人工知能学会
第28回知識流通ネットワーク研究会
SIG-KSN-028-01

EARS と構造化分析を用いた要求レビュ手法の提案

山本修一郎
名古屋大学　名誉教授

東　弘之
株式会社　ベリサーブ

愛知県名古屋市千種区不老町

東京都千代田区

A Requirements Review Approach using EARS and Structured Analysis

Shuichiro Yamamoto
Nagoya University Professor Emeritus

Hiroyuki　Azuma
VeriServe　Corp.

Furo-cho, Chikusa-ku, Nagoya Aichi Japan

Chiyoda-ku, Tokyo

概要

プロセス仕様（PSpec）を自然言語で記述する構造化分析手法では自然言語記述の曖昧性が発生する．一方，要求構文を規定する EARS (Easy Approach to Requirements Syntax)では，処理やデータ間の関係を定義できないという問題があった．本稿では，要求仕様を EARS 構文で記述し，EARS 記述間の関係をデータフロー図で構造化することによる要求レビュ手法を提案する．また，EARS 構文に出現するデータ間の関係を分析する方法についても考察する．

Abstract

The Structured Analysis (SA) method of describing the process specification (PSpec) in natural language causes ambiguity in the natural language description. On the other hand, EARS (Easy Approach to Requirements Syntax), which defines the requirements syntax, has a problem that the relationships between processes are difficult to clearly describe.

In this paper, we propose the requirements review method consists of EARS and SA. The approach validates requirements statements by using EARS syntax and data flows in DFD diagrams.

1. はじめに

日本語で記述した要求仕様には，曖昧性があるため，システム開発の手戻りが発生するという問題がある．本稿では，要求構文を限定する EARS[1-3]と，データフローに基づいて階層的に機能要求を明確化する構造化分析手法(Structured Analysis, SA)[4]を統合することによって，要求レビュを効率化する手法を提案する．

以下では，まず 2 節で構造化分析と EARS，要求レビュを説明する．次いで，3 節で EARS と構造化分析を用いた要求レビュ手法を提案する．さらに，提案手法の具体的な適用事例を 4 節で説明する．5 節で考察を述べ，6 節でま

とめと今後の課題を述べる．

2. 関連研究

以下では，先行研究として，構造化手法，EARSならびに要求確認手法を説明する．

2.1 構造化手法

DeMarcoによる構造化手法では，データフロー図(Data Flow Diagram)を用いて入力データを出力データに変換する機能を表すプロセスを段階的に分解する．それ以上分解できない最下位のプロセスに対してプロセス仕様(PSpec)を記述する．PSpecでは自然言語を用いて入力データから出力データを生成する機能を記述する．

構造化分析では，データ辞書でデータ構造を定義する．データの構造を連接‘+’，選択‘|’，反復‘*’記号を用いて明確に記述することができる．

2.2 EARS

EARS（Easy Approach to Requirements Syntax）はロールスロイス社の Marvin らが提案している自然言語(English)で要求を記述するための簡易構文テンプレートである[1,2]. この方法は，要求定義の専門家ではない発注者（ステークホルダ）が 5 種類のテンプレート構文を用いることにより，自然言語による要求定義が持つ曖昧性や検証不能性などの多様な問題を回避できるように考案されたものである．Marvin らは，この手法を航空機エンジン制御システムの要求記述に適用することにより，英語を用いた要求記述の多くの問題を解消できたと報告している．5 種類の EARS 構文を表 1 に示す.

表 1　EARS 構文

EARS	構文
基本型	〈システム〉 が 〈処理〉する
事象型	〈期待事象〉が発生した場合，〈システム〉が〈処理〉する
例外型	〈例外事象〉が発生した場合，〈システム〉が〈処理〉する
状態型	〈システム〉が〈状態〉にある限り，〈処理〉する
選択型	〈システム〉に〈性質〉がある場合，〈処理〉する

2.3 要求確認

要求のレビュには，チェックリスト，欠陥分類，シナリオ，ステークホルダの立場，ユースケースに基づく手法などがある[5].

IEEE std.830[6]では，要求仕様が持つべき特性として，正当性，無曖昧性，完全性，一貫性，順位付け，検証容易性，修正容易性，追跡性 を提示している．

ISO/IEC/IEEE std.29148[7]では，個別要求特性と要求集合特性を提示している．個別要求特性には，①必要性，②実装自由性，③無曖昧性，④一貫性，⑤完全性，⑥単一性,⑦実現性，⑧追跡性，⑨検証性がある．要求集合特性には，①完全性，②一貫性，③獲得容易性，④限定性がある．セーフウェア(Safeware) [8]では，要求仕様に含まれている情報が，ソフトウェアの望ましい挙動と望ましくない挙動を設計者が判断する上で不十分であるとき，要求仕様があいまいであるとしている．あいまいでない要求仕様が完全であることになる．

システムが動作する状況下で，ソフトウェアの安全な挙動を規定するために要求仕様が完全である必要がある．

要求の測定指標として，SMART が知られている．TOGAF[9]では，目標が達成可能で測定可能であるように定義されていることを保証する方法として，SMART を(Specific, Measurable, Actionable, Realistic, Time-bound)としている．ISACA[10]の SMART は特定された(Specific)，測定可能な(Measurable)，達成可能な(Attainable)，現実的な(Realistic)，タイムリー(Timely)である．

Kotonya と Sommerville[11]が VORD(Viewpoint-oriented Requirements Definition)を提案している．VORD はユーザ課題と組織の関心事に着目するサービス指向手法である．VORD では顧客がシステムサービスと直接対話する直接的視点と，顧客がシステムサービスと直接対話することのない間接的視点を区別する．間接的視点にはシステム開発の視点，システムが組織に与える影響の視点，外部環境に与える影響の視点などがある．

Leite と Freeman は VORV(Viewpoint-oriented Requirements Validation)を提案している[11]. VORV では，視点（Viewpoint）は対象を見ている場所である．パースペクティブ(Perspective)は現実の特定の側面に従って観測された事実の集合である．ビュウ(View)は視点とパースペクティブを統合として定義される．VORV では，まずデータ，プロセス，アクタからなるパースペクティブを is-a と part-of 関係でビュウを洗練することにより，不一致点のリストを作成する．次に，これらのパースペクティブをビュウとして統合する．

IPA の非機能要求グレード[12]では，「非機能要求」について，ユーザと開発者との認識の行き違いや，互いの意図とは異なる理解を防止することを目的として，非機能要求を網羅的にリストアップして分類している．非機能要求グレードによって，重要な非機能要求から要求レベルを段階的に設定して非機能要求を確認できる．

大林ら[13]は要求逸脱分析表を用いて要求をテスト設計者の立場からレビュする方法を提案している．要求逸脱分析表では，要求仕様ごとに，パラメータ，ガイドワード，逸脱とその重大性を表形式で網羅的に分析することができる．パラメータには，システム，対象，イベント事象，入力，出力，機能，応答事象を記述する．パラメータ逸脱を分類するガイドワードには，なし，以外，部分，冗長，遅い，早いなどがある．テスト項目を選択する基準として重大性の高さを用いる．

Anu[14]らは，人的な要求仕様の誤りを，1)誤記，2)一貫性の欠如，3)要求仕様標準やテンプレート等の誤用，4)文法誤りに分類している．Walia と Carver による要求誤り分類(RET, Requirement Error Taxonomy)では，要求誤りを人的誤り，手段誤り，文書誤りに分類している[15]．Anu[16]らによるプランに基づく人的誤りの定義では，正しいプランの適用を不注意で間違えた lapse, 正しいプランの適用を記憶違いで間違えた slip, 間違ったプランを選択した mistake に分類して, 要求プロセスにおける人間行動の誤りを整理している．要求文書の人的誤りは次の通り．1)Lapse:ユーザ要求の転記ミス，用語誤り，2)Slip:ユーザ要求の抜け，同義語，3)Mistake:誤解，要求標準誤り，要求プロセス誤り，文法誤り，前提誤り，NFR 抜け．また，要求確認のための人的誤りとしては，確認知識が不完全，確認知識の適用誤り，レビュアの役割知識の不足，非機能要求誤りの識別知識が不十分などを指摘している．

Porter ら[17]は要求仕様書のレビュシナリオに基づく要求確認がチェックリストよりも優れていることを実験的に明らかにしている．要求確認シナリオには，データの一貫性確認，不正機能確認，機能欠落確認がある．

Basili ら[18]は，要求文書の Perspective-Based Reading (PBR)の有効性を実験的に確認している．PBR の前提は，設計者，試験者，利用者など異なる役割の経験者が，それぞれの成果物の作成可能性に基づいて要求文書を評価することにより，広い範囲で要求文書を確認できることである．

林[19]らは熟練者の知見に対応付けた要求整理モデルを用いて設計上の懸念事項を要求仕様書から摘出する手法を提案している．

2.4 開発文書確認

Parnas と Weiss[20]は，設計レビュのためのレビュプロセス，レビュ体制と役割，レビュの視点を提案している．とくに，レビュ参加者の役割と確認項目の関係を定義した．具体的には，①装置専門家，②装置プログラマ，③航空機プログラマがそれぞれ，①前提の妥当性，②前提の十分性とアクセス機能の正確性，③前提と機能の一貫性を確認する必要があるとした．

Gilb と Graham[21]は，矛盾がないこと，明確であること，具体的であることなど，15 個のソフトウェア開発文書規則を提示している．

コードのための直行欠陥分類 Orthogonal defect classification computational code (ODC-CC)[22]では，誤りを欠落 (missing)，誤り (wrong)，不要 (superfluous)，矛盾 (inconsistent)，曖昧(obscure)に分類している．

ユーザ文書の確認項目をデータ入力，データ出力，エラーメッセージ，データ項目間の関係である．また開発対象の確認項目は，機能記述，デフォルト，前提，限界，事前条件，事後条件である．

3. EARS＋SA による要求レビュ手法

EARS と SA による要求レビュ手順を図 1 に示す. まず，レビュ対象の要求仕様を EARS 構文で書き直す．この定型化過程で，機能要求の曖昧性を摘出して修正できる．次に，EARS によって定型化された機能要求文に基づいて SA を適用することにより，DFD とデータ辞書を作成する．この過程でデータフローとデータ辞書に出現するデータ項目の不一致を摘出して修正する．

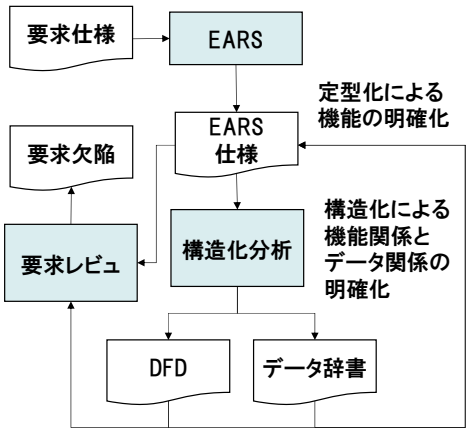

図 1　提案手法

4. 具体例

以下では提案手法を「酒屋の在庫問題」[23-25]に適用する．

4.1 酒屋の在庫問題

酒屋問題の機能記述は以下の通りである．この機能記述は「酒屋問題」そのものではなく，玉井[25]によるユースケース記述による．

P1)顧客からの注文依頼が発生した場合, システムが注文を受付ける
P2)注文の在庫状況をシステムに照会する
P3)システムは，在庫あり注文であれば，受付係に提示する
P4)注文が在庫不足であれば, 受付係に在庫不足を提示する
P5)システムが，未出荷注文であることを記録する
P6)積み荷票をシステムに登録する
P7)システムは入荷により在庫不足注文の在庫不足が解消された場合，受付係に通知する
P8)在庫不足解消注文に対する出庫指示を通知する
P9)注文依頼に基づき，出庫指示書を作成する
P10)出庫指示書を倉庫係に通知する
P11)出庫指示により空きコンテナが生じる場合，空きコンテナ番号を倉庫係に通知する
P12)システムが在庫データを更新する

4.1 EARS による要求文の確認

酒屋問題の記述(P1-P12)に対する EARS の適用結果は以下の通りである．ここで，EARS を適用するために，機能記述の表現を書き直していることを注意しておく．機能の表現を変更した場合「修正前＝＞修正後」で示した. また，適用した EARS 構文の種別を[EARS 型]で示している．

P1)顧客からの注文依頼が発生した場合, システムが注文を受け付ける　[事象型]
P2)注文についての在庫状況をシステムに照会する　＝＞注文照会があった場合，システムが在庫状況を提示する　[事象型]
P3)システムは，在庫あり注文であれば，受付係に提示する＝＞在庫のある注文である限り，受付係に提示する　[状態型]
P4)注文が在庫不足であれば，受付係に，在庫不足を提示する
＝＞　在庫不足の注文である限り，受付係に在庫不足を提示する　[状態型]
P5)システムが，未出荷注文であることを記録する
＝＞　在庫不足の注文である限り，未出荷注文であることを記録する　[状態型]
P6)積み荷表をシステムに登録する
＝＞システムが，積み荷表を登録する　[基本型]
P7)システムは在庫不足注文が, 入荷により在庫不足が解消された場合，受付係に通知する
＝＞入荷により在庫不足注文の在庫不足が解消された場合，受付係に通知する　[事象型]
P8)在庫不足解消注文に対する出庫指示を通知する　[基本型]
P9)注文依頼に基づき，出庫指示書を作成する
＝＞注文依頼を受け付けた場合，出庫指示書を作成する　[事象型]
P10)出庫指示書を倉庫係に通知する　[基本型]
P11)出庫指示による空コンテナが生じる場合，空コンテナ番号を倉庫係に通知する　[事象型]
P12)システムが在庫データを更新する　[基本型]

なお，酒屋問題に対する EARS の適用では，[例外型]と[選択型]を使うことはなかった．

4.3 構造化分析

EARS に基づいて作成した初期 DFD を図 2 に示す.

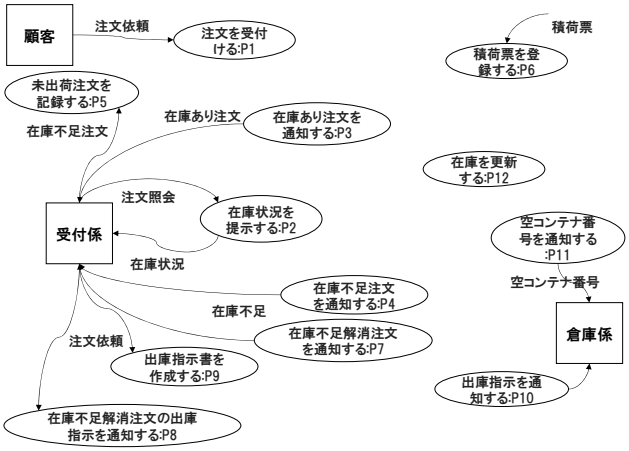

図 2　酒屋の在庫問題の初期 DFD

EARS 記述だけでは，システムが管理するデータを明記していないので，この DFD ではデータストアが欠落している．したがって，要求レビュでは欠落しているデータストアを補完して DFD を改訂する．
また，データ辞書は以下の通りである．

積荷票＝コンテナ番号+日付+(品名+数量)*
出庫依頼＝品名+数量+送り先
出庫指示書＝注文番号+送り先+
(コンテナ番号+品名+数量+空コンテナマーク)*
在庫不足リスト＝送り先＋品名＋数量
出庫依頼票＝品名＋数量＋依頼者
在庫＝(品名+在庫量)*+
(コンテナ番号+(品名+数量)*+空コンテナマーク)*

4.5 要求レビュ

(1)データストア記述の欠落

まず，初期 DFD にデータストアを追加する．改訂した DFD を図 3 に示す．

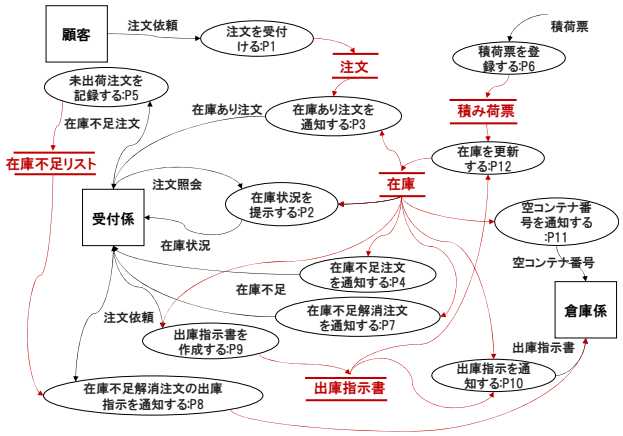

図 2　酒屋の在庫問題の改訂 DFD

データストアはシステムの要素であるから，EARS 記述では，データストアに対して選択型で次のような記述を追加する．
注文を保持する場合，注文を管理する
ここで，日本語としての自然さを考慮して選択型の「持つ」を「保持する」に変更した．

(2)機能記述の不足

依頼された酒が在庫不足かどうかをシステムが提示するためには，倉庫内のコンテナにある銘柄ごとの在庫量を知る必要がある．このためには，コンテナ搬入時に，銘柄ごとの在庫量を算出する必要があるが，この記述がない．
依頼された酒が在庫不足でなければ，在庫があることはわかる．しかし，出庫指示書を作成するためには，依頼された酒を格納する複数のコンテナから，依頼された数だけの酒を出庫するために，どのコンテナを選ぶかを判断する必要がある．
また，空コンテナをシステムが検出するのであれば，その記述が必要である．どのようにしてコンテナを空にするかは出庫指示に依存するので，依頼された酒を格納するコンテナが複数ある場合，どのコンテナを選択するかを定義する必要がある．
出庫指示書をシステムが作成するのであれば，どのようにしてコンテナを空にするかを含めてコンテナの選択基準を明確に定義する必要がある．
たとえば，以下のようにコンテナ出庫条件を定義する必要がある．
【コンテナ出庫条件】
C1)日付が最も古いコンテナを出庫する
C2)積載本数が少ないコンテナを出庫する
C3)積載本数が 0 なら，空コンテナマークを付ける
C4)リコール商品を出庫する
C5)期限切れ商品を出庫する
また，出庫指示書作成可能な限界を超えて大量の注文を受付けることはできない．「酒屋の在庫問題」では，受付係が注文受付と出庫指示の通知を担当しているので，大量の注文受付が発生すると，出庫指示を出せなくなる可能性が高い．
P12 では、「システムが在庫データを更新する」とあるが、在庫データの更新条件が不明確である。出庫するまでは、その銘柄は倉庫にあるから在庫数は減少しないと考えられる。また、出庫指示書を作成した時点で在庫数を削減する可能性もある。しかし、この場合、注文中だが出庫はまだという仕掛中の在庫の扱いが問題になる。

(3)機能の曖昧性

次の 3 機能の関係が曖昧である．
・在庫不足解消注文の出庫指示を通知する:P8
・出庫指示書を作成する:P9
・出庫指示を通知する:P10
たとえば，P8 で出庫指示書を作成することが明記されていない．

(4)データの曖昧性

次のようなデータの曖昧性がある．
・送り先と依頼者の関係が曖昧である
・注文番号を作成する契機がない
・商品の製造年月日を考慮していない
・商品の消費期限が不明である

(5)例外要求の欠落

「酒屋の問題」の EARS 記述では，例外型が出現しなかった．この理由は，「酒屋の問題」記述で例外が考慮されていなかったためである．EARS の例外型に対する機能要求の抽出例を以下に示す．
【入庫例外】倉庫に搬入できるコンテナ数の上限が不明である．無限にコンテナを搬入できる倉庫はない．
【在庫例外】特定銘柄の在庫がない場合，倉庫がコンテナで一杯になっていると，空コンテナを搬出するまで必要な銘柄を搭載する新たなコンテナを搬入できない．
【出庫例外】注文を受付けてから出庫するまでの時間制約が不明であるため，所定時間内に出庫指示できない可能性がある．
【注文例外】搬入予定がない銘柄の注文への対応取扱い銘柄を管理していないから，在庫がない銘柄で搬入予定がある銘柄についての注文を受付けることができない．
【注文取消し】注文の取消し機能がない
現実には注文の取消しが発生する可能性がある．注文されたが，出庫指示中や出庫指示後，倉庫から出庫後に，受付けた注文が取消されることへの対応が必要である．注文が取消されるもう一つの可能性として，商品の消費期限が切れた場合や商品がリコール対象になった場合がある．これらの問題に対応するためには，商品の名称や製造年月日を管理する必要がある．
【機能の不一致】出庫指示書の作成と倉庫係への出庫指示の通知の不一致が生じる可能性がある．たとえば，作成した出庫指示書が倉庫係に通知されない可能性がある．
【データの不一致】積荷票を登録したが，在庫更新が抜けていると，物理的な在庫があるにも，システム上は在庫不足が発生する可能性がある．
【システム障害】システム障害の状態についての要求

が欠落している．たとえば，次の記述が必要にある．
　システムが障害状態にある限り，障害対応する．
この記述に基づいて，システムの障害状態をどのようにして検知するかについての要求欠落があることを検出できる．

(6)コンテクストの欠落
　コンテナと積荷票を酒屋に納入する搬入元が不明である．要求記述では「コンテナが搬入される」ことが記述されているだけである．また，前述したように，倉庫内に格納可能なコンテナ数が不明である．
　さらに，受付係と倉庫係には，勤務時間があるはずだから，業務時間内に完了できない仕事についての扱いが不明である．たとえば，受付係の場合，出庫指示書を作成したが，倉庫係への出庫指示の通知が済んでいないかもしれない．このような仕掛中の仕事についての要求が不明である．

(7)データの一貫性
　仕掛中業務の発生やシステム中断が発生した場合，注文，積荷票，在庫，出庫指示書などのシステムで管理されるデータ間の一貫性を確認する機能が必要である．しかし，要求記述にはこのようなデータの一貫性を保証する機能の記述がない．

(8)人間による過失の扱い
　システム障害の原因の多くは，システム運用における人間作業の過失である．しかし，「酒屋の在庫問題」では，運用における過失対策についての要求記述が不明である．

5. 考察
　上述した方法についての考察を以下で説明する．

5.1 有効性
　提案した要求のレビュ手法を，対象とした酒屋の在庫問題に効果的に適用できた．EARS と SA を用いたことにより，要求文とその依存関係を効率的にレビュできることを明らかにした．
　EARS では，データストアを明確に識別していない．EARS 記述から DFD を作成することで，データストアの欠落が認識できるので，EARS 記述の目的語（名詞）から管理すべきデータ項目の欠落を摘出できる．EARS の選択型テンプレートを用いて，データストアをシステムが持つことから，データストアの管理処理の欠落を摘出できる．
　また，EARS の例外テンプレートを用いることにより，例外要求の欠落を摘出できる．
　提案手法では，要求文とその依存関係の確認を分離できるため，要求確認を系統的に実施できることから確認経費と工数を低減できる可能性がある．

5.2 適用性
　本稿で示したように，EARS は日本語文章にも適用できる．また SA はデータとその変換を分析するので入力を出力に変換するシステムに適用できる．したがって，本手法の適用範囲は広い．

5.3 文法要素と EARS 型
　EARS 型に，文法要素である主語，述語，目的語を対応付けると，表 2 の通りである．
　EARS 型の文で，主語，述語，目的語を識別することで必要な EARS 型の文を抽出できる可能性がある．
　たとえば,「システムが注文を受付ける」という文では，「注文」が目的語である．システムに「注文」というデータがあることで,「システムが注文を持つ場合，注文を管理する」という選択型の文が必要であることが分かる．また，選択型 EARS 文の結果節の動詞からシステムの機能を抽出できる．さらに，動詞がシステムの状態を変化させることに着目すると，EARS 文の動詞からシステムの状態を抽出できる可能性がある．

表 2　文法要素と EARS 文型

文法要素	主語	動詞	目的語
EARS 文型	状態型, 選択型	基本型,事象型, 例外型	状態型, 選択型
開発対象	システム	機能，操作	データ

　「酒屋問題」で見たように，要求は部分的に記述されていることが多い．EARS 型を構成する主語，動詞，目的語に着目することにより，必要な要求文を追加して要求記述を補完できる．今後，文法要素に基づく EARS 文の補完手法を明らかにする必要がある．たとえば、以下の手順で、EARS 記述の初期要求から反復的に最終的な抜けのない要求を獲得できる。
　【文法要素に基づく EARS 要求集合の補完】
　[手順 1]EARS で要求を記述した文の集合を E とする。最終的な要求集合 R を E とする
　[手順 2]E の要素文から、主語、述語、目的語の集合 S(E),V(E),O(E)を抽出する
　[手順 3]
　S(E)の要素の状態に対して、状態型の要求文 $E_{S(E)}$を作成する
　V(E)の要素の状態に対して、事象型、例外型の要求文 $E_{V(E)}$を作成する
　O(E)の要素の状態に対して、選択型の要求文 $E_{O(E)}$を作成する
　[手順 4] $E_{S(E)}$、$E_{V(E)}$、$E_{O(E)}$のすべての要素が E の中にあれば、それ以上、追加すべき要求文がないので、終了する
　[手順 5] E の中にない $E_{S(E)}$、$E_{V(E)}$、$E_{O(E)}$の要素の集合を E とする。R を E と R の和集合 E∪R とし[手順 2]へ戻る
（手順終わり）
　この手順で初期要求に追加される要求記述は要求記述の抜けである。すなわち、この手順は、要求抜けを摘出するレビュ手法であると考えられる。

5.4 レビュのスコープ
　丸山[24]が指摘したように,「設計前に制約が確定しているわけではない」という点では,「酒屋の在庫問題」でも，本稿で明らかにしたように，EARS における例外制約についての記述が不足している．つまり，品質に影響を与える機能要求を確定するためには要求の確認作業を反復する必要がある．この反復作業を本稿で提案した手法で実施できることが判明した．要求の確認作業を要求レビュで実施することができれば，要求を実装しなくても発見できる要求制約がある．つまり，実装後に要求制約の誤りを検出して要求を変更するという手戻りを排除できることになる．
　1980 年代に設計法を比較するために提案された「酒屋の在庫問題」では，受付係が「注文受付」と倉庫係への「出庫指示」を担当している．もし，大量の「注文受付」が発生すると,「出庫指示」を出せなくなる可能性が高いことが分かる．このように,「酒屋の在庫問題」を業務プロセスの問題として捉えることもできる．
　「酒屋問題再考」では，開発工程や意思決定の視点を考慮した,「酒屋問題」に変わる新たな共通問題を作成することが提案されたが，前述したように「酒屋問題」をこれらの視点から再考することができる．また,「酒屋問題再考」では，業務プロセスを含むビジネスアーキテクチャについて言及していない．ビジネスアーキテクチャの視点から

「酒屋問題」を分析することも興味深い．

ビジネスモデルとしては，倉庫を持たないビジネスモデルも考えられる．また，顧客管理，推奨商品案内，適正在庫の予測などが重要になっている．アーキテクチャ設計やテスト設計を容易化する要求仕様かどうかを確認することも重要である．

5.5 限界

本稿では，EARS と SA を統合した要求レビュ手法を提案した．しかし，酒屋の在庫問題に対して適用しただけである．今後，他の事例にも適用するとともに，定量的な有効性の評価が必要である．また，酒屋の在庫問題への適用は，解説に基づいて筆者らが提案手法による要求レビュを後付けで実施している．要求レビュへの提案手法の適用性を客観的に評価するためには，提案手法を実際のプロジェクトの初期段階から適用して評価する必要がある．

また，本稿では本手法が要求レビュに対する最適な手法であるかどうかについては評価していない．したがって，他の要求レビュ手法と比較して有効性を明らかにする必要がある．たとえば，既存の要求レビュ手法と本手法を定量的に比較評価する必要がある．

本稿では EARS と SA に基づく要求レビュで欠陥を摘出できることを明らかにした．しかし要求欠陥の分類については整理していない．今後，本手法で摘出できる要求欠陥を分類していく必要がある．また，この要求欠陥分類に基づいて，要求欠陥パターンをデータベース化できる可能性がある．要求欠陥パターンと要求記述の関係を明らかにすることができれば，要求欠陥を自動的に摘出できる可能性がある．

また、文法要素に基づく EARS 要求の完備化手順では、要求文の具体的な追加規則については説明していない。今後、これらの追加規則を具体化して評価していく必要がある。

さらに，本稿で提案した手法を実際に展開するためには，本手法の研修教材を開発することにより，教育上の課題を解決していく必要がある．

6. まとめと今後の課題

本稿では，EARS と SA を用いて要求を効果的にレビュする手法を提案した．また，提案した手法を酒屋の在庫問題に適用することにより，要求をレビュできることを明らかにした．しかし，提案手法が実際のシステム開発における要求仕様のレビュに適用できることについては確認していない．また，提案手法と従来手法を定量的に比較して有効性を客観的に評価していない．さらに、考察で提案した文法要素に基づいて EARS 記述を補完する手法の規則の具体化と有効性を明らかにする必要がある。

今後，本稿で提案した要求レビュ手法の客観的な有効性を明らかにする必要がある．また，他の事例にも適用して，有効性を明らかにしていく必要がある．

本稿では，EARS と SA を統合した．SA 以外にも機能間の関係をモデル化する手法には，UML[26]や ArchiMate[27]がある．したがって，EARS と UML による要求レビュ手法や EARS と ArchiMate による要求レビュ手法を考案できる．また，これらの統合型要求レビュ手法の有効性を比較評価する研究についても検討する予定である．

参考文献

[1] Marvin, A., Wilkinson, P., Harwood,A., Novak,M.,EARS:Easy Approach to Requirements Syntax,RE2009, pp.317-322, 2009
[2] Marvin,A.,Wilkinson,P.,BIG EARS(The Return of Easy Approach to Requirements Syntax), RE2010, pp.277-282, 2010
[3] Mavin, A., Wiklkinson, P., Ten Years of EARS, IEEE Software, September/October, pp.10-14, 2019
[4] DeMarco,T., Structured Analysis and System Specification, Prentice Hall, 1978. (高梨智弘，黒田純一監訳) 構造化分析とシステム仕様，日経マグロウヒル, 1987.
[5] 山本修一郎,要求開発の基礎知識～要求プロセスと技法入門,近代科学社 Digital, 2019
[6] IEEE，ソフトウェア要求仕様に対する推奨プラクティス-IEEE Std 830-1998
[7] ISO/IEC/IEEE 29148:2011, Systems and software engineering —Life cycle processes — Requirements engineering, 2011
[8] Nancy Leveson, Safeware– System Safety and Computers, Addison-Wesley, 1995, 松原友夫監訳，セーフウェア，翔泳社，2009
[9] The Open Group, TOGAF standard Version 9.2, 2018
[10] ISACA，公認 IT ガバナンス専門家 CGEIT レビュ―マニュアル第 7 版, ISBN 978-1-60420-489-6, 2015
[11] Kotonya, G., Sommerville, I., Requirements Engineering, John Wiley, 1998
[12] IPA 非機能要求グレード，https://www.ipa.go.jp/sec/softwareengineering/std/ent03-b.html
[13] 大林英晶，森崎修司，渥美紀寿，山本修一郎，逸脱分析を用いた要求仕様からのテスト項目抽出法, 情報処理学会論文誌，vol.57, No.4, pp.1262-1273, 2016
[14] Anu, V., Walia, G., Hu, W., Carver, J., Bradshaw, G., Using A Cognitive Psychology Perspective on Errors to Improve Requirements Quality: An Empirical Investigation, 2016 IEEE 27th International Symposium on Software Reliability Engineering, pp.65-76, 2016
[15] Walia, G., Carver, J., A systematic literature review to identify and classify software requirement errors, Inf Softw Technol 51, pp.1087–1109, 2009
[16] Anu; V., Hu, W., Carver, J., Walia, G., Bradshaw, G., Development of a Human Error Taxonomy for Software Requirements: A Systematic Literature Review, Information and Software Technology, vol.103, pp.112-124, 2018
[17] Porter, A., Votta, L., and Basili, V., Comparing detection methods for software requirements inspections: a replicated experiment, IEEE Transactions on Software Engineering, vol.21, No.6, pp.563-575, 1995
[18] Basili, V., Green, S., Laitenberger, O., Lanubile, F., Shull, F., *et al.*, The empirical investigation of Perspective-Based Reading, Empirical Software Engineering, vol.1, No.2, pp.133-164, 1996
[19] 林香織，鈴木亜矢香，藤戸貴大，藤元謙次，小林展英，仕様レビューにおける熟練者の知見の活用に向けた取組み，人工知能学会 第27回知識流通ネットワーク研究会 SIG-KSN-027-03, pp.1-6, 2020
[20] Parnas, D., Weiss, D., Active design reviews: principles and practices, ICSE '85: Proceedings of the 8th international conference on Software engineering, pp.132–136, 1985
[21] Gilb, T., and Graham, D., Software Inspection, Addison-Wesley Professional, 1993
[22] Kelly, D., Shepard, T., A Case Study in the Use of Defect Classification in Inspections, CASCON '01: Proceedings of the 2001 conference of the Centre for Advanced Studies on Collaborative research, pp.1-14, 2001
[23] 山崎利治，共通問題によるプログラム設計技法解説，情報処理，vol.25, No.9, pp.934, 1984
[24] 丸山勝久，情報処理，vol.54, No.9, pp. 886-pp.839, 2013
[25] 玉井哲雄，ソフトウェア工学の基礎，岩波書店，2004
[26] Object Management Group, Unified Modeling Language, https://www.omg.org/spec/UML/About-UML/
[27] The Open Group. ArchiMate® 3.1. Specification. C197. 2019.