84

業務アプリケーション改修時の XDDP 適用事例[1]
～品質の見える化による、効果的なプロセス改善の実践～

1. 概要

弊社が提供する地方公共団体向けパッケージシステム（以下、公共系システム）は政策変更による制度改正など、多様かつ頻繁な変更要求が発生する。その一方で住民生活に影響を与えないよう、ミスのない確実な変更も求められる。こうした状況に対応するため、既存システム改修等の派生開発に特化した開発アプローチである派生開発プロセス（XDDP2）[1][2]の手法を適用した。

導入に先立ち、品質状況を見える化する自社ツール「品質カルテ」を活用し実態を把握した。さらにプログラム発注先やドキュメント群をつなぐ「ハブ」として XDDP のプラクティスを採用、結合テスト以降の不具合削減と納期・コスト遵守に成功した。

XDDP は組込みシステムを中心に普及している開発手法であるが、業務アプリケーションの分野においても品質改善効果を得られることが確認できた。ここでは本取り組みの内容ならびに、本取り組みを通じて得られた知見について紹介する。

2. 取り組みの目的

2.1. 解決すべき課題

公共系システムの特徴について以下に述べる。

- 制度改正への対応
  国の政策により行われる制度改正に対し、システムの仕様を確実に追従させる必要がある。閣議決定から施行までの期間が短い場合は迅速な改修が必要となり、不具合による手戻りが命取りになることがある。
- 業務が多岐に渡り、独自制度も多い
  地方自治体には国で決められたもの以外にも、都道府県独自、市区町村独自など、多岐に渡る業務や制度が存在し、それぞれの自治体の業務や制度に合ったシステムを提供する必要がある。
- 社会的な影響が大きい
  公共系システムは法律や財産、国や自治体の政策を取り扱うため、法令違反や給付金等の計

[1] 事例提供：株式会社両備システムズ 技術開発センター 品質管理グループ 河内 一弘 氏

[2] XDDP: eXtreme Derivative Development Process の略

算ミスがあってはならない。給付漏れや課税額、支給額の誤りなどは住民生活に与える影響が極めて大きく、リリース後に障害が発生することは許されない。

前述のとおり、公共系システムは多様かつ頻繁な変更要求が発生する一方で高い信頼性が求められる特徴がある。しかし、テスト工程に入った後での不具合発生による手戻りが多く発生していながら具体的な根本原因が掴めず、有効な対策が打てないまま対応に追われる状況が続いていた。

また、弊社パッケージは長年にわたるバージョンアップを重ねており、社内的にも新規開発より既存システム改修案件の比率の方が高くなっている。しかし、社内の標準プロセスは新規開発用の V 字モデルをベースに作られたものであり、標準プロセスどおりに開発しても十分な品質確保に結びつかないケースも散見されていた。

さらに、弊社はプログラミングと単体テストを協力会社に発注するケースがあるが、受け入れテスト時の品質に課題があった。これらの状況を解決し、生産性を維持しながら既存システム改修時の品質を向上させることが組織の課題となっており、品質管理部門がプロセス改善活動に乗り出すこととなった。

2.2. 本取り組みの目標

本取り組みの目標として、以下を設定した。

- 既存システム改修における品質低下要因を明らかにする。
- 生産性を落とすことなく既存システム改修時の品質を向上させる。
- 実施した施策にどのような効果があるかを明らかにするとともに、継続して効果を上げるための仕組みを確立する。

2.3. 取り組み前の状況

弊社の公共系システムは、1990 年代～2000 年代初頭にかけてメインフレームからオープン系への移行を行って以降、Web アプリケーション化などの変遷を得ながらも、基本的には当時開発したシステムを約 20 年に渡り改修や機能追加を重ねながら提供している。こうした派生開発のスタイルは、経年での税率計算処理など維持すべき行政制度を持つ公共系システムにおいては、いわゆる「枯れた（成熟した）」業務ロジックを実装したモジュールで構成されるため信頼性を確保する面で有効な反面、処理の複雑化や大規模化による保守性の低下を引き起こしていた。弊社システムは全国約 1700 の自治体のうち 500 を超える自治体で稼動しており、各自治体の独自業務への対応、制度改正への対応を個別に行う案件が多く存在する。モジュールの共通化やパラメータでの処理切り替えで保守性を向上させる取り組みも進めているもののまだ十分とはいえず、システム本体の業務ロジック改修で対応するケースも多く、負担が増えているのが現状である。

また、システム改修は自社ですべての工程を実施する場合もあるが、プログラミングや単体テストといった一部の工程を協力会社に依頼するケースも多く、この場合の品質確保も課題となっていた。

3. 取り組みの対象、適用技術・手法、評価、計測

3.1. 対象

今回の事例で対象としたシステムについて、以下に述べる。

案件１）制度改正対応案件

全国約 120 の自治体に提供しているシステムにおける、手当金支給制度改正に伴う改修案件である。本制度は第 2 子以降の手当加算額を増額するものであるが、第 2 子、第 3 子以降で加算額が異なる、年収に応じた手当額を支給する必要があるなど、算出条件が複雑なものとなっており、慎重なシステム改修が必要であった。

開発期間	:	2016 年 2 月 ～ 2016 年 10 月
開発工数	:	約 14 人月
全体規模	:	約 300 Kstep
改修規模	:	約 15 Kstep
開発体制	:	設計および結合テスト以降を自社で実施し、プログラミングと単体テストを協力会社へ発注。

案件２）システム移行案件

クライアント・サーバ版システムを Web システムに移行するにあたり、単純に旧システムを Web システムに移行するだけではなく、別の業務システムから一部機能を取り込み、一つのシステムに統合した形で提供する案件である。

当システムには各自治体の業務に合わせた設定項目が多数あり、修正時にはこれらのパラメータがどこに影響しているか理解して修正しないと予期せぬ箇所に影響が発生することがあり、修正には細心の注意が必要であった。

また、当システムはこれまで社内メンバーによる開発が中心であったが、今回は社内の開発担当者が別の案件に従事していたため、協力会社に開発を依頼することになった。

しかしながら、過去の外注案件はコードレビューで指摘が繰り返され、検収時のチェックで手戻りが多発する事例が見られており、受け入れ時の品質が不安視されていた。

開発期間	:	2016 年 12 月 ～ 2017 年 6 月
開発工数	:	約 20 人月

全体規模　：　約 2500 Kstep
改修規模　：　約 22 Kstep
開発体制　：　基本設計と結合テスト以降を自社で実施し、詳細設計～プログラミング～単体テストを協力会社に発注

3.2. 方法論（どんな技術・手法を用いて実施したか）

本事例は、XDDP のプラクティスを用いた品質改善事例であるが、最初から XDDP ありきで取り組んでいたわけではない。どのような経緯で XDDP 採用に至ったかを説明する。

プロセス改善の方法論として、PDCA サイクルを用い、最初に「あるべき姿」を計画し、その姿に現状を合わせていくスタイルがある。

しかし、今回は既存システム改修において品質問題が発生しているという状況に直面しているものの、その本質や原因が明らかになっておらず、具体的な計画を立てづらい状況であった。このため、「現在、何が起きているのか」、現状を明らかにした上で解決策を検討するという PDCA の変形版として知られている「CAPDo アプローチ[3]」を採用することとした。

表 84-1 PDCA サイクルと CAPDo サイクル

	PDCA サイクル		CAPDo サイクル	
1	Plan	「あるべき姿」を達成する計画を立案	Check	現状の問題点を把握し、計測する。
2	Do	計画に基づいて実行	Act	現状を元に改善策を立案する
3	Check	計画どおりの効果が出ているかを計測	Plan	改善策を元に具体的な施策を計画する。
4	Action	チェック結果に基づいた是正	Do	計画に基づき施策を実行する。

3.3. 「品質カルテ」を用いた品質状況の把握

弊社では品質メトリクスの集計ツール「品質カルテ（図 84-1）」の導入を進めていた。これは、弊社独自に開発したツールでレビュー・テスト記録をインプットに品質メトリクス[4][5]を集計し、工程別のサマリ情報を出力する機能を持ち、定量的な品質評価や予測を行うことが可能となる。

これまでも開発時の品質記録としてレビュー記録やテスト記録は利用していたが、様式が統一されておらず各部門で独自に様式を作成していた。また、複数ファイルに跨る記録の集計は手作業に頼っていたため、記録を残すだけになることが多く、それを用いた品質傾向の分析は十分に行われているとはいえない状況であった。品質カルテでは、所定様式を用いれば各々のファイルに分かれた品質記録をまとめて読み込み、品質診断レポートを出力する機能を有しており、開発現場や品質管理担当者は品質記録を集計する手間から解放される。

今回は、本ツールを活用して対象システムの現状把握および、施策実施後の効果測定を行うこととした。

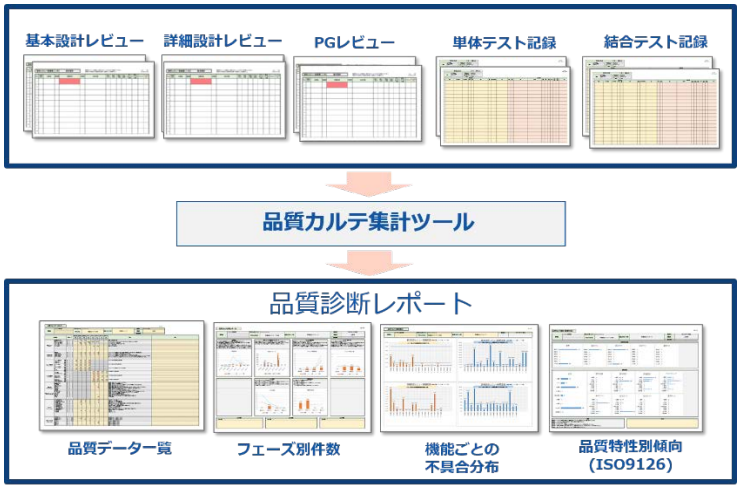

図 84-1 品質カルテ

品質カルテで集計可能な主なメトリクスは以下である。

表 84-2 品質カルテで集計可能なメトリクス

測定対象	指標	単位	説明
設計時品質	レビュー指摘密度	欠陥数／頁	設計量に見合った数の不具合が検出できていること。
	レビュー参加人数	参加人数／回数	適切な人数、体制でレビューできていること。
	レビュー時間密度	分／頁	レビューに適切な時間をかけていること。
テスト時品質	テスト密度	項目数／KLOC	開発量に見合った数のテストを実施できていること。
	障害発生密度	欠陥数／KLOC	開発量に見合った数の不具合が検出できていること。
	障害検出数	欠陥数／項目数	テスト量に見合った数の不具合が検出できていること。
検出不具合の内訳	品質特性別検出数	検出不具合件数	ISO/IEC9126 で定義されている 6 つの品質特性（機能性、信頼性、使用性、効率性、保守性、移植性）別に指摘件数を集計し、その偏りや分布を表示する。
	欠陥モジュールの偏り	検出不具合件数	どのモジュールや機能に指摘が集中しているのか、その分布をグラフで表示する。
フェーズ間推移		検出不具合件数	開発フェーズ別の不具合件数をグラフ化して表示
検出した工程／検出すべき工程		検出不具合件数と工程	不具合を検出した工程と、本来検出すべき工程の件数を表示

3.4. 状況分析と対応策の検討

品質カルテを用いて開発状況を見える化した結果、以下の状況であった。

- 協力会社が実施する単体テストのテスト密度および障害発生密度は基準値に収まっていた。
- 自社での受入テストを兼ねて実施する結合テストの段階で不具合検出件数が急増していた。

つまり、単体テストの段階では品質に問題なしと判断されているにもかかわらず、その後の結

合テストで品質問題が発覚する状況になっていた。

この状況に対して、バグ票の内容を精査したり、結合テスト担当者へのヒアリングを行った結果、検出不具合の傾向として設計者の意図と異なる実装がされていたり、実装が漏れていたりするケースが多いことが判明した。

この結果を踏まえ、設計担当者、プログラミング担当者双方を交えたヒアリングを実施し、起きている状況を調査した。その結果、要求を仕様化する段階のプロセスが曖昧で、要求漏れや仕様漏れが発生し、十分な設計情報がプログラミング担当者に伝わっていないことが、結合テスト以降の不具合に繋がっていることが原因と結論づけた（図 84-2）。

起きている問題は、以下の２パターンに大別される。

① 要求を実現するための仕様が曖昧

１点目は、業務要求の仕様への落とし込みができていないケースである（図 84-2）。

設計者は顧客ヒアリングを元に設計書を作成するが、顧客要求をシステムの仕様としてどのような形で実現するかの記載が十分でなく、いわゆる「行間の広い」設計書のまま、プログラミング担当者に渡されていた。その結果プログラミング担当者は自分なりの解釈で実装してしまうことになり、結合テストの段階で初めて問題が発覚するというものである。

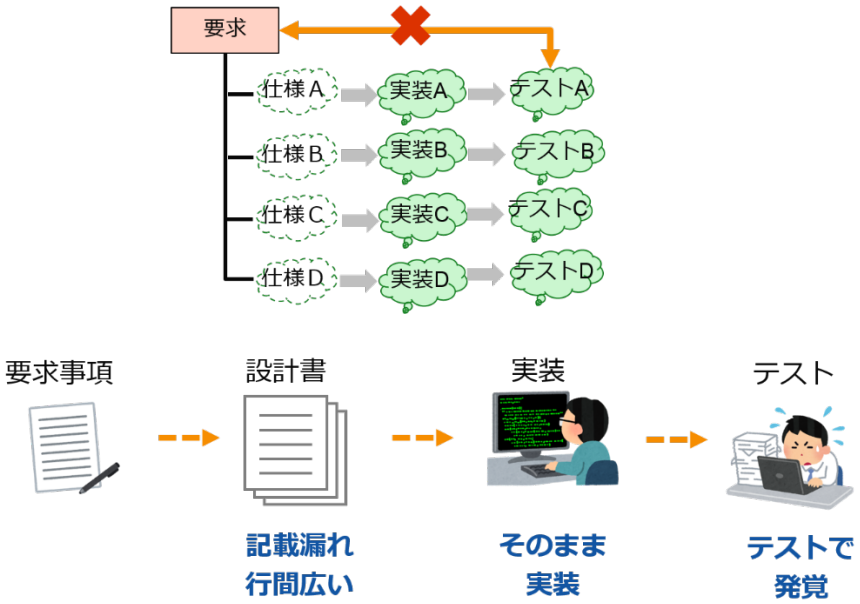

図 84-2 要求を実現するための仕様が曖昧

② 要求が曖昧なまま、一部の仕様だけが実装される

２点目は、一部のシステム仕様だけがピンポイントで実装者に伝わり、顧客要求を満たすだけのシステム仕様が満足されていないケースである（図 84-3）。

これは、顧客要求の明確化が不十分なことに起因しており、必要な機能の一部が実装されなか

ったり、処理性能や異常時のリカバリーといった非機能要求を満たすことができないなどの問題が、結合テストやシステムテストの段階で発生するというものである。

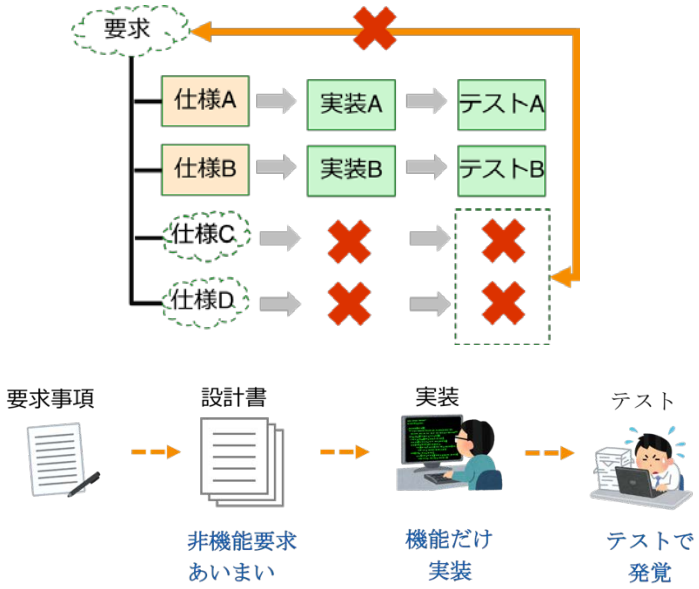

図 84-3 要求が曖昧なまま、一部の仕様だけが実装される

この 2 パターンに共通する原因はいずれも「要求と仕様のトレーサビリティが確保できていない」ことである。よって、このトレーサビリティを確保しやすい開発手法を取り入れることで、多くの不具合を未然に防ぐことができ、品質を向上させることができるのではないかと考えた。

3.5. XDDP の導入

この問題を解決する方法として浮上したのが、「USDM 記法による変更要求仕様書」「トレーサビリティマトリクス」「変更設計書」の三点セットにより要求と仕様のトレーサビリティを確保する開発手法である派生開発プロセス（XDDP）の導入である。

XDDP は組込みシステムの開発では導入事例があるものの、弊社の公共系システムといった業務システムへの適用事例は聞いたことがなかった。しかし、組込みシステムと公共系システムは表 84-3 で示す共通点があり、トレーサビリティ確保においては実装技術に依存する要素は少ないと判断したことから、弊社システムの現状や開発スタイルに合わせてテーラリングすれば適用可能と判断し、導入のための計画と準備を進めることとした。

表 84-3 組込みシステムと公共系システムの共通点

	組込みシステム	公共系システム
既存システムの改修頻度	高い（旧製品を元に新製品開発）	高い（制度改正が多い）
改修の難易度	高い（法規制やハードウェア、関連製品への影響など）	高い（法制度や他顧客への影響など）
ユーザからの品質要求	高い（人命に影響）	高い（住民生活や財産に影響）
トレーサビリティ確保の必要性	高い（法規制やリスク要因とのトレーサビリティを保証する必要がある）	高い（法制度やリスク要因とのトレーサビリティを保証する必要がある）

4. 取り組みの実施、及び実施上の問題、対策、工夫

4.1. 計画、準備（課題解決に対するアプローチ）

過去に適用実績のない方法論を取り入れることになるため、社内の現状にフィットさせ、かつ確実に効果を上げられるようにすることを念頭に、計画と事前準備を行った。

プロセス改善の現場で起こりがちな「進め方を計画したものの、現場の理解を得られず手法を取り入れてもらえなかった」「形だけの様式利用に終始し、実際の品質向上に結びつかなかった」というケースを未然に防ぐため、以下のような活動を行った。

① ステークホルダーに趣旨を説明

導入にあたり、開発プロジェクトの管理職やリーダーといったキーパーソンに現在の品質状況と XDDP の導入の必要性を説明した。変更要求仕様書など新たな様式を作成したり、一部のプロセスの変更を行うこととなり、従来の開発案件とは異なる進捗傾向が現れることが予想されるため、事前に上司の理解を得ておくことが必要と考えたためである。

さらに、初めての取り組みで手厚いフォローが必要になることが予想されたため、品質管理担当者がプロジェクトに関与し、必要に応じアドバイスをすることについても予め同意を得るようにした。

② コミュニティとの連携による情報収集

社内に XDDP の有識者が存在しなかったため、社外事例も含めた情報収集を行うこととした。

XDDP については派生開発推進協議会（AFFORDD）という、派生開発に関する情報交換や研究会活動を行うコミュニティが存在する。当協議会の Web サイトや公開資料を閲覧するとともに、当協議会の関西部会（AFFORDD 関西）の会合に参加し、協議会メンバーの体験談に基づくアドバイスを得るなどの活動を行った。

また、当協議会は研究活動の成果として様々な資料を公開している。初心者向けにわかりやす

くまとめられている資料も多く、開発担当者に以下の資料を印刷し配布するなどの活用を行った。

> 困ってませんか？派生開発　～XDDP はじめの一歩～
> http://affordd.jp/tech_documents/affordd-t3_20130524.pdf
>
> USDM 小冊子 基礎編
> http://affordd.jp/tech_documents/affordd-t2-usdmtext-basic_1.3.pdf
>
> USDM 小冊子 補足編
> http://affordd.jp/tech_documents/affordd-t2-usdmtext-appendix_1.3.pdf

③ 様式およびプロセスのテーラリング

XDDP で定められている変更要求仕様書とトレーサビリティマトリクスは、「要求」「理由」「説明」「仕様グループ」「仕様」「トレーサビリティ」といった記載項目が存在するが、現場の開発メンバーが導入しやすくなるよう、開発メンバーと協議しながら様式の一部変更を行った（図 84-4）

本来、トレーサビリティマトリクスには、対象システムの全モジュールを記載することになっているが、今回の対象システムは総ステップ数が 300Kstep を超えるため、膨大な量となってしまう。今回のシステムは業務毎にモジュールが分かれており、異なる業務のモジュール間の呼び出し関係は限定的であることから、トレーサビリティマトリクスの一覧性を重視して、関連する業務のモジュールだけを記載することとした。

また、変更仕様書へのリンクは仕様記載部とトレーサビリティマトリクス記載部の交点に記載するが、すべての交点にリンクを記載すると量が多くなりすぎること、今回は既存の設計書が業務単位に存在し実装モジュールの特定が可能だったことから、仕様単位に設計書へのリンクを書く列を設けることとした（図 84-4）。

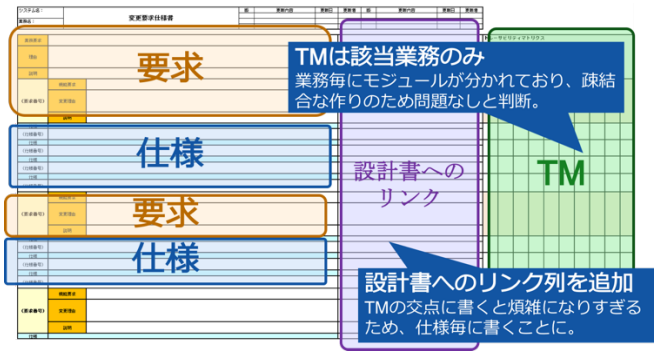

図 84-4 様式の見直し

④ プロセスのテーラリング

XDDP のプラクティスのうち、今回は「変更要求仕様書」と「PFD によるプロセス設計」を採用することとした。(図 84-5)。変更設計書については、今回案件は既存の設計書が存在していたことから、そのまま活かすこととし、変更要求仕様書からリンクすることとした。今回は協力会社に詳細設計とプログラミングを発注するため、お互いの「担当プロセス」「変更要求仕様書の記載箇所」を色分けし、それを基に担当箇所の合意に用いるなど、担当範囲と責任の所在が明確となるような工夫を行った。

これらの工夫により、変更要求仕様書をドキュメントや開発担当者間のコミュニケーションを促進する「ハブ」として位置づけ、開発に不可欠な存在に位置づけるようにした（図 84-6)。

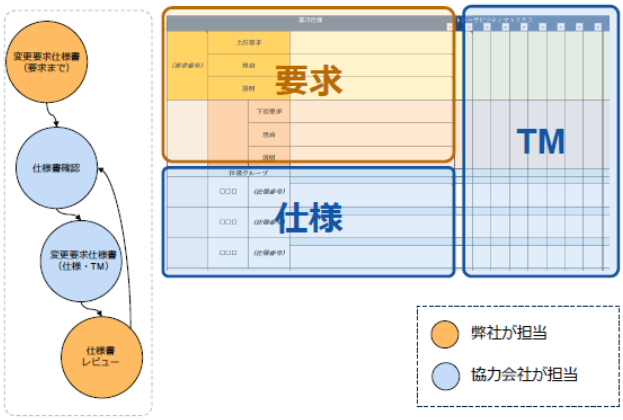

図 84-5 協力会社との執筆分担を明確化

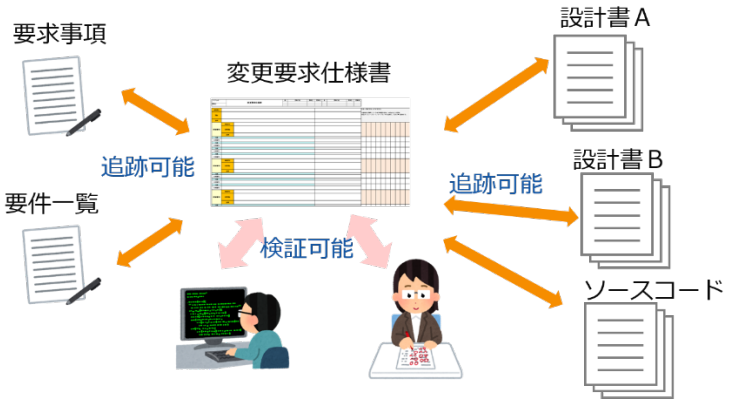

図 84-6 変更要求仕様書と既存ドキュメントの関係

4.2. 実施（課題解決に向けた具体的な取り組み内容、活動）

開発期間中も、計画通りの実施がされているかどうかを定期的にフォローを行った。

① 品質管理担当者の設計レビューへの参加

品質管理担当者が XDDP 推進者の立場で開発部門の設計レビューに参加した。

変更要求仕様書の「要求」「理由」「説明」「仕様」などの記載欄に適切に記載するにはある程度の経験やノウハウが必要となるが、初めての使用であることに加え、開発担当者は業務要求の理解や設計作業に集中しているため書き方まで意識が行き届かない可能性があった。このため、品質管理担当者が第三者的な視点でレビューに参加し、変更要求仕様書の記載や表現に着目したチェックを行った。この結果、「仕様部分に要求事項を書いてしまう」「仕様を定量的かつ検証可能なように書けていない」「適切な理由が書けない」などの問題を指摘することができ、記載レベルの統一を図ることができた。

本レビューでは、品質管理担当者が業務要求についての質問を行うことで、開発担当者の中で新たな気づきが生まれ、より深い視点での検証を促進し、その結果検討漏れや記載漏れが見つかるなどの副次的効果も得られた。

② 開発リーダーとの定期連絡会

弊社側、発注先側のリーダーを交えた定期的な状況確認会を実施した。これは、XDDP の導入によって進捗遅れや業務上の混乱などの問題が発生していないか確認し、早期に解決するためのものであった。これにより、開発担当者だけでなく、リーダーから見た視点でも開発プロセスの進行状況を捉えることができるようになった。また、弊社側が認識していなかった受注先側での課題や意識のズレなどの問題を、早期に検出し解決することができた。

5. 達成度の評価、取組みの結果

5.1. 達成度の評価

開発完了時点で、達成度の評価を実施した。

品質カルテでの測定で問題となっていた、結合テスト以降の不具合検出が削減されているかどうかを確認した。

案件１）制度改正対応案件

結合テスト以降の不具合検出数が適用前に比べ 432 件→8 件と大幅な減少がみられた。

表 84-4 不具合件数の減少（制度改正対応案件）

	設計～単体テスト	結合テスト以降	合計
XDDP 適用前	250 件	432 件	682 件
XDDP 適用後	55 件	8 件	63 件

案件２）システム移行案件

結合テスト以降の不具合検出数が 28 個→2 個に減少し、大幅な品質向上効果が認められた。

表 84-5 不具合件数の減少（システム移行案件）

	設計～単体テスト	結合テスト以降	合計
XDDP 適用前	513 件	28 件	541 件
XDDP 適用後	244 件	2 件	246 件

導入の結果、2 案件とも結合テスト以降での不具合件数を削減できた。

不具合件数といった定量的な面以外に定性的な効果を確認するため現場担当者へのヒアリングを実施した。こちらについても概ね好評であり、次回以降も XDDP を適用したいという声が大半であった。

現場担当者のコメントを以下に示す。

・仕様が明らかになっているので、コーディングしやすかった（プログラミング担当者）。
・いつもは結合テストでトラブルが続出するのに、全然不具合が出なくて逆に心配になった（結合テスト担当者）。
・協力会社からの「よい質問」が増えた。仕様理解が進み、さらに踏み込んだ点まで質問できる余地が増えたためと捉えている。
・レビューにおいて「コーディング規約違反」といった保守性に関する指摘の割合が増えた。コードの品質が上がり、保守性のチェックを行う余裕が生まれたためではないか（開発リーダー）。
・変更要求仕様書に変更点が一覧形式でまとめられているため、出荷時のリリースノートを作成するのが楽になった（システムエンジニア）。

5.2. 取り組みの結果

本取り組みで得られた結果の考察を以下に述べる。

① XDDP を導入しても納期・コストへの悪影響はなかった

XDDP は、変更要求仕様書などの成果物を新たに作成することから、当初は工数増による納期やコストへの影響や、開発担当者の負担が懸念されていた。しかし、実際には当初計画どおりのコスト、納期でのリリースを達成できており、目立った工数増加も見られなかった。また、現場の開発担当者からも負担が増えたなどの不満の声は聞かれなかった。

この理由を明らかにするため現場担当者へのヒアリングを実施したところ、「設計やプログラミングに取り掛かる前の下準備に要する手間や混乱が減った」といったコメントが得られた。これは、従来型の開発では設計段階における既存プログラムの調査や、担当者間での質疑応答が繰り返されるといった、「目に見えにくく属人化されたプロセス」、いわゆる「裏プロセス」による工数の浪費が発生していたことが考えられる。これらの裏プロセスがプロセス設計で整理されたプロセスに置き換わり、かつ変更要求仕様書に要求事項や仕様が整理されて書き込まれたことから、プロセスが見える化され無駄がなくなったことが好影響をもたらしたものと思われる（図 84-7、図 84-8）。

また、レビュー工数の増加は見られたものの、これは新たに作成した変更要求仕様書および、こちらに関連する箇所のレビュー工数であると考えられる。情報が整理されたドキュメントをレビューすることで設計段階での不具合検出を行いやすくなり、後工程での手戻りが減ったため、トータルでの工数増加を防ぐことができたと考えられる。

定性的な面でも開発担当者の負担感は増加していないことが確認でき、「次もこの方法で開発したい」とのコメントを得ることができた。これはトータルでの工数が増えていないことに加え、開発中の混乱や手戻り発生によるストレスから解放されたことも寄与していると思われる。

さらに、現時点では出荷後の大きなトラブルも起きていないことから、品質改善に効果があったものと判断している。

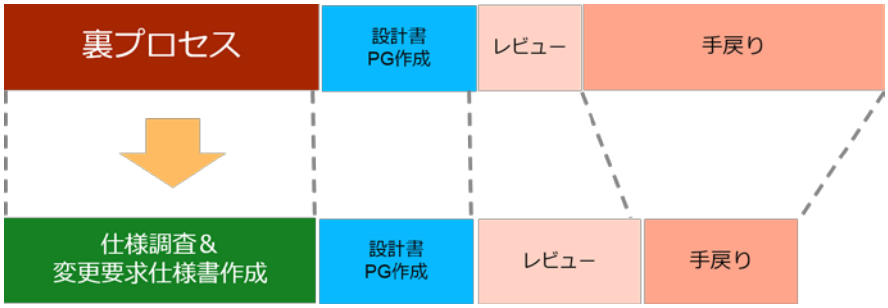

図 84-7 裏プロセスによる工数圧迫

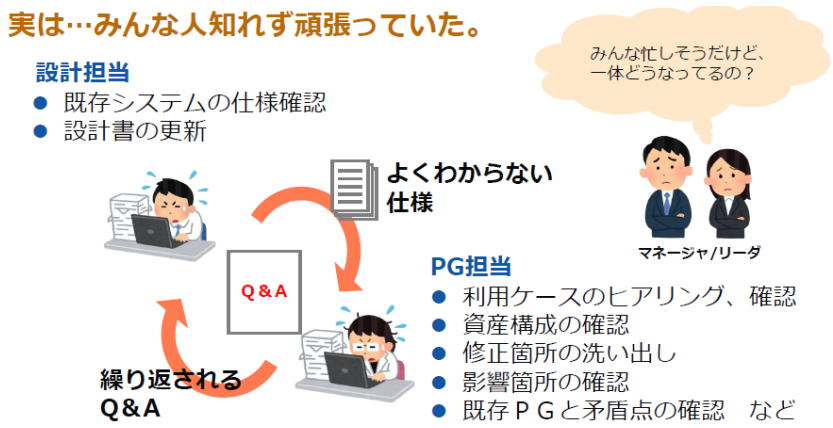

図 84-8 裏プロセスの内容

② XDDP 導入で得られる改善効果

XDDP 導入効果を明らかにするため品質カルテを検証したところ、バグ発見工程比率に変化が現れていた（図 84-9）。制度改正案件においては、以前は開発中検出不具合のうち 63%が結合テスト以降で検出されていたが、XDDP 導入以降は 11%まで減少した。これは、変更要求仕様書により要求と仕様のトレーサビリティが確保され、かつ仕様の漏れも発生しにくくなったため、設計情報が確実に実装されるようになったこと、ドキュメント化によりレビュー段階での早期検出が可能になったことが理由であると判断している。

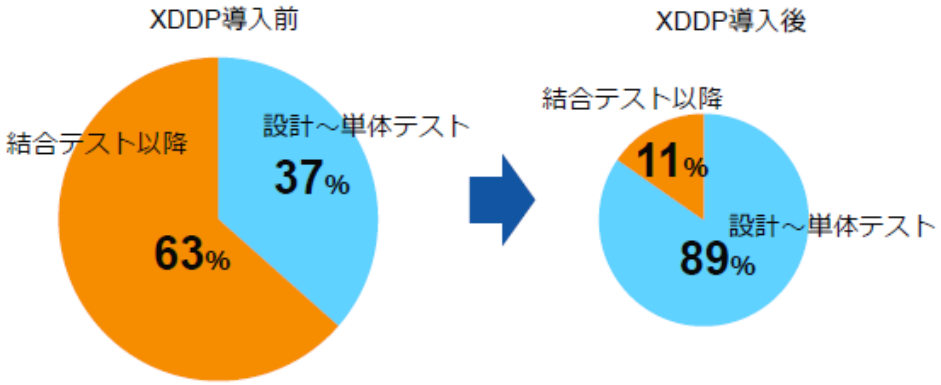

図 84-9 バグ発見工程比率の変化

6. 今後の取り組みと考察

6.1. 取り組みで得られた知見

今回の取り組みにより、プロセス改善を成功させるためには、以下の取組みが必要との知見を得ることができた。

① 見える化環境を作り、課題を明確にする

今回は、XDDP のプラクティスを導入する前に「品質カルテ」で品質メトリクスを収集し、起きている問題点を定量化したことで、ステークホルダーにプロセス改善の動機付けを促すことができた。また、事前に計測を行っておくことにより、施策実施後の効果確認を行いやすくなるメリットもあった。

本事例では品質カルテには自動集計機能を取り入れたが、ツールなどを活用し品質データ収集に伴う現場の負担感を減らすことも、スムーズなプロセス改善には重要との教訓も得られた。

② ステークホルダー間の相互理解重視

プロセス改善のアプローチとして、組織トップや上司の指示で行う「トップダウンアプローチ」と現場主導で進める「ボトムアップアプローチ」の二通りがあるが、今回は両者を組み合わせたアプローチとなった。具体的な問題点を把握したり、成果に結びつけたりするには現場の協力やモチベーションが不可欠である一方、改善活動のためのリソースを確保するためには上司の理解と承認が必要となる。今回のケースでは、現場から挙がった問題点を定量化し、上司に説明することで活動リソースを確保することができた。

今回の事例では、プログラミングを発注している協力会社のメンバーにも事前説明を行い、趣旨を理解してもらうなど、すべてのステークホルダーの理解を得て改善活用を進めていくことが重要であるとの教訓を得た。

③ プロセスの妥当性を事前に検証する

開発プロセスやツール、ドキュメント様式の使用を事前に決めていても、それらが品質作り込みの勘所に作用しない限り品質を向上させることはできない。既存プログラムの改修においては、改修元システムの状況やドキュメントの整備度合い、有識者の所在などがプロジェクト毎に異なり、一律のプロセスを杓子定規に適用するだけでは十分な成果を収めることが難しい。案件ごとの特徴を捉え、関係者の意見も聞きながらテーラリングを進めるという地道な作業が重要であるとの教訓を得た。

④ 段階的な適用

プロセス改善を行う場合、組織内の全てのプロジェクトで一斉に始めてしまうケースがある。しかし、どのプラクティスが品質問題の解決にどのように作用するかの見極めがついていない状態で着手すると施策が空振りし工数が無駄になってしまう場合がある。さらに複数のプロジェクトで同時多発的に問題が発生した場合、少ない推進メンバーでは十分なケアを行うことができない。今回の事例では最初は少数のプロジェクトを対象に試行する形で始め、対象プロジェクトのメンバーと密に連絡を取りながら進めていくこととした。これにより、きめ細かなサポートが可能となり品質向上に結びついたとともに、現場からのフィードバックを多く得ることができたため、今後の展開にも役立つノウハウを多く得ることもできた。

6.2. 現在の状況と今後の取り組み

社内に XDDP を根付かせるために、以下の施策を実施している。

① 社内外への成果発表／社内説明会の開催

実践中の事例について、派生開発推進協議会主催の「派生開発カンファレンス 2017」で事例発表を行った（図 84-10）。この内容を社内にも発表し、認知度を高める取組みを行った。

この取り組みにより、社内での XDDP の認知度が高まり、2017 年第 2 四半期以降、導入プロジェクトは 2→6 に増加しており、効果が出ているといえる（図 84-11）。

図 84-10 派生開発カンファレンス 2017 での事例発表

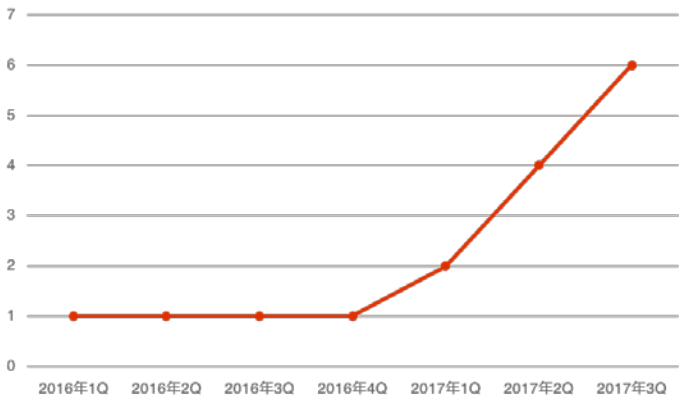

図 84-11 XDDP 導入プロジェクト数の推移

② XDDP 導入サポートの実施

派生開発は、改修元システムの状況やドキュメントの整備度合い、有識者の所在などがプロジェクトによって異なるため、現場の声を聞きながら細かなプロセステーラリングを行っていく必要がある。このため、品質管理部門を推進部門とした開発プロジェクトへの XDDP 導入サポート活動を実施している（図 84-12）。社内への紹介以降、既存システムの改修に悩むプロジェクトからの声が多く寄せられ、問い合わせが増えている状況である。新たに導入サポートを行ったシステム案件においても、結合テスト以降の不具合削減に成功し、コスト、納期共に遵守できており、先行 2 プロジェクトと同様の結果が得られている（図 84-13）

当プロジェクトには XDDP だけでなく、Selenium WebDriver を用いたテスト自動化の取り組みも並行して推進しており、テスト効率化によって空いた工数を更なる品質向上や、新しいビジネスの創出に割り当てるなどの成果を目指していく。

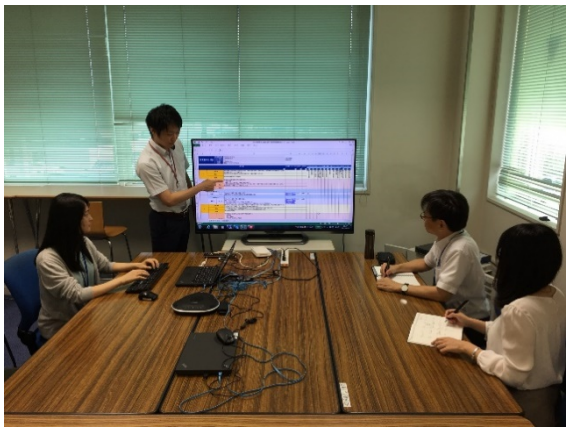

図 84-12 XDDP 導入支援の様子

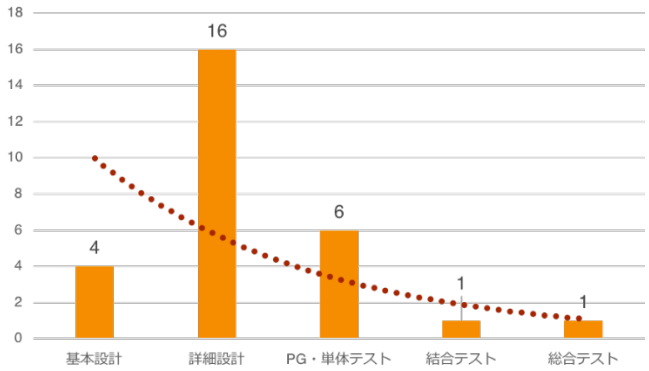

図 84-13 XDDP 導入後におけるフェーズ毎の検出不具合数

6.3. 課題

社内での適用プロジェクトが増加してきた状況であるが、現時点では以下の課題があると判断している。

① 導入サポートができる人材の不足

プロジェクトの現状分析やプロセスのテーラリングが必要となるが、XDDP という過去に社内になかった方法論を導入したこともあり、導入ノウハウと経験を有する人材がまだ少ない状態である。現状のままでは、導入サポート工数の不足から適用プロジェクト数が頭打ちになったり、現場へのケアが手薄になったり、一旦導入したやり方が元に戻ってしまうプロジェクトが出てくる懸念がある。これについては、以下の対策を予定している。

- XDDP 導入経験者を増やし、自グループ内の推進役を担ってもらう
- 品質課題の把握やプロセス改善を行うワーキンググループの立ち上げ
- XDDP 導入のコツやポイントを記載した「XDDP 導入ガイド」の整備

② 有識者がいないプロジェクトへの対応

今後、有識者が存在しないプロジェクトや十分な保守ドキュメントが揃っていないプロジェクトについても品質を改善する何らかの手立てが必要と考えている。現時点では結論が出ていないが、以下の検討を進めていく予定である。

- ソースコード静的解析ツールの選定と活用
  人の能力と経験に頼るソースコード解析では、開発規模が大きい場合に対応できないため、メソッドの呼び出し関係を解析するツールを導入するなど取り組みを進めていく。
- テスト自動化ツールなどを活用した仕様化テストの導入
  保守ドキュメントが残されていないレガシーシステムにおいては、対象システムを実際に動作させ、その仕様を明らかにしていく仕様化テスト(Characterization Testing)という手法が知られている[6]。現在、弊社ではテスト自動化の取り組みを進めており、このノウハウ

を応用して、レガシーシステムの仕様抽出ができないか取り組んでいきたいと考えている。

- コミュニティとの情報交換

  現在、派生開発推進協議会ではレガシー化したシステムの仕様を抽出する「スペックアウト」の方法が研究されている。これらの研究会との情報交換等により、有効な方法を見つけていきたいと考えている。

6.4. おわりに

いくつかの課題はあるものの、組込みシステムを中心に普及してきた XDDP の手法は、地方自治体向けシステムといった業務アプリケーションの改修においても有効であることが確認できた。

今後は、地方自治体向けシステムにとどまらず、弊社グループが手がけている医療、流通、文教等の各システムに普及させ、さらなる品質向上を図っていく所存である。

－以上－

参考文献

[1] 清水吉男：[入門＋実践]仕様が書けていますか？要求を仕様化する技術・表現する技術、技術評論社、2010

[2] 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、技術評論社、2007

[3] 独立行政法人情報処理推進機構　ソフトウェア・エンジニアリング・センター：プロセス改善ナビゲーションガイド～なぜなに編～、オーム社、2007

[4] 独立行政法人情報処理推進機構　ソフトウェア・エンジニアリング・センター：定量的品質予測のススメ～IT システム開発における品質予測の実践的アプローチ～、オーム社、2008

[5] 独立行政法人情報処理推進機構　ソフトウェア・エンジニアリング・センター:
続 定量的品質予測のススメ～IT システム開発における定量的品質管理の導入ノウハウと上流工程へのアプローチ～、オーム社、2011

[6] マイケル・C・フェザーズ：レガシーコード改善ガイド、翔泳社、2009

独立行政法人情報処理推進機構 技術本部 ソフトウェア高信頼化センター（IPA/SEC）