アジャイル開発方式の適用による パッケージソフトウェアの品質向上

Improving Packaged Software Quality by Applying Agile Software Development

● 松浦豪一

あらまし

近年，パッケージソフトウェアに対するお客様の要望は多様化している。株式会社富士通マーケティングでは，これまでパッケージソフトウェアFUJITSU Enterprise Application GLOVIAを活用したソリューションを提供してきたが，お客様からのフィードバックや要望に対し，素早く対応できないことが課題であった。その原因は，ウォーターフォール方式によってパッケージソフトウェアを開発しており，仕様確定や工程完了判断までに期間やコストを必要としていたためである。この課題を解決するため，アジャイル開発方式を適用した開発プロセスの改善に取り組むことになった。アジャイル開発方式のプラクティスであるタイムボックスやテストファーストの適用により，早期に問題を把握し改善することが可能となり，製品の品質を向上できた。また，プログラムの振る舞いを変更せずにプログラム構造だけを修正するリファクタリングにより，生産性を向上できた。更に，ユーザーストーリーを使った計画ゲームにより，商談の獲得率も向上できた。

本稿では，パッケージソフトウェア開発におけるアジャイル開発方式の適用について述べる。

Abstract

Customer needs regarding packaged software have been growing increasingly diverse in recent years. While offering solutions based on its packaged software FUJITSU Enterprise Application GLOVIA, Fujitsu Marketing Limited has not been able to quickly respond to customer feedback and requests. This is because the waterfall model, being employed for software package development, is a time-consuming and costly practice to determine specifications and judge when a process is complete. In order to overcome this challenge, we have decided to adopt agile software development to improve our software development processes. By employing timeboxing and a test-first approach, the method allows us to detect and address problems at an early stage, leading to better program quality. It also helped to enhance productivity through refactoring, which allowed us to reconfigure programs without changing their behavior. Furthermore, by adopting the planning game method based on user stories, we have increased the chance of having successful business contracts. This paper explains the agile software development applied in packaged software development.

ま え が き

近年，パッケージソフトウェアに対するお客様の要望は多様化している。株式会社富士通マーケティング（以下，富士通マーケティング）では，これまでパッケージソフトウェアFUJITSU Enterprise Application GLOVIAを活用したソリューションを提供してきたが，お客様からのフィードバックや要望に対し，素早く対応できないことが課題であった。その原因は，ウォーターフォール方式によってパッケージソフトウェアを開発しており，仕様確定や工程完了判断までに期間やコストを必要としていたためである。この課題を解決するため，アジャイル開発方式を適用した開発プロセス改善に取り組むことになった。

本稿では，まず富士通マーケティングのパッケージビジネスの推進体制を紹介する。次に，ウォーターフォール方式の課題とその解決策であるアジャイル開発方式について述べ，最後に効果を紹介する。

パッケージビジネスの推進体制

本章では，パッケージソフトウェアビジネスの推進体制について説明する。パッケージソフトウェアビジネスは以下の3チームで推進され，これらが共同して製品企画・開発・検査を行っている。

(1) 拡販チーム

パッケージソフトウェアを販売するために，商品の価値をアピールすることに注力し，商談発掘，提案を行っている。

(2) 開発チーム

パッケージソフトウェアを開発し，品質を確保する。開発に当たっては，機能・コスト・実現性を重視し，決められたコストの中で要件を満たすことを目指している。

(3) インプリメントサポートチーム

提供前にパッケージソフトウェアを検査し，その結果を踏まえてお客様のシステムに適用することにより，課題解決に注力する。

ウォーターフォール方式の課題

本章では，ウォーターフォール方式によるパッケージソフトウェアの開発方法とその課題について説明する。ウォーターフォール方式の工程は，**図-1**に示すとおり，要件定義，設計，プログラミング，テスト，出荷前検査の5段階に分類され，各工程が上流から下流に流れるように進められる。お客様の要望から要件を定義し，機能を設計することでソフトウェアの仕様を確定する。要件定義から設計までの工程は順次行う。設計は，外部仕様とシステム仕様に分かれており，確定した仕様を基に，プログラミング，テストの工程を順次行う。また，各工程で品質分析を行い，工程完了を判断する。テストでは，設計内容が仕様どおりであるかを検証し，外部仕様設計とシステムテスト，システム仕様設計と結合テストが対応付けられている。

以下に，ウォーターフォール方式では開発期間を短くできない理由を説明する。

(1) 要件定義・外部仕様定義

ウォーターフォール方式の開発では，お客様の要望に対応する仕様を明確にした上で関係者の合意を得るため，要件定義および外部仕様定義に1か月程度の期間が必要になる。パッケージソフトウェアの開発においては，開発チームが仕様を検討し，拡販チームとインプリメントサポートチームがレビューを行い，承認を得る必要がある。

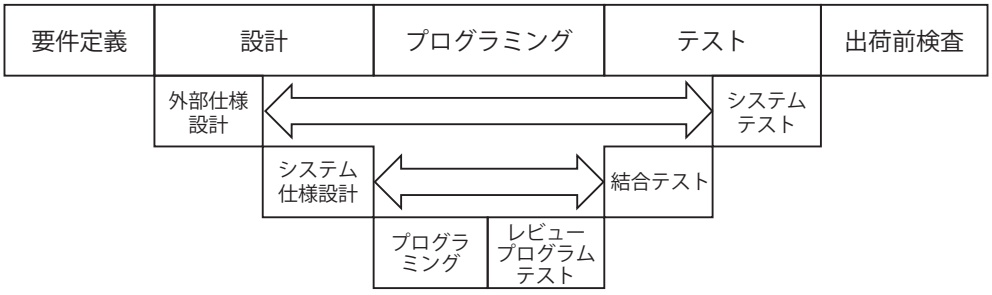

図-1　ウォーターフォール方式の工程定義

（2）工程完了判断

ウォーターフォール方式の開発では，工程完了を判断するために作業を止める必要があり，開発期間短縮のボトルネックとなっている。したがって，工程完了を判断し，検査に合格するまで次の作業に進めない。開発期間全体が長ければ数日の作業停止は吸収できるが，期間が1か月といった短期の場合では，1日の作業停止でも進捗に大きな影響を与えてしまう。

アジャイル開発方式による解決策

本章では，前章の課題を解決するためアジャイル開発方式を適用したパッケージソフトウェアの開発について説明する。

アジャイル開発方式は，2週間から4週間の単位で期間を区切り，繰り返し開発を行う。開発段階の成果物をお客様に提供し，フィードバックをもらい，開発内容，プロセスなどの全般的な課題を認識し，解決策を学習する。この学習により開発者の能力を向上させ，品質を向上させる開発方式である。

アジャイル開発方式には，実践者の経験知を集めた技法であるプラクティスが無数に存在する。今回は，パッケージソフトウェア開発に適合するプラクティスに着目した。

プラクティスの内容およびパッケージソフトウェア開発での実践内容について以下で説明する。

● タイムボックス(1)の設定

アジャイル開発方式では，問題を早期に把握するために作業にタイムボックス(注)を設定する。ウォーターフォール方式では，開発内容に合わせて期間を設定するが，アジャイル開発方式ではタイムボックスに合わせて開発項目を小さくする。具体的には，実装する機能を減らすことで開発期間を短縮する。例えば，設計からテスト完了までを2週間から4週間で行う。この開発期間は，プロジェクトの開始から完了まで常に固定して行う。開発期間に実施するタスクやテストについても，タイムボックスを設定する。タイムボックスは，最終的な製品提供までの期間より短いため，早期に問題を把握できる。

ここで，パッケージソフトウェア開発に対して設定したタイムボックスについて説明する。

（1）システムテスト（5日間）

パッケージ開発完了時点の品質を確認するために，システムテストを実施する。その工数を10人日と設定し，二人で作業するため期間を5日間とした。テストが不合格であった場合は，お客様への提供は中止する。システムテストの内容は，以下のとおりである。

・業務フローテスト
・過去障害の再発防止テスト
・新機能確認テスト
・テスト担当者による探索テスト

（2）リリース単位の開発期間（20日間）

成果物のリリース単位を20日間と設定する。これは，1か月に1回の提供を実現することを想定している。したがって，1か月以内に提供できるように開発内容を調整する。この開発期間の成果物を含むパッケージソフトウェアの全機能に対して，システムテストを行う。リリース単位で品質を評価することで，ウォーターフォール方式の工程完了の判断を代替できる。

（3）項目ごとの開発期間（5日間）

開発項目一つあたりの期間を5日以内とする。その際，リリース単位の開発期間を越えないようにする。開発完了までに設計，プログラミング，テストのレビューを行うため，2日に1回は別のメンバーがチェックすることになる。これにより，問題を早期に把握することを狙っている。

（4）タスク

開発項目に合わせてタスクを洗い出し，各タスクの所要時間を2時間以下にする。これにより，計画精度の向上を狙っている。

パッケージソフトウェア開発にタイムボックスを適合する理由は，以下のとおりである。

・テストにより発生する問題を早期に把握できる。
・問題を早期に把握することにより，チームのメンバーが問題とその解決策を学習するため，プロダクトの品質が向上する。

● テストファースト(2)

アジャイル開発方式では，短期間でプログラムを作成するために，テストを作成してからプログ

（注） 固定の時間枠を使った時間管理の方法である。時間に合わせて作業内容を変更するため，問題を把握しやすいという特長がある。

ラムを開発するテストファースト（**図-2**）というプラクティスを利用する。図中①のように，外部仕様設計・システム仕様設計と同時にテスト設計を実施する。その後プログラミングを実施し，結合テストを行う。テスト設計を先行することにより，開発時の抜けや漏れを防止する効果がある。図中②のように，結合テストで問題が発生し設計をやり直す場合は，外部仕様設計，システム仕様設計，およびテスト設計に戻り，①と同じ手順を実施する。アジャイル開発の場合，開発期間が短く，開発規模も小さいため，比較的容易にテストを先に設計できる。

また，パッケージソフトウェア開発に対して，開発項目の設計と同時に結合テストを設計する。そのため，プログラム開発前に設計書とテスト仕様書のレビューを実施する。併せて，テスト設計の手法に要因分析表[3]を採用した。

パッケージソフトウェア開発にテストファーストを適用する理由は，結合テスト以降で問題が発生しないよう，問題を早期に把握するためである。

● リファクタリング[2]

アジャイル開発方式では，プログラム構造の複雑性を低減させるために，プログラムの振る舞いを変更せずにプログラム構造だけを修正するリファクタリングを行う。これにより，同じプログラムを何度も修正することによって生じるプログラム構造の複雑化を避けることができる。

パッケージソフトウェア開発において，リファクタリングの適用は，開発計画時にお客様の要望が多いプログラム群に対して優先して行う。手順は以下のとおりである。

(1) 対象のプログラム群に対して，機能を網羅するテストを設計する。
(2) 上記テストを利用し，プログラムが正しく動作することを確認する。
(3) プログラムの構造を修正し，修正後のプログラムが修正前と同じ動作をすることを確認する。

パッケージソフトウェア開発にリファクタリングを適用する理由は，以下のとおりである。

- リファクタリングによりソースプログラムの可読性が高まり，品質が向上する。
- テストを完了したプログラムが複雑でないため再利用しやすくなり，生産性の向上が期待できる。
- リファクタリングする際に機能全般をテストするため，レベルダウンの防止につながる。

● ユーザーストーリーを使った計画ゲーム[4]

計画ゲームとは，一定の期間で行う作業を簡単な作業単位に分割するもので，関係者を集め，開発項目ごとの見積もりや優先順位を決める。計画ゲームを実施するときに，お客様の要望をユーザーストーリーという形式でまとめる。ユーザーストーリーの例を以下に示す。

「私は，運用管理者として，データの取り消し機能がほしい。なぜなら，ハードディスクの容量が枯渇しないように保守したいから」

このように開発項目をユーザーストーリーで表現することで，お客様の要望とそれによって得られる価値を明確にできる。お客様の要望に基づいて仕様を作成し，その合意を得ることに比べて，要望のみで判断するため短期間で合意できる。アジャイル開発方式では，開発後の修正を許容するため早期に設計を確定する必要がない。したがって，開発チームもお客様の要望を実現するための工夫が容易になる。

計画ゲームをパッケージソフトウェア開発に適用する理由は以下のとおりである。

- 開発者がお客様の要望の価値を理解して開発を進められる。
- 要望に合致させるため，開発者が臨機応変に製品の仕様を変更できる。
- お客様の要望が具体的でなくても，開発に着手でき，作業を繰り返す中で要望を明確にできる。

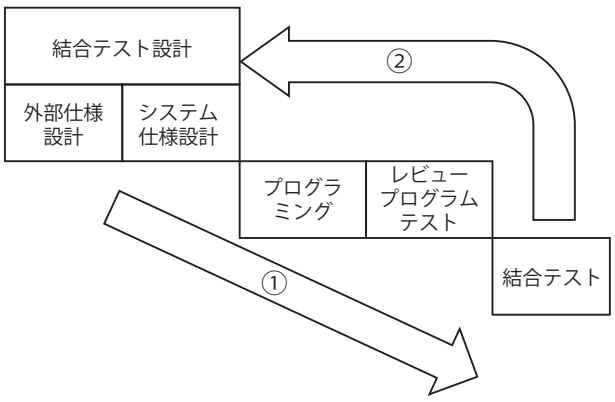

図-2　テストファーストの作業の流れ

表-1　タイムボックス手法の採用前後の障害検出率

	障害検出率（件/ks）	増減率（%）	出荷後障害検出率（件/ks）	増減率（%）
採用前	12.8	－	0.30	－
採用後	8.9	−37	0.11	−63

表-2　テストファーストの採用前後の障害検出率

	障害検出率（件/ks）	増減率（%）	出荷後障害検出率（件/ks）	増減率（%）
採用前	12.8	－	0.30	－
採用後	13.0	＋4	0.17	−44

効　　果

本章では，前章で説明した四つの解決策を実行した具体的な効果[5]を説明する。

(1) タイムボックスの設定

タイムボックス採用前後の障害検出率を**表-1**に示す。開発中の障害検出率は37％減の8.9件/ks（キロステップ）に減少し，更に出荷後の障害検出率は63％減の0.11件/ksへと大幅に減少した。タイムボックス設定の徹底により，設計不足・テスト不足などの問題を早期に把握し，再設計や再テスト設計を行うことで品質を高めることができた。

(2) テストファースト

テストファーストの採用前後の障害検出率を**表-2**に示す。開発中の障害検出率は4％増の13.0件/ksとわずかに上昇したが，出荷後の障害検出率は逆に44％減の0.17件/ksに大幅に減少した。これは，障害が事前に検出されていることを示している。

(3) リファクタリング

リファクタリングした部分の開発前後で障害検出件数の増加が見られないため，製品品質を悪化させることなく，構造上の問題を解決できた。

リファクタリングによる生産性の評価を**表-3**に示す。機能あたりの開発規模はリファクタリングしたものが最小で，かつ工数も最小である。これは生産性が高いことを示している。

(4) ユーザーストーリーを使った計画ゲーム

ユーザーストーリーを使った計画ゲームを採用することにより，商談の提案率・内示率が高まり，製品売上高が向上した。具体的には商談化・提案・内示の各件数の比率（商談の獲得率）が改善している。商談の獲得率が向上したのは，機能追加がタイムリーに実施され，商談活動に結びついたことが挙げられる。

表-3　リファクタリングによる生産性評価

	生産性（ks/人月）	機能あたりの工数（人月/機能）	機能あたりの規模（ks/機能）
未適用	1.2	0.29	0.35
テストファースト適用後	1.7	0.19	0.32
リファクタリング適用後	1.1	0.19	0.20

表-4　計画ゲームによる商談の獲得率

	商談化（件数）	提案（件数）	内示（件数）
V1	24	4	1
V2	18	4	1
V3	10	3	1
V4	5	3	1

拡販施策やサポート部門からの情報を素早く製品に反映することにより，短期間に製品の品質を向上させることができた。その結果，デモンストレーションと併せた提案がお客様の要望に合ったものになり，商談の獲得率に結びついた。**表-4**は製品のバージョンごとの商談獲得率を表している。1社の内示を得るための提案件数と商談件数であり，商談化と提案の件数が少ないほど商談の獲得率が高いことを示している。

む　す　び

本稿では，パッケージソフトウェア開発にアジャイル開発方式のプラクティスを適用することによる品質向上について述べた。パッケージソフトウェアに対する要望はますます多様化しており，お客様起点の素早い解決が求められている。更なる改善のためには，アジャイル開発方式の適用をパッケージソフトウェア開発のみならず，パッケージソフトウェアを利用したお客様システムの構築に拡大することが必要不可欠である。富士通マーケティングでは，パッケージソフトウェア開発にアジャイル開発方式を適用した経験を活かし，アジャイル開発方式の教育を行っている。パッケージソフトウェアを利用したお客様システムの構築にお

いても，アジャイル開発の教育を含めて対応することで，お客様のICTパートナーとしての使命を果たしていく。

参考文献

(1) K. Schwaber et al.：The Scrum Guide by Ken Schwaber and Jeff Sutherland，the originators of Scrum.
http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-JA.pdf
(2) ケント・ベックほか：エスクストリームプログラミング．第1版第1刷，平成27年6月25日，p.181，ISBN978-4-274-21762-3.
(3) 松浦豪一：ソフトウェアテストシンポジウム2014. テスト設計のタイミングと手法の変更による、品質向上と生産性向上ー順序を入れ替え、繰返し考えるー.
http://jasst.jp/symposium/jasst14tokyo/pdf/D2-1.pdf
(4) ケント・ベックほか：XPエクストリーム・プログラミング実行計画．第1刷，2001年，p.142，ISBN4-89471-341-1.
(5) 独立行政法人情報処理推進機構：先進的な設計・検証技術の適用事例報告書 2015年度版．パッケージソフトウェア開発プロセス改善による品質向上と生産性向上．p.10.
http://www.ipa.go.jp/files/000049407.pdf

著者紹介

松浦豪一（まつうら　ひでかず）
（株）富士通マーケティング
GLOVIA事業本部
サポート・サービス統括部
パッケージソフトウェアを利用したお客様システムの構築，およびアジャイル開発方式の教育に従事。