第１分科会　（チョコザイルチーム）

顧客との合意後の仕様齟齬検出意義とメトリクスについて

～仕様齟齬の予兆検出および最適な対処方法選択の提案～

主査　　：三浦　邦彦（矢崎総業株式会社）
副主査　：中森　博晃（パナソニック ファクトリーソリューションズ株式会社）
　　：山田　淳（東芝ソフトウェア・コンサルティング株式会社）
研究員　：市川　孝裕（株式会社インテリジェンス ビジネスソリューションズ）
　　海野　和由（矢崎総業株式会社）
　　小林　道央（株式会社インテック）
　　山本　久代（サントリーシステムテクノロジー株式会社）

研究概要

ソフトウェア品質は上流工程で担保することが大前提である．しかしながら，ウォーターフォール型の開発プロセスにおいて，上流工程で顧客と十分に仕様を整合したはずであるにも関わらず，開発工程が完了し，受入れ検査を実施した際に，顧客と開発チームでの仕様の解釈違い（仕様齟齬）が発覚するケースがある．そこで我々は，然るべき「開発の特性」と「メトリクス」を定義し，モニタリングを実施すれば，開発の途中で仕様齟齬の予兆を検知できるのではないか，また，予兆を検知した後，プロジェクトが狙う品質(Quality)・コスト(Cost)・納期(Delivery)，および開発チームの変化への適応能力に応じて，どのように対処すべきかを定義した「対処パターンと判断基準」を準備しておくことにより，最適な対処方法を合理的に選択できるのではないかという二つの仮説を立てた．

仮説を検証するため，研究員の所属各社のソフトウェア開発関係者を対象にアンケート調査を実施し，本研究で定義した「開発の特性」,「メトリクス」の有効性と実現性を確認した．また,「対処パターンと判断基準」の有効性を確認し，条件が整えば実現性がある事を確認した．

1.　はじめに

我々のソフトウェア開発現場では，ウォーターフォール型の開発プロセスが主流である．上流工程では顧客からの要求を確定し，設計工程に落とし込む必要があり，そのために要求分析に時間を費やしている．しかしながら，開発が完了し，受入れ検査で顧客が初めて完成品を確認した際に，顧客の意図通りにできていないこと，即ち顧客と開発チームでの仕様の解釈違い（仕様齟齬）が発覚するケースがある．その要因として，以下の３つが考えられる．

・顧客側要因：経験・実績，新規取引，プロジェクト関与度，窓口・担当交代等
・開発チーム側要因：業界知識・製品ドメイン知識不足・誤解釈等
・製品要因：新規製品・新規機能，仕様複雑性等

本研究では，上記問題を引き起こす要因となる「顧客」,「開発チーム」,「製品」のそれぞれの特性を「開発の特性」として定義し，開発の中で測定すべき「メトリクス」と併せて，仕様齟齬の予兆を詳細設計以降の早期段階で検知することを第一の目的とした．そして，予兆を検知した後で，プロジェクトが狙う品質・コスト・納期，および開発チームの変化への適応能力に応じて，どのように対処すべきかを定義した「対処パターンと判断基準」を用いることにより，適切な対処方法に切り替えて仕様齟齬を解決することを第二の目的とした．

2.　研究の背景

ウォーターフォール型の開発において，顧客と開発チームが開発の上流工程で仕様を整合し，その後の各工程のマイルストーンレビューで双方が成果物を確認しているにも関わらず，最終的な成果物が，顧客の意図と合わないケースがある．即ち，通常の中間成果物ならびにＱ＆Ａだけでは埋まらない溝がある．

その対応策として，開発の初期段階からアジャイル開発を導入し，顧客と開発チームが頻繁に現物確認を行えば，この問題は解決できる可能性がある．しかしながら，ウォーターフォール型の開発が定着している組織にとって，アジャイル開発への全面的な移行は短期間では難しく，リスクが高い．数多あるプロジェクトの中で，ウォーターフォール型で十分対応できているプロジェクトの方が多いという現実の中では，その決断は難しい．

また，上流工程にて仕様齟齬を防止する様々な手法が提案されおり，一定の効果を上げているが，仕様齟齬が発生するケースは皆無ではない．上流工程において仕様を整合する手法が本来的ではあるが，現実には上流工程で仕様齟齬を全て防止できていない．外部設計以降の工程において仕様齟齬を早期発見，対応する手法についての提案は少ない．

本研究では，ウォーターフォール型で開発を開始し，開発の上流工程で顧客と開発チームが仕様を整合した後，さらに開発を進めていく中で，仕様齟齬の予兆が検出できたところで，プロジェクト内で許容できる範囲で適切な対処方法に切り替えを行えば，仕様齟齬を解決できるのではないかと考えた．そのために，以下の方針を立てた．

方針１）できるだけ早い段階で，仕様齟齬の予兆を検知する．

方針２）仕様齟齬の予兆を検知した場合，仕様齟齬の有無を確認するために対処すべき方法を迅速に，且つ，合理的に判断し，実行する．

3.　研究目標

２．研究の背景で提示した２つの方針を実現させるため，我々は以下の方策を考えた．

・方策１）プロジェクトの開発工程内で集計して，仕様齟齬の予兆を迅速に総合的に判断できるような「開発の特性」と「メトリクス」を提案する．

・方策２）仕様齟齬の予兆を検知した後，そのプロジェクトが狙う品質・コスト・納期のバランス，および開発チームから見たプロセスの変更難度から，最適な対処パターンを合理的に選択できる表「対処パターンと判断基準」を提案する．

4.　研究内容

4．1　「開発の特性」,「メトリクス」,「対処パターンと判断基準」の作成

4．1．1　「開発の特性」,「メトリクス」,「対処パターンと判断基準」について

仕様齟齬の予兆を判断するために研究員の経験から有用と考えられる「開発の特性」，「メトリクス」をブレインストーミングにより抽出後，選択し整理した．また，予兆が実際に仕様齟齬であるかどうかを確認するための対処パターンおよびその際に選択すべき対処パターンの判断基準として「対処パターンと判断基準」を作成した．

「開発の特性」は,「顧客」,「開発チーム」,「製品」に分類して整理した.「開発の特性」の用途としては，予兆の判断材料として利用するほか，予兆検知後の対処方法パターン決定の判断材料，予兆検知後の対処の優先順位決定においても利用する．

「メトリクス」は，予兆の判断材料として利用する.「メトリクス」は開発中に継続的に取得しつづける点で，主として上流工程の終了時に一度取得する「開発の特性」と異なっている.「メトリクス」は定量的であるのに対して,「開発の特性」は定性的な側面を含むものであるという違いもある.「開発の特性」,「メトリクス」と分けているが，実運用時には図１のように１つのファイルとして，一覧することで，予兆を総合的に判断することを想定している．

「対処パターンと判断基準」は，プロジェクトが狙う品質・コスト・納期，および開発チームの変化への適応能力に応じて，どのように対処すべきかを整理したものである．

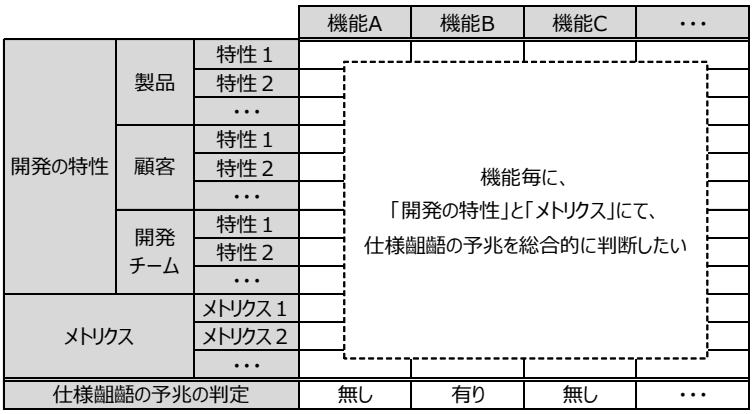

図 1.「開発の特性」と「メトリクス」利用イメージ

4．1．2　本手法の適用方法

ウォーターフォール型の開発といっても各社･各プロジェクトにて多種多様であるため，本手法の適用対象プロジェクトを下記に定めている．過度に具体的なプロジェクト像を定めることによる本手法の適用範囲を狭めることを防止するため，認識を共有する上で最低限必要と思われる点のみ定めている．表 1 のプロジェクトへの本手法適用時の全体像を図 2 に整理した．「開発の特性」と「メトリクス」については，4．3　検証結果の表 4 と表 5 に示す．

表 1．適用対象プロジェクト

開発期間	開発規模（最大時）	開発対象	開発プロセス	開発形態
半年～1 年	10 人／月以上の工数	エンタープライズ	ウォーターフォール型	受託開発

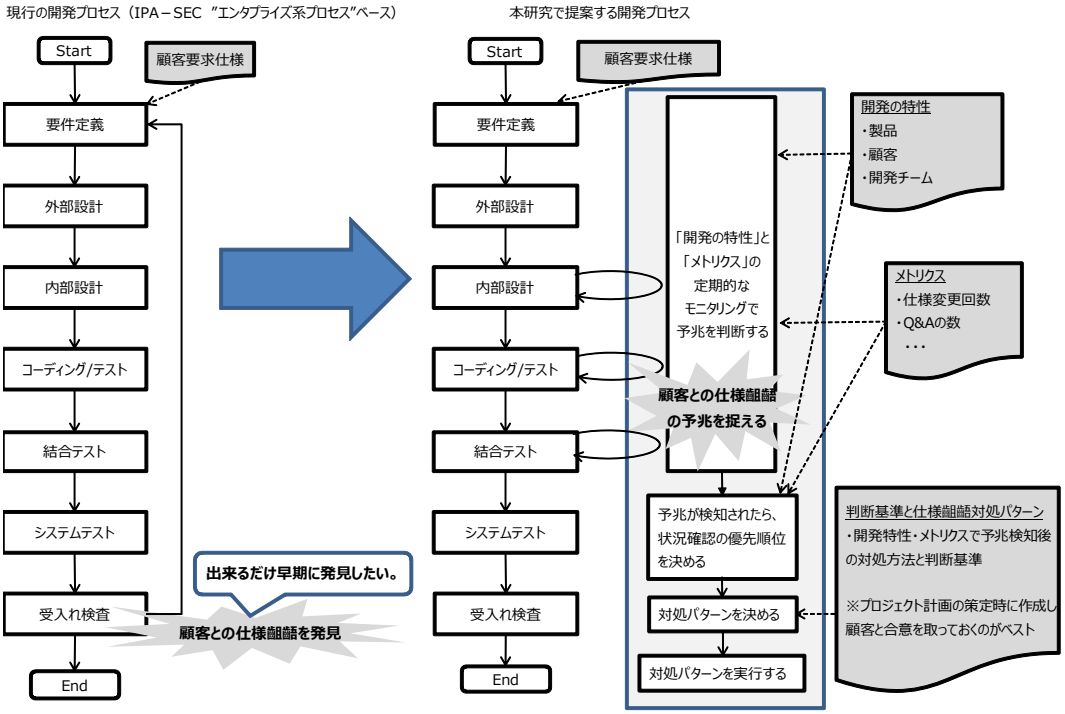

図 2．本手法の適用イメージ

4．2　検証手法

4．2．1　アンケートの実施

仮説を検証するため，研究員所属各社でアンケート調査を実施した．

アンケート対象者は，ソフトウェア開発経験者とした．（回答者数 計 4 社 43 名）

4．2．2　アンケート内容の工夫点

(1) 各メトリクスに対する評価

評価を明確にするために設問毎に回答選択肢を設定，中間値をなくし 4 択にした．選択肢のうち肯定的な 1)または 2)に回答した人の割合を好感度とし，好感度が 70%以上の設問項目を是と判断した．

表 2．設問毎の回答選択肢

No	回答選択肢			
1	1)非常に有効である	2)ある程度有効	3)あまり有効でない	4)全く有効でない
2	1)非常に必要である	2)ある程度必要である	3)あまり必要でない	4)全く必要でない
3	1)容易に実現できる	2)実現できる	3)実現が困難	4)実現不可能
4	1)非常に妥当である	2)ある程度妥当である	3)あまり妥当でない	4)全く妥当でない
5	1)非常に考慮すべき	2)ある程度考慮すべき	3)あまり考慮すべきでない	4)全く考慮すべきでない

(2) アンケート対象者のプロファイル（属性）

アンケートでは下記のプロファイルについて質問した．

表 3．アンケート対象者のプロファイル（属性）

No.	質問事項	回答選択肢
1	現在の役職	担当者，リーダ職，管理職，役員
2	開発経験	5 年未満，5 年以上 10 年未満，10 年以上 15 年未満，15 年以上
3	マネジメント（管理）経験	5 年未満，5 年以上 10 年未満，10 年以上 15 年未満，15 年以上
4	開発分野	エンタープライズ系，組込み系
5	開発プロセス経験（複数選択可）	ウォーターフォール型，スパイラルモデル，プロトタイプモデル，スクラム
6	仕様齟齬経験	経験がある，経験がない

4．3　検証結果

4．3．1　「開発の特性」の検証

「開発の特性」の各項目について，その有効性に対する好感度の高い順に整理した結果を表 4 に示す．全ての項目で 70%以上の高い好感度を得ており，妥当であると判断できる．

表 4．「開発の特性」の検証結果

目的	ID	対象	開発の特性	好感度	目的別の好感度順位
予兆を判断するため	C1	製品	製品の新規性（新規／保守／継続）	97.7%	1
	C7	顧客	開発内容の理解度	97.7%	1
	C9	顧客	開発途中での担当者の交代の有無	97.7%	1
	C11	開発チーム	経験・実績	97.7%	1
	C3	製品	顧客内で仕様についての意見が分かれていた機能があるか	95.3%	5
	C4	顧客	経験・実績	95.3%	5
	C13	開発チーム	業界知識・製品ドメイン知識	95.3%	5
	C6	顧客	性格（意見をコロコロ変える等）	90.7%	8
	C8	顧客	開発への関与度、協力度	90.7%	8
	C10	顧客	窓口は一本化されているか	90.7%	8
	C5	顧客	新規顧客／継続顧客	88.4%	11
	C12	開発チーム	性格（先入観が強い、慎重　等）	88.4%	11
	C14	開発チーム	開発途中での担当者の交代	88.4%	11
	C2	製品	技術の新規性（既存技術のみ／新規技術あり）	86.0%	14
予兆検知後に、状況確認の優先順位を決めるため	C16	製品	仕様が複雑な機能はどれか	100.0%	1
	C15	製品	コア機能はどれか	97.6%	2
	C17	製品	顧客が特に気にしている機能・フィーチャーはどれか	97.6%	2
	C2	製品	技術の新規性（既存技術のみ／新規技術あり）	80.5%	4
予兆検知後に、対処パターンを決めるため	C7	顧客	開発内容の理解度	97.6%	1
	C8	顧客	開発への関与度、協力度	92.7%	2

4．3．2　「メトリクス」の検証

「メトリクス」の各項目について，その有効性と実現性に対する好感度の高い順に整理した結果を表 5 に示す． M8 以外のメトリクスは，有効性と実現性ともに 70%以上の高い好感度を得ており，妥当であると判断できる．

表 5.「メトリクス」の検証結果

ID	測定単位	メトリクス	有効性に対する好感度	実現性に対する好感度	好感度順位
M2	予兆検知の単位	仕様変更回数（多さ）	97.7%	97.6%	1
M9	プロジェクト全体	プロジェクト全体での仕様変更回数（多さ）	97.7%	95.2%	2
M4	予兆検知の単位	開発組織内レビューまたは顧客レビューの工数・時間（少なさ、短さ）	88.4%	88.4%	3
M5	予兆検知の単位	Q&Aの数（少なさ）	83.7%	85.7%	4
M1	予兆検知の単位	関連するQ&Aの数（多さ）	83.7%	85.7%	4
M6	予兆検知の単位	前の工程の内容だったQ&Aの数（多さ）	81.4%	85.4%	6
M7	予兆検知の単位	Q&Aの数の変化（増加量大）	74.4%	83.3%	7
M3	予兆検知の単位	仕様の確定が遅れた機能か（有無）	74.4%	83.3%	7
M8	予兆検知の単位	Q&Aの数に対する仕様変更回数（少なさ）	69.0%	71.4%	9

4．3．3　「対処パターン」の検証

「対処パターン」について，その有効性と実現性に対する好感度の高い順に整理した結果を表 6 に示す． P2 は，有効性と実現性ともに高い好感度を得ており，妥当であると考える．一方，P1 は，有効性と実現性ともに低い好感度であり，妥当であるとは言えない．P4，P5，P3 は，有効性に対する好感度が非常に高いが，実現性に対する好感度が低い．そのため，アンケート回答者のプロファイル毎にどのような傾向があるかの詳細分析を実施した．

表 6.「対処パターン」の検証結果

ID	対処パターン	有効性に対する好感度	実現性に対する好感度	好感度順位
P2	テストケースやテストシナリオを作成して顧客に確認する	93.0%	88.4%	1
P4	気になる機能のみをスケジュール前倒しでプロトタイプ（育てて最終的に使うもの）を作成し顧客にデモ実施	97.7%	67.4%	2
P5	気になる機能のみをスケジュール前倒しでモックアップ（最終的に捨てるもの）を作成し顧客にデモ実施	90.7%	65.1%	3
P3	気になる機能のみをスケジュール前倒しで実装（完成版）をおこない顧客にデモ実施	86.0%	46.5%	4
P1	顧客に詳細設計書を再度レビューしてもらう	41.9%	39.5%	5

詳細分析は，統計的な分析（フィッシャーの正確確率検定で P 値<0．05）と，好感度 70%以上の主なプロファイルの考察により実施した．表 7 に詳細分析の結果を示す．

P4 と P5 の実現性は，表 7 に示すようにプロトタイプモデル，スクラムの経験者からは実現性に対して 70%以上の好感度を得ている．一方，ウォーターフォール型開発しか経験がない人に限れば，P4， P5 の有効性に対しては 90%以上の好感度であり，実現性に対しては 60%の好感度である．ウォーターフォール型開発しか経験がない人のコメントからは，実現性を低く評価した理由として，技術的に機能を切り出して動作可能な状態にすることの難しさが挙げられていた．従って，プロトタイプモデル，スクラムなどのウォーターフォール型開発以外の開発経験者を有する開発チームなど P4， P5 を実現することが可能である状況においては，対処パターンの有益な選択肢と考える．

P3 の実現性は，統計的にプロトタイプ開発プロセスの経験者から好感度を得ている．また，現在の役職が担当者からの好感度を得ているため，課題は技術的要因ではなく，コスト面や顧客との調整面であることが推測される．さらに対処パターンの実現性に関するアンケートにおいて否定的な回答者の理由をテキストマイニングした結果（図 3）から，担当者から役職が上がるほどコストや費用に対する言及が増えていくことがわかった．コスト面でのリスクを考慮して実現性を低く評価していると考えられる．P3 を実施するためには確かに追加コストが必要になるが，受入れ検査段階にて仕様齟齬が発覚して手戻りが発生することを考えれば，最終的なコストが低く抑えられる可能性がある．そのため予兆から仕様齟齬が強く疑われるケースにおいては，P3 は有益な選択肢として考える．

従って，予兆を検知した場合，まずはP2テストケースやテストシナリオを作成して顧客に確認するパターンにより仕様齟齬の有無を確認することを検討し，P3, P4, P5について，対処パターンの実現性と予兆の確度によって，選択肢として比較検討するのが望ましい．

表 7.「対処パターン」の詳細分析

ID	対処パターン	分析対象	統計的な相関があるプロファイル	P値	考察	
P4	気になる機能のみをスケジュール前倒しでプロトタイプ（育てて最終的に使うもの）を作成し顧客にデモ実施	実現性	なし	–	統計的な傾向	なし
					好感度70%以上の主なプロファイルの考察	・「プロトタイプモデル」開発プロセス経験者の72.7%が肯定的であった。 ・「スクラム」開発プロセス経験者の77.8%が肯定的であった。
P5	気になる機能のみをスケジュール前倒しでモックアップ（最終的に捨てるもの）を作成し顧客にデモ実施	実現性	なし	–	統計的な傾向	なし
					好感度70%以上の主なプロファイルの考察	・現在の役職が「リーダ」職の75%が肯定的であった。 ・「スパイラルモデル」開発プロセス経験者の71.4%が肯定的であった。 ・「プロトタイプモデル」開発プロセス経験者の72.7%が肯定的であった。 ・「スクラム」開発プロセス経験者の77.8%が肯定的であった。
P3	気になる機能のみをスケジュール前倒しで実装（完成版）をおこない顧客にデモ実施	実現性	現在の役職は担当者	0.03768	統計的な傾向	・現在の役職が「担当者」の人は肯定的であり、「担当者」以外の人は否定的であった。
			–	–	好感度70%以上の主なプロファイルの考察	・現在の役職が「担当者」の85.7%が肯定的であった。 ・「プロトタイプモデル」開発プロセス経験者の72.7%が肯定的であった。
P1	顧客に詳細設計書を再度レビューしてもらう	有効性	なし	–	統計的な傾向	なし
					好感度70%以上の主なプロファイルの考察	・好感度70%以上のプロファイルが存在しなかった。
		実現性	なし	–	統計的な傾向	なし
					好感度70%以上の主なプロファイルの考察	・好感度70%以上のプロファイルが存在しなかった。

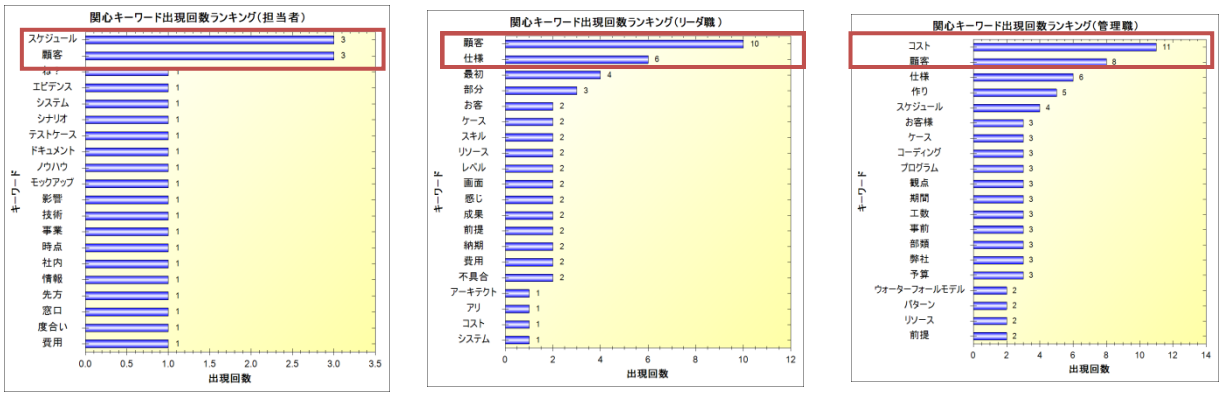

図 3. P3のテキストマイニングの結果

4．3．4　「判断基準」の検証

「判定基準」について，その妥当性に対する好感度の高い順に整理した結果を表8に示す． J6以外は非常に高い好感度を得ており，妥当であると考える． J6については，アンケート回答者のプロファイル毎にどのような傾向があるかの詳細分析を実施した．

表 8.「判定基準」の検証結果

ID	判断基準	好感度	好感度順位
J3	顧客が確認できる状態になるまでの期間の短さ	100.0%	1
J7	他の工程、全体スケジュールに与える影響度の低さ	95.3%	2
J8	顧客との調整（協力の引き出し易さ、顧客の時間の融通の）のし易さ	95.3%	2
J9	仕様齟齬が見つかった場合の無駄の少なさ	93.0%	4
J2	齟齬の発見のしやすさ＝「顧客に見せるモノ」の精度の高さ	92.9%	5
J1	仕様齟齬を確認するためのモノを作る作業のコストの安さ	90.7%	6
J4	顧客側にかかる作業負荷の軽さ	90.7%	6
J5	開発側に必要な技術・ノウハウの難度の低さ	88.4%	8
J6	ウォーターフォール型開発からの移行のし易さ	67.4%	9

表 9 に詳細分析の結果を示す．J6 の妥当性に関しては，統計的な傾向は見られなかったものの，ウォーターフォール型開発以外の開発プロセス経験者から 70%以上の好感度を得ているため，「判断基準」の対象として残すべきと考える．

表 9.「判定基準」の詳細分析

ID	判断基準	分析対象	統計的な相関があるプロファイル	P値	考察	
J6	ウォーターフォール型開発からの移行のし易さ	妥当性	なし	–	統計的な傾向	なし
					好感度70%以上の主なプロファイルの考察	・「プロトタイプモデル」開発プロセス経験者の90.9%が肯定的であった。 ・「スパイラルモデル」開発プロセス経験者の85.7%が肯定的であった。 ・「スクラム」開発プロセス経験者の88.9%が肯定的であった。 ・現在の役職が「管理職でない」人の72.7%が肯定的であった。

4．3．5　各「判断基準」に対する「対処パターン」の優位順の検証

各「判断基準」に対する「対処パターン」の優位順について，その妥当性に対する好感度の高い順に整理した結果を表 10 に示す． J6 以外は高い好感度を得ており，妥当であると考える． J6 については，アンケート回答者のプロファイル毎にどのような傾向があるかの詳細分析を実施した．

表 10．各「判断基準」に対する「対処パターン」の優位順の検証結果

表中の英字表記は、判断基準に対する各対処パターンの優位順（A:最も高い→D：最も低い）を示している。

			対処パターン					優位順(A～D)に対する好感度	好感度順位
			P1	P2	P3	P4	P5		
ID	分類	判断基準	①顧客に詳細設計書を**再度レビュー**をしてもらう	**②テストケースやテストシナリオ**を作成して顧客に確認する	③気になる機能のみをスケジュール前倒しで**実装（完成版）**をおこない顧客にデモ実施	④気になる機能のみをスケジュール前倒しで**プロトタイプ（育てて最終的に使うもの）**作成し顧客にデモ実施	⑤気になる機能のみをスケジュール前倒しで**モックアップ（最終的に捨てるもの）**を作成し顧客にデモ実施		
J3	D:納期	顧客が確認できる状態になるまでの期間の短さ	–	A	D	C	B	90.7%	1
J9	C:コスト	仕様齟齬が見つかった場合の無駄の少なさ	–	A	D	C	B	90.7%	1
J8	変更難度	顧客との調整（協力の引き出し易さ、顧客の時間の融通の）のし易さ	D	A	A	B	A	81.4%	3
J7	変更難度	他の工程、全体スケジュールに与える影響度の低さ	–	A	C	D	B	81.0%	4
J1	C:コスト	仕様齟齬を確認するためのモノを作る作業のコストの安さ	–	A	B	D	C	79.1%	5
J2	Q:品質	齟齬の発見のしやすさ ＝「顧客に見せるモノ」の精度の高さ	D	A	A	B	C	78.6%	6
J4	変更難度	顧客側にかかる作業負荷の軽さ	–	D	A	C	A	76.7%	7
J5	変更難度	開発側に必要な技術・ノウハウの難度の低さ	–	A	A	D	A	72.1%	8
J6	変更難度	ウォーターフォール型開発からの移行のし易さ	–	A	A	D	A	66.7%	9

表 11 に詳細分析の結果を示す．J6 の妥当性は，統計的な傾向として，「エンタープライズ系」の人は肯定的であることは言える．

表 11．各「判断基準」に対する「対処パターン」の優位順の詳細分析

ID	判断基準	分析対象	統計的な相関があるプロファイル	P値	考察	
J6	ウォーターフォール型開発からの移行のし易さ	妥当性	組み込み系ソフトウェア開発への携わり	0.0251	統計的な傾向	・主に携わっているソフトウェア開発の分野が「組み込み系」の人は否定的であり、「エンタープライズ系」の人は肯定的であった。
			–	–	好感度70%以上の主なプロファイルの考察	・「プロトタイプモデル」開発プロセス経験者の90.9%が肯定的であった。 ・「スパイラルモデル」開発プロセス経験者の85.7%が肯定的であった。 ・現在の役職が「リーダ職」の73.3%が肯定的であった。

5．　終わりに

5．1　研究成果，振り返り

プロジェクトの開発工程内で集計して，仕様齟齬の予兆を迅速に総合的に判断できるような「開発の特性」を 17 項目，「メトリクス」を 8 項目提示することができた．

「開発の特性」,「メトリクス」の監視開始タイミングはそれぞれ異なっている.「開発の特性」「メトリクス」による予兆の検知を実運用する際には，表 12 に示す開発工程の段階に応じて監視を開始していくと良い.

「開発の特性」と「メトリクス」の関係について,「開発の特性」により，仕様齟齬を疑う基準となる「メトリクス」の閾値が異なってくる．例えば，Q&A の数が同じであっても，顧客側の特性として実績・経験が少ない場合は仕様齟齬の予兆となり，実績・経験が多い場合に予兆とみなす必要がないなどのことが起こりうる．また表 12 に示す通り,「開発の特性」にてリスクとなる特性に応じて，重点的にどの「メトリクス」に注目するのが良いのかを変化させていくことで，より効果的に予兆検知できると考える.

表 12.　「開発の特性」,「メトリクス」の監視開始タイミングと「開発の特性」のリスクに応じた重要「メトリクス」

推奨順位	監視開始タイミング	開発の特性				メトリクス	
		ID	対象	特性名	リスク要因	ID	メトリクス名
1	要件定義	C1	製品	製品の新規性（新規／保守／継続）	新規	M4	開発組織内レビューまたは顧客レビューの工数・時間（少なさ、短さ）
						M5	Q&Aの数（少なさ）
2	要件定義	C7	顧客	開発内容の理解度	低い	M3	仕様の確定が遅れた機能か（有無）
						M4	開発組織内レビューまたは顧客レビューの工数・時間（少なさ、短さ）
3	外部設計	C9	顧客	開発途中での担当者の交代の有無	有り	M2	仕様変更回数（多さ）
4	要件定義	C11	開発チーム	経験・実績	少ない	M4	開発組織内レビューまたは顧客レビューの工数・時間（少なさ、短さ）
						M5	Q&Aの数（少なさ）
5	外部設計	C3	製品	顧客内で仕様についての意見が分かれていた機能があるか	有り	M2	仕様変更回数（多さ）
						M3	仕様の確定が遅れた機能か（有無）
6	要件定義	C4	顧客	経験・実績	少ない	M3	仕様の確定が遅れた機能か（有無）
						M4	顧客レビューの工数・時間（少なさ、短さ）
7	要件定義	C13	開発チーム	業界知識・製品ドメイン知識	少ない	M4	開発組織内レビューまたは顧客レビューの工数・時間（少なさ、短さ）
						M5	Q&Aの数（少なさ）

また，仕様齟齬の予兆を検知した後，そのプロジェクトに最適な対処パターンを合理的に選択できる,「対処パターン」を 4 項目,「判断基準」を 8 項目，各「判断基準」に対する「対処パターン」の優位順を 8 項目提示することができた.

5．2　課題と今後の展望

今回のソフトウェア開発関係者へのアンケート調査によると 83.7%の人が，顧客との合意後に仕様齟齬が発覚したという経験をしている．また，本研究のように開発の途中段階で仕様齟齬を検知，対応する手法について研究していく意義について,「ある」または「ある程度ある」と回答した人は 100%であった．本研究に留まらず，さらに深掘りしてこのテーマについて研究していく必要があると考える.

本研究では，仕様齟齬の予兆を迅速に総合的に判断できるような「開発の特性」と「メトリクス」を提示した．しかしながら，「開発の特性」の最適な定量化方法については，各社・各プロジェクトの状況により異なるため示しておらず，利用者に委ねる形とした.

今後の課題として，「開発の特性」と「メトリクス」をどのように組み合わせれば効果的に予兆を判断できるかを実プロジェクトでの試行を通じて予兆を判断し，最適な対処パターンを選択できたかを検証することが挙げられる.

6.　参考文献

[1]鷲崎弘宜，メトリクスによるプロダクトの品質把握と改善-Goal-Question-Metric(GQM)法のコツ，および，組織目標との整合，早稲田大学グローバルソフトウェアエンジニアリング研究所，2013
[2]IPA 独立行政法人情報処理推進機構: ソフトウェア開発データ白書の活用
[3]アジャイル開発「スクラム」におけるプロジェクトオーナーの「勘所」～QCD 問題を検知するための「メトリクス」と「勘所性」の提言～日本科学技術連盟 SQiP 研究会第一分科会，2015