研究コース 3 （Q Chans グループ）

高リスク不具合を検出する軽量なテスト優先度付け手法の提案

Proposal of Light Test-Prioritization Method to detect high risk bug.

リーダー：西田　尚弘（株式会社日新システムズ）
研 究 員：飯沼　真一（ソーバル株式会社）
　　糸川　喜裕（ブラザー工業株式会社）
　　中根　拓也（株式会社インテック）
主　査　：喜多　義弘（東京工科大学）
副 主 査：上田　和樹（日本ナレッジ株式会社）
アドバイザー：秋山　浩一（富士ゼロックス株式会社）

研究概要
テストマネジメントをおこなう上で，何らかの基準によってテストの優先度付けを行う必要がある．優先度付け手法の一つにリスクベースドテストが存在するが，リスク把握など，テスト実施に必要な情報を揃えるために手間がかかる．本稿では，高リスク不具合をより軽量な手法で検出することを目的とし，機能ごとのテストケース数の変化，不具合数からテスト優先度付けを行う手法（LTP-Method）を提案する．プロジェクト情報としてすでに揃っている情報のみで推定するため，新規情報の収集が不要な分，リスクベースドテストと比較して軽量で導入が容易である点に優位性がある．実製品のソフトウェアに対して LTP-Method を適用し，本手法の導入容易性，高リスク不具合の早期検出に関する有効性について検証する．

Abstract
In testing management, we need to decide priority of test cases with something rules. Risk-based-testing is one of the methods for deciding priority of test cases. It takes a long time to collect risk-information which is needed for using risk-based testing. In this research, we propose the Light-Test-Prioritization Method (LTP-Method) which detects the high risk bugs from the number of bugs and test cases each function. LTP-Method is easier to introduce than Risk-based-test because it uses only existing project information. We apply LTP-Method to the product software and verify to ease of use and effectiveness for early detection for high risk bugs.

1. 研究の背景

ソフトウェア開発では，対象製品をゼロから作り上げる新規開発よりも，既存製品をベースに開発を行う派生開発の割合が高くなっている．開発全体に対して派生開発が占める割合は 93%，新規開発は 7%である．派生開発では，図 1-1 に示すように開発工数が少ない場合が多い[1]．図 1-1 は 155 件の派生開発のソフトウェアに対して，開発工数（テスト工数を含む）を調査した表である．派生開発では，開発はベースのソフトウェアに対して追加機能を作れば良いが，テストでは機能追加した個所が，既存機能に対してデグレードしていないか確認するため，既存機能を含めたソフトウェア全体をテスト必要がある．しかしながら，図 1-1 で示したように十分な工数は与えられないため，何らかの方法でテストの優先度付けをする必要がある．派生開発では，機能追加，機能拡張を行った新機能部分と，修正を行っていない既存機能部分に分けることができる．新機能部分が最優先でテストされるのは当然である．既存機能は，新機能テスト後に残されたテスト工数の中で優先

度を付けてテスト実施することになる．そこで，本研究では派生開発での既存機能に対するテスト優先度付け手法について検討を行う．算出した優先度に従ってテストしたと仮定した場合に，どの程度不具合が早期検出できるかという有効性を確認する．

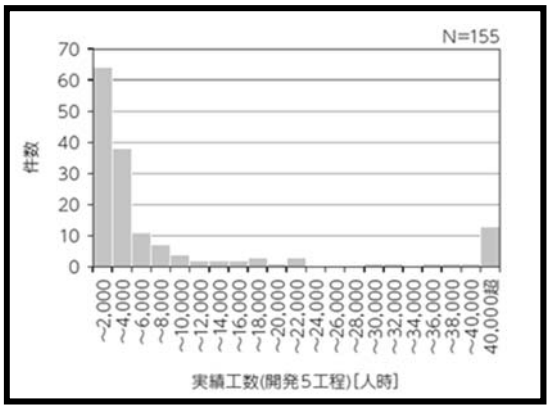

図 1-1 派生開発における工数分布

2. 先行研究

テスト優先度付けの手法として，リスクベースドテスト[2][3]の考え方が広く知られている．リスクベースドテストは対象ソフトウェアが潜在的に持つリスクに基づいて実施されるため，ソフトウェアの特性に対して深い理解が必要になり，必要に応じて任意のソフトウェア単位でのリスクを関係者にヒアリングすることもある．しかしながら，ヒアリング対象の開発者やプロジェクトマネージャーの時間が取れないなど，ヒアリングして情報取得することが難しい場合がある．しかしながら，ヒアリング関係者が共通の知識レベルを持っていない場合には，個人の主観による回答がなされる状況になり,バラツキのあるヒアリング結果が得られる可能性が高まる．

3. 課題解決のためのアプローチ

本研究では，リスクベースドテストの課題と捉えた，ヒアリングに要する時間を削減しつつ，個人の主観に依存しないテスト優先度付け手法を検討する．前述の課題を解決するため，提案するテスト優先度付け手法はプロジェクト情報から取得できる定量的な情報を用いる．優先度付け手法の適用対象は，派生開発での既存機能のテストを対象とし，新機能は最優先でテストされるため本手法の適用外とする．

本研究の最終的なゴールは，潜在している高リスク不具合を効率よくテストで検出することである．本年度はそのゴールに向かう前段階として，各テストケースという詳細な単位ではなく，任意の規則に従って複数のテストケースをまとめた，もう少し粗い単位でのテスト優先度付けを研究のスコープとする．本研究での任意の規則はソフトウェア機能群（機能群とは関連する複数の機能のまとまりのこと．サブシステムとも呼ぶ.）単位とし，ソフトウェア機能群単位でテストケース優先度付けを行うこととする．任意の規則にソフトウェア機能群単位を選択した理由は，リスクベースドテストでも機能群単位での優先度付けがなされている研究[3]がある点，4 人の研究員が属するそれぞれの組織で，テストケースや，不具合管理の大分類の単位としてソフトウェア機能群を用いていた点の 2 点による.これらの理由からテスト優先度付けの単位としてソフトウェア機能群単位を採用する．

3.1 本手法が応えるべき課題

本研究では，テスト優先度付け手法を検討し，特に高リスク不具合を検出するテストを優先的に実施することを重視する．まず，高リスク不具合の定義に関して,「ユーザ使用目的への影響」,「ユーザの経済的被害」から影響レベルを算出したシステム障害影響評価スケール[4]を参考にする．本研究での高リスク不具合の定義は，表 3-1 に示す階級 5～6 に当たる不具合である．ある不具合に対して,「ユーザ使用目的への影響」,「ユーザの経済的被害」で異なる階級が割り当たる場合は，より階級が高い方をその不具合のリスク階級と定義する．不具合の中でも特に高リスク不具合に着目した理由は，高リスク不具合が発生した場合の経済損失リスク[5]がより大きくなるためである．高リスク不具合を市場流出させることなくテストで検出することは，経済損失リスク抑制のために重要である．

3.2 本手法が応えるべき課題

本研究では，テスト優先度付け手法を検討し，特に高リスク不具合を検出するテスト

表 3-1　不具合内容に対するリスク度の定義（N/A は Not applicable を表す）

階級	ユーザ操作への影響	ユーザに対する経済的被害の状況
0	不具合に気付かない	N/A
1	一部のユーザがシステムの動作に違和感を覚える	N/A
2	ユーザの多くが不具合に気付く	N/A
3	ユーザの多くが不具合に気付きクレームを訴える人がいる	一部のサービスが停止し，ユーザの一部で軽微な経済損が出る可能性がある
4	ユーザの目的が一部達成できなくなる	一部のサービスが停止し，ユーザの一部で経済損が出る可能性がある
5	ユーザの目的が達成できなくなる	一部または全部のサービスが停止し，ユーザの一部で多大な経済損が出る可能性がある
6	N/A	一部または全部のサービスが停止し，多くのユーザで多大な経済損が出る可能性がある

を優先的に実施することを重視する．まず，高リスク不具合の定義に関して，「ユーザ使用目的への影響」，「ユーザの経済的被害」から影響レベルを算出したシステム障害影響評価スケール[4]を参考にする．本研究での高リスク不具合の定義は，表 3-1 に示す階級 5～6 に当たる不具合である．ある不具合に対して，「ユーザ使用目的への影響」，「ユーザの経済的被害」で異なる階級が割り当たる場合は，より階級が高い方をその不具合のリスク階級と定義する．不具合の中でも特に高リスク不具合に着目した理由は，高リスク不具合が発生した場合の経済損失リスク[5]がより大きくなるためである．高リスク不具合を市場流出させることなくテストで検出することは，経済損失リスク抑制のために重要である．

4. 提案　高リスク不具合を検出する軽量なテスト優先度付け手法（LTP-Method）

本章では，ユーザに多大な影響を与える高リスク不具合を効率的に検出するためのテスト優先度付け手法（Light Test-Prioritization Method to detect high risk bug：LTP-Method）を提案する．LTP-Method は，テスト現場でテストマネージャーがテスト優先度付けをする場面を想定しており，特に高リスク不具合を効率的に検出することを目的とする．

LTP-Method ではテスト優先度付けのために，高リスク不具合が潜在する機能群を推定する．この目的を達成するためには高リスク不具合を潜在率の高い順に推定する必要がある．推定対象に因果関係があると考えられる因子から，推定対象の将来的な値を推定する手法として回帰分析が存在する．回帰分析手法にはロジスティック回帰分析や，比例ハザード分析，重回帰分析[6]など様々な手法が存在する．本研究では，高リスク不具合潜在期待値を優先順位の重み付け（点数化）として量的に扱う目的から，定義した高リスク不具合のみを推定する必要がある．これには，量的な目的変数について，複数の説明変数から予測式を作れる特徴を持つ重回帰分析の使用が適切である．

LTP-Method では重回帰分析を用いて，高リスク不具合潜在の期待値に関連する因子ごとの係数を求め，求めた係数を用いて高リスク不具合を効率的に検出するためのテスト優先度付を行う手法を提案する．LTP-Method 構築のための手順を図 4-1 に示す．図 4-1 の各項目の詳細について以降の章で記述する．　高リスク不具合潜在の期待値を推定するために，ソフトウェア品質を推定できる要因について調査し[7][8]，研究員の属する組織での実情をヒアリングした結果，表 4-1 に示すように複数の要因を洗い出すことが出来た．その中から，LTP-Method ではモデル構築のためのデータ収集を容易にすることを狙い，テスト現場で取得可能な情報を重回帰分析の因子に採用する．

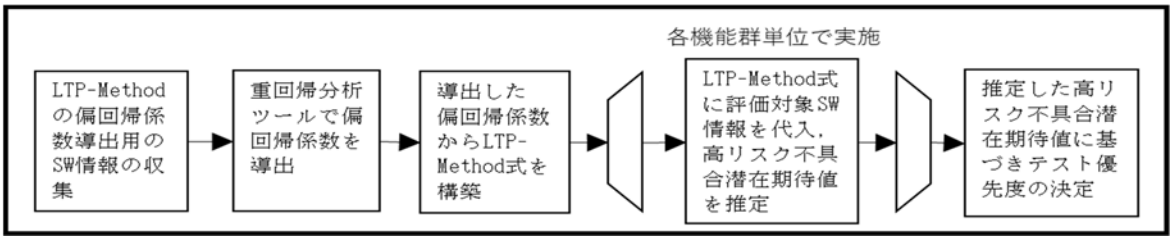

図 4-1 LTP-Method 実施手順

表 4-1　ソフトウェア品質を推定できる要因

No	要因	分類
1	旧 SW での高リスク不具合数	不具合数
2	旧 SW での全不具合数	不具合数
3	旧 SW での高リスク不具合を除く，その他の不具合数	不具合数
4	新 SW でのテストケース数	テストケース数
5	旧 SW でのテストケース数	テストケース数
6	新 SW に対する旧 SW からのテストケース数の増加割合	テストケース数
7	テスト日程	日程
8	ソフトウェア規模	ソフトウェア情報
9	プロセス成熟度	プロセス

4.1　LTP-Method 構築のために必要な情報

派生開発では，テスト現場でこれまでのソフトウェアバージョンの評価結果が保存管理されている場合が多い．ソフトウェアの特性は新旧ソフトウェアで類似性が高いため，旧ソフトウェアバージョン(以下,旧 SW)での不具合情報を新ソフトウェアバージョン(以下,新 SW）におけるテスト優先度付け手法に活用することは有効である．これは，既存機能への機能追加はベースになるソフト品質の影響を大きく受けるためである．

LTP-Method 構築のために，歴代の SW が存在する製品において，評価対象の新 SW の直近 1,2 世代前の SW 情報を準備する．

4.2　LTP-Method 構築のための説明変数選定

LTP-Method ではテストケース数，不具合数を説明変数として採用する．これらは，テスト現場で取得容易な情報であるため情報取得のための工数が少なくすむ点，定量的な情報のため取得情報に関して属人性を解消できる点がメリットである．表 4-1 で列挙したように，テストケース数，不具合数共に新旧 SW で選択できる情報にはいくつかの種別がある．採用する説明変数（No.1，No.3，No.6）と採用理由を表 4-2 にまとめる．

テストケース数に関しては，表 4-2 に記載した理由より，No.6（新 SW に対する旧 SW からのテストケース数の増加割合）が高リスク不具合の推定に有効と考える．この考えは，旧 SW でのテストケースを新 SW に引き継いでいることを前提としている．新 SW ごとにテストケースを新規作成している場合は，旧 SW のテストケースとの関連性が無くなり，増加割合を求めることに意味を持たないためである．また，本提案では既存機能を対象とするため増加割合を求める際に分母がゼロになることはない．

不具合数に関しては，表 4-1 に記載した No.1～3 の不具合数の分類のうち，どの要因を優先して採用すべきか判断が難しい．そこで，表 4-1 の No.1～3 の不具合数の分類に対して,エクセルの分析ツールで高リスク不具合と不具合数に関する要因の相関係数を求める．求めた相関係数は表 4-3 にまとめる．表 4-3 より No.1（旧 SW での高リスク不具合数）が最も相関が強いため説明変数に採用とする．次いで No.2（旧 SW での全不具合数）の相関が高いが，全不具合数には，採用済みの高リスク不具合数を含み，因子間の関連を持ってしまうため採用はできない．よって，次いで相関が高い，No.3（旧 SW での高リスク不具合

表 4-2　LTP-Method 説明変数の選択理由

No	説明変数	選択理由
1	旧 SW での高リスク不具合数	新旧 SW で不具合傾向は類似性があると推測できるため，旧 SW で高リスク不具合が多かった機能群は，新 SW でも高リスク不具合が多いと考える
3	旧 SW での高リスク不具合を除く，その他の不具合数	旧 SW で不具合数が大きい機能群は品質が低く，新 SW でも高リスク不具合が含まれる確率も高いと考える
6	新 SW に対する旧 SW からのテストケース数の増加割合	テストケースの追加は旧 SW で品質が低かった機能群や，新 SW での機能追加など，不具合が入り込みやすい機能群に対して行われる．そのため，テストケースの増加と高リスク不具合潜在の期待値は関連があると考える

表 4-3　高リスク不具合と要因の相関係数

No	要因	相関係数
1	旧 SW での高リスク不具合数	0.59
2	旧 SW での全不具合数	0.52
3	旧 SW での高リスク不具合を除く，その他の不具合数	0.51

を除く，その他の不具合数）を説明変数に採用する．

なお，No.7（テスト日程）については，製品全体の日程は取得可能でも，機能群単位での詳細な日程は取得が難しいため不採用とする．No.8（ソフトウェア規模），No.9（プロセス成熟度）については，関係者へのヒアリングが発生するため，本提案の前提であるテスト現場での容易な情報取得という観点から外れるため除外する．

表 4-3 で示す 3 つの因子を説明変数として採用し，機能群ごとの高リスク不具合潜在の期待値推定に用いる．高リスク不具合の推定式を式①で定義する．

$$Y = \alpha_1 X_1 + \alpha_2 X_2 + \alpha_3 X_3 + \alpha_0 \cdots ①$$

Y：高リスク不具合が潜在する期待値，X_1：旧 SW での高リスク不具合数，
X_2：旧 SW での不具合数，X_3：新旧 SW 間でのテストケース数の増加割合，
α_0：回帰定数，$\alpha_1, \alpha_2, \alpha_3$：偏回帰係数

本提案では機能群ごとのテスト優先度付けを行うため,上記式を各機能群単位で実施する．次章で，各因子間の重みづけについて検討する．

4.3　LTP-Method 構築のための偏回帰係数，回帰定数の算出方法

偏回帰係数の検討にはエクセルの重回帰分析ツールを用いる．エクセルの分析ツールを選択した理由は，テスト現場での適用が容易であると考えたためである．

分析ツールの Y軸（目的変数）を 1 世代前の旧 SW での高リスク不具合数，X軸（説明変数）に 2 世代前の旧 SW での高リスク不具合数，2 世代前の旧 SW での不具合数，1 世代前と 2 世代前との SW 間でのテストケース数の増加割合として，エクセルの重回帰分析ツールを実行することで偏回帰係数と回帰定数を導く．偏回帰係数導出に用いた元データの詳細は付録 1-1，付録 1-2，付録 1-4，付録 1-5，機能群ごとの偏回帰係数は付録 2-1，付録 2-2 に記載している．

4.4　LTP-Method を用いた高リスク不具合潜在期待値の算出方法

高リスク不具合潜在期待値を算出するため，式①に対して 4.3 章で導いた偏回帰係数，回帰定数を代入する．また，1 世代前の高リスク不具合数をX_1に，1 世代前の高リスク不具合の除くその他の不具合数をX_2に，1 世代前の SW から新 SW へのテストケース増加割合を

表 5-1 偏回帰係数と回帰定数

	切片	旧SWでの 高リスク不具合数	旧SWでの不具合数	新旧SW間での テストケース数の増加割合
製品A	0	0.69183	0.113809	0.509831
製品B	0	0.604606	0.375137	0.128912

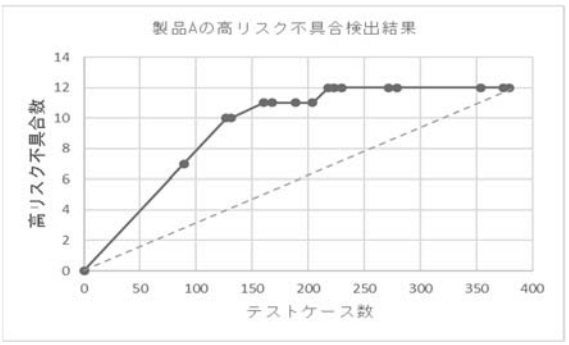

図 5-1 製品 A への LTP-Method 適用結果

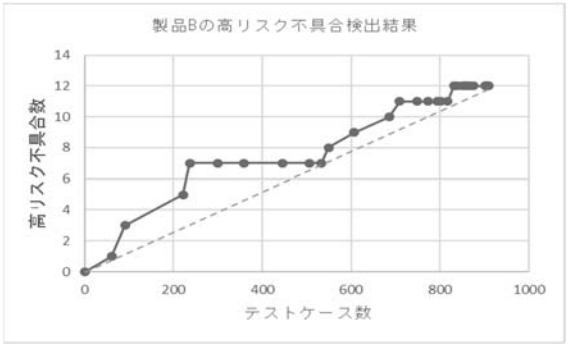

図 5-2 製品 B への LTP-Method 適用結果

X_3に代入する．上記③より評価対象の新 SW の各機能群に対してY値を求め，高リスク不具合が潜在する期待値を求める．新 SW 情報の期待値を付録 1-3，付録 1-6 に記載する．

5. LTP-Method 有効性のための実験と評価

5.1 評価の目的

LTP-Method に旧 SW の不具合情報を入力すれば，一意に新 SW の高リスク不具合潜在の期待値を推定できる．LTP-Method の適用をソフトウェア機能群単位で実施することで，機能群単位の高リスク不具合潜在確率を推定でき機能群間でのテスト優先度付けに有用な情報が取得できることを実証する．

5.2 実験手順

LTP-Method の有効性評価として，まず LTP-Method の偏回帰係数と回帰定数を導く．その手段として，エクセルの重回帰分析ツールを用いる．重回帰分析ツールへの入力データとして，歴代の SW が存在する製品において 1,2 世代前の SW 情報を使用する．

本研究で偏回帰係数，回帰定数決定のために用いた情報は，3 世代分の派生開発を行った製品 A，同じく 3 世代分の派生開発を行った製品 B の 2 製品分の SW 情報を使用する．

5.3 実験結果

重回帰分析ツールを用いて求めた製品 A，製品 B の偏回帰係数，回帰定数は表 5-1 になる．導いた偏回帰係数と回帰定数を使用し，LTP-Method の式を作成する．

製品 A 用 LTP-Method：$Y = 0.691830 \times X_1 + 0.113809 \times X_2 + 0.509831 \times X_3$・・・②

製品 B 用 LTP-Method：$Y = 0.604606 \times X_1 + 0.375137 \times X_2 + 0.128912 \times X_3$・・・③

式②を用いて，製品 A の新 SW に適用した結果を図 5-1 に示す．また，式③を用いて，製品 B の新 SW に適用した結果を図 5-2 に示す．機能群ごとの適用結果については付録 3-1，付録 3-2 に記載している．

図は X 軸に実施テストケースの累積数，Y 軸に実際のプロジェクトで機能から検出された高リスク不具合数を設定する．LTP-Method を適用し，高リスク不具合潜在の期待値（Y）が大きい機能群から順にテストを行う際の高リスク不具合検出数の実績値を実線で示し，優先度付け手法を適用せずテストを行う際の高リスク不具合検出の期待値を点線で示す．

表 6-1　1 機能群当たりの LTP-Method 構築，適用時間

手法	SW 情報収集に要する工数	偏回帰係数算出に要する工数	手法適用に要する工数	合計工数
リスクベースドテスト	20～120 人分（2 人分の工数）	-	N 人分	20～120＋N 人分
LTP-Method	5 人分	2 人分	1 人分	8 人分

なお，優先度付け手法を適用しない場合は，ランダムに高リスク不具合が検出されると仮定し，図の始点と終点を結ぶ線で示す．

図 5-1，図 5-2 の結果より，製品 A，B 共に LTP-Method を適用した場合は，適用しない場合と比べて高リスク不具合の検出数が多い．このことから，LTP-Method を適用した結果得られた優先順位に従ってテストを行えば，どのタイミングにおいても LTP-Method を適用しない場合の期待値よりも多くの高リスク不具合が検出できることが確認できる．

しかしながら，本実験での LTP-Method の検定結果より，予測はできるものの，他の製品やプロジェクトで本 LTP-Method を適用するには，信頼性が低いことも併せて分かった．検定結果については，付録 4-1 に記載している．

6. 考察

6.1　製品 A，B への LTP-Method の適用結果

図 5-1 より製品 A では約 50%のテストケース数のテストを行うと，12 件中 11 件の高リスク不具合が検出される．対して，図 5-2 より製品 B では約 50%のテストケース数のテストを行うと 12 件中 7 件の高リスク不具合が検出される．製品 A と比較して製品 B の検出精度が劣る理由は，製品 B 用の LTP-Method を学習した旧 SW が，製品プラットフォームの変更直後の製品であるためである．高リスク不具合が各機能群で平均的に発生しており，テスト対象の新 SW と不具合の発生傾向が異なるためと考える．本研究では，LTP-Method を学習したサンプル数が 1 セット分のみであること，LTP-Method に採用した説明変数が研究員の取得可能な情報であるという制限があったため十分な要因抽出ができなかった．そのため，学習サンプルとは異なる傾向を示した機能群が正確に推定できなかったと考えられる．学習サンプルを増やし偏回帰係数の学習を重ねることや，説明変数の見直しにより推定精度と回帰式の信頼性向上が期待できるかを検証することが今後の課題である．

6.2　リスクベースドテストと比較した LTP-Method の導入容易性

今回，LTP-Method 構築，適用のために研究員が要した時間の内訳を表 6-1 に示す．表に記載する工数は，1 機能群のテスト優先度決定に要する工数である．比較のため，研究員の経験によるリスクのヒアリングに関する時間を記載する．ヒアリング関係者は少なくとも質問者，回答者の 2 人は工数が必要なため表には 2 人の工数の加算値を記載する．なお，研究員はリスクのヒアリングに関する経験のみ有し，リスクベースドテストを用いてテスト優先度付けを実施した経験は無いため，リスクベースドテスト実施の工数は N 人分と仮の数値を表に記載する．表 6-1 より，リスクベースドテストに比べ LTP-Method は要する工数が少ないため，工数観点で LTP-Method の導入容易性が確認できた．

6.3　リスクベースドテストと比較した LTP-Method の手法安定性

LTP-Method では，新旧 SW 情報の不具合数，テストケース増減数という定量的な情報のみを用いてテスト優先度付けを行うため，リスクベースドテストにおけるリスク判断などの属人性は介在しない．そのため，テスト関係者の経験によらず同一条件下では常に一定のテスト優先度付けを行うことができる点で手法安定性の有効性について確認できた．しかしながら，プラットフォーム変更や，リファクタリングなどにより，テスト対象の新 SW の傾向が旧 SW と変わる場合は，旧 SW 情報を用いた高リスク不具合潜在の期待値推定

は推定精度が低下する．そういった場合は，開発関係者に新 SW の品質をヒアリングするリスクベースドテストに有効性があるため，状況に応じて併用することが望ましい．

7. おわりに

7.1 まとめ

ソフトウェア開発で多くの割合を占める派生開発を対象とし，テスト工数削減の対象になることがある既存機能に対して，テスト優先度付け手法の研究を行った．先行技術を調査して，リスクベースドテストのヒアリングに関する工数がかかる点，ヒアリング結果に属人性がある点が分かり，それを課題と捉えた．これらの課題を解決するため，プロジェクト管理されている定量的な SW 情報を用いることで，軽量にテスト優先度付けを行う手法の LTP-Method を考察した．LTP-Method では不具合が流出した際の経済損失リスクが大きい，高リスク不具合の検出を主眼に置いた．

実験により，LTP-Method を適用した結果得られた優先順位に従ってテストを行えば，LTP-Method を適用しない場合の期待値よりも多くの高リスク不具合が検出できることが確認できた．また，LTP-Method ではモデル構築のための SW 情報収集に関する時間は 1 機能群当たり 5 分と，リスクベースドテストと比較して手法適用のための事前準備時間が少ないことが確認できた．併せて，LTP-Method 構築に必要な情報は定量的な SW 情報のみを用いるため，属人性が排除されており，手法が安定的に使用できると考えられる．

7.2 今後の課題

本研究での残課題は以下の 2 点である．

1 点目は，本研究に使用できるデータセットが 2 製品分しか準備できなかったため，LTP-Method の有効性確認が十分に検証できなかったことである．今回評価した 2 製品では有効性が確認できたが，更にサンプル数を増やして，説明変数の選択，偏回帰係数の算出に関する妥当性について検証を行うべきである．また，本研究ではデータが準備できなかったリスクベースドテスト手法を適用した際の結果と比較検証をすることも今後の課題である．

2 点目は，リスクベースドテストと LTP-Method の併用の検討である．本研究では LTP-Method の学習サンプル数が少なく，学習用の旧 SW の高リスク不具合発生状況が，評価対象の新 SW と異なっていた場合に LTP-Method の推定精度が低下した結果が得られた．その課題を解決するため，大規模な変更や影響範囲が大きい変更が入り，旧 SW と不具合傾向が変わる可能性がある機能群は，リスクベースドテストを適用してリスク把握した上で優先的なテスト実施を検討し，それ以外の機能群に対して LTP-Method を適用することを検討する．評価対象 SW の傾向に合わせて 2 つの手法を組み合わせることで，各手法を単独で使用するよりも効率的に高リスク不具合の検出ができることを検証する必要がある．

8. 参考文献

[1] 情報処理推進機構(IPA)，“組込みソフトウェア 開発データ白書 2017”

[2] 町田 欣史，リスクベースドテストにおけるリスク分析の妥当性評価手法の提案

[3] 西森雅峰，金田重郎，芳賀博英，井田健太，佐々木亮太，“タグを用いたリスクベースドテストの効率化”、情報処理学会 全国大会講演論文集 第 72 回 pp.345-346

[4] 村松 昭男，“組込みソフトウェア開発向け品質作り込みガイド ESQR 解説”，P11

[5] 情報処理推進機構(IPA)，“「海外における IT 障害の影響及び対応策に関する事例調査」報告書”，pp.105-108

[6] 木下英明，“多変量解析の”からくり“をどう捉えるか その２ 重回帰分析”，活水論文集 第 48 集 pp.47-48

[7] 情報推進機構 技術本部 ソフトウェア・エンジニアリング・センター，“「定量的管理基盤メトリクス分類表有効性調査」－概要調査報告書―”

[8] 情報処理推進機構(IPA)，“メトリクス分析手法を用いた試験品質向上の取組み”

付録 1　機能群ごとのテストケース数及び不具合情報

付録 1-1　製品 A の 2 世代前の SW 情報

	テストケース数	不具合数	高リスク不具合数
機能1	3	5	1
機能2	45	32	4
機能3	15	23	1
機能4	88	23	2
機能5	12	7	0
機能6	19	3	0
機能7	82	80	3
機能9	11	3	0
機能10	31	40	0
機能11	1	1	0
機能12	3	2	0
機能13	33	30	4
機能14	8	7	0
機能15	4	7	1
機能16	1	3	0
機能17	3	0	0
機能18	4	9	0
機能19	20	21	0
機能20	1	4	1
Total	384	300	17

付録 1-2　製品 A の 1 世代前の SW 情報

	テストケース数	不具合数	高リスク不具合数
機能1	7	0	0
機能2	19	9	2
機能3	16	11	1
機能4	6	2	0
機能5	13	3	0
機能6	12	2	0
機能7	23	15	1
機能8	2	1	0
機能9	17	5	1
機能10	15	1	0
機能18	5	12	0
機能19	14	9	0
Total	149	70	5

付録 1-3　製品 A の評価対象の新 SW 情報

	テストケース数	不具合数	高リスク不具合数
機能1	5	2	0
機能2	38	10	3
機能3	75	3	0
機能4	42	10	0
機能5	15	2	0
機能6	8	4	0
機能7	89	34	7
機能8	20	0	0
機能9	21	0	0
機能10	29	10	1
機能12	5	0	0
機能14	14	3	1
機能18	7	3	0
機能19	4	0	0
機能20	7	9	0
Total	379	90	12