

テストプロセスの改善を考慮したペア開発の改良と評価

喜多 義弘
長崎県立大学
kita@sun.ac.jp

池田 暁
クオリティアーツ
akira.ikeda@quality-arts.com

水田 真之介, 永尾 直樹, Daniel Jia Qin Goh
株式会社 NDKCOM
{s-mizuta, nagao.naoki, daniel}@ndkcom.co.jp

要旨

ソフトウェア製品の品質を向上させるため、ソフトウェア開発におけるテストプロセスを改善することは有効な手段の1つである。テストプロセス改善では、第三者視点による不具合検知の向上や、製品設計とテスト設計の分担による効率化を図るため、開発チームから独立したテスト組織を編成することが推奨されている。しかし、テスト技術者の人材が不足しているソフトウェアベンダーにとっては、そのことが改善活動の大きな障壁となっている。

我々は以前、テスト組織を編成せずともテストプロセスの改善を可能にすることを目的とした、少人数による小規模開発手法である「ペア開発」を提案した。しかし、このペア開発には抽象的な概念や課題があり、実用には至っていなかった。そこで本論文では、ペア開発の実用化とそれによりテストプロセスが改善できることを目的とし、ペア開発の改良を行い、その有用性について評価する。

1. はじめに

ソフトウェア開発において、ソフトウェア製品の品質を高めるために、テストプロセスを見直し、改善する動きが高まっている。テストプロセスを改善することは、

製品の品質向上につながるだけでなく、テストを効率よくかつ効果的に行うことにより工数とコストの削減にもつながる。

テストプロセスの改善には、第三者視点による不具合検知の向上や、製品設計とテスト設計の分担による効率化を図るため、開発チームから独立したテスト組織を編成することが推奨されている [1]。しかし、テスト技術者の人材不足により独立したテスト組織を編成できない場合があり、そのことが改善活動の大きな障壁となっている。

また、人材不足は製品開発の技術者においても深刻であり、開発規模によっては開発チームを編成せず、技術者が単独で開発工程の全てを一貫して行うこともある。この方法は第三者が関わりにくい状況になりやすく、製品に対する認識がずれたり、視野が狭まったりすることにより、誤った製品を開発することに繋がってしまう。

我々は以前、テスト組織を編成せずともテストプロセスの改善を可能にすることを目的とし、少人数による小規模開発手法である「ペア開発」[2]を提案した。しかし、このペア開発には抽象的な概念や課題があり、実用には至っていなかった。

そこで本論文では、ペア開発の実用化とそれによりテストプロセスが改善できることを目的とし、ペア開発の改良を行う。さらには、ペア開発を実際の開発現場に適用し、従来の開発手法と比較することにより、ペア開発の有用性について評価する。

2. 関連研究

2.1. 技術者単独または少人数による開発

ソフトウェア開発において、開発プロジェクトごとにチームを組むことが一般的である。しかし、ソフトウェア製品やプロジェクトの規模、または技術者の人材不足により、チームを組まずに技術者単独で開発を行うことがある。2名程度の少人数で行う場合もあるが、主となる技術者が全ての工程を一貫して行い、作業量が多い工程に補助の技術者を割り当てることが多い。

1人の技術者が工程を一貫して開発することは、技術者自身のペースや技術力に合わせやすく、開発全体を見通しやすいメリットがある。しかしながら、技術者の主観によって製品に対する認識や観点が偏りやすく、視野が狭まることで潜在する欠陥に気づかなかつたり、欠陥が入り込みやすかつたりと製品の品質に関わるデメリットもある。

2.2. テストプロセスの改善

テストプロセスの改善を行うための指針の1つとして、TPI NEXT[1]がある。これは、ビジネス主導によるテストプロセス成熟度評価とそれに基づく改善を枠組みとしたプロセス改善モデルである。

TPI NEXTにおいて、テスト活動に必要な16個の要素を定めているが、今回我々は要素の1つである「テスト組織」に着目した。テスト組織については、「独立した存在としての組織」として定義され、テスト関連の成果物やサービスを明確にし、テストに対して専任であり、責任を持つこととしている。しかしながら、人材不足が起こっている現場においては、開発チームから独立したテストチームを組織することは難しい現状がある。

TPI NEXTに関連する既存研究として、TPI NEXTの導入方法[3]やその支援手法[4]は提案されているが、人材不足によりテスト組織の編成が困難である場合の対策については提案されていない。

そこで本研究においては、まずテストプロセスの改善を行うためのフレームワークを作ることを考える。想定している開発環境として、前節で挙げた「技術者単独または2名程度の少人数で行っている開発環境」を想定

し、テストプロセスの改善を見据えつつ、その環境に適した開発方法を考える。

2.3. 従来の開発モデルとその問題点

従来の開発モデルとして、ウォーターフォール型開発モデルをベースとするV字モデルやW字モデルがある。これらの開発モデルは、設計工程とテスト工程が分離していることから、それぞれの役割に分担しやすい。さらにW字モデルにおいては、製品設計を行いながらテスト設計も並行して行うため、製品設計にテストの観点を取り入れやすく、製品設計の漏れや誤りを早期に発見しやすい。

しかしながら、各開発モデルには問題点がある。V字モデルにおいては、テスト工程が進むほど当工程で発見した欠陥の修正による手戻りは大きく、コストの増加や後期の圧迫が発生する。一方、W字モデルにおいては、開発設計の初期段階では開発する製品の要求や要件が十分に整理されていないため、初期のテスト設計が難しい。これらの解決方法として、テスト設計を行う者（以下、テスト設計者）が製品設計を行う者（以下、製品設計者）と密なコミュニケーションを取る、または共同で要求分析や要件定義を行い、製品やそのテストに対する互いの認識を合わせておく必要がある。

2.4. ペア開発

我々は以前の研究[2]にて、テスト組織を編成せずともテストプロセスの改善を可能とすることを目的として、ペア開発を提案した。ペア開発は、V字モデルやW字モデルと同様に、ウォーターフォール型開発モデルをベースとする。

図1に、ペア開発の全体の流れを示す。ペア開発では、製品設計者とテスト設計者の2名で開発を行う。要求分析から要件定義までは2名が共同で行い、要求仕様書および要件定義書をそれぞれ作成する。次に製品設計者は基本設計および詳細設計を行い、各設計書を作成する。一方、テスト設計者はこれらの書類を基に各テスト設計書および各テストケースを作成する。コーディングの工程に入ると、2名で分担してソースコードを作成する。テスト工程では担当したソースコードをそれぞれ交換し、テストを行う。

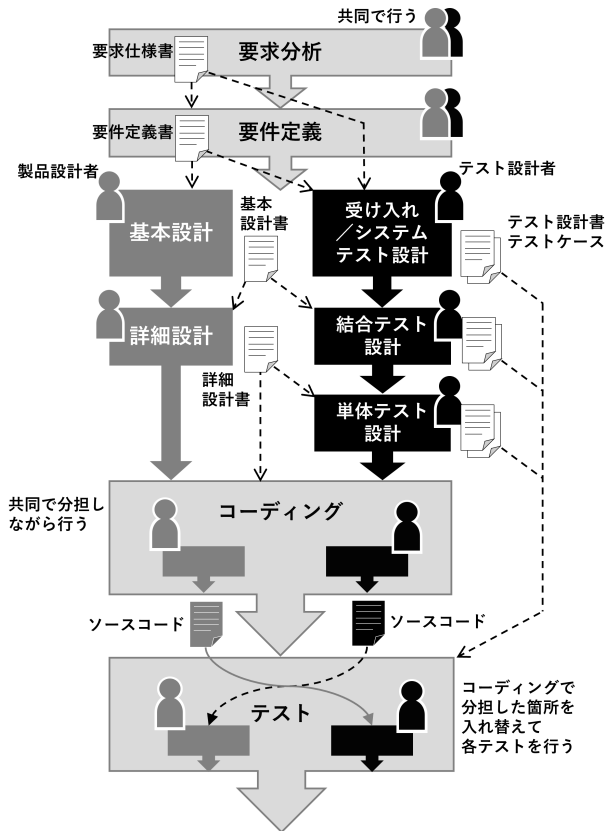


図 1: 従来研究 [2] におけるペア開発

当論文におけるペア開発の問題点を以下に3つ示す。

- コーディング以降の工程が抽象的であり、コードレビューの有無やテストレベルごとの分担が不明である。
- 製品設計およびテスト設計を並行に進めることは、製品に対する互いの認識が曖昧なまま設計工程を進めることに繋がる。
- 製品設計およびテスト設計における各設計書の妥当性を確認する機会がない。

特に最後の項目である、「各設計書の妥当性を確認する機会がない」ことについて、当論文では、設計の際にペア間で密にコミュニケーションを取ることによって妥当性を確認することを目指したが、設計書を直接確認し合う（レビューし合う）ことにより、互いの観点到触れ、さらなる気づきや認識を与えようとする。

3. ペア開発の改良

本論文では、従来研究 [2] にて提案したペア開発について、実用化とそれによりテストプロセスを改善することを目的として、ペア開発の改良を行う。特に 2.4 節で挙げた問題点に対する改良を行う。

改良後のペア開発について、設計フェーズでの流れを図 2 に、実装・検証フェーズでの流れを図 3 にそれぞれ示す。ここでは説明の便宜上、全体の流れを設計フェーズと実装・検証フェーズの 2 つに分けているが、実際には一貫して行うことを想定しており、設計フェーズの最後に実装・検証フェーズへと移る。

3.1. 設計フェーズの流れ

まず、図 2 の設計フェーズにおいて、要求分析および要件定義を製品設計者およびテスト設計者の 2 名が共同で行う。共同で行う狙いとして、製品に対して互いの認識を合わせ、齟齬を出さないことを目指す。

次に、製品設計者による基本設計および詳細設計を行う。このとき、各設計書をテスト設計者が随時レビューを行い、指摘があれば製品設計者に対してその都度フィードバックする。ここでのレビューの狙いは、テスト設計者によるテスト観点を製品設計に取り入れることにより、製品設計の抜け漏れをなくすことを目指す。また、それと同時にテスト設計者においては、製品に対しどのようなテストが必要かをテスト設計前に把握できることも見込まれる。

製品設計が進み、基本設計書および詳細設計書が完成に近づいたら、テスト設計者はそれらの書類を基に各テスト設計書およびテストケースをそれぞれ作成する。このとき、各テスト設計書およびテストケースは製品設計者が随時レビューを行い、指摘があればテスト設計者に対してその都度フィードバックする。ここでのレビューの狙いは、製品設計者の製品に対する認識にテスト設計が合っているかを確認し、2 名ともにテスト工程で行うべきテストを事前に把握することを目指す。また、この時点で製品設計の抜け漏れを発見した場合は、ただちに基本設計書および詳細設計書を修正し、その旨をテスト設計者と共有し、テスト設計にも反映する。

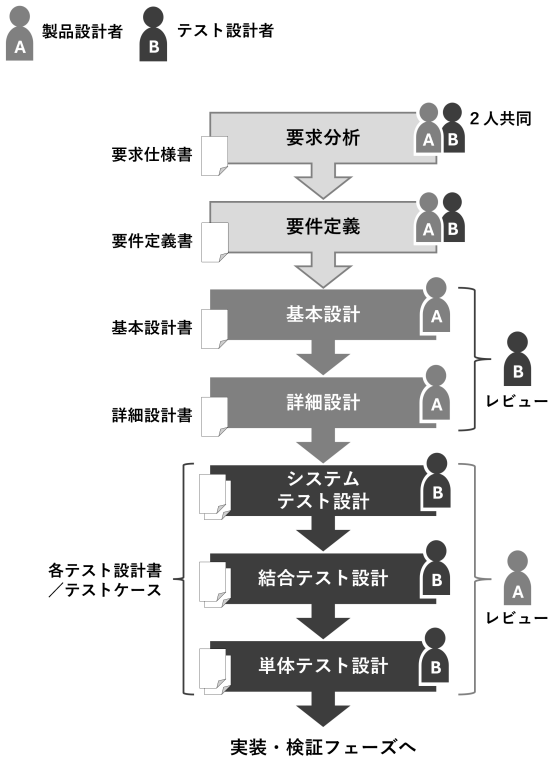


図 2: 改良後のペア開発 (設計フェーズ)

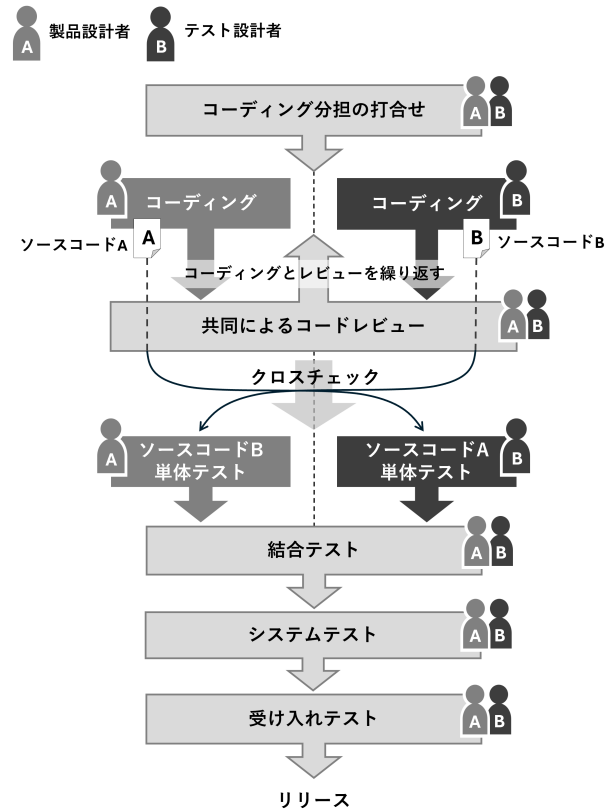


図 3: 改良後のペア開発 (実装・検証フェーズ)

3.2. 実装・検証フェーズの流れ

製品設計およびテスト設計における各設計書が完成した後、実装・検証フェーズへ移る。図3において、まず製品設計者およびテスト設計者の2名で打合せ、コーディングの分担を決める。分担による負荷の割合は、各自のコーディングスキルや当工程に対するエフォートなどを考慮して、極端な偏りがない程度で任意に決めるものとする。

コーディングは分担により進めていくが、コードレビューは2名での共同で行う。共同で行うことの狙いは、コードの妥当性確認に併せて、製品の認識について互いの齟齬の有無を確認することを目指す。コードレビューは任意のタイミングで頻繁に実施し、コーディングとコードレビューを繰り返すことを想定している。

各ソースコードができた後、作成したソースコードを交換し合い、互いに相手のソースコードに対して単体テストを実施し、クロスチェックを行う。

これにより、テストの客観性を担保する。両者の単体テストが終了した後、結合テスト以降のテスト工程は共同で行う。また、各テストレベルにおいて不具合などを発見した場合は、適宜修正を入れ、コードレビューを行う。そして必要であれば単体テストからやり直すことも考慮する。

4. ペア開発の有用性評価実験

4.1. 実験概要

ペア開発の有用性を評価するために、実際のソフトウェア開発においてペア開発と従来の開発手法を比較する。具体的には、製品化する Android 向けアプリケーションを各開発手法で並行して開発する。

図4に、本実験における各開発手法の流れを示す。図中の左側は本提案手法であるペア開発とし、右側は少数の技術者による従来の開発手法を想定する。

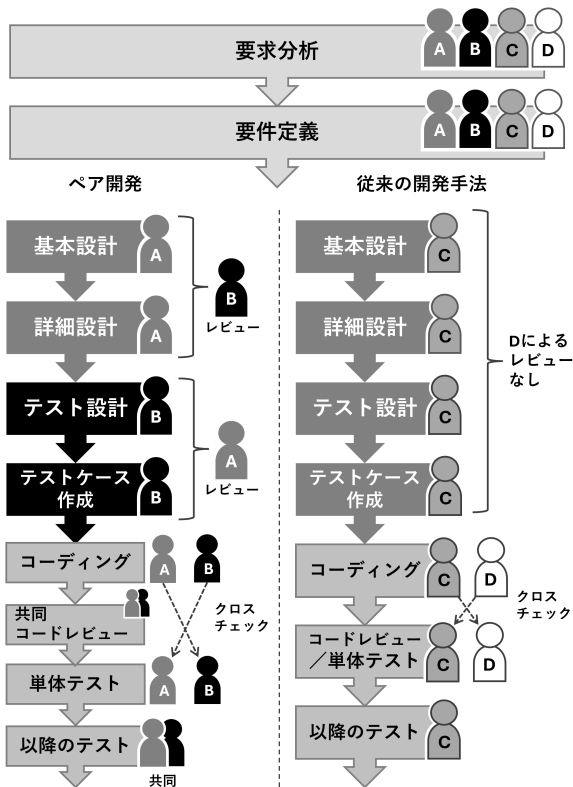


図 4: 評価実験における各開発手法の流れ

要求分析および要件定義までは共通で行い、以降の工程はそれぞれの開発で分かれて行い、それぞれの製品を開発する。設計が異なるため見た目や UI の違いはあるが、同じ要件で作るため実装する機能や入出力は同じ製品である。

被験者として、ペア開発に 2 名の技術者（技術者 A と B）、従来の開発手法に 2 名の技術者（技術者 C と D）をそれぞれ割り当てる。各被験者は、普段から従来の開発手法で開発を行っており、開発の経験も同程度有している。図 4 より、要求分析および要件定義は被験者の 4 人が共同で行い、顧客の要求や製品の要件などは各自認識しているものとする。

ペア開発として、技術者 A が製品設計者に、技術者 B がテスト設計者にそれぞれ分かれ、本提案どおりに各工程を行う。単体テストにおいては、相手がコーディングで担当した部分をテスト対象として、クロスチェックを行う。

一方、従来の開発手法として、チームを編成せず、技

術者 C が全ての工程を単独で一貫する方法をとる。作業量が多くなるコーディングや、客観的な観点が必要なコードレビューおよび単体テストには、補助となる技術者 D も共に行うが、開発全体の管理は全て主となる技術者 C が行う。コードレビューおよび単体テストにおいては、相手がコーディングで担当した部分をレビューやテスト対象として、クロスチェックを行う。

これらの開発により出てきた各種設計書やテストケースなどの成果物を比較し、各開発の違いを分析する。

4.2. 実験結果

表 1 に、各開発における工程ごとの所要工数を示す。単位は人時で表し、工程は基本設計から単体テストまでを表している。表より、ペア開発の方が従来の開発よりも全体的に工数がかかっていることが確認できる。特に、基本設計、テスト設計、コードレビューと単体テストに工数を多くかけている。また、従来の開発では行われていない設計レビューおよびテスト設計レビューがペア開発で行っているため、その分の工数もかかっている。一方、コーディングは従来の開発の方がペア開発よりも工数がかかっていることも確認できる。

表 2 に、各開発におけるアプリの表示画面ごとのテストケース数を示す。この表では、テストケース数を比較するための一例として、当実験期間で実装した 4 つの画面を対象にしている。表より、ペア開発の方が従来の開発よりも多くのテストケースを作成されていることが確認できる。

さらに、表示画面ごとのテストケース数の割合に着目すると、ペア開発では機能表示画面 A および機能表示画面 B でテストケースを多く作成されているのに対し、従来の開発では機能表示画面 B に偏ってテストケースを多く作成されていることも確認できる。

表 3 に、表 2 で数を示したテストケースにおいて発見した不具合を示す。テストケースが多い分、ペア開発の方が従来の開発よりも不具合を発見していることが確認できる。

表 1: 工程ごとの所要工数 (人時)

	ペア開発	従来の開発
基本設計	26.0	9.5
詳細設計	18.0	21.0
設計レビュー	4.0	-
テスト設計	32.5	11.0
テスト設計レビュー	6.0	-
コーディング	73.0	101.0
コードレビュー and 単体テスト	32.0	8.5
合計工数	191.5	151.0

表 2: 表示画面ごとの単体テストのテストケース数

	ペア開発		従来の開発	
タイトル画面	19	(5.5%)	4	(4.9%)
メニュー画面	63	(18.3%)	13	(15.8%)
機能表示画面 A	113	(32.8%)	19	(23.2%)
機能表示画面 B	150	(43.4%)	46	(56.1%)
合計	345	(100%)	82	(100%)

表 3: 表 2 のテストケースにおいて発見した不具合数 (発見した不具合数 / テストケース数)

	ペア開発		従来の開発	
タイトル画面	2	/ 19	2	/ 4
メニュー画面	3	/ 63	0	/ 13
機能表示画面 A	20	/ 113	2	/ 19
機能表示画面 B	15	/ 150	10	/ 46
合計	40	/ 345	14	/ 82

5. 考察

5.1. 実験結果の考察

表 1 より、ペア開発の方が従来の開発よりも工数がかかった理由として、製品設計およびテスト設計において互いにレビューを行ったことが関わっていると考えられる。レビューそのものの工数が含まれることも当然では

あるが、レビューにより担当の技術者単独では気づかなかった観点を得ることができたと考えられる。また、レビューによるコミュニケーションを円滑に行うため、レビューをする側は相手の成果物を理解しようとし、一方、レビューを受ける側は自身の成果物を理解しやすくするために伝える工夫や整理を行う。これにより、製品に対して互いに理解を深め合い、自身の成果物に対して不足している部分を補おうとしたため、従来の開発よりも工数がかかったと考えられる。

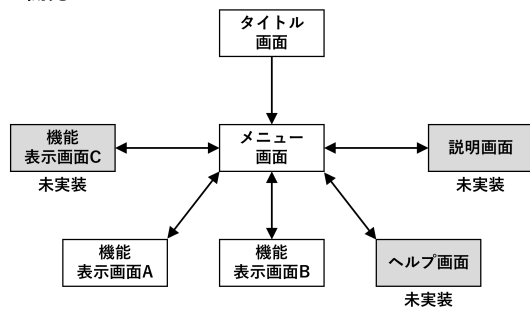
従来の開発においては、製品設計とテスト設計は単独の技術者が行うとした場合、自身の思いのまま設計を行うことができる。そのため、他とのコミュニケーションを必要以上にとることなく工程を進めることができる。しかしながら、そのことにより第三者による客観的な観点が入りにくくなり、自身の認識にずれが生じていても気づかないことがある。このことから、従来の開発では設計書の可読性の低さや抜け漏れが出やすくなることが予想される。

表 2 より、ペア開発の方が従来の開発よりもテストケースを多く作成できた理由として、前述したとおり、互いの成果物への理解を深めたことにより、必要なテストを把握できたためと考えられる。一方、従来の開発において作成したテストケースは、機能表示画面 B に関するテストケースが全体の半数を占める。このように偏ったテストケースになった原因として、被験者である技術者 C の認識の偏りにあると考える。

その一例として、図 5 に、ペア開発および従来の開発において詳細設計時に作成した画面遷移図の違いを示す。図より、ペア開発での画面遷移図は各画面への遷移を端的に表現しているのに対し、従来の開発での状態遷移図は、以下のような違いが見られる。

- 説明画面やヘルプ画面への遷移は考慮されていない。
- 機能表示画面 B のみ抽象度が低く偏っている。
- ペア開発の画面遷移図には見られない遷移（破線矢印）があり、複雑になっている。

ペア開発



従来の開発

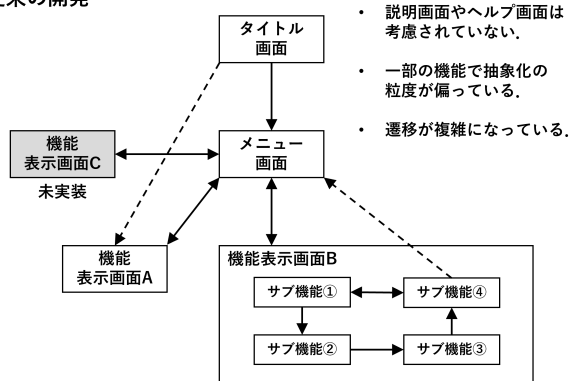


図 5: 各開発における詳細設計時の画面遷移図の違い

ペア開発においては、設計でレビューを行うことにより、互いの認識を合わせやすく、開発するものがイメージしやすくなる。また、認識の異なる部分や指摘する部分をレビューの段階で気づき、相手に伝えることで補正を円滑に入れることができる。そのため、詳細設計時に整理された画面遷移図を出すことができたと考える。

一方、従来の開発では、すべての設計を単独で行ったことから、自身の認識のみであるため視野が狭まったと考えられる。特に、図 5 中の機能表示画面 B については、他の画面よりも具体的な流れになっていることから、この部分に視野や意識が集中した結果、ヘルプ画面や説明画面が画面遷移図から抜け、複雑な遷移になったと考えられる。

さらに、当実験でペア開発を行った被験者の技術者 A および技術者 B から、ペア開発における実用上のメリットおよびデメリットについて意見を伺った。それぞれを以下に示す。

● ペア開発のメリット

- 分担がはっきりしているため、自身の作業に集中できる。
- 成果物に担当した技術者の観点がでるため、製品に対する責任感が出る。
- 異なる観点でレビューすることで、違和感に気づきやすく、欠陥の早期発見につながる。

● ペア開発のデメリット

- 互いの予定を合わせる必要がある。
- ペアの選定が難しい。
- ペア間の信頼関係が重要である。

ペア開発のメリットについては、各項目が製品の品質向上に結び付いている。また、製品設計とテスト設計で分かれるため、テスト担当者が独立した形になる。そのため、当初の目的である「テストプロセス改善」について従来の開発よりも適した開発手法であると考えられる。

次にペア開発のデメリットである、「互いの予定を合わせる必要がある」および「ペアの選定が難しい」について考える。ペア開発を円滑に進めるポイントとして、ペアを組む者同士のコミュニケーションが重要であり、そのために互いのスケジュールや進捗を合わせる必要がある。異なる部署や他の業務を兼任している場合は、開発にかかるエフォートの低い方に合わせる必要があるため、開発期間全体が間延びすることに繋がってしまう。また、ペア間で技術力や進捗に差がある場合は、作業量が偏ってしまったり、遅れている作業がボトルネックとなり全体の進捗にさらに遅れが生じたりとトラブルを招きやすくなる。このことから、ペア開発を行うには出来る限り「同程度の技術力を持つ者同士」「同じ部署かつ同じ職種である者同士」で組むことが望ましく、進捗も常に同一行程を進むように合わせる必要がある。

デメリット最後の項目である「ペア間の信頼関係が重要である」について考える。従来、単独で行っていた技術者にとって、一貫して行っていた工程の一部をペアの技術者に任せることは、自身から見えない部分が出てくることを意味している。これは、見えない部分に対する品質の担保だけでなく、ペアとの認識のずれによる自身の成果物の不具合に対する不安も含んでいる。これら

の不安を払拭するためにも、ペア間で積極的かつ密にコミュニケーションを取る必要がある。

当実験は、被験者 4 名（うちペア開発実施者 2 名）であるため、被験者の技術力や環境に依存し、偏った実験結果であることも考えられる。そのため、被験者や開発対象を増やし、より客観的で平準な実験を行うことが今後求められる。

5.2. テストプロセス改善における考察

ペア開発は、テストプロセス改善を行うためのフレームとして考案した開発手法である。そのため、従来の手法よりもテストプロセス改善が行いやすいものでなくてはならない。そこで TPI NEXT[1] におけるテストプロセスの達成推奨項目をどれだけ満たしているかについて、当実験で行ったペア開発と従来の手法を対象に測定し、比較した。

図 6 に、ペア開発における TPI NEXT[1] のテストプロセス達成推奨項目を示す。このマトリックスは、TPI NEXT が推奨するテストプロセスの項目を示しており、その達成の度合いをテストプロセスの「成熟度」として表現している。アルファベット文字はクラスタと呼ばれ、1 つの改善ステップの役割を果たすために複数のキーエリアのチェック項目をグループ化したものである [1]。これらのアルファベット文字が達成すべき順番を示しており、キーエリア全体において「A」から順に達成していくことを推奨している。そのマトリックス上に、従来の手法で既に達成済みの項目に加え、ペア開発により新たに達成した項目、およびペア開発を行うことで達成しやすくなると考えられる項目をそれぞれ示す。

図 6 より、ペア開発によって達成できた項目が 2 項目あることを確認できる。その 2 項目は以下のとおりである。

- 【1. 利害関係者のコミットメント】利害関係責任者を決定し、テスト担当者に周知している。
- 【14. テストケース設計】テストケースには、以下の説明項目を含む。a) 開始時の状況, b) 変更プロセス=実施するテストアクション, c) 予測される結果。

- X 従来の手法で既に達成済みの推奨項目
(Xは任意のアルファベット文字、以下同じ)
- X 従来の手法では達成されず
ペア開発により新たに達成した推奨項目
- X ペア開発を行うことで達成しやすくなると
考えられる推奨項目

キーエリア	成熟度レベル										
	初期レベル	コントロールレベル			効率化レベル				最適化レベル		
1 利害関係者のコミットメント	A	B	B	C	F	H	H	K	M	M	
2 関与の度合い	A	B	C	E	H	H	J	L	L		
3 テスト戦略	A	A	B	E	F	F	H	K	L		
4 テスト組織	A	D	D	E	I	I	J	J	K	L	L
5 コミュニケーション	B	C	C	D	F	F	J	M	M		
6 報告	A	C	C	F	G	G	K	K			
7 テストプロセス管理	A	A	B	B	G	H	J	K	M		
8 見積もりと計算	B	B	C	C	G	H	I	I	K	L	L
9 メトリクス	C	C	D	G	H	H	I	K	K		
10 欠陥管理	A	A	B	D	F	F	H	J	K	L	L
11 テストウェア管理	B	B	D	E	I	I	J	L	L	L	
12 手法の実践	C	D	E	F	H	J	J	M	M		
13 テスト担当者のプロ意識	D	D	E	E	G	G	I	I	K	K	M
14 テストケース設計	A	A	E	F	I	I	J	K	K	M	
15 テストツール	E	E	E	F	G	G	I	L	M	M	
16 テスト環境	C	D	D	E	G	H	J	J	L	M	M

図 6: ペア開発における TPI NEXT[1] のテストプロセス達成推奨項目

前者は、ペア開発によってテスト担当者を置き、さらに要求分析や要件定義にテスト担当者も参加することにより、利害関係者を把握し、その責任者を知ることができたと考えられる。後者は、製品設計やテスト設計をペア間でレビューし合うことにより、テストケースとして必要な要素を漏れなく挙げ、ペア間で互いに把握しやすいうように明記したと考えられる。

次に、ペア開発を行うことで達成しやすくなると考えられる項目について考える。これらは当実験において達成の兆しは見られたものの、「A」から順に達成していくことを考慮したときに順を飛ばしていたり、当実験だけでは達成が断言できなかったりと、不確定なものを対象としている。項目数が 3 項目以上増えているキーエリアとして「1. 利害関係者のコミットメント」、「2. 関与の度合い」、「4. テスト組織」、「5. コミュニケーション」、および「6. 報告」が挙げられる。つまり、ペア開発によってこれらのキーエリアが改善でき、テストプロセス改善

に有用であることが考えられる。

しかしながら、テストプロセス改善はキーエリア全体を通してバランスよく行っていくことが前提であるため、今回改善できていないキーエリアについて、ペア開発と組み合わせることが可能な新たな改善手法を検討する必要がある。

5.3. 今後の課題と展望

ペア開発は、コーディングの前にテスト設計を行うため、テスト駆動開発 [5] の考え方をを用いることができる。また、コーディングの部分ペアプログラミングに置き換えることもできる。これらにより、ペア開発はアジャイル開発 [6] との親和性が高いと考えられるが、ペア開発のベースがウォーターフォール開発モデルであるため、アジャイル開発への移行は従来と同等の時間とコストがかかる見込みである。

ペア開発では、要求分析や要件定義において各設計者の認識を十分に取ることに重点を置くため、アジャイル開発モデルに近い形への派生を考慮した場合も、この点を重視する必要がある。各設計者の密なコミュニケーションが実現可能である場合は、要求分析や要件定義の工程をより軽量に改良できる余地があると考えられる。

ペア開発により、製品設計とテスト設計を分離することで、開発から独立したテスト組織を編成せずとも疑似的にその形を実現することを目指した。しかしながら、このペア開発によってテストプロセス改善が可能であるかを未だ検証できていない。そのため、ペア開発を導入した上で TPI NEXT のテストプロセス成熟度評価 [1] やその改善活動を行い、ペア開発の有用性について検証する必要がある。

さらにペア開発は、初級または中級の技術者向けに OJT (On the Job Training) を行うことに向いていると考える。当初は初級者がベテランの技術者と組んでペア開発を行うことを想定していたが、「同程度の技術力を持つ者同士」で組むことが望ましいことから、初級または中級の技術者同士がペアを組み、その監視役をベテランの技術者が担うことで実用できると考える。また、ベテランの技術者は監視だけでなく、製品設計およびテスト設計のレビューも担当することにより、自身が有するノウハウを教え伝えることもできると考える。

6. おわりに

我々は以前、テスト組織を編成せずともテストプロセスの改善を可能にすることを目的とし、少人数による小規模開発手法である「ペア開発」[2] を提案した。しかし、このペア開発には抽象的な概念や課題があり、実用には至っていなかった。

そこで本論文では、ペア開発の実用化とそれによりテストプロセスが改善できることを目的とし、ペア開発の改良を行った。製品設計者とテスト設計者としてペアを組み、設計時のレビューやコーディング後のコードレビューや単体テストにおけるクロスチェックなどの改良を経て、実用性を向上することを目指した。ペア開発の有用性を確認するために実際のアプリ開発を通して実験を行い、その結果、従来の開発手法よりも有用な設計と多くのテストケースを作成することができ、その有用性を確認することができた。さらにペア開発によって達成しやすくなるテストプロセスの項目を照らし合わせた結果、テストプロセス改善についても有用であることを示した。

このペア開発を通じて、テストプロセスの改善を行っていき、その効果を測定することが今後の課題である。

謝辞

本研究において、有益な助言をいただいた特定非営利活動法人ソフトウェアテスト技術振興協会 (ASTER) の坂静香氏に深く感謝を申し上げます。

参考文献

- [1] A. V. Ewijk, et al., “TPI NEXT® Business Driven Test Process Improvement,” Sogeti Nederland B.V. 2013, 訳 藪田和夫, 湯本剛, 皆川義孝, “TPI NEXT® ビジネス主導のテストプロセス改善,” 株式会社トリフォリオ, 2015.
- [2] 喜多義弘, 池田暁, 他 7 名, “テストプロセス改善を考慮した少人数によるソフトウェア開発手法の提案,” ソフトウェア・シンポジウム 2024(SS2024), pp.71-77, 2024.

- [3] 高野愛美, 河野哲也, “品質保証部門におけるテストプロセス改善モデル初期導入に関する取り組み,” ソフトウェア品質シンポジウム 2016, 2016.
- [4] 河野哲也, 山崎崇, 佐藤徳尚, “TPI NEXT による現場主導のテストプロセス改善を支援するための手法の提案,” ソフトウェア・シンポジウム 2017, 2017.
- [5] K. Beck, “Test Driven Development by Example,” Pearson Education, Inc., 2003, 訳 和田卓人, “テスト駆動開発,” 株式会社オーム社, 2017.
- [6] 内閣官房情報通信技術 (IT) 総合戦略室, “アジャイル開発実践ガイドブック,” 2021.