Vol.2019-SE-201 No.3
2019/3/7

ユースケース記述中の不吉な臭いの体系化と検出

関 洋太朗[a)] 林 晋平[b)] 佐伯 元司[c)]

概要：
ユースケース記述の品質向上は，ソフトウェア開発において品質向上や開発コスト削減に関わり重要である．しかし，現実的にはユースケース記述の品質問題は発生しており，品質の低い箇所を手動で検出する作業には手間がかかる．本論文では，記述中の品質の低い箇所を不吉な臭いとして体系的に定義するとともに，その検出の自動化を試みる．不吉な臭いの定義をするために，8 つのユースケース記述中の低品質箇所を調査し，その要因と出現箇所に基づき分類を行った．また，その分類に基づき計 61 種類の臭いを体系的に定義し，GQM 法を用いて記述の特徴を数値化するメトリクス及び不吉な臭いと判定するための述語を開発し，定義した臭いのうち 25 種類を検出するツールを実装した．

1. はじめに

ソフトウェア開発プロセス中の要求理解ステップにおいて，ユースケースモデルが広く用いられている [1]．ソフトウェア開発者は，ユースケースモデルを使用することで，システム設計案の妥当性がチェック可能になり，その設計が全ての要求を満たしていることを保証できるようになる．他にも，ユースケースモデルはテスト計画や開発スケジュールの作成，ユーザーガイドの記述など，ソフトウェア開発の幅広い範囲で用いられる [1]．上記のように，ユースケースモデルはソフトウェア開発において幅広い役割を担っている．

それゆえに，ユースケースモデルの品質を高めることは，高品質のソフトウェアを低コストで開発する上で重要である．しかし，ユースケースモデルの品質問題は頻発している．その原因の一つとして，ユースケースモデルの記述部であるユースケース記述は自然言語で記述され，公式の記述のためのガイドラインやチェックリストが存在しない点が挙げられる．ユースケースモデリング手法において，各ユースケースをどれくらい詳細で具体的に記述すべきかのガイドラインがなく [2]，実際に低品質なモデルが数多く作成されている [3]．

このユースケース記述のガイドライン及びチェックリストが存在しないという課題に対し，多くの非公式なガイドラインとチェックリストが提案されている．Phalp らは，7Cs と名付けられたユースケース記述に対する自然言語で記述されたチェックリストを提供した [4]．また，Törner らは，欠陥の特徴ごとに分類された評価基準と，その欠陥の位置を特定するための質問リストを提供した [5]．

しかし，これらの手法における問題点として，実際にユースケース記述に対してのチェックを行う際に，各項目を手動で検査しなければならないという点がある．人手での検査は，問題点の見逃しや，各適用者ごとに判断基準が異なることによる評価結果の揺れなどが発生し得る [4]．他にも，実際に適用するにあたり無視できない時間がかかるという指摘もある [4]．

本論文の目的は，ソースコード中の設計や実装に問題のある箇所を示す “不吉な臭い”[6] をユースケース記述に対して定義し，不吉な臭いの大まかな種類とその発生位置によって分類されたユースケース記述における不吉な臭いのリストを作り，その検出を自動化することである．

本論文の以降の構成を以下に示す．2 章では，本論文で紹介する不吉な臭いの考案のために行った分析について説明する．3 章では，ユースケース記述の不吉な臭いを実際に検出するためのメトリクスの導出方法について説明する．4 章では，提案手法の有効性を検証するための評価実験について説明し，その結果について考察を行う．5 章では，既存研究で提案されている不吉な臭いのチェックリストとの比較を行う．6 章では，本論文のまとめと今後の課題について述べる．

[1] 東京工業大学
Tokyo Institute of Technology, Meguro, Tokyo 152–8550, Japan
[a)] seki.y.af@se.cs.titech.ac.jp
[b)] hayashi@c.titech.ac.jp
[c)] saeki@se.cs.titech.ac.jp

ⓒ 2019 Information Processing Society of Japan

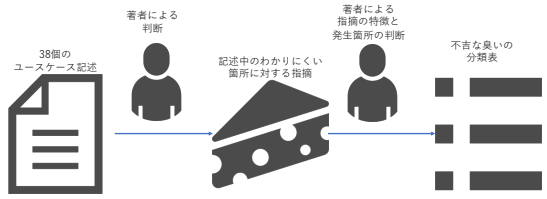


図 1 不吉な臭いの分類考案のための調査の手順

2. 不吉な臭いの調査

2.1 不吉な臭いの分類考案のための調査

本論文では，ユースケース記述の不吉な臭いを，意味が一意に定まらない，内容を理解するのに手間がかかるなどといった問題を抱えている記述中の箇所と定義する．つまり，ユースケース記述中の各開発者によって異なる解釈をし得る箇所や，記述が難解で理解に時間がかかる箇所などが，ユースケース記述の不吉な臭いとなる．

ここでは，ユースケース記述の不吉な臭いの分類を考案する際に行った調査について説明する．本調査は，ユースケース記述の不吉な臭いの種類とその分類方法を考案するために行われた．上記の目的を達成するために，実際のユースケース記述における不吉な臭いにはどのようなものがあるのか，また，それらの不吉な臭いはどのように分類することができるかを調査した．

2.1.1 データ収集

検索サイト Google[*1]でキーワード『ユースケース記述』を用いて検索した結果に含まれるウェブページから取得可能なユースケース記述から，以下の 2 条件を満たす 38 個を対象とした．これらのユースケース記述は，広範囲のドメインをカバーするように選ばれた．

- より多くの人が理解できるように，自然言語で記述されていること
- 不吉な臭いの発生している箇所を指摘しやすいように，項目ごとに欄が別れていること

2.1.2 調査方法

調査の概略を**図 1** に示す．以下の手順で調査を行った．

(1) ユースケース記述中のわかりにくい箇所を，その理由とともに指摘する．
(2) 指摘をその特徴ごとに分類する．
(3) 指摘をその発生箇所ごとに分類する．

上記の手順を，38 個のユースケース記述に対して著者らのうち 1 名が適用した．ここでは，実際に不吉な臭いを検出する際に確認すべき箇所をわかりやすくするために，不吉な臭いの特徴のみではなく指摘箇所でも分類を行った．

2.1.3 調査結果

調査の結果，曖昧さ，誤り，不適切な粒度，冗長，欠落，誤った位置の 6 種類の指摘理由とユースケース，項目，文，単語の 4 種類の発生箇所に分類が行えた．各分類についての説明を**表 1**，**表 2** に示す．例えば，“ユースケース記述にアクタ欄が存在せず，記述中のどの語がアクタなのかわからない” という指摘があった場合，この指摘はアクタ欄という “項目” が “存在しない” ことによる不吉な臭いであるとみなし，“項目の欠落” として分類した．

この分類を元に，縦軸を指摘理由の特徴，横軸を指摘箇所として，**表 3** で示すテンプレートを作成した．以降，本論文では不吉な臭いを分類する際この形式のテンプレートに，対応する不吉な臭いを挿入していった．

[*1] https://www.google.co.jp

表 1 指摘理由の特徴

指摘理由の特徴	説明
曖昧さ	意味が一意に定まらず，理解に労力がかかる．
誤り	記述が正しくない，矛盾している．
不適切な粒度	記述が大雑把すぎるか，反対に詳細すぎる．
冗長	記述に余分な部分がある．
欠落	記述に必要な部分が欠けている．
誤った位置	本来記述されるべき項目とは異なる箇所に記述されている．

表 2 指摘箇所の説明

指摘箇所	説明
ユースケース	ユースケースモデル全体に問題がある．
項目	アクタ欄，概要欄など特定の項目に問題がある．
文	記述中の特定の文に問題がある．
単語	記述中の特定の単語に問題がある．

表 3 不吉な臭いの分類テンプレート

指摘理由の特徴 \ 指摘箇所	ユースケース	項目	文	単語
曖昧さ				
誤り				
不適切な粒度				
冗長				
欠落				
誤った位置				

2.2 不吉な臭い

2.2.1 不吉な臭いの一覧

本論文で提供する不吉な臭いの一覧を**表 4** に示す．本論文では，既存研究のうち Phalp ら [4] のチェックリストおよび Torner ら [5] のチェックリストにおける問題点であった，適用する際に確認すべき箇所がわかりにくいという問題を解決するために，事前調査で得られた分類法を用いて不吉な臭いを発生箇所ごとに分類した．また，Anda ら [7] のリストの問題点であった，どのような不吉な臭いが実際に各分類に割り当てられるか記述されていないという点を解決するために，実際に割り当てられる不吉な臭いを表に記載した．以下に，表 4 の不吉な臭いのうち，特に頻出していた “アクタ欄の欠落” と “長すぎる文” を具体的に取り上げて説明する．

ⓒ 2019 Information Processing Society of Japan

表 4　各分類に当てはめられた不吉な臭い

特徴	ユースケース	項目	フロー	文	単語
曖昧さ			• 順序立たないフロー • 発生元不明の例外フロー • 発生元不明の代替フロー	• 不明瞭な実行可能性 • 発生元不明の動作結果 • 一意に定まらない文	• 代名詞の使用 • 省略された単語 • “アクタ” アクタ • 説明のない動作主 • 複数概念を指す同一単語 • 属性の省略
誤り		• 事前条件を満たさないフロー • 満たされない事後条件 • 記述内容を表さないユースケース名 • 条件の過不足 • アクタの過不足		• 矛盾した文章 • 条件を無視した動作	
不適切な粒度	• 複数状況を語るユースケース	• 同一原因で複数の例外フロー • 同一原因で複数の代替フロー • 異なる役割の単一アクタ • 同じ役割の複数アクタ		• 長すぎる文 • 短すぎる文 • 複数動作を行う文 • 特定の文の過剰修飾 • 特定の文の過少修飾	• 既出単語の省略 • 既出単語への情報付加
冗長			• 同一役割の複数フロー • 事後条件と関係のないフロー • 状態フロー	• 同一名詞の繰り返し • 文の過剰修飾	• 単語の余分な修飾
欠落	• 独立した単独で実行できないユースケース	• アクタ欄の欠落 • 例外フロー欄の欠落 • 代替フロー欄の欠落 • 事前条件欄の欠落 • 事後条件欄の欠落 • ユースケースの概要欄の欠落 • ユースケース名欄の欠落	• 例外の考慮不足 • 選択肢の考慮不足	• 戻り先のない例外フロー • 原因不明の例外フロー • 戻り先のない代替フロー • 原因不明の代替フロー • システム動作の記述不足 • システム情報の記述不足	• 欠落した動作対象 • 欠落した操作手順 • 出所不明の情報
誤った位置		• メインフロー中の事前条件 • メインフロー中の事後条件 • メインフロー中の例外フロー • メインフロー中の代替フロー			
表記揺れ					• 同義語の使用

2.2.2　不吉な臭い：アクタ欄の欠落

名前　アクタ欄の欠落

発生箇所　項目

特徴　欠落

症状　実際にユースケースに関係しているアクタを特定する際に，アクタがリストアップされていないため手間がかかる．

検出方法　この臭いを検出するためには，Actor 欄が正しく存在するかを調べれば良い．

2.2.3　不吉な臭い：長すぎる文

名前　長すぎる文

発生箇所　文

特徴　不適切な粒度

症状　対象の文が情報を持ちすぎているか，余分な記述をしており理解時の妨げになる．

検出方法　この臭いを検出するためには，文の長さを算出し，その値が外れ値を超えていないかを調べれば良い．

ⓒ 2019 Information Processing Society of Japan

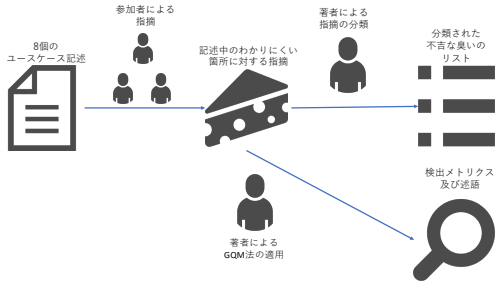


図 2　検出メトリクス考案のための調査の手順

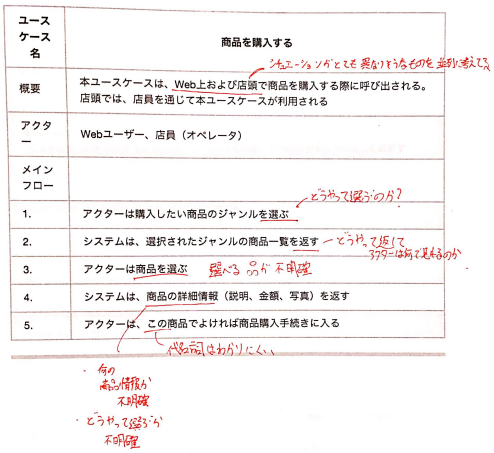


図 3　実際の指摘の例:ユースケース記述は http://www.itmedia.co.jp/im/articles/0307/05/news002.html より引用

3.　検出メトリクスの導出

ここでは，不吉な臭いを検出するためのメトリクスを考案し，それらを分類するために行なった事前調査について説明する．本調査は，前節で考案した分類に当てはまる実際の不吉な臭いを特定し，その不吉な臭いの検出方法を考案する目的で行われた．各分類に当てはまる不吉な臭いを特定するために，本調査ではユースケース記述のわかりにくい箇所をその理由とともに挙げさせ，それらの指摘を前節の分類方法に当てはめて分類を行った．また，不吉な臭いの検出方法を考案するために，指摘を元に GQM（Goal, Question, Metric）法 [8] を用いてメトリクスを考案した．

3.1　調査方法

図 2 に調査方法の概略を示す．以下の手順で調査を行った．

(1) ユースケース記述中のわかりにくい箇所を，その理由とともに指摘させる．
(2) 指摘を前節で考案した分類に当てはめる．
(3) 指摘が前節の分類に当てはめることができなかった場合，その発生箇所及び指摘理由を元に新たに分類基準を考える．
(4) 指摘を元に，GQM 法を適用してメトリクスを考案する．

本調査では，前節で用いた 38 個のユースケース記述から 8 つを選び，それらをテストデータとして利用した．また，手順 1 では，ユースケース記述および情報工学に関する知識を持つ 8 名が，ガイドラインなどを用いずに指摘を行った．その際には，特に時間制限を定めず，紙に記述されたユースケース記述に対して図 3 のように指摘を記述させた手順 2，3 は著者らのうち 1 名が行った．

手順 4 においては，図 4 に示すように GQM 法を用いてメトリクスの考案を試みた．実際に GQM 法を適用した際の手順は次の通りである．

(1) 最上位のゴールとして “項目の曖昧さ” という様に不吉な臭いの分類を，サブゴールとして最上位のゴールに分類される不吉な臭いをそれぞれ記入する．

表 5　各分類に当てはめられた指摘の数

指摘理由の特徴	ユースケース	項目	フロー	文	単語
曖昧さ	0	10	0	14	15
誤り	0	17	1	11	0
不適切な粒度	5	3	0	6	1
冗長	0	0	3	11	2
欠落	1	10	5	22	8
誤った位置	0	6	0	0	0
表記揺れ	0	0	0	0	8

(2) 各サブゴールについて，その不吉な臭いを検出するためにどの様な点を調べればいいかを，Question として記入する．
(3) 各 Question に対して，実際に確認を行うためのメトリクスを考案する．

例えば図 4 に例示した GQM は，以下のようなプロセスに従って記述された．

(1) “項目の曖昧さ” を検出するために，各フローが正しく番号付けされているかを確認する必要がある．
(2) 各フローが正しく番号付けされているかを確認するために，“フローの番号は 1 ずつ大きくなっているか？” という Question が発生する．
(3) “フローの番号は 1 づつ大きくなっているか？” という Question に答えるためには，全てのフローの番号が 1 づつ大きくなっているかを確認するメトリクスが必要であるとわかる．

3.2　調査結果

調査結果として，それぞれの分類に当てはめられた指摘の数を表 5 に示す．表 5 に “フロー” という列と “表記揺れ” という行が追加されていることからわかる通り，この調査を通して新たに 1 種類の指摘理由と，1 箇所の指摘箇所が発見された．フローという範囲の指摘は，2 つ以上のフローが矛盾しているなどの，フローの一部または全体で問題が発生していることを示す．また，表記揺れという特

ⓒ 2019 Information Processing Society of Japan

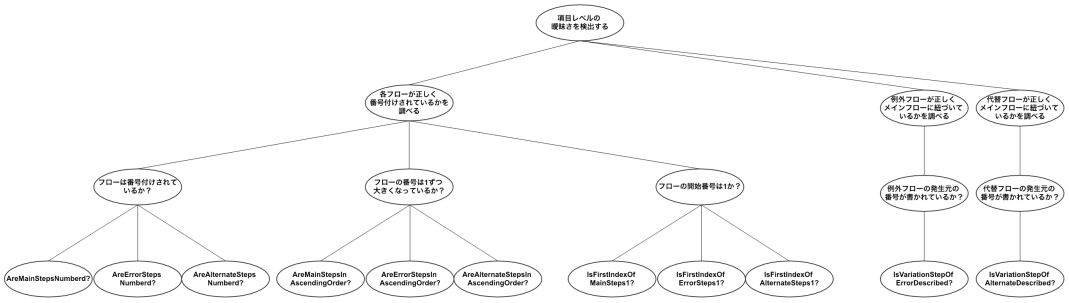


図 4 GQM 法による “項目の曖昧さ” に分類される “順序立たないフロー” に対するメトリクス考案の例

表 6 不吉な臭いの検出に用いるメトリクス

Metric	概要
CountOfPronoun	代名詞の数
CountOfWord(*word*)	単語 *word* の数
CountOfErrorWithSameReason	同じ発生原因の例外フロー数
CountOfAlternateWithSameReason	同じ発生原因の代替フロー数
LengthOfSentence	文の長さ
CountOfVerb	動詞の数
CountOfModifier	修飾語の数
CountOfNoun(*noun*)	名詞 *noun* の数

徴の指摘は，同じ意味の単語を複数種類使い分けている，語尾が文ごとに異なるといった文章中の表記揺れに関する問題が発生していることを示す．

さらに，表 5 のうち実際に指摘があった箇所について GQM 法を用いたところ，25 種類の不吉な臭いに対して **表 6**，**表 7** に示す 36 個のメトリクス及び述語が考案された．

4. 評価

提案手法を評価するために以下のリサーチクエスチョン（RQ）を定めた．

RQ: 実装した自動検出ツールは，2 つ目の調査で指摘された不吉な臭いのうち，メトリクスを考案したものについては正しく検出できているか？

RQ を確認することによって，ツールに組み込んだメトリクスによって正しく不吉な臭いの検出を行えるかが調べられる．

4.1 調査方法

3 章での調査の結果をテストデータとして用いて，不吉な臭いとして指摘のあった箇所をツールが正しく検出できるかを調べる．テストデータとしてユースケース記述をツールに入力し，ツールの出力する結果と事前調査で指摘のあった箇所とを比較することで，ツールの性能を評価する．なお，前述の調査で指摘のあった不吉な臭いのうち，ツールでサポートされている不吉な臭いのみを対象とし，その他の自動検出をサポートしていない不吉な臭いに関しては比較を行わない．

4.2 結果

自動検出ツールが不吉な臭いと判断し，実際に不吉な臭いであったものを True Positive（TP），自動検出ツールが不吉な臭いとして判断しなかった一方で実際には不吉な臭いとして指摘されているものを False Negative（FN），自動検出ツールが不吉な臭いと判断したが実際には不吉な臭いとして指摘されていなかったものを False Positive（FP）として，式 1 により Precision と Recall を算出する．

$$Precision = \frac{|TP|}{|TP| + |FP|} \qquad Recall = \frac{|TP|}{|TP| + |FN|} \quad (1)$$

Precision，Recall を各ユースケース記述ごとに求めたものを表 8 に示す．

4.3 考察 : 各ユースケース記述ごとの Precision，Recall

実験の結果より，本論文で提案するメトリクスを用いることで，3 章のユースケース記述のうち，ユースケース 1，2，3，4，5 のものまでは高い Recall を達成している．一方で残りの 3 つの 6，7，8 に関しては Recall が非常に低くなっている．

このうちユースケース 7，8 については，“アクタ” アクタとして指摘した箇所のほとんどが誤検出であったことが原因である．一方で，“アクタ” アクタはユースケース 1，2 でも同様に検出されているが，これらのユースケースでは誤検出ではなく実際に “アクタ” アクタが指摘されている．前者が実際には “アクタ” アクタとして指摘されなかった原因として，ユースケース 7，8 にはアクタ欄が存在し，かつアクタが 1 種類しか存在しない点が考えられる．アクタ欄があり，かつアクタがただ一つしか存在しないことにより，“アクタ” と記述されても読者はどのアクタを指しているかを判断できるため，不吉な臭いとしては指摘されなかったと推測される．

また，ユースケース 6 については，ツールでの検出対象となる不吉な臭いが存在しなかったため，$TP = \emptyset$ になり Recall も定義より 0 となった．

Precision については，ユースケース 6 を除く全てのユースケースにおいて 1.00 になっているので，本ツールを用いることで対応している不吉な臭いを高い確率で検出できる

表 7　不吉な臭いの検出に用いる述語

述語	概要
AreMainStepsNumberd?	メインフローは全て番号付けされているか？
AreErrorStepsNumberd?	例外フローは全て番号付けされているか？
AreAlternateStepsNumberd?	代替フローは全て番号付けされているか？
AreMainStepsInAscendingOrder?	メインフローの番号は全て 1 づつ大きくなっているか？
AreErrorStepsInAscendingOrder?	例外フローの番号は全て 1 づつ大きくなっているか？
AreAlternateStepsInAscendingOrder?	代替フローの番号は全て 1 づつ大きくなっているか？
IsFirstIndexOfMainSteps1?	メインフローの最初の番号は 1 か？
IsFirstIndexOfErrorSteps1?	例外フローの最初の番号は 1 か？
IsFirstIndexOfAlternateSteps1?	代替フローの最初の番号は 1 か？
IsVariationStepOfErrorDescribed?	例外フローの発生元は記述されているか？
IsVariationStepOfAlternateDescribed?	代替フローの発生元は記述されているか？
IsActorColumnExist?	アクタ欄は存在するか？
IsMainColumnExist?	メインフロー欄は存在するか？
IsErrorColumnExist?	例外フロー欄は存在するか？
IsAlternateColumnExist?	代替フロー欄は存在するか？
IsPreConditionsColumnExist?	事前条件欄は存在するか？
IsPostConditionsColumnExist?	事後条件欄は存在するか？
IsOverviewColumnExist?	概要欄は存在するか？
IsNameColumnExist?	ユースケース名欄は存在するか？
IsRejoinLocationOfErrorExist?	例外フローの戻り先は記述されているか？
IsRejoinLocationOfAlternateExist?	代替フローの戻り先は記述されているか？
IsReasonOfErrorExist?	例外フローの発生原因は記述されているか？
IsReasonOfAlternateExist?	代替フローの発生原因は記述されているか？

表 8　各ユースケース記述ごとの Presicion，Recall

ユースケース	Precision	Recall	ツールによる指摘の数	実際の不吉な臭いの数	ツールに対応した実際の不吉な臭いの数
1	0.737	1.00	19	26	14
2	0.700	1.00	10	23	7
3	0.923	1.00	13	15	12
4	0.818	1.00	11	16	9
5	0.778	1.00	9	30	7
6	0.00	N/A	10	19	0
7	0.222	1.00	9	17	2
8	0.250	1.00	8	29	2

と考える．また，ユースケース 6 については実際にツールの対応している不吉な臭いとして指摘されたものがなかったため，TP，FP がともに 0 になり，Precision の算出は行えなかった．

4.4　考察 : 各不吉な臭いの特徴ごとの Precision，Recall

Recall については，“冗長な文” 以外の分類において 1.00 となっており，実際の不吉な臭いを検出ツールを用いることで高精度での検出することが可能と考える．

Precision については，“文の欠落”，“曖昧な項目” において非常に高い数値を示しており，これらの不吉な臭いに対しては高精度での検出が可能と考える．また，項目の欠落に関しても 0.7 弱と高い数値を示しており，この不吉な臭いに関しても高精度での検出が可能と考える．

一方で，ツールでの指摘のうち “曖昧な単語” については 50%，“不適切な文の粒度” に関しては 25%ほどしか，実際に不吉な臭いとして指摘されているものはなかった．“曖昧な単語” において Precision が低い原因としては，ユースケース 7，8 における “アクタ” アクタの誤検出の多さが原因であると考える．また，“不適切な文の粒度” において Precision が低い原因としては，各フロー内で実際の動作例などを示している場合，単語数や文の長さは著しく増加するが，実際に記述を読む段階では理解の妨げにはならない可能性がある．

4.5　妥当性の脅威

内的妥当性については，3 章の調査参加者の判断基準と，時間経過による判断基準の変化が脅威となる．しかし，調査において時間制限を設けず，不吉な臭いの指摘順序も指定をせず何度でも他の記述を参考に指摘をし直してよいとしたので，時間経過による判断基準の変化による脅威は小さいと考えられる．また，参加者ごとの判断基準の相違に

ⓒ 2019 Information Processing Society of Japan

表 9　各不吉な臭いの分類ごとの Presicion，Recall

不吉な臭いの分類	Precision	Recall	ツールによる指摘の数	実際の不吉な臭いの数
曖昧な項目	0.833	1.00	12	10
曖昧な単語	0.522	1.00	23	12
不適切な項目の粒度	1.00	1.00	1	1
不適切な文の粒度	0.292	1.00	24	7
冗長な文	N/A	N/A	0	0
項目の欠落	0.696	1.00	23	16
文の欠落	1.00	1.00	4	4

ついても，参加者全員が要求工学に関する一定の知識を有しているため，脅威は軽減されていると考えられる．

外的妥当性については，本実験で自動検出ツールの性能を調査する際に，8 つのユースケース記述のみを対象にしているため，他のユースケース記述でも同様の精度で検出できるとは限らない．そのため，より多くのユースケース記述に対して実験を行う必要がある．

5. 関連研究

本章では本論文に関連する研究を述べる．関連する研究の中でも，ユースケース記述の不吉な臭いに関するものについて焦点を当てる． ユースケース記述の不吉な臭いの検出についての研究は多く存在する [4], [5], [7].

Phalp ら [4] は，ユースケース記述の品質を測定するために，7Cs と名付けられたチェックリストを提供している．このチェックリストは，ユースケース記述の品質と文章の理解しやすさに関する既存研究から，ユースケース記述における望ましい品質とは何かを調査することで作成された．

Torner ら [5] は，ユースケース記述の不吉な臭いの評価基準をリストにまとめ，不吉な臭いの影響度合いで順序づけを行った．このリストは 12 の評価基準からなり，それぞれの評価基準には，判断に使用するための質問が自然言語で記述されている．12 の評価基準は，判断できる不吉な臭いの特徴から 7 種類に分類される．

Anda ら [7] は，ユースケースモデルにおける欠陥を，欠陥の発生する箇所と特徴によって分類する手法を提案し，実際にその手法に基づいて分類された不吉な臭いをチェックリストとして提供している．提供されたチェックリストでは，不吉な臭いの発生箇所ごとに，それぞれの特徴に対応した不吉な臭いを検出するための質問が自然言語で記述されている．

本論文で提供する不吉な臭いのリストでは，既存研究のチェックリストでは記載されていなかった “不適切な項目の粒度，ユースケースの欠落” に分類される不吉な臭いに対応させることができる．一方で，Torner らの研究における，複数のフローを取り出して別のユースケースに切り出すべき状態を表す “C7: Use Case Decomposition”，Phalp らの研究における，例外フロー及び代替フローが正しく実行できるかを問う “Consideration of Alternatives: Viable”，Anda らの研究における，余分なアクターの存在を示す 1.3，システムとは関係のないユースケースの存在を示す 2.2，アクタの説明と実際のやりとりとの乖離を示す 2.4，事前条件の詳細さが適切でないことを示す 3.5，ユースケース間での矛盾を示す 4.3 という不吉な臭いは，本論文のリストには記載されておらず，検出することができない．

6. おわりに

6.1 まとめ

本論文では，ユースケース記述に含まれる不吉な臭いの体系化と，それらの検出を自動化した．また，体系化された不吉な臭いのリストを既存研究のものと比較して有用性を示し，不吉な臭いの自動検出ツールについても，実際に多くのユースケース記述において高い精度で検出が可能であることを示した．

一方で，ツールによる自動検出ができる不吉な臭いの数が少ない，既存研究では検出することができた不吉な臭いの一部が検出できないといった課題も残った．

6.2 今後の課題

6.2.1 自動検出可能な不吉な臭いを増やす

現時点のツールでは，提案した 59 個の不吉な臭いのうち，自動検出に対応しているものは 23 個と，その割合は 39%に留まる．未対応となった不吉な臭いの多くは，“例外の考慮不足” のように実際にその文章を読み取っていかないと評価ができないなどの理由で自動検出は難しいと考えられるが，“既出単語の省略” など一部のものについては意味解析などによって検出できる可能性がある．

6.2.2 調査の対象を増やす

本論文では，調査の際に 7 個のユースケース記述を対象にアンケート調査を実施したが，本論文では検出されなかった不吉な臭いが他のユースケース記述には存在する可能性がある．したがって，より多くのユースケース記述を対象に調査を行なうことで新たな不吉な臭いが発見される可能性がある．

6.2.3 検出された不吉な臭いに対する改善案の提示

本論文では，不吉な臭いを検出することにより改善すべき点は提示されるが，具体的にどのようにして修正を施す

ⓒ 2019 Information Processing Society of Japan

べきかは提示されていない．各不吉な臭いに対して具体的な改善方法を提供することができれば，不吉な臭いを検出するだけでなく同時にその箇所を修正することも可能になる．

謝辞 本研究の一部は，日本学術振興会科学研究費補助金（JP15K15970，JP18K11237，JP18K11238）の助成を受けた．

参考文献

[1] Geri, S. and Jason, W.: *Applying Use Cases: A Practical Guide*, Addison Wesley Longman (1998).
[2] A. Issa, A.: Utilising Refactoring To Restructure Use-Case Models, *Lecture Notes in Engineering and Computer Science*, Vol. 2165 (2007).
[3] El-Attar, M. and Miller, J.: Improving the quality of use case models using antipatterns, *Software & Systems Modeling*, Vol. 9, No. 2, pp. 141–160 (online), DOI: 10.1007/s10270-009-0112-9 (2010).
[4] Phalp, K. T., Vincent, J. and Cox, K.: Assessing the quality of use case descriptions, *Software Quality Journal*, Vol. 15, No. 1, pp. 69–97 (2007).
[5] Tørner, F., Ivarsson, M., Pettersson, F. and Öhman, P.: Defects in Automotive Use Cases, *Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering*, ISESE '06, New York, NY, USA, ACM, pp. 115–123 (online), DOI: 10.1145/1159733.1159753 (2006).
[6] Fowler, M.: *Refactoring: Improving the Design of Existing Code*, Addison-Wesley (1999).
[7] Anda, B. and Sjøberg, D. I. K.: Towards an Inspection Technique for Use Case Models, *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering*, SEKE '02, New York, NY, USA, ACM, pp. 127–134 (online), DOI: 10.1145/568760.568785 (2002).
[8] Basili, V. R.: Software modeling and measurement: the Goal/Question/Metric paradigm, Technical report (1992).

ⓒ 2019 Information Processing Society of Japan