1 であること，単一の問題が品質に大きな影響を及ぼすこともあるため，頻出しない項目の重要性が低いわけではないことに注意されたい．

5.2 Q1 への回答

5.2.1 分析方法

Q1 には，例題として用意した要求仕様書を reqchecker に入力し，得られた指摘と正解とを比較することにより，適合率及び再現率を算出して答える．適合率 P と再現率 R は，要求仕様書中に存在する問題点の集合 C とツールの出力した問題点の集合 D を用いて以下のように求める．

$$P = \frac{|C \cap D|}{|D|}, \qquad R = \frac{|C \cap D|}{|C|}$$

5.2.2 結果と考察

得られた適合率，再現率を表 4 に示す．ここで，$|C|$ は問題点の個数，$|D|$ は reqchecker による問題点検出数を表す．また，適合率，再現率における NA は分母が 0 のため計算できなかったことを表す．

これから，以下のことがわかる．まず，NA であったものを除いて，13 項目中 8 項目で適合率が，14 項目中 7 項目で再現率が 0.8 以上だった．また，全体として，正解とされた 158 の問題点を適合率 0.60，再現率 0.71 で検出できた．品質特性ごとにみれば，適合率が 0.8 を上回った 8 項目中 4 項目は「非あいまい性」，2 項目は「追跡可能性」に関するものであった．また，再現率が 0.8 以上であった 7 項目のうち 2 項目は「非あいまい性」，2 項目は「追跡可能性」に関するもので

表 4　適合率と再現率
Table 4　Precision and recall values.

		R_{winery}				R_{airline}				R_{meeting}				全体			
品質特性	項目	$\|C\|$	$\|D\|$	P	R	$\|C\|$	$\|D\|$	P	R	$\|C\|$	$\|D\|$	P	R	$\|C\|$	$\|D\|$	P	R
非あいまい性	a	2	1	1.00	0.50	2	0	NA	0.00	1	0	NA	0.00	5	1	1.00	0.20
	b-1	6	5	0.80	0.67	6	1	1.00	0.17	7	5	1.00	0.71	19	11	0.91	0.53
	b-2	12	21	0.52	0.92	3	15	0.20	1.00	4	11	0.27	0.75	19	47	0.36	0.89
	c	5	5	1.00	1.00	3	1	1.00	0.33	4	4	1.00	1.00	12	10	1.00	0.83
	d	5	7	0.57	0.80	17	21	0.33	0.41	12	9	0.67	0.50	34	37	0.46	0.50
	e	2	2	1.00	1.00	2	1	0.00	0.00	5	2	1.00	0.40	9	5	0.80	0.44
完全性	c-2	1	1	1.00	1.00	1	1	1.00	1.00	1	1	1.00	1.00	3	3	1.00	1.00
無矛盾性	b	1	0	NA	0.00	0	0	NA	NA	0	0	NA	NA	1	0	NA	0.00
検証可能性	a	1	8	0.13	1.00	4	11	0.36	1.00	3	9	0.22	0.67	8	28	0.25	0.88
変更可能性	a-2	1	1	1.00	1.00	1	1	1.00	1.00	1	1	1.00	1.00	3	3	1.00	1.00
	b	1	0	NA	0.00	1	0	NA	0.00	0	1	0.00	NA	2	1	0.00	0.00
	c	3	3	0.67	0.67	2	1	1.00	0.50	3	3	0.67	0.67	8	7	0.71	0.63
追跡可能性	a	1	1	1.00	1.00	0	0	NA	NA	0	0	NA	NA	1	1	1.00	1.00
	c	14	14	1.00	1.00	14	14	1.00	1.00	6	6	1.00	1.00	34	34	1.00	1.00
合計		55	69	0.68	0.85	56	67	0.49	0.59	47	52	0.62	0.68	158	188	0.60	0.71

あった．このことから，reqchecker は特に非あいまい性，追跡可能性について高精度での検出能力をもっていることが示唆される．

検出に失敗する例もあった．非あいまい性 a について，「成田空港からアムステルダム、パリ、ロンドン」といった表現で，列挙部が「かつ」として結合しているのか，「または」として結合しているのかで複数の解釈がとれる要求文があったが，これに対応する複数解釈パターンを準備できておらず，検出に失敗した．無矛盾性 b に関しては，要求文中に「(」「)」が含まれていたために構文解析に失敗し，定義文における定義語が正しく特定できず，同一の定義語であると検出できなかった．変更可能性 b に関しては，ある要求文と同一の内容が他の要求文の一部に表れる場合（包含関係）を想定しておらず，正しく検出できなかった．そのうちの一つは，航空機の予約時に入力する情報について同じく述べた二つの要求文を検出すべきだったが，一方の要求文が更にユーザインタフェースについても述べていたため，要求文全体では語の類似度が高くならず，検出に失敗した．

5.3　Q2 への回答

5.3.1　分析方法

被験者 3 名を募り，2 種類の要求仕様書を配布し，それぞれ reqchecker を使用した場合と使用しなかった場合で，要求仕様書中に存在する品質特性に関する問題点の発見及びその修正を行わせた．被験者には，事前配布の資料による自習と講義形式での説明により，あらかじめ要求仕様書の品質特性に関する理解を促した．被験者は情報工学を専門とした学生 3 名であり，

表 5　被験者への割り当て表
Table 5　Assignment to the subjects.

	1 回目（不使用）	2 回目（使用）
被験者 A	R_{meeting}	R_{airline}
被験者 B	R_{meeting}	R_{airline}
被験者 C	R_{airline}	R_{meeting}

表 6　アンケート回答結果
Table 6　Result of questionnaire.

質問項目	回答	平均値
reqchecker が便利だったか	3, 4, 4	3.67
非あいまい性について役に立ったか	5, 5, 5	5
完全性について役に立ったか	4, 5, 5	4.67
無矛盾性について役に立ったか	2, 3, 4	3
検証可能性について役に立ったか	3, 4, 4	3.67
変更可能性について役に立ったか	2, 3, 5	3.33
追跡可能性について役に立ったか	3, 4, 4	3.67

表 5 に示すように作業を割り当てた．作業には 1 要求仕様書につき 40 分を与えた．

要求仕様書修正作業の実施後，各被験者にアンケート調査を行った．アンケートでは，各品質特性及び全体に対して reqchecker が役に立ったかを尋ねた．具体的には，1）reqchecker が便利だったか，2）非あいまい性，3）完全性，4）無矛盾性，5）検証可能性，6）変更可能性，7）追跡可能性について役に立ったかを 5 段階評価で尋ねた．また，選択式以外にも，reqchecker が作業に便利だったかなどを自由形式で尋ねた．

5.3.2　結果と考察

アンケート回答の結果を表 6 に示す．「回答」の列は 3 名の回答を昇順に並べたもので，「平均値」は三つの値の平均値を表している．全項目において回答の平均値は 3 以上となっており，reqchecker 全体についても

個々の品質特性についても好評価を得た．

特に，非あいまい性は平均 5，完全性は平均 4.67 と高い評価を得た．Q1 への回答での分析によれば，非あいまい性に関する指摘は 1 例題平均で 37 あり，そのうちの 53.1%が正しかった．指摘の数が多く，またその半数強が正しかった非あいまい性は，被験者によく受け入れられたものと考える．また，完全性については，reqchecker は仕様書が用語の定義の記載を含むかを調べるのみであったものの，役に立ったと評価された．これは，被験者が用語の定義の有無について意識しておらず，指摘により意識漏れに気付いたからだと考える．いずれも，該当の品質特性において reqchecker が特に有用だったことを示唆している．

自由形式の設問に対して，幾つか興味深い回答を得た．まず，reqchecker が作業に便利だったかどうかに対しては，「（たたき台として）便利だった」という回答を 1 名から得た．これは，作業の取っ掛かりとなる点をツールが示してくれることが便利だったと解釈できる．また，ほか 2 名も「（不便な点もあるが）概ね便利だった」と答えた．

一方，幾つかの懸念点も指摘された．まず，「検出されなかった部分の品質特性項目について疎かになる」という意見があった．これは，たとえツールの出力が不完全であり得るという認識があっても，ツールが検出箇所を強調表示するために，実際には強調されなかった部分の問題点に気付きにくくなるからと考える．また，検証可能性 a や変更可能性 c に関して「問題箇所がわかりにくい」という意見もあった．これら 2 項目は他とは異なり，他の品質特性で検出した問題点をそのまま項目の問題点として扱って出力するため，要求文中のどこが問題となっているかのマーキングを詳細に行わない．対応箇所をマーキングするような改善がツールの使用性向上に貢献すると考える．

5.4 議　　論

利用した例題はいずれも予約システムに関するものであり，他の問題領域の仕様書に対する適用可能性は本予備評価からは明確でなく，外的妥当性への脅威が残る．しかし提案手法は，問題領域固有の要求文の性質を用いておらず，使用している類義語辞書やパターンも問題領域によらない一般的なものである．したがって，我々はこの脅威は深刻ではなく，提案手法は他の問題領域の仕様書に対しても同様の適用可能性をもつと考えている．また，例題は公開されており，その適切性を第三者が検証することもできる．

しかし，利用した例題には無矛盾性 b「定義が矛盾している」に関する問題点が一つしか含まれておらず，reqchecker がその検出に失敗したため，提案手法における無矛盾性 b の検出可能性が不明確なまま残った．これを解消するため，矛盾した定義を含む例題に対して提案手法を適用した．この例は自然言語による仕様記述についての論文 [18] で紹介されているもので，「プログラムの入力は文字のストリームである．」「したがって，入力は区切りによって分けられた単語列であり，…」という文を含む．これらはいずれも「入力」を定義しているが，定義内容が異なるため矛盾している．実際に適用したところ，この二重定義を検出できたため，reqchecker は無矛盾性 b に関しても一定の検出能力を有していると考える．

現在のところ，reqchecker を用いることにより要求仕様書の品質を向上できるか，については検証ができていない．Q2 への回答で得られたデータは被験者 3 名分に過ぎず，これらを用いて定量的な評価を行うことは現実的ではない．今後，より現実的な評価を行っていきたい．また，よりツールが広く使われるよう，検出規則のチューニングを行っていきたい．

6. 関 連 研 究

要求仕様書の分析技術，問題点特定技術は幾つか存在するものの，日本語自然言語で記述された要求仕様書の問題点を，要求仕様書の品質特性と関連付けた形で特定し出力するツールは，著者らの調べた限り公開されていない．

自然言語で記述された要求仕様書を形式仕様に変換することで，矛盾やあいまいさを検出する手法がある [13]．この手法では自然言語記述を形式仕様に変換するために一度 UML 等に書き直す必要があり，そのためには，制限された自然言語により要求仕様が記述されている必要がある．文に使える記述が制限されてしまうので，本論文が扱う問題とは異なっている．また，要求仕様書をモデル化することにより形式的に問題点を検出する手法もある [19]～[21] が，これらも同様に本手法とは扱っている問題が異なる．

自然言語処理によりあいまいさを自動検出する研究は複数ある [11], [12]．例えば，修飾関係のあいまいさの問題点 [11] や代名詞のあいまいさの問題点 [12] を自動検出する手法がある．また Kiyavitskaya らは，自然言語で記述された文や語における様々なあいまいさについて論じ，メトリクスも提案している [10]．自然言

語の要求文に対して自然言語処理を施して問題を検出するというアプローチは我々と共通しているが，目指す方向性が異なっている．reqchecker は様々な種類の問題点を検出し提示することで要求分析者による問題点発見，分析を支援することを目指している一方，これらの手法はあいまいさに特化して精密な分析を行っている．

要求仕様書に限らず，一般的な自然言語文書に対して問題点を指摘するツールは既に存在している．例えば，日本語自然言語文の解析ツール ClearDoc がある [22]．このツールは，日本語文書中のあいまいな記述や複雑な文章をから特定でき，ソフトウェア仕様書に適用できる．しかし，検出項目の網羅性，IEEE 830 に基づく問題点の分類を与えない点などが reqchecker とは異なる．一方，ClearDoc ではあいまいさのメトリクスが定義されており，その値で文を並べ替えるなど，問題の度合いを活用できる機能がある．reqchecker にもこういった機構を導入することは有益と考える．

提案手法は要求仕様書の自然言語記述を分析して問題点を特定するが，同様の処理は，要求仕様書に限らず，自然言語により表現された要求記述一般にも適用できると考える．すなわち，表現に自然言語を用いるような要求獲得法，例えばインタビューからの要求抽出法 [23], [24]，ゴール記述に自然言語を用いるゴール指向分析法 [6], [25]，シナリオ分析法などの結果にも提案手法の分析の一部を適用可能と考える．これらの手法は獲得される要求の完全性の向上などに貢献するため，要求記述の語彙・構文的な特徴のみに基づき仕様書の品質向上を支援する提案手法と組み合わせることにより，得られる要求仕様の品質をより向上できる．

提案手法における仕様書の分析は特定の問題領域に特化しない一般的なものである一方，問題領域固有の知識を用いて要求の品質を向上させる手法もある．例えば，問題領域の語彙や語彙間の関係，それらの上の推論規則をオントロジーとして知識化して活用することにより，欠落した要求の補填などを支援するツールが提案されている [26]．同様に，問題領域の語彙やそれらの関係をまとめたシソーラスを用いた要求獲得法とその支援ツールも開発されている [27]．このような問題領域の知識も要求の完全性向上などに利用できる．

7. むすび

本論文では日本語要求仕様書にある品質特性に関する問題点の検出手法を提案し，それを自動化するツールとして要求仕様書のチェッカー reqchecker を実装した．例題への適用及び被験者実験により行った reqchecker の予備評価は良好であり，特に一部の品質特性項目については高い有用性が示唆された．

予備評価で示されたように，reqchecker には幾つかの問題点が残っている．これらの問題への対処など，今後も継続的なソフトウェア進化を促進していきたいと考えている．また，**4.** で述べたように reqchecker はオープンソースソフトウェアとして公開されており，有志による有益な拡張も歓迎する．

さしあたっての今後の課題を以下に示しておく．

- パターンの拡充等による検出可能な品質特性項目の範囲の拡大．
- 不要な問題点指摘の出力を抑制するコマンドの用意，マーキングの改善などによる使用性の向上．
- 仕様書規模，問題数，被験者数の拡大による，規模の大きな評価実験の実施．
- 品質特性に基づく出力の分類の有用性に関する実験的評価の実施．

文　献

[1] T. Clancy, “The standish group report,” 1995. Chaos report, 1995.

[2] 妻木俊彦，白銀純子，要求工学概論 —要求工学の基本概念から応用まで，近代科学社，2009.

[3] 大西　淳，郷健太郎，要求工学 —プロセスと環境トラック，共立出版，2002.

[4] G. Gabrysiak, H. Giese, A. Seibel, and S. Neumann, “Teaching requirements engineering with virtual stakeholders without software engineering knowledge,” Proc. 5th International Workshop on Requirements Engineering Education and Training, pp.36–45, 2010.

[5] G. Gabrysiak, H. Giese, and A. Seibel, “Why should I help you to teach requirements engineering?,” Proc. 6th International Workshop on Requirements Engineering Education and Training, pp.9–13, 2011.

[6] A. van Lamsweerde, “Goal-oriented requirements enginering: A roundtrip from research to practice,” Proc. 12th IEEE International Requirements Engineering Conference, pp.4–7, 2004.

[7] 要求工学ワーキンググループ，“要求定義で困ってませんか？要求仕様書の品質に関する研究成果報告，” 2007. http://www.selab.is.ritsumei.ac.jp/~ohnishi/RE/rewg-tr1v2.pdf

[8] The Institute of Electrical and Electronics Engineers (IEEE), “830-1998 – IEEE recommended practice for software requirements specifications,” 1998.

[9] A.M. Davis, Software Requirements: Objects, Functions and States, revised edition, Prentice Hall, 1993.

[10] N. Kiyavitskaya, N. Zeni, L. Mich, and D.M. Berry,

"Requirements for tools for ambiguity identification and measurement in natural language requirements specifications," Requirements Engineering, vol.13, no.3, pp.207–239, 2008.
[11] H. Yang, A. Willis, A.D. Roeck, and B. Nuseibeh, "Automatic detection of nocuous coordination ambiguities in natural language requirements," Proc. 25th IEEE/ACM International Conference on Automated Software Engineering, pp.53–62, 2010.
[12] H. Yang, A. deRoeck, V. Gervasi, A. Willis, and B. Nuseibeh, "Extending nocuous ambiguity analysis for anaphora in natural language requirements," Proc. 18th IEEE International Requirements Engineering Conference, pp.25–34, 2010.
[13] R. Drechsler, M. Soeken, and R. Wille, "Formal specification level: Towards verification-driven design based on natural language processing," Proc. Forum on Specification and Design Language, pp.53–58, 2012.
[14] 情報通信研究機構，"日本語 WordNet"．http://compling.hss.ntu.edu.sg/wnja/
[15] H.M.S. Wen, G.H. Eshley, and F. Bond, "Using WordNet to predict numeral classifiers in Chinese and Japanese," Proc. 6th International Conference of the Global WordNet Association, 8 pages, 2012.
[16] 工藤　拓，松本裕治，"CaboCha/南瓜: Yet another japanese dependency structure analyzer"．http://taku910.github.io/cabocha/
[17] 工藤　拓，松本裕治，"チャンキングの段階適用による日本語係り受け解析，" 情処学論，vol.43, no.6, pp.1834–1842, 2002.
[18] バートランド・メイヤー，"仕様をきちんと書くには，" IEEE ソフトウェア'88，pp.139–154, 岩波書店，1988.
[19] B.P. Erik Kamsties and D.M. Berry, "Detecting ambiguities in requirements documents using inspections," Proc. 1st Workshop on Inspection in Software Engineering, pp.68–80, 2001.
[20] M. Urbieta, M.J. Escalona, E.R. Luna, and G. Rossi, "Detecting conflicts and inconsistencies in web application requirements," Proc. 11th International Conference in Current Trends on Web Engineering, pp.278–288, 2012.
[21] S. Zafar, "A light-weight formal approach for modeling, verifying and integrating role-based access control requirements," Proc. 16th Asia-Pacific Software Engineering Conference, pp.257–264, 2009.
[22] PROVEQ, "ドキュメント検証ツール ClearDoc"．http://www.proveq.jp/verification/document/cleardoc
[23] 山中隆敏，埜口　元，谷藤史門，古宮誠一，"インタビューによる要求抽出作業を誘導する方法の提案，" コンピュータソフトウェア，vol.28, no.1, pp.230–247, 2011.
[24] J. Kato, S. Komiya, M. Saeki, A. Ohnishi, M. Nagata, S. Yamamoto, and H. Horai, "A model for navigating interview processes in requirements elicitation," Proc. 8th Asia-Pacific Software Engineering Conference, pp.141–148, 2001.
[25] A. van Lamsweerde, Requirements Enginieering: From System Goals to UML Models to Software Specifications, Wiley, 2009.
[26] M. Kitamura, R. Hasegawa, H. Kaiya, and M. Saeki, "A supporting tool for requirements elicitation using a domain ontology," Software and Data Technologies, CCIS vol.22, pp.128–140, 2008.
[27] 加藤潤三，佐伯元司，大西　淳，海谷治彦，林　晋平，山本修一郎，"要求獲得のためシソーラス構築支援，" 情処学論，vol.57, no.7, pp.1576–1589, 2016.

（平成 29 年 4 月 18 日受付，8 月 16 日再受付，
10 月 2 日早期公開）

林　　晋平　（正員）

2004 北海道大学工学部情報工学科卒．2006 東京工業大学大学院情報理工学研究科計算工学専攻修士課程了．2008 同専攻博士後期課程了．現在，同大学情報理工学院助教．博士（工学）．ソフトウェア変更やソフトウェア開発環境の研究に従事．

有賀　　顕

2011 東京工業大学工学部情報工学科卒．2013 同大学大学院情報理工学研究科計算工学専攻修士課程了．在学時は要求分析やソフトウェア開発教育の研究に従事．現在，株式会社ユー・エス・イー所属．

佐伯　元司　（正員）

1983 東京工業大学大学院工学研究科情報工学専攻博士後期課程了．同大学助手，助教授を経て，現在，同大学情報理工学院教授．工学博士．要求工学やソフトウェア開発技法等の研究に従事．