変更の影響範囲を特定するための「標準調査プロセス」の提案

Unified investigating process to Detect the range Affected by specification changes

アンリツエンジニアリング株式会社 プロトコル技術部
ANRITSU ENGINEERING CO.,LTD. Protocol Engineering Dept.
○多田 一成[1)] ○宇田 泰子[2)] ○川井 めぐみ[3)] ○伊藤 友一[4)]
○Kazunari Tada ○Yasuko Uda ○Megumi Kawai ○Tomokazu Ito

Abstract

In derivational development, developers need to meet frequent requirements of changes on time. Detecting the range affected by specification changes efficiently and without omission is a must to achieve the goals of quality, cost and delivery (QCD) and therefore to succeed in our business. Now we took a survey from our developers and found out that each developer finds the parts in range affected by specification changes need to be changed and keeps records of those in various ways. In this way, we cannot expect stable high quality. We compared XDDP（eXtreme Derivative Development Process）, which is supposed to be an effective approach in derivational development with development process that has been done in the field. And we found that there is a difference in the search of the source code. In addition, we found that there is a problem with this search method. In order to solve these problems, we defined a "standard survey process" for detecting the range of influence changes and summarized the use of the research process as a "Survey Process Guide". We tried this procedure in a project whose tests have failed, and we succeeded in reducing the number of errors in finding the range affected by specification changes.

1. はじめに

既存システムへの機能追加や部分的な変更を行う派生開発においては，様々な内容の変更要求に対して迅速に対処することが求められる．しかし，実際は開発工程の後半に発覚する不具合によって手戻りを繰り返すケースが発生している．特に変更による影響範囲を特定できないことに起因する不具合が後を絶たない.

そこで派生開発の現場で発生している不具合事例を分析したところ，影響範囲の調査が担当者に依存し，その結果に差が生じていることがわかった．そして，多くの担当者が調査の一環でソースコードを調べる際に，無意識に辿っているコードリーディングの順番（以降，探索ルート）に問題があることがわかった．問題の探索ルートでは，処理や構造を把握する調査で，無意識に変更箇所や影響範囲を特定する調査を行うなど，調査目的の異なる調査であっても区別なく調査を行ってしまい，変更による影響範囲の特定漏れを引き起こしやすい状況を作り出している．また，調査結果としてメモ書きしか残せていないため，レビューにおいて調査の抜け漏れに気付きにくいこともわかった．プロジェクトの QCD を確保しながら，様々な内容の変更要求に対処するためには，効率よく漏れなく，変更による影響範囲を特定できるようにしなければならない．そこで我々は，調査方法および調査結果の人によるバラツキ(属人性）を排除した調査プロセスの水

1) アンリツエンジニアリング株式会社　第一事業本部　プロトコル技術部
Protocol Engineering Dept. 1st Business Div. ANRITSU ENGINEERING CO.,LTD.
神奈川県厚木市恩名 5-1-1 〒243-0032 Tel:046-296-6697 e-mail:Kazunari.Tada@anritsu.com
5-1-1 Onna, Atsugi-shi, Kanagawa, 243-0032 Japan
2) アンリツエンジニアリング株式会社　ANRITSU ANGINEERING CO.,LTD.
3) サントリーシステムテクノロジー株式会社　Suntory System Technology Limited
4) ＴＩＳ株式会社　TIS Inc.

準を高めるために，調査を「事前調査」「変更箇所調査」「影響調査」の 3 つのステージに分類し，探索ルートを内容に含む「標準調査プロセス」を定義した．そして，現場で有効に使用するために，「調査プロセスガイドライン」としてまとめた．検証の結果，「標準調査プロセス」を適用することで，適切な調査のステージで適切な調査を行うことができた．また，レビューのインプットになる調査結果も残すことができ，調査の抜け漏れを防ぐことができた．

以降，第 2 章では調査における現状の問題点を分析し，課題を整理する．第 3 章では，解決策の「標準調査プロセス」と「調査プロセスガイドライン」を提案する．第 4 章では，「標準調査プロセス」と「調査プロセスガイドライン」の適用結果を述べる．第 5 章では，提案の評価を行う．

2. 現状分析

2.1 不具合事例の分析

我々は，自身の現場でどのような不具合が発生しているのかを知るために，システムテストで発生した 115 件の不具合事例を集め，以下の 4 つのカテゴリに区分し分析した．カテゴリ別の発生件数とカテゴリ別の手戻り工数を図 1 に示す．

(A) 変更要求の問題
　変更要求を正しく捉えなかったことに起因する不具合
(B) 変更内容の問題
　変更仕様を正しく設計できなかったことに起因する不具合
(C) 調査方法の問題
　変更の影響範囲の特定，変更による影響を正しく洗い出すことができなかったことに起因する不具合
(D) 実装の問題
　正しく実装できていなかったことに起因する不具合

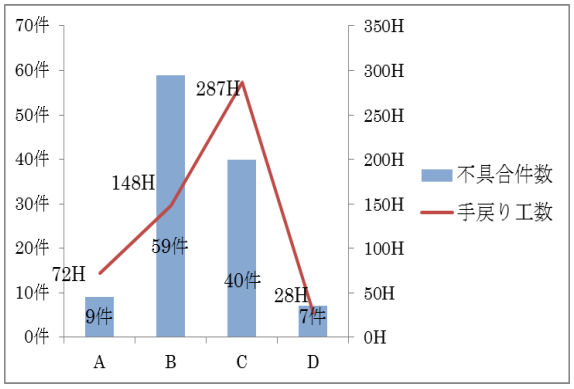

図 1　不具合件数と手戻り工数

この分析結果から，不具合件数の第 1 位が「変更内容の問題」の 59 件，第 2 位が「調査方法の問題」の 40 件，手戻り工数の第 1 位が「調査方法の問題」の 287 時間，第 2 位が「変更内容の問題」の 148 時間となることがわかった．「変更内容の問題」の発生原因の多くは「ソースコード上で変更箇所を見つけ次第に変更し，十分な変更仕様の検討が行われていない」ことが原因であることから，既存の技術である XDDP[1][2][3]を適用することで解決可能であることがわかっている．そこで我々は，不具合件数が「変更内容の問題」に対して若干少ないが，手戻り工数が第 1 位の「調査方法の問題」を解決することで，プロジェクトの品質・コスト・納期（QCD）に与える影響の低減を図ることにした．

2.2 変更の影響範囲の調査プロセスの実態

派生開発において変更の影響範囲をどのように調査しているのか，その実態を明らかにするため，我々の現場の開発メンバー27 名を対象にアンケートを実施した．アンケートでの質問とその回答からわかったことを以下に示す．

(1) どのような調査方法で調査を行っているのか？
　基本の構造を理解するために行う調査や変更箇所を特定するための調査，変更による影響範囲を特定するための調査を一緒に行っている人が多い．
(2) 調査結果をどのように記録しているか？
　調査結果を文書化していない．または，文書化していても記載内容が人によって様々である．記載すべき内容が決まっていないため，記載すべきことに抜け漏れがある．

(3)調査範囲を決めて調査を行っているか？
　調査範囲を明確に設定せずに，調査を開始している人が多い．
(4)調査において，ムダだと感じている点はあるか？
　調査のたびに，同じ箇所を調査した経験のある人が多い．
(5)調査不備による不具合の発生経験の有無は？
　約 60%の人が経験ありと回答しており，半数以上を占めている．
(6)調査の途中で気になる処理を見つけた場合,その処理を調査することはあるか？
　構造を把握する調査中に，変更箇所らしきもの見つけ，変更箇所の調査に移るなど，無意識に調査目的が変わることがある人が多い．

以上から，派生開発の現場で行われている影響範囲の調査は個人の力量に多く依存していることがわかった．これが「調査方法の問題」に起因する不具合を引き起こしていると考えられる．

2.3　先行研究

変更の影響を受ける範囲を特定する調査プロセスの問題を解決するために，開発プロセスである XDDP[1][2][3]の技術や先行研究[4]が参考にならないか確認した．

まず，XDDP では，変更箇所の調査において適当な設計資料が残されていない場合，“スペックアウト”と呼ばれる方法を用いてソースコードを調査する過程で調査資料を残し，見つけた変更箇所は変更要求仕様書に書き出して残している．これにより，ソースコードを探す過程で該当する箇所を見つけ次第に変更してしまう状況を回避できる．しかし，調査資料の残し方やその内容に関しては，書籍等でも紹介するに留めている．

次に，大規模複雑化したソフトウェアでの変更の影響を受ける箇所の見落としを防止し，影響を受ける箇所の調査の属人性を低減する手法がある[4]．これは階層化したトレーサビリティマトリクスを活用し，影響を受ける箇所を表現する方法と変更パターンごとの影響を受ける箇所の調査方法が論じられている．この手法は，変更要求・テスト項目・ソフトウェア構造の関係を可視化する手法であり,調査プロセスの実施タイミングが異なるため適用することが困難と判断した．

上記の結果から，調査プロセスの問題を解決する方法は書籍等で紹介されているが，現場で適用できるほど具体化はされていない．そこで，我々は，調査の問題にフォーカスを絞り，現場で活用できる具体的な施策として，変更による影響範囲の特定漏れを低減するための調査プロセスについて検討した．

3.　解決策

3.1　初心者と熟練者と XDDP の調査プロセスのギャップ

前章で，担当者ごとに調査プロセスが異なることがわかった．調査プロセスの相違点を明らかにするために，実務経験の少ない担当者（以降，初心者），実務経験が多い担当者（以降，熟練者）と，派生開発で効果的な手法とされる XDDP の 3 つの調査プロセスを比較した．

その結果，それぞれの調査プロセスには，探索ルート，調査資料の残し方，ソースコードの理解の仕方に違いがあることがわかった．特に初心者は，それぞれの関数(モジュール)の役割を確認することをせず，変更箇所を探すことによって影響箇所の特定漏れを起こしやすい．また，調査の結果を残していない，あるいは成果物をレビューしても調査目的・内容・範囲などがわからず，調査の抜け漏れに気付きにくい．そこで，3 つの調査プロセスのソースコードの解析の仕方を確認したところ，探索ルートの違いにより，調査の範囲，調査資料の残し方，ソースコードの理解の仕方に違いが出ることがわかった．このことから，影響箇所の特定漏れを起こす原因は，探索ルートであるという結論に至った．

3.2　探索ルートの問題

ここでは，3.1 で述べた探索ルートの問題点を示す．図 2 は問題がある探索ルート A（破線）とそうでない探索ルート B（実線）を示している．探索ルート B は XDDP でも勧めている方法で，調

査している関数内に新たな関数が現れても，個々の関数ごとに関数の最後まで調査する．探索ルート A は，関数が現れるたびに，その関数の中に次々と入って調査を行う．探索ルート B より影響範囲の特定漏れを起こしやすい探索ルート A の問題点 4 つを述べる．

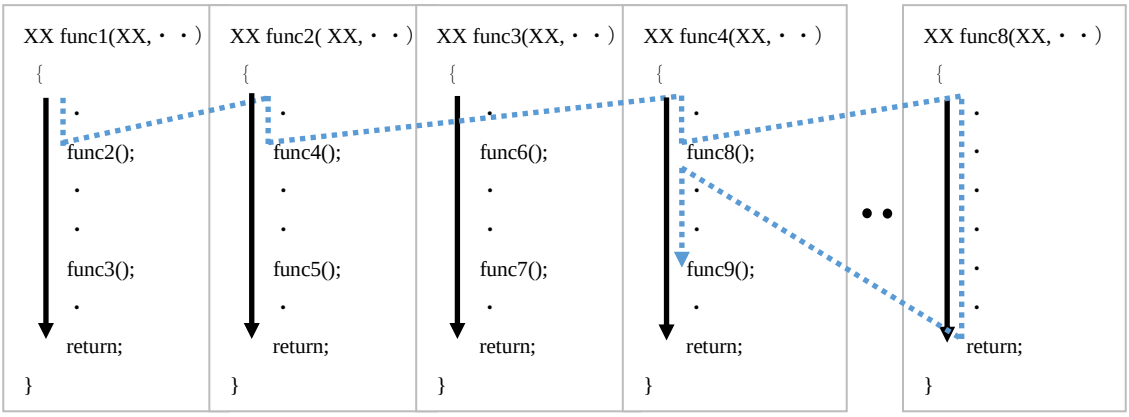

図 2　ソースコードの探索

(1) 問題点 1(狭い範囲しか調べられないことによって起こる問題)

探索ルート B がソースコードの処理を構造「面」で捉えることに対して，探索ルート A は処理を順番「線」で捉える．そのため，探索ルート A の調査範囲は B に比べると狭く，他の変更箇所に気付けないという問題に繋がる．調べた範囲が限定的で，より効果的な変更箇所が他にあることに気付けないこともある．また，何度も同じ関数を往来するなど探検隊のような状態に陥り，ソースコードを辿ることに時間がかかり，調査に使える工数内に必要な調査を十分できないということも起きてしまう．

(2) 問題点 2(調査目的の違う調査を一緒に行うことによって起こる問題)

変更箇所や修正方法によって影響を受ける箇所が変わるため，「変更箇所をすべて特定」した後に「影響を受ける箇所を特定」する必要がある．探索ルート A の調査では，目的を意識せず「変更箇所を特定」しながら「影響を受ける箇所を特定」する．そのため，後で変更箇所が変わることで，それによる影響を受ける箇所も変わることに気付かず，影響を受ける箇所を特定できないという問題が発生する．

(3) 問題点 3(構造を把握しないことによる問題)

一般に，機能を実現するために 10 数個から数 10 個の関数がそれぞれの役割を分担して組み立てられている．この状況で探索ルート A の調査を実施すると，関数群の役割を把握できないため，変更に対する影響を受ける箇所を特定しづらい．

(4) 問題点 4(調査結果についての問題)

探索ルート A では，調査中の関数の中に新たな関数が現れると，その関数の中に入るため，図 3 の辿った経路を忘れないようにするために記録したメモしか残らない．そのため調査結果がレビューや次の派生開発にも利用できる調査資料にならない．

一方，探索ルート B では，1 つの関数ごとに処理を構造「面」で捉えソーススコードを広く浅く調査するので，探索ルート A よりも早い段階で各関数がどのような機能を担っているか見えてくる．変更によってどの関数が影響を受け，どの関数が影響を受けないかがわかりやすくなり，そのため，どの関数を調査しなければならなかを切り分けできるので，無駄な調査工数が発生しなくなる．そして処理を構造「面」で捉えることで，図 4 の関数の構造図や PAD[5]を残すことが容易である．また構造図を作成しながら，関数の入力(引数)やコメントも記載しやすい．図 4 で func3 の右上に記載した●は，func3 を調査した際に PAD を作成したことを示している．func5 の下に記載したグランドマークは，調査した結果，関数が無いことを示している．そのため，func5

は調査済みということがわかる．func4，func6，func7 はグランドマークがついていないので，まだ調査していないことがわかる．調べた範囲を図として表現することで，調べ終わった関数とそうでない関数を明確に切り分けることができる．様々な情報を構造図上に記載するので，各関数の影響範囲も確認しやすい．

このように，探索ルート B は A に比べて，変更箇所や影響範囲の特定漏れを起こしにくく，短い時間で広い範囲を調査することができ，信頼性・活用性・再利用性の高い調査結果を残せることがわかった．

以下が内部データ保持の処理
func1()
//受信データ設定
func2()
次に，各大項目レベルでの値格納処理に飛ぶ
func4()
・

図 3　探索ルート A の成果物(メモ)

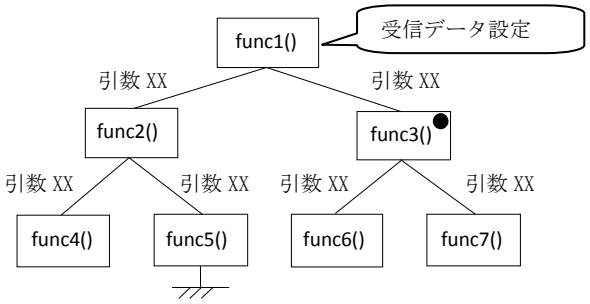

図 4　探索ルート B の成果物(構造図)

3.3　標準調査プロセスの定義

探索ルート A を辿る場合は，明確に調査の目的を意識していない可能性が高い．調査目的の異なる調査を区別なく行うため，影響範囲の特定漏れを起こしやすいことがわかった．そこで，明確に調査の目的を意識させるため，調査を目的別に 3 つのステージ「事前調査」「変更箇所調査」「影響箇所調査」に分類し，各調査での探索ルートを定めた「標準調査プロセス」を定義した．以下に調査ステージの内容を示す．現段階では，処理の流れなどの静的な構造の理解を目的としており，動的な振る舞い(例：状態遷移，タイミングチャート)の理解については対応していない．

表 1　標準調査プロセスの 3 つの調査ステージ

No	調査ステージ	説明
1	事前調査	・依頼内容を理解するために実施する，予め知っておくべきシステムの構造や動作を把握する調査，調査範囲を決めて調査し，成果物を残す．この調査では変更箇所は探さない．構造や動作を把握することに徹する． ・構造図を作成する際には，1 関数に対して大体 1 分程度の時間となる．実績値（前回の結果）がある場合は，その値を利用する．その値と調査対象の関数の数と調査に使える工数から，何関数を調査できるかを計算し，調査範囲を決定する．
2	変更箇所調査	・変更要求を実現するために必要な，システムの変更箇所を特定する調査（XDDP でいう「スペックアウト」に相当する） ・ソースコードを追って理解しながら変更箇所を探す． ・ひととおり変更仕様が抽出できたら調査を終了する
3	影響調査	・システムの変更によって，影響が及ぶ箇所を特定する調査（XDDP でいう「スペックアウト」に相当する） ・変更箇所があるとは限らない． ・変更方法や変更箇所によって影響の仕方が変わるため，変更箇所調査が完了した後に実施する．

このように表 1 の調査ステージと探索ルート B でソースコードを調査することで，担当者によって差があった調査方法が統一され，適切な調査ステージで適切な調査が行える．またレビューに耐えうる調査結果を残すことができ，調査の抜け漏れを防ぐことができる．

3.4　調査プロセスガイドラインの作成

プロセスの定義だけでは，実際の現場でどのように活動すれば良いかわかりづらい．そのため，

「標準調査プロセス」をより具体的な活動指針として明文化した「調査プロセスガイドライン」[6]を作成した．以下にその構成を示す．

表 2　調査プロセスガイドラインの構成

No.	項目	説明
1	調査プロセス定義	調査プロセス PFD，プロセス定義書，プロセスアクティビティで明確に調査プロセスを定義する
2	調査プロセスガイドラインの適用方法	実際のプロジェクトで適用する際の手順を記載している
3	調査プロセスチェックポイント	XDDP，熟練者のノウハウ，アドバイス，調査時のチェックポイントを記載している
4	調査プロセステンプレート	事前調査実施時に使用するテンプレート(調査テーマ記入シート)とその記載例を記載している
5	調査実施例	実際の調査での成果物の残し方，探索ルートを記載している

「調査プロセスガイドライン」の No.1 で明確にプロセスを定義しているので，初めて調査する人も「標準調査プロセス」に則って調査を行うことができ，変更による影響範囲の特定漏れを低減する効果が期待できる．さらに No.3 の調査プロセスチェックポイントで熟練者の調査ノウハウをチェックポイントとして記載しているので，初心者育成の効果も期待できる．以下に「調査プロセスガイドライン」を用いた際の調査の概要を説明する．ステップの詳細は，No.1 の調査プロセス PFD，プロセス定義書，プロセスアクティビティを参照すること．調査担当者が初心者の場合，「調査プロセスガイドライン」を活用できるよう，事前にトレーニングすることを推奨する．

(1)ステップ 1

「調査プロセスガイドライン」の No.2 の調査プロセスガイドラインの適用方法に従い，変更要求の調査目的や調査内容に応じて，パターンを決定する．

【基本パターン 1】

・変更依頼を受け，システム構造の理解や変更箇所を特定する場合の基本パターン

・担当者が対象システムの知識や保守経験が浅い場合は，「事前調査」から始める

【基本パターン 2】

・担当者が構造や処理の流れ（現行仕様）を理解している，または理解できる資料がある場合は，「事前調査」を省略できる

表 3　各パターンによって実施するステップと調査

基本パターン	ステップ 2	ステップ 3		
		事前調査	変更箇所調査	影響調査
1	●	●	●	●
2	－	－	●	●

●：実施する　－：実施しない

以降，表 3 の基本パターンに従い，各ステップを実行する．

(2)ステップ 2

No.4 の調査プロセスのテンプレート（調査テーマ記入シート）を記載する．ここでそれぞれの欄を記載した後，内容の妥当性をレビューで確認する．これにより，調査目的，調査範囲，調査時間が明確になり，事前調査が発散しないようになる．

(3)ステップ 3

No.3 のプロセスチェックポイントと No.5 の調査実施事例（探索ルートと成果物）を参考にする．プロセスチェックポイントには，調査プロセス PFD とプロセス定義書に対応した PFD-No に対するチェックポイントとアドバイスを記載している．調査実施事例（探索ルートと成果物）には，XDDP の調査プロセスで用いられている探索ルート B とその調査結果の残し方，影響範囲の特定漏れを起こしやすい探索ルート A を記載している．これらを基に，

ソースコードを効率よく漏れなく調査することによって，変更による影響範囲の特定漏れを低減し，有用な調査資料を残すことができる．

4. 解決策の検証

4.1 標準調査プロセスと調査プロセスガイドラインの検証方法

「標準調査プロセス」と「調査プロセスガイドライン」の有用性を判断するために，次の 3 点について検証した．なお，検証は過去の不具合事例に対して行うものとする．ただし，「調査プロセスガイドライン」はドラフト版である．

(1)調査のステージを意識して調査を行うことができるか

(2)不具合事例の原因となった影響を受ける範囲や変更箇所を特定できるか

(3)メモ書きではない調査結果が残せるか

検証範囲は 2.1 で「調査方法の問題」に分類したシステムテストでの不具合事例 40 件の内，変更による影響範囲が無知見領域に存在した事例 2 件，知見領域で，変更による影響範囲に気付けなかった事例 3 件を対象とした．これら 5 件は実プロジェクトで探索ルート A を用いて調査した結果，変更箇所を特定することができなかった．事例 1 と事例 2 から 5 は別プロジェクトで，いずれも組み込みソフトウェアで発生した不具合である．検証方法は，不具合が発生した変更要求に対し，我々自身が「標準調査プロセス」と「調査プロセスガイドライン」を用いて，シミュレーションを行い，調査結果を残すまでを実施するという方法で検証を行った．

また，実プロジェクトにおいて，初心者に「標準調査プロセス」と「調査プロセスガイドライン」を適用し，事前調査を実施した．

注：知見とは調査対象のソースコードを調査したことがあり，一部動作を把握している状態を示す．

無知見とは調査対象のソースコードを全く調査したことがないことを示す．

4.2 標準調査プロセスと調査プロセスガイドラインの有用性の検証結果

5 件の不具合についてシミュレーションした結果を表 4 に示す．

表 4 標準調査プロセスと調査プロセスガイドラインの有用性の検証結果

不具合事例	変更の影響範囲	調査ステージ	結果	成果物	調査工数	構造図作成工数
事例 1	無知見	事前調査	特定できた	構造図	5 時間	5.0 時間(64 関数)
事例 2	無知見	事前調査	特定できた	構造図	4 時間	1.4 時間(70 関数)
事例 3	知見	事前調査 変更箇所調査	特定できた	構造図 フローチャート	3 時間	1.8 時間(104 関数)
事例 4	知見	変更箇所調査	特定できた	フローチャート	2 時間	作成していない
事例 5	知見	事前調査 変更箇所調査	特定できなかった	構造図 フローチャート	1 時間	0.5 時間(28 関数)

表中の「変更の影響範囲」は無知見領域に存在したか，知見領域に存在したかを意味する．また「調査ステージ」には実施した調査ステージを記載する．

影響範囲を特定できた 4 件の事例については，「事前調査」ステージで基本的な構造や動作を理解するための調査を行い，成果物として構造図やフローチャートを残せた．無知見領域に影響範囲が存在した事例では，これらの調査結果から影響を受ける箇所を特定することができた．知見領域に影響範囲が存在した事例では，調査資料を基に「変更箇所調査」ステージで行った調査において，変更箇所を特定することができた．事例 5 では，特定できた事例と同様に構造図やフローチャートを残せたが，変更箇所を特定できなかった．

5 つの事例を基に，構造図に記載した関数と構造図の作成時間から，1 時間あたり構造図に約 30 関数を記載できることがわかった．ただし，フローチャートの作成時間は除く．

また，実プロジェクトで，初心者に「標準調査プロセス」と「調査プロセスガイドライン」を適用した結果，7 時間で 350 関数を調査し，構造図を残すことができた．

4.3 標準調査プロセスと調査プロセスガイドラインの有用性についての考察

検証の結果，調査のステージを意識することで，混同されていた構造を理解する調査，変更箇所を特定する調査，影響範囲を特定する調査を分けて行うことができ，混同していることで気付けなかった変更による影響範囲の特定漏れを低減することができると考えられる．シミュレーションでの結果より，今までは探索ルートAを用いて何度も同じ関数を調査し(探検隊の状態)多くの時間を投入したにもかかわらず影響範囲を特定することができなかったが，探索ルートBを用いて影響範囲を特定することができた．このことは探索ルートBが探索ルートAより処理や構造を把握するために非常に効率が良いことを示す．

また，メモ書きでない調査結果を残すことができたことから，レビューのインプットとなると言える．これらのことから，「標準調査プロセス」と「調査プロセスガイドライン」は変更による影響範囲の特定漏れの低減に効果があるといえる．さらに，構造図を作成する時間がわかったため，これを事前調査の見積りに使うことができると考えられる．また，実プロジェクトでの初心者の適用結果から，「標準調査プロセス」は技術的に難しくなく，「調査プロセスガイドライン」は現場で有効に活用できることがわかった．しかし，表4の事例5では，変更による影響範囲を特定できなかった．この事例は状態遷移の制御が行われている箇所に影響範囲が存在したケースである．この場合，状態遷移マトリクスを作り，その後状態遷移ダイアグラムを起こした上で影響範囲を特定する必要がある．そのため今回の「標準調査プロセス」と「調査プロセスガイドライン」の適用範囲外となり，影響範囲を特定できなかったと考えられる．

5. 本研究のまとめ

5.1 研究成果

変更による影響範囲の特定漏れを低減するために，「標準調査プロセス」を定義し，これをうまく活用するための「調査プロセスガイドライン」を考案した．そして「標準調査プロセス」と「調査プロセスガイドライン」を過去の不具合事例でシミュレーションを用いて検証し，また実プロジェクトで適用した結果，以下の効果を確認した．

- 適切な調査のステージで適切な調査を行える
- 処理の流れなど静的なケースにおける，変更による影響範囲の特定漏れを低減できる
- 調査結果を残すことができ，レビューが可能となる
- 初心者でも技術的に問題なく調査を行うことができる

5.2 今後の進め方

我々の考案した「標準調査プロセス」と「調査プロセスガイドライン」は，効果があることがわかったが，動的な振る舞いの調査方法についてはまだ着目できていない．今後は，実際のプロジェクトでの運用件数を増し有効性を高め，また動的な振る舞いの理解を目的とする調査方法についても着目し，「標準調査プロセス」と「調査プロセスガイドライン」を改良していきたい．

6. 参考文献

[1] 清水 吉男，“「派生開発」を成功させるプロセス改善の技術と極意”，2007 年
[2] 清水 吉男，“[入門＋実践]要求を仕様化する技術・表現する技術
-仕様が書けていますか？“，2005 年
[3] 清水 吉男，“派生開発プロセス[XDDP]のポイント -XDDP の考え方を知る-“，
SQiP 第 6 分科会[派生開発]特別講演資料，2014 年
[4] 白川 智也，“派生開発における影響箇所の気づき“，ET-WEST2014 AFFORDD セッション資料，2014 年
[5] 二村 良彦・他，“「PAD(Problem Analysis Diagram)によるプログラムの設計及び作成」”，情報処理学会論文誌，1980 年
[6] 宇田 泰子・他，“調査プロセスガイドライン”，2014 年度 SQiP 研究会第 6 分科会，2015 年