Vol.2013-SE-179 No.35
2013/3/12

業務観点でのレビューを目指した不具合情報の分析

森崎 修司[1,2]　松本 健一[1]

概要：ソフトウェアが担う業務に特化した欠陥種別の検出を目的とするレビューは汎用的な欠陥種別の検出を目的としたレビューよりも修正コスト低減効果やスケジュール遅延リスク低減効果が大きいことが期待される．本論文では，金融業務を担うソフトウェアの開発に携わる熟練者へのヒアリングから得られた欠陥種別 “日付に関する欠陥” をもとに不具合情報を分析した．レビューでの検出が可能であった不具合 488 件のうち 86 件が日付に関する不具合であり，これらの不具合が他の不具合と比較して，スケジュール遅延リスク低減効果，修正コスト低減効果が大きく，統計的に有意な差があることがわかった．また，業務に特化した欠陥種別特定の計算機支援を目的とし，不具合情報の自由記述に含まれる単語を熟練者に提示したところ，支援が有用であるという意見を得た．

1. はじめに

ソフトウェア品質を向上するための施策として，ソフトウェアレビューが提案されており [3]，これまで様々なレビュー手法が提案されている [1][6][8]．ソフトウェアレビューの主要な効果は，欠陥の早期発見により修正コストが低減したり，スケジュール遅延リスクが低減したりすることにある．これまでに提案されているレビュー技法 (リーディングテクニック) の多くは，“データ定義に不整合はないか”，“テストできるか” といった汎用的な欠陥種別の検出を想定し網羅的な欠陥検出により，修正コスト低減効果やスケジュール遅延リスク低減効果を大きくすることを目的としている．

レビューによるリスク低減やコスト低減は，レビューで検出するほうがテストで検出するときと比較して，スケジュール遅延リスクが小さくなる欠陥や修正コストが小さくなる欠陥を検出することによりもたらされる．レビュー対象の規模が大きくなると，汎用的なシナリオで網羅的に欠陥を検出するコストが，低減される修正コストよりも大きくなったり，低減されるスケジュール遅延リスクに見合うものでなくなる場合がある．近年，レビュー対象の規模が大きくなり，網羅的な欠陥検出や汎用的なシナリオによる欠陥検出がリスク低減やコスト低減につながりにくくなってきている．

Denger らは，レビューで使うシナリオや実施方法をテーラリングする手法を提案し，最終成果物の品質を意識しレビューで検出すべき欠陥種別を決定することを勧めている [2]．Denger らの提案手法を用いれば，原理的には，リスク低減，コスト低減につながる欠陥種別を検出できるようテーラリングすることができるが，その言及や欠陥種別の特定方法に関する言及もない．

本研究の目的は，レビューにおいて，ソフトウェアが担う業務に特化した欠陥の検出に注力することにより，スケジュール遅延リスクの低減や欠陥の修正コストの低減につながるかどうか実証的に評価し，欠陥種別のシステマチックな特定に向けて，計算機支援が可能かを評価することにある．具体的には，ソフトウェア開発に携わる熟練者にヒアリングし，スケジュール遅延リスク低減，修正コスト低減につながる可能性のある欠陥種別を得る．過去にソフトウェア開発において検出，収集された不具合情報を用いて，得られた欠陥種別が存在するかどうか確認し，存在する場合にはリスク低減，修正コスト低減の効果があるか評価する．また，その欠陥種別が検出できるようなシナリオを作成することができるかどうかも調べ，レビューにおける検出可能性を検討する．次に，欠陥種別の特定の計算機支援が可能か評価することを目的とし，不具合情報の自由記述に含まれるキーワードを熟練者に提示し，ヒアリングの際に得られた欠陥種別以外の欠陥種別の特定ができるかどうか評価する．

[1] 奈良先端科学技術大学院大学 情報科学研究科
Graduate School of Information Science, Nara Institute of Science and Techonlogy, Takayama, Ikoma, Nara, 630-0192, Japan
[2] 静岡大学 情報学部
Faculty of Informatics, Shizuoka University, Johoku, Naka, Hamamatsu, Shizuoka, 432-8011, Japan

表 1 対象とする不具合情報 $\mathcal{D}$

不具合 ID	項目 l_1^c	$\cdots$	項目 l_m^c	項目 l_{m+1}^f	$\cdots$	項目 l_{m+u}^f
1	d_{11}^c	$\cdots$	d_{m1}^c	$d_{(m+1)1}^f$	$\cdots$	$d_{(m+u)1}^f$
2	d_{12}^c	$\cdots$	d_{m2}^c	$d_{(m+1)2}^f$	$\cdots$	$d_{(m+u)2}^f$
$\vdots$						
n	d_{1n}^c	$\cdots$	d_{mn}^c	$d_{(m+1)n}^f$	$\cdots$	$d_{(m+u)n}^f$

表 2 不具合情報 $\mathcal{D}$ に含まれる項目

項目	分類	説明	本評価での目的
混入工程	l^c	混入された工程	レビューでの検出可能性の判断
修正工数	l^c	修正に要した工数	早期検出によるコスト低減の判断
重大度	l^c	不具合が与える影響の大きさ	早期検出によるリスク低減の判断
機能分類	l^c	検出された機能カテゴリ	不具合の検出場所に偏りの確認
不具合の説明	l^f	不具合の再現方法，現象，修正方法等	欠陥種別の分類，及び，キーワード分析

2. 関連研究

Personal Software Process[4] は，過去に自身が間違った内容に応じてチェックリストをテーラリングすることを推奨している．これまでの自分の間違いに特化することにより欠陥をより多く検出すれば，スケジュール遅延リスク低減や欠陥の修正コスト低減を間接的に実現できる場合があるが，リスク低減や修正コストによって過去の間違いを重みづけする等の言及はない．

これまでに提案されているリーディングテクニックのいくつかでは，何らかの方法で欠陥種別を想定し，その検出手順をシナリオとして定義している．それらの多くは，本論文で想定するようなリスク低減，修正コスト低減につながる欠陥の検出が原理的には可能であるが，言及がない．

Perspective-based reading [1] では，レビュー対象のソフトウェアとの利害関係者の立場を想定し，その立場で欠陥検出を試みる．たとえばテストエンジニアを立場とし，要求文書をレビュー対象とした場合，要求が実現できたかどうかをどのようにテストするかを考えながらレビューするといった具合である．役割は，網羅的な欠陥検出や効率的な欠陥検出に主眼が置かれている．リスク低減や修正コスト低減を加味することは原理的には可能であるが，立場と関連するリスク低減や修正コスト低減に限定される．

Defect-based reading [6] では，レビュー対象に存在している可能性のある欠陥種別を想定し，その欠陥種別を検出するためのシナリオを作成し，レビューアに割当てる．文献 [6] では，シナリオ例として，“誤り”，“漏れ”，“曖昧さ”が挙げられている．要求文書をレビュー対象とした “誤り” の検出では，要求文書内での項目どうしや要求文書と外部の定義の間で整合性がない部分を検出するための手順がシナリオとして定義されている．欠陥種別として，本論文で想定するリスク低減や修正コスト低減につながる欠陥を設定することはできるが，言及はない．

Usage-based reading [8] では，ソフトウェアのユースケースに沿ってレビュー対象を読み進め，問題のある箇所を検出する．ユースケースはソフトウェアで支援する業務に従って作成され，優先順位を付けた上でレビューされるため，ユーザにとって優先順位の高い部分からレビューできる．本研究で想定する業務観点によるレビューはユースケースとして表現することができる場合があるが，ユーザの優先順位とリスク低減や修正コスト低減が完全に一致するとは限らない．

3. 評価方法

3.1 対象とする不具合情報と欠陥種別

本研究で対象とする不具合情報 $\mathcal{D}$ を表 1 に示す．不具合情報は，ソフトウェアのテストの最中に検出され，記録された不具合 D_i の集合である．不具合 D_i には不具合の再現方法，症状，重大度，検出日，報告者といった項目 l の情報 d を付与する．評価では，後述する必須項目以外の項目に特に前提はおかないが，入力が選択肢や数値に限定される項目 l^c と報告者が任意の内容を記述できる自由記述式の項目 l^f に分類できるものとする．重要度，報告者，検出日は l^c であり，再現方法や症状は l^f である．

本研究において不具合情報 $\mathcal{D}$ に含まれる項目を表 2 に示す．“混入工程” はレビューで検出できた可能性のある不具合かどうかを調べるために用いる．“修正工数” は修正コスト低減の効果があるかどうかを調べるために用いる．“重大度” はスケジュール遅延リスクの大小を調べるために用いる．重大度の大きな不具合は影響範囲が大きく，スケジュール遅延リスクが大きくなるからである．“機能分類” は欠陥種別が特定の機能に偏っているかどうかを調べるために用いる．

本研究で対象とする欠陥種別は，特定の観点 (着眼点) において分類できる欠陥の集合を指す．欠陥種別 T は，ある一定の着眼点において 1 件以上の不具合 D_i を抽象化したものである．また，欠陥種別 T の説明を t とし，T をレビューで検出するためのシナリオを S とする．

表 3 対象とした不具合情報とソフトウェアの概要

対象ソフトウェア	対象業務	外国債券の取引入力，記帳管理
	開発期間	2 年 (基本設計〜リリース)
	開発プロセス	ウォータフォール
不具合情報	収集フェーズ	結合テスト (サブシステム間結合)
	入力項目数	28 項目

表 4 評価で利用した不具合情報の項目

項目名	形式	説明
現象	l_f	不具合の現象
原因	l_f	不具合混入の原因
対策	l_f	不具合の修正方法
対応工数	l_c(数値)	不具合修正の工数 (人日)
原因分析	l_c(33 項目)	不具合混入の原因 (レビュー不備，レビューア不十分，レビュー観点不備を含む)
重大度	l_c(3 項目)	A(全体に大きな影響), B(特定の機能が使用不可)，C(機能は使用可)
テストイベント	l_c(28 項目)	テスト種別
機能カテゴリ	l_c(14 項目)	機能分類

表 5 不具合の件数

分類	件数
日付に関する不具合	86
日付に関する不具合ではない	401
不明	1
レビューで検出できたと判断された不具合	488

3.2 仮説と評価手順

早期検出によるリスク低減が期待できるシナリオ，早期検出による修正コスト低減が期待できるシナリオが可能かどうかを実証的に評価する．仮説とその評価手順を以下に示す．

仮説 1　業務に特化した欠陥種別 T_s が存在しレビューの観点 (着眼点) になり得る

仮説 1-1　熟練者から業務に特化した欠陥種別 T_s が得られる

不具合情報 $\mathcal{D}$ を収集したソフトウェア開発プロジェクトの全体を俯瞰するような立場にある熟練者にヒアリングする．早期修正によるスケジュール遅延リスク低減，修正コスト低減につながる欠陥種別 T_s が存在するか聞き，存在すればその内容も聞く．

仮説 1-2　熟練者から得られた T_s が不具合情報 $\mathcal{D}$ に存在する

欠陥種別 T_s が不具合情報 $\mathcal{D}$ に存在しているか確認する．対象とする不具合は混入工程が不具合情報 $\mathcal{D}$ の収集フェーズに V 字モデルにおいて対応するレビュー (R) 以前のものとする．たとえば，結合テストで収集された不具合情報であれば，R は設計レビューとなる．

仮説 1-3　T_s に該当する不具合とその他の不具合の間で重大度，修正工数に差がある

混入工程が R 以前であり，T_s に該当しない不具合と T_s に該当する不具合を比較し，重大度，修正工数に差があるか調べる．

仮説 1-4　T_s を検出するためのシナリオ S_s を作成できる

T_s をレビューで検出することを目的としてシナリオ S_s を作成し，レビューアがそれをもとに T_s を検出できるか熟練者に検討してもらう．

仮説 2　計算機支援により業務に特化した欠陥種別を不具合情報から特定できる

仮説 2-1　欠陥種別 T_s を表すキーワードが l^f に多く含まれる

不具合の現象や原因をはじめとする自由記述の項目 l^f に含まれる単語に欠陥種別 T_s を表すキーワード $w_1, w_2, \cdots w_n$ が含まれているか調べる．

仮説 2-2　l^f に高頻度で現れる単語の提示が欠陥種別 $T'_s, T''_s, \cdots$ の特定の支援となる

不具合情報 $\mathcal{D}$ の自由記述項目 l^f に含まれる単語を抽出し，出現頻度上位の単語を熟練者に提示することにより，T_s 以外の欠陥種別 $T'_s, T''_s, \cdots$ 特定のための判断材料になるか検討してもらう．

4. 評価

4.1 対象とした不具合情報と熟練者

対象とした不具合情報は，大手銀行の外貨有価証券システムの再構築において検出，蓄積された不具合情報である．対象とした不具合情報の収集フェーズは機能間結合テストである．対象システムと不具合情報の概要を表 3 に示す．また，今回の評価において利用した不具合情報の入力項目を表 4 に示す．

ヒアリングした熟練者は，大手銀行のシステム部に所属し，長年にわたり金融システムの開発に携わっている者である．対象業務に関する豊富な知識と経験，システム開発に関する豊富な知識と経験，システムに求められる品質とその確保について熟知している．

表 6　重大度の割合

分類	重大度 A	重大度 B	重大度 C
日付に関する不具合	2(2.3%)	59(68.6%)	25(29.1%)
日付に関する不具合ではない	4(1.0%)	190(47.4%)	207(51.6%)
不明	0(0.0%)	1(100.0%)	0(0.0%)
レビューで検出できたと判断された不具合	6(1.2%)	250(51.2%)	232(47.5%)
不具合情報全体	(2.9%)	(39.8%)	(57.3%)

4.2　熟練者への欠陥種別 T_s のヒアリング (仮説 1-1)

業務に特化した欠陥種別 T_s が存在するという回答が得られ，T_s として “日付に関する欠陥” が挙げられた．理由は，対象システムである外国債券の取引，記帳管理において，利息の起算日，利払い日，償還日をはじめとする日付に関する誤りをレビューで検出することが，レビューによるスケジュール遅延リスクを効果的に低減すると推測されたからである．対象ソフトウェアは外国債券を取り扱うものであり，現地時間の日付や日本時間の日付等，細かい観点でのレビューが必要になる．

4.3　欠陥種別 T_s の存在確認 (仮説 1-2)

T_s “日付に関する欠陥” を不具合情報 $\mathcal{D}$ の l^f を目視により調べたところ，T_s に該当する欠陥 D_i が存在した．件数をまとめたものを表 5 に示す．対象不具合情報に含まれる不具合のうち，“原因分析” の項目においてレビューでの検出漏れと記録された不具合 488 件を対象とした．488 件の “現象”, “原因”, “対策” 項目を目視し，日付に関する不具合を 86 件 (約 18%) 確認した．86 件は，日付の解釈の間違いや日付の定義が誤っているといったものを含み，たとえば “期日に前回ファクター償還日の前営業日を設定する際に，本来考慮しなくてもよい休日を考慮している” があった．また，488 件のうち 1 件のみ記述だけでは判断できないものがあった．

4.4　T_s に該当する不具合とその他の不具合の重大度，修正工数の差 (仮説 1-3)

日付に関する欠陥を早期検出することにより，スケジュール遅延リスク低減につながるかどうかを調査するために不具合の重大度の割合を調べた．結果を表 6 に示す．表中の値は不具合の件数を表している．カッコ内の%は表の各行の合計を 100%としたときの割合を示している．T_s と T_s 以外の不具合とを比較すると，重大度 B の不具合の割合が大きく，重大度 C の割合が小さかった．

日付に関する欠陥を早期検出することにより，修正コストの低減につながるかどうかを調査するために不具合の平均修正工数を調べた．結果を表 7 に示す．レビューにおいて検出できた可能性のある不具合の修正工数と比較すると，T_s の修正工数の平均のほうが大きかった．

レビューで検出できたと判断された不具合 488 件から不明の 1 件を除いた 487 件において，日付に関する不具合とそれ以外の不具合の間で，重大度，修正工数に統計的に有意な差があるかを調べるために検定を実施した．結果を表 8 に示す．重大度の比較では，帰無仮説を “日付に関する不具合とそれ以外の不具合の間で，重大度の割合に違いはない” とし，フィッシャーの直接確率検定により検定した．修正工数では，帰無仮説を “日付に関する不具合とそれ以外の不具合の間で，修正工数の違いはない” とし，U 検定により検定した．重大度，修正工数ともに有意水準 1%において，統計的に有意な差があった．

表 7　修正工数の平均

分類	修正工数の平均
日付に関する不具合	2.10
日付に関する不具合ではない	1.68
全件 (「原因」にレビューを含む)	1.75
不具合管理表全件 (参考)	2.21

表 8　検定結果

比較群	検定	p 値
重大度	フィッシャーの直接確率検定	0.00029
修正工数	U 検定	0.00516

4.5　T_s を検出するためのシナリオ S_s (仮説 1-4)

レビューにおいて T_s を検出するためのシナリオ S_s が作成可能か熟練者とともに評価した．レビューアに “日付に関する欠陥を検出する” というシナリオを与えたとしても，全てのレビューアが日付に関する欠陥を検出できるわけではないという意見を得た．また，日付に関する欠陥を詳細化し，分類や具体例を提示すれば検出できるかもしれないという意見を得た．そこで，T_s に該当する不具合 86 件をさらに分類した．その結果を表 9 に，細分類ごとの不具合件数を表 10 に示す．シナリオ S_s に細分類を付加すれば，業務知識を持つレビューアであれば，T_s に該当する欠陥が検出可能だろうという意見を得た．細分類 “特異日”, “期間の定義”, “日付チェック” が多く，これら 3 細分類で日付に関する不具合の約 64%の件数となることがわかった．熟練者との議論により，これらの細分類ごとにシナリオ (S_{s1}, S_{s2}, S_{s3}) を設定することもでき，効率化につながるだろうという意見を得た．

S_s に確認すべき場所を限定して，さらに検出が容易になるかどうかを調べるために，日付に関する不具合が特定

表 9 細分類

分類名	説明
期間の定義	期間の算出方法，開始日，終了日の定義の誤り 例) 未検閲の場合の利払い日 5 営業日以内の取り扱いが誤っている．
特異条件	特殊な条件の組合せの考慮漏れ 例) ブローカーの適用開始日が過去日付の場合の考慮が漏れている．
特異日	特殊な日の取り扱いの誤り 例) 特定条件下での利払い日
日次処理	日次処理の実施タイミングやチェックの誤り 例) オフバラの公正価値の日次登録チェックが誤っている．
日付チェック	日付の妥当性確認の考慮漏れ 例) 過去の記帳日に関する妥当性確認が漏れている．
日付の定義	時差や休日補正などの誤り 例) 金利更改日の定義の誤り
その他	上の分類に該当しないもの

表 10 細分類ごとの件数，重大度，対応工数

細分類	総数	重大度 A	重大度 B	重大度 C	対応工数の平均（人日）
期間の定義	15(17.4%)	0	12	3	2.47
特異条件	8(9.3%)	0	6	2	2.03
特異日	25(29.1%)	2	17	6	1.90
日次処理	3(3.5%)	0	2	1	2.67
日付チェック	15(17.4%)	0	11	4	1.93
日付の定義	11(12.8%)	0	6	5	2.27
その他	9(10.5%)	0	5	4	1.94

の機能に限定されるかどうかを調べた．対象ソフトウェアでは機能数が非常に多いため，機能を分類した機能カテゴリによって不具合の件数を集計した．結果を表 11 に示す．機能カテゴリにおいては，日付に関する不具合が “マスター系”，“業務共通系” に多く含まれていた．上述の細分類に加え，これらの場所を優先的に調べることをシナリオ S_s に含めることにより，効率的に日付に関する欠陥を検出できる可能性がある．

4.6 T_s を表すキーワードの自由記述項目 l^f における出現頻度 (仮説 2-1)

T_s を表すキーワード w を，“日”，“期間”，“期限” とし，不具合情報 $\mathcal{D}$ の項目 “現象”，“原因”，“対策” に含まれるかを調べた．これら 3 つの項目のうち 1 項目以上に 3 つのキーワードのいずれかが含まれていれば，自由記述項目に T_s を表すキーワードが含まれている，とした．

T_s に該当する不具合に対する適合率，再現率を求めた結果を表 12 に示す．上述のキーワード 3 つを少なくとも 1 つ含む D_i の約 85%が T_s に該当していた．また，T_s に該当する不具合のうち約 31%は 3 つのキーワードを 1 つも含んでいなかった．

4.7 l^f に出現する単語による欠陥種別特定支援 (仮説 2-2)

対象不具合情報の項目 “現象”，“原因”，“対策” の自由記述のテキストを日本語処理のフリーソフトウェアである mecab に与え，名詞と判断された単語の出現頻度を集計した．出現頻度は，D_i に k 回出現する場合も k 件の D_i に 1 回ずつ場合も同じである．項目 l^f ごとに出現頻度上位 20 件の単語と出現回数を集計した結果を表 13 に示す．

これらのキーワードが欠陥種別 $T_s', T_s'', \cdots$ を特定するために役立つかどうか，熟練者に確認したところ，“取引”，“銘柄” といった単語と自身の経験から，特定のパターンの実行順序を推測でき，そのパターンにおいて日付に関する欠陥種別以外の欠陥 $T_s', T_s'', \cdots$ を想定できるだろうという意見を得た．他にも “異動”，“通貨”，“解除” といった単語から一連の業務が想定でき，その業務に特化した欠陥種別を想像できるという意見も得た．

5. 考察

5.1 ヒアリングにより得られた業務に特化した欠陥種別 T_s

熟練者からスケジュール遅延リスク低減につながる可能性のある欠陥種別としてヒアリングした “日付に関する欠陥” を不具合情報から調べたところ，設計レビュー以前で検出できていた可能性のある不具合のうち，2 割弱にあたる 86 件の不具合が日付に関する不具合であった．熟練者との議論において，これらの不具合全てが検出できるとは限らないが，“特異日”，“日次処理” といった細分類や例示を加え，欠陥検出シナリオにすることによって，レビューでの検出の可能性が高まるだろうという意見が得られた．これまでに報告されている欠陥検出シナリオは，セキュリティ，要求機能の全てを網羅といった，汎用的な欠陥種別を想定したものが中心であったが，本評価により，これまで言及されたことがなかった，ソフトウェアが担う業務に特化した欠陥種別を想定したシナリオを作成することができることを示せた．

評価対象の不具合情報では，レビューで検出できたと考えられる不具合のうち，日付に関する欠陥のほうが，その他の欠陥と比較して，スケジュール遅延リスクが相対的に大きい不具合 (重大度 A, B) が多く含まれていた．修正工

表 11 機能カテゴリの分類

機能カテゴリ	日付に関する不具合	日付に関する不具合ではない
TBA・OTC 取引系	0	5
オフバラ取引系	0	7
その他現物系	2	17
マスター系	30	144
移行・切替処理	2	17
株式取引系	0	14
基盤・シス共	0	1
期中系	2	6
業務共通系	24	39
決済系	1	22
現物取引系	6	54
財務系	9	29
他システム I/F	4	19
帳票系	6	27

表 12 単語有無による分類

単語	適合率	再現率
「日，期間，期限」	84.9%	68.9%

数に関しても，日付に関する欠陥のほうが，相対的に大きくなっていた．また，重大度，修正工数の両方で，有意水準 1%で統計的有意差があった．ソフトウェアが担う業務に沿った欠陥種別は，そのソフトウェアにとって重要であったり複雑であったりする可能性が大きいことを示しているといえる．そのため，これらをレビューで検出すべき欠陥種別として設定することで，効果的，効率的に修正コスト低減，スケジュール遅延リスク低減につながる可能性が大きい．

評価において，日付に関する欠陥が特定の機能 (機能カテゴリ) に偏っていることが明らかになった．特定の業務が特定の機能に偏ることは一般的なソフトウェア設計のプロセスやアーキテクチャから容易に想像がつくが，本評価において実証的に示せた．特定の機能に偏っていることがわかれば，レビュー対象箇所を絞り込むことが可能になる．上述のスケジュール遅延リスク低減効果，修正コスト低減効果に加え，レビューの効率化をさらに進めることができる．また，文献 [7] に示されるような Focused, Customizable 等の条件を満たすリーディングテクニックのシナリオとして業務特化のシナリオを加え，汎用的な欠陥種別を検出するためのシナリオと併用することもできる．

評価では欠陥種別が不具合情報に含まれているかどうかを調査し，スケジュール遅延リスク低減や修正コスト低減につながる可能性が大きいことを示したが，実際にシナリオを用いてレビューで日付に関する欠陥が検出できるか評価することは今後の重要な課題の 1 つである．また，日付に関する欠陥検出のためのシナリオを用いることにより，用いなかった場合に検出できていた欠陥の検出に影響を与えるか評価することも今後の重要な課題の 1 つである．

5.2 不具合情報による欠陥種別特定の支援

本評価では，熟練者へのヒアリングによって欠陥検出によるスケジュール遅延リスク低減，修正コスト低減の効果が高い欠陥種別である “日付に関する欠陥” を得た．この欠陥種別を表わすようなキーワードが不具合情報に含まれているかどうかを調べたところ，“日”，“期間”，“期限” といった単語が不具合の原因，現象，対策の自由記述に含まれていた．この結果は，不具合情報の自由記述に含まれる単語から欠陥種別の特定ができる可能性を示している．

また，不具合情報の自由記述の項目に現れる単語を示すことにより，熟練者の欠陥種別特定支援が可能であることを示唆する結果が得られた．抽出した単語の中には熟練者からヒアリングした日付に関する欠陥を表わすキーワード “日” が含まれており，他にも “銘柄”，“取引”，“異動”，“通貨”，“解除”，“拠点” といった業務に特化した欠陥種別を特定するために必要となる可能性のある単語も含まれていた．また，熟練者による評価において，これらの単語をもとに具体的な欠陥種別を想起できるという意見を得た．今回は mecab の実行結果から得た集計結果をそのまま提示したが，一般語を除く等の方法を用いて，より強力な支援が期待できる．

6. おわりに

レビューにおける欠陥の早期検出がもたらす，スケジュール遅延リスク低減効果，欠陥修正コスト低減効果を大きくすることを目指し，ソフトウェアが担う業務に特化した欠陥種別の検出が可能か実証的に評価した．具体的には，熟練者へのヒアリングで得た業務に特化した欠陥種別が不具合情報に存在するか確認し，リスク低減，コスト低減を他

表 13 出現頻度上位の単語

現象		原因		対策	
単語	出現頻度	単語	出現頻度	単語	出現頻度
ため	222	チェック	120	よう	286
データ	96	メッセージ	63	チェック	195
時	95	抽出	63	場合	113
場合	93	よう	60	項目	99
チェック	92	書	50	登録	75
日	88	変更	39	一覧	73
設計	68	考慮	38	コード	69
画面	62	金	34	メッセージ	56
取引	61	残	33	仕様	53
設定	58	拠点	32	エラー	51
エラー	58	後	32	ファクター	47
処理	58	解除	30	出力	44
漏れ	56	リスト	27	拠点	43
表示	52	ところ	26	取得	35
銘柄	48	もの	25	訂正	34
部	47	ITB	25	情報	32
値	45	共通	24	異動	32
基本	45	存在	24	通貨	31
残高	44	状態	21	作成	31
検閲	44	押下	20	解除	30

の欠陥と比較した．また，その欠陥種別をレビューで検出するためのシナリオの妥当性を検討した．さらに，不具合情報に含まれるキーワードを抽出し提示することにより，ヒアリングで得た以外にも，業務に特化した欠陥種別の特定支援が可能か検討した．

評価では，大手銀行の外国債券システムを対象とした．システム開発の熟練者に業務に特化した欠陥種別をヒアリングし，“日付に関する欠陥” を得た．システム開発の結合テスト実施時に収集された不具合情報を調査したところ，レビューで検出できていた可能性のある不具合 488 件のうち 86 件が日付に関する欠陥であった．86 件の不具合をレビューで検出することにより，スケジュール遅延リスク低減効果が期待できるかを，不具合の影響範囲である重大度の分類で比較した．レビューで検出できた可能性のある他の不具合と比較して重大度大，中の不具合の比率が大きいことがわかった．また，不具合情報に含まれる修正コストに関しても 86 件の不具合のほうがレビューで検出できた可能性のある他の不具合と比較して大きいことがわかった．

レビューアに “日付に関する欠陥を検出する” というシナリオをわたすことによって，86 件の不具合を検出できるか検討した．単純に日付に関する欠陥の検出と指定するだけでは難しいが，特異日，日次処理といった細分類を付与したり例示したりすることにより，検出可能となるという意見を熟練者から得た．これまでのレビュー技法 (リーディングテクニック) では，汎用的な欠陥種別を想定しこれを検出するためのシナリオを定義しているが，ソフトウェアの業務に特化した欠陥種別を検出するためのシナリオを作成すれば，より効果の大きいレビューにつながることを示せた．

一般に業務とそれを実現する機能の間には強い関連があるため，レビュー対象を限定できることが予想されるが，今回の不具合情報を用いて，このことを実証的に評価することができた．優先してレビューする機能カテゴリをシナリオに加えることにより，さらなる効率化につながることが期待される．

リスク低減や修正コスト低減の効果が期待される欠陥種別の特定を計算機によって支援できるか明らかにするために，不具合情報の自由記述に含まれる単語の出現頻度を調査した．キーワード “日”，“期間”，“期限” を症状等の自由記述部分に含む不具合と，目視により日付に関する不具合と判断した不具合の適合率，再現率を調べたところ，適合率 7 割弱，再現率 8 割強であった．また，自由記述の中に出現する出現頻度上位の単語の中に，業務に特化した欠陥種別の特定を支援する単語が含まれていた．

謝辞 熟練者をはじめとして，データのご貸与，コメントをくださった方々に御礼申し上げる．本研究の一部は，文部科学省「次世代 IT 基盤構築のための研究開発」の委託に基づいて行われた．また，文部科学省科学研究補助費（基盤研究 B:課題番号 23300009）による助成を受けた．

参考文献

[1] V. Basili, S. Green, O. Laitenberger, F. Lanubile, F. Shull, S. Sorumgard, M. Zelkowitz: The Empirical Investigation of Perspective-based Reading, Journal of Empir-

ical Software Engineering, vol.2, no.1, pp.133-164(2006)
[2] C. Denger and F. Shull: A Practical Approach for Quality-Driven Inspections, IEEE Software, vol. 24, no. 2, pp. 79-86(2007)
[3] M. E. Fagan: Design and Code Inspection to Reduce Errors in Program Development, IBM Systems Journal, vol. 15, no. 3, pp. 182-211(1976)
[4] W. Humphrey: A Discipline for Software Engineering, Addison-Wesley Longman Publishing(1995)
[5] A. Porter and L. Votta: An Experiment to Assess Different Defect Detection Methods for Software Requirements Inspections, In Proceedings of the 16th International Conference on Software engineering, pp. 103-112(1994)
[6] A. Porter, L. Votta, V. Basili: Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment, IEEE Transactions on Software Engineering, vol.21, no.6, pp. 563-575(1995)
[7] F. Shull, I. Rus and V. Basili: How Perspective-Based Reading Can Improve Requirements Inspections, IEEE Computer, vol. 33, no. 9, pp. 73-79(2000)
[8] T. Thelin, P. Runeson, B. Regnell: Usage-based Reading: An Experiment to Guide Reviewers with Use Cases, Information and Software Technology, vol. 43, no. 15, pp. 925-938(2001)