銀行勘定系システム更改における要件定義の工夫

～リバースエンジニアリングへの挑戦～

（株）さくらケーシーエス

■ 執筆者Ｐｒｏｆｉｌｅ ■

2001 年　（株）さくらケーシーエス入社
システムコンサルティング担当

黒田　晃弘

■ 論文要旨 ■

銀行の勘定系システムパッケージシステムが保守期限を迎えるため、システムのアップグレードを行うこととなった。システムの基本機能のみを利用しているのであれば、パッケージシステムの機能は保証されている。しかし、本事例では標準機能にカスタマイズを実施し、機能追加を行っている。そのため、カスタマイズ機能や標準機能とカスタマイズ機能の連携について確認する必要がある。

パッケージを利用した勘定系システムでは、リバースエンジニアリングツールを利用しただけでは、現行システムの正確な把握が困難である。そこで、解決策としてリバースエンジニアリングツールと業務要件から業務処理フローを解析する新しい仕組みを開発したので、その方法を提案したい。この方法により、パッケージ基本機能とカスタマイズ機能について、業務要件を可視化することが可能となる。

■ 論文目次 ■

■ 図表一覧 ■

１．はじめに

銀行の勘定系システムは 10 数年利用することが多い。導入当初はシステムの内容を理解している人が多く在籍している。従って、システムの機能改善や障害に対する対応も迅速に対応できる。

しかし、システムが安定して機能改善も一段落すると、銀行はシステム部門の要員を削減する。その結果、システムの内容を理解している要員が銀行内部に少なくなっていくものである。既存機能のシステムの大規模な改修やシステムのリプレースが必要になると、既存のシステムを理解している有識者が少なく、そのプロジェクトを実施するうえで、有識者の不足という事態になる。このような場合に、リバースエンジニアリングを適用することにより、システム機能の可視化を行い、システムに関するノウハウ者の育成が試みられている。ここでいう、リバースエンジニアリングとは、既存のシステムを可視化するプロセスをいう。既存の仕様書、設計書をリバースエンジニアリングツールなどにより、現在の機能と整合性のとれたドキュメントする。この可視化されたドキュメントにより、対象システムの仕組み、構成要素、業務処理を理解することである。

しかし、ツールを用いただけでは、現行仕様の把握と要件定義が十分にできることは少ないのが実情である。特にパッケージシステムを用いた場合、パッケージの基本機能とカスタマイズ機能を融合させたドキュメントを作成するのことは経験的に難しいとを多くのエンジニアが感じていることである。この点が多くのプロジェクトで課題となっている。

今回の事例では、現行の銀行勘定系システムパッケージシステムが保守期限を迎えるため、システムのバージョンアップにより更改を行うこととなった。システムの基本機能のみを利用しているのであれば、パッケージシステムの機能は保証されている。しかし、本事例では標準機能にカスタマイズを実施し、機能追加を行っている。そのため、カスタマイズ機能や標準機能とカスタマイズの連携について確認する必要がある。銀行は、勘定系システムの既存機能をリバースエンジニアリングにより可視化し、テスト計画書の作成及びテスト実施を行う計画である。

筆者はこのプロジェクトにおける業務コンサルタントとして参加した。筆者の役割は、現行業務プロセスを明らかにする方法論の開発とその方法論による業務フローの可視化である。

プロジェクトで銀行へのヒアリングや現行システムのドキュメント有無の確認を行った。その結果、パッケージ基本機能に関するドキュメントはパッケージベンダーより提供されている。しかし、カスタマイズ機能については、ドキュメントがあまり整備されていないことが判明した。銀行内部に機能全体を理解している有識者がほとんどいないため、現行資料をもとにシステムのバージョンアップのテスト計画書を作成することは難しい。そこで、リバースエンジニアリングにより、システムの機能と利用されている業務を可視化し、システムのテスト計画書を作成することにした。

今回、パッケージを利用した勘定系システムでは、リバースエンジニアリングツールを利用しただけでは、現行システムの調査は十分にできない。そこで、ツールと業務要件から業務処理フローを解析する新しい仕組みを開発したので、その方法を提案したい。この方法により、パッケージシステムとカスタマイズ機能について、テスト計画の作成が可能となり、システムのバージョンアップが可能となる。

２．適用対象のシステムの概要

本事例のシステムは銀行の勘定系システムである。この勘定系システムが、銀行のシステム全体の中でどのような位置づけであるかを示す。

２．１銀行システムの概要

銀行のシステムとは、預金や融資、送金等を行う勘定系システムと、銀行間の取引を行う市場系取引システム、銀行全体のバランスシートや損益を管理する会計レポートシステム、国内の銀行へ送金する国内送金システム、外国銀行へ送金する外国送金システム、銀行内部の取引や収益を分析する情報系システム及びそれらのシステムを接続するインターフェースシステムから構成されている。

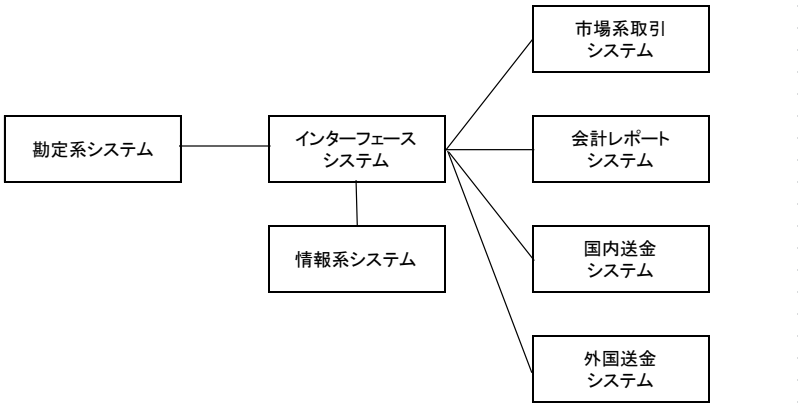


図.1 システム全体の概要

２．２勘定系システムの概要

勘定系システムのパッケージは、一般的に顧客、流動性預金、定期性預金、融資、送金、貿易業務を基本機能として具備している。本事例では、パッケージシステムを銀行が導入して、基本的な業務はパッケージ基本機能で実現している。銀行固有の要望や商品・サービスの対応が必要な機能については、画面、帳票、手数料機能、国内・外国送金機能、インターフェース機能をカスタマイズにより機能追加している。つまり、勘定系システムは、パッケージ機能と追加機能から構成されている。

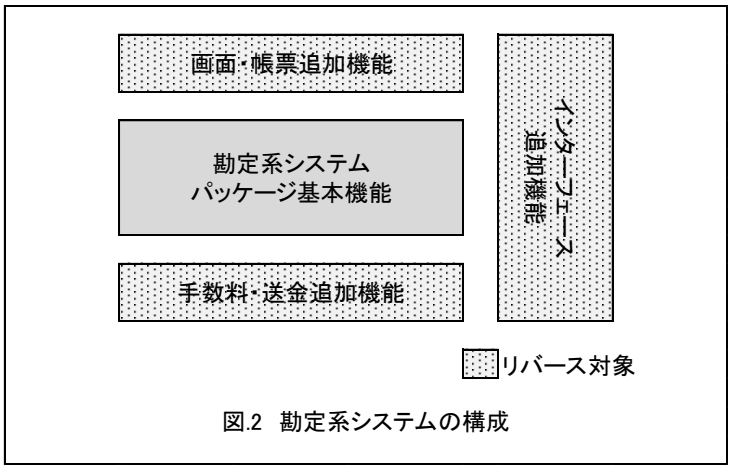


図.2 勘定系システムの構成

２．３リバースエンジニアリングの対象機能

勘定系システムパッケージの画面機能、帳票機能、手数料機能、送金機能及びインターフェース機能のうち、カスタマイズ機能をリバースエンジニアリングの対象とした。パッケージ基本機能については、製品著作権の問題より、ツールによるリバースエンジニアリングには適さないため、対象から外した。

３．リバースエンジニアリングプロセス

本事例で適用したリバースエンジニアリングプロセスの手順を示す。一般的なリバースエンジニアリングの方法[1] [2]を採用している。図.3 リバースエンジニアリング実施手順の①から⑤までを「３．１作成ドキュメントの作成」から「３．５リバースエンジニアリング実施」に処理内容を簡記する。なお、多くのプロジェクトの場合、リバース処理プロセスをツールにより実行するが、本事例でもこのプロセスをツールを用いて実施した。

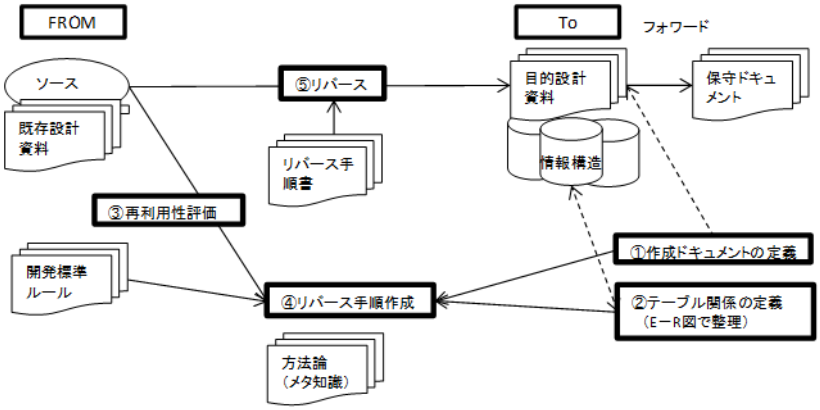


図.3 一般的なリバースエンジニアリングの実施手順

３．１作成ドキュメントの定義

このプロセスでは、どのようなドキュメントがシステムを可視化する上で必要となるのかを定義する。

３．１．１モジュール仕様書

各モジュールの処理概要を記載する。入出力のデータ、起動条件やプロセスの概要を記載する。

表.1 モジュール仕様書 ひな形

モジュール名	
使用言語	
入力データ	
出力データ	
起動条件	
プロセス概要	

[1] リバースエンジニアリング手法の一提案,稲田満,情報処理学会 49 回大会他に加筆

[2] インフォメーションエンジニアリング I ,James Martin,1989

３．１．２モジュール間制御遷移チャート

各モジュールの処理の流れを記述することにより、処理機能を表す。

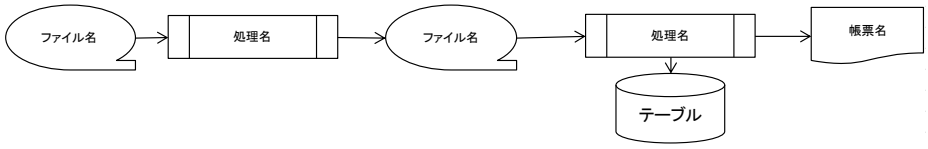


図.4 モジュール間制御遷移チャート　ひな形

３．１．３インターフェース仕様書

他システムとのインターフェースのファイル項目を明らかにすることにより、どのようなデータが授受されているかを明らかにする。

表 2 インターフェース仕様書　ひな形

No	項目名	属性	桁数
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

３．２テーブル関係図の定義

作成ドキュメントよりテーブル間の関係を定義する。

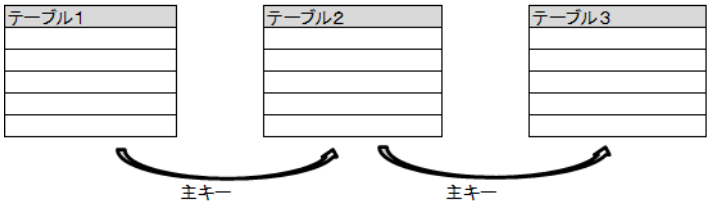


図 5 テーブル関係図　ひな形

３．３再利用性評価

既存の設計ドキュメントから作成ドキュメントに再利用可能なものを調査する。再利用可能なものは、継続して利用する。

３．４一般的なリバースエンジニアリングの手順作成

開発時に作成した標準化ルール等を用いて、作成ドキュメントの作成手順を作成する。有識者が対応するものと、非有識者が対応するものを分けて作成する。ツールを使って資料を作成する手順と人間が手作業で作成する手順をまとめる。

３．５リバースエンジニアリング実施

既存のドキュメントとリバースエンジニアリングツールを使用して、ソースコード解析を実施する。

４．リバースエンジニアリングの試行

全体のシステム機能について調査する前に、実際にシステムの一部機能を表現できるか、非有識者がその機能を理解できるかを検証する。追加機能として開発した国内送金機能をリバースエンジニアリングの対象として実施した。

４．１作成ドキュメントの定義

４．１．１モジュール仕様書

モジュールは小さい機能や小さい機能が組み合わさった処理単位で作成されている場合がある。そのため、最小限の機能でモジュール仕様書を作成した場合、数が多すぎてシステム全体としての意味が分からなくなる。一方、主要業務処理単位にした場合、システム全体は理解しやすくなるが、個別の機能が表現しにくくなる。

そこで、国内送金業務をカスタマイズ機能の資料ではなく、パッケージの基本機能のカテゴリーに沿ってモジュール仕様書を作成することとした。国内送金機能のモジュール仕様書作成単位を次のように分類した。

表.3 モジュール仕様書作成単位

業務名	パッケージ機能の機能分類	今回対象
国内送金	共通	
	送金仕向(送金送信)	
	送金被仕向(送金受信)	○
	手形仕向	
	手形被仕向	

４．１．２モジュール間制御遷移チャート

ツールにより、モジュール間のデータ遷移を作成する。アクセステーブルを明らかにして、データの流れとプロセス上の処理をフローチャーに作成する。

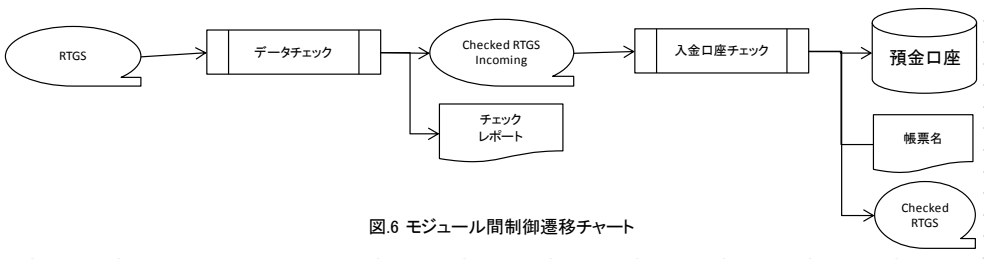


図.6 モジュール間制御遷移チャート

４．１．３インターフェース仕様書

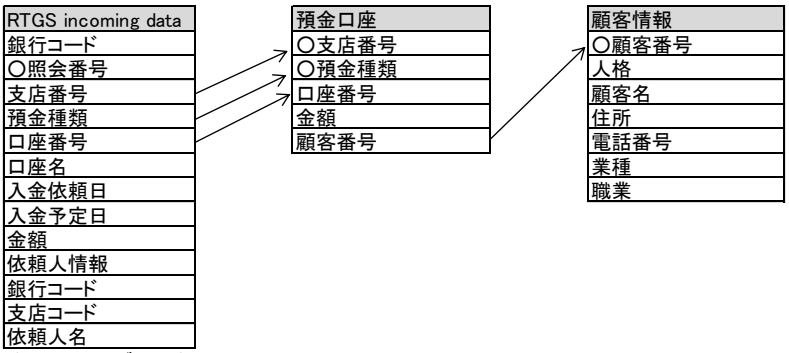
インターフェースの仕様書は、ソースコード上に定義されているデータレイアウトよりインターフェースフォーマットを作成する。

表.4 インターフェース仕様書

No	項目名	属性	桁数
1	銀行コード	N	4
2	支店番号	N	3
3	預金種類	N	2
4	口座番号	N	7
5	口座名	N	40
6	入金依頼日	D	8
7	入金予定日	D	8
8	金額	N	13
9	依頼人情報		–
10	銀行コード	N	4
11	支店コード	N	3
12	依頼人名	C	40

４．２テーブル関係図の作成

モジュールがアクセスするテーブルについて、テーブル間の関連を定義する。



注）〇はテーブルの主キー

図.7 テーブル関係図

４．３再利用性定義

パッケージシステムの基本機能に関するドキュメントは、業務の基本機能を理解するための資料として利用する。また、バージョンアップの標準機能とカスタマイズ機能の連携処理に関する機能解析に利用する。

パッケージ追加機能については、現行システムに関する資料が存在しないため、現行システム資料の再利用はできなかった。

４．４今回のリバースエンジニアリングの手順作成

当初導入時のカスタマイズ開発に関する標準化ルールが整備されていない。そのため、プログラムの構成やプログラムのネーミングルールをいくつか確認することにより、仮の標準化ルールを作成する。

既存の設計書も存在しないため、ユーザーからの業務ヒアリングと一般的な業務経験より、仮の業務処理フローを作成する。

機能単位のモジュールのソースを入力として、リバースエンジニアリングツールを実行する。ツールの実行により、モジュール仕様書、モジュール間制御遷移チャート、インターフェース仕様書、テーブル関係図を作成する。

各作成ドキュメントより、システムテストや保守業務に利用する業務処理フローとシステムテスト計画書を作成する。

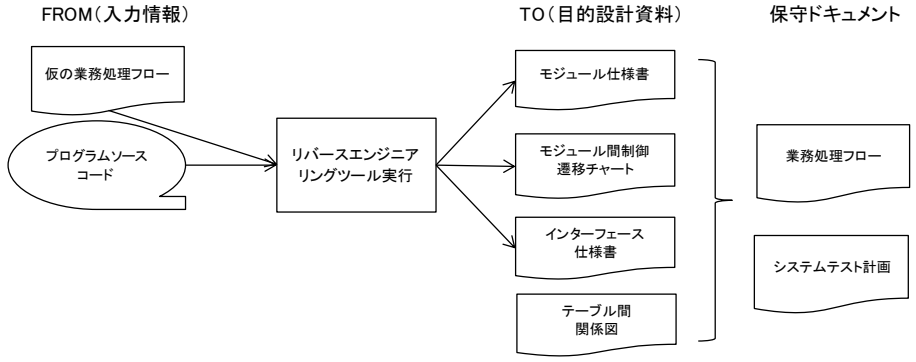


図.8 今回のリバースエンジニアリングの実施手順

４．５リバースエンジニアリングの実施

定義したモジュール仕様書の作成単位ごとに、実施手順を実行した。この単位で処理を実施すると、入出力のデータや起動条件、プロセスの概要を解読できた。その情報により、モジュール仕様書、モジュール間制御遷移チャート、インターフェース仕様書、テーブル間の関係図を一部の処理を除き、一旦作成することができた。

表.5 モジュール仕様書

モジュール名	SKIN010
使用言語	RPG
入力データ	RTGS incoming data
出力データ	RTGS checked incoming data
起動条件	新しいファイルのフォルダ受信時
プロセス概要	入力データの属性チェック

RTGS:Real-Time Gross Settlement

５．リバースエンジニアリングを適用して判明した問題

５．１カスタマイズ機能の解析

送金受信機能のモジュールをツールで解析を実行すると、送金受信データのチェック及び、入金口座のチェックという処理の流れは可視化することはできた。しかし、送金機能の後半処理である顧客預金口座への入金処理は、パッケージの基本機能により実現しているため、後続機能を可視化することはできなかった。モジュール間制御連携チャートはリエンジニアリングツールによりある程度理解可能な資料の作成は可能である。そのため、カスタマイズした機能がすべて追加機能であれば、モジュール間制御連携チャートを理解することにより、処理機能を理解することが可能となる。

しかし、追加送金機能のソースコードを解析しただけでは、業務機能を表すことができないことが分かった。なぜならば、パッケージに追加機能を開発して、新しい業務要件を実現するためには、追加開発機能だけではなく、パッケージの基本機能を理解しないとテストの計画を立てることができないことが判明した。

これは、カスタマイズ機能は、機能を追加して新しい業務処理を構築していると想定していたためである。つまり、個別業務を追加する場合、個別業務単位で機能を開発してると想定していたためである。この点が、当初の想定と相違していた。

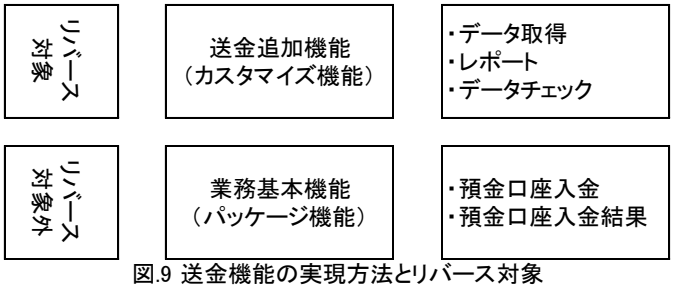


図.9 送金機能の実現方法とリバース対象

５．２業務フローの解析

リバースエンジニアリング手順は、既存システムに設計書や利用マニュアルがあることが前提となっている。しかし、今回の事例では、現行システムの設計資料が存在しない。また、ユーザーのマニュアルも十分にメンテナンスされていない。そのため、一部の設計資料を再作成しても、正確な業務処理フローを作成することができない。パッケージの基本機能については、ソースコードを用いてリバースエンジニアリングを実施するわけではない。つまり、一部の処理フローは判明するが、処理機能全体を解明することはできない。

５．３リバースエンジニアリングの評価

現行カスタマイズ機能のソースコート、仮の業務処理フロー及び仮の標準化ルールを入力としたリバースエンジニアリングの実施では、十分に現行処理機能を解析することはできない。なぜならば、現行のカスタマイズ機能は、パッケージの基本機能とカスタマイズした追加機能により十減されているからである。

プログラムソースを中心としたリバースエンジニアリングだけでは、基本機能の解析はできない。従って、正確な業務処理フローやシステムテスト計画書を作成することは難しいのである。

６．改良したリバースエンジニアリング手順実施

ツールを利用した方法では、システム機能の全体を可視化することはできない。そのため、モジュールから業務機能を解析するのではなく、業務フローから業務プロセスを再定義し、それをもとに基本機能とカスタマイズ機能を可視化し、テスト範囲を定義する。新しいリバースエンジニアリング実施プロセスを図.10 に示す。

図.10 リバースエンジニアリング実施プロセス

一般的な開発プロジェクトでの要件定義では、業務処理フロー作成、システム化機能抽出、システム機能開発といったプロセスとなる。ここまでのプロセスは通常の要件定義と同様である。

ただし、このプロセスだけでは、どの機能が基本機能であるのか、また、どのプロセスが追加機能であるのかを分類することができない。そこで、パッケージシステムのリバースエンジニアリングを実施するプロセスとして、基本機能と追加機能を分類するプロセスを追加した。そして、分類した機能ごとに機能解析を実施し、最後に業務フロー上の機能との整合性チェックを行う手順とした。

つまり、パッケージの基本機能については、パッケージのマニュアル及びパラメータの定義より、処理機能を理解する。パッケージのカスタマイズ機能については、ソースコードより、処理機能を理解する。これらの処理機能と業務処理フローとの整合性をチェックすることにより、正確な業務処理フローを完成させる手順とした。

この新リバースエンジニアリング実施手順により、解析対象とした送金受信機能を再解析した。

６．１業務フロー作成

各業務処理単位に業務のフローを作成する。業務ユーザーより、業務の流れをヒアリングし、業務プロセスフローを作成する。ユーザーが送金受信処理を行う上で行っているプロセスを抽出して、プロセスを連結することにより作成する。

送金受信データを外部システムから勘定系システムへ取り込む。勘定系システムは入力されたデータの形式チェックを行う。次に形式チェックを完了したデータをもとに、送金データの入金顧客口座が存在するかのチェックを行う。その後、入金顧客口座が存在したデータは顧客口座に自動的に入金される。同時に入金結果レポートが作成される。業務担当者は、入金通知対象の顧客について、入金通知書を作成する。

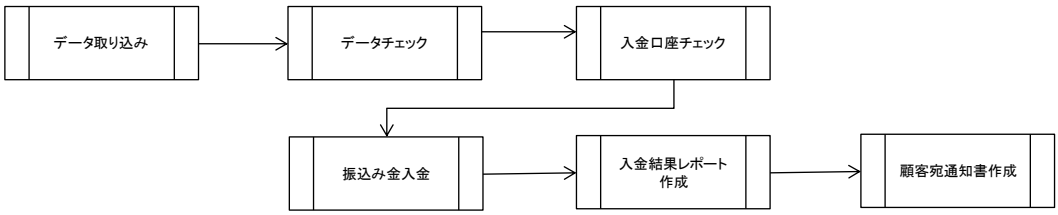


図.11 業務プロセスフロー

６．２システム化機能抽出

業務フローの処理プロセスの中から、システムで実現している処理プロセスを抽出する。取り込み処理と顧客宛通知書作成処理はユーザーがマニュアルで処理を行っているものである。システム処理のみを抽出したものが図.12 である。

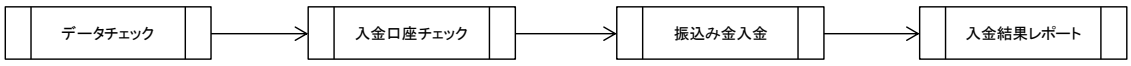


図.12 システム化業務プロセスフロー

６．３基本機能と追加機能の分類

システム化業務プロセスについて、各処理がパッケージの基本機能で実現されているか、あるいは、カスタマイズ機能で実現されているかを分類する。各処理に紐づくカスタマイズモジュールがあるかどうかを確認することにより、カスタマイズした追加した機能であると分類する。カスタマイズモジュールがない処理機能については、基本機能として分類する。

この分類の精度を上げるため、パッケージのマニュアルの記載機能を確認し、当該処理が基本機能であることを確認する。

表.6 基本機能と追加機能分類表

システム化プロセス	データチェック	入金口座チェック	振込み金入金	入金結果レポート
基本機能/追加機能	追加機能	追加機能	基本機能	基本機能

６．４基本機能解析

パッケージに基本機能として備わっている機能について、業務フローの前後処理とパッケージのマニュアル記載事項より、機能を解析する。

６．４．１振込み金入金機能

預金口座への振込み入金機能は、預金入金でも使う機能である。振込み金入金は預金の入金機能と同じ機能を使うため、基本機能である。

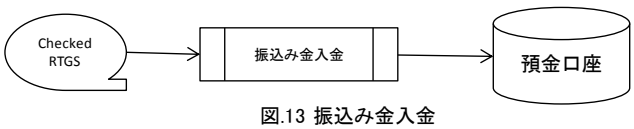


図.13 振込み金入金

６．４．２入金結果レポート作成

預金の口座振替処理や融資実行処理でも預金口座への入金確認処理を作成している。よって、入金結果レポート作成処理も基本機能である。

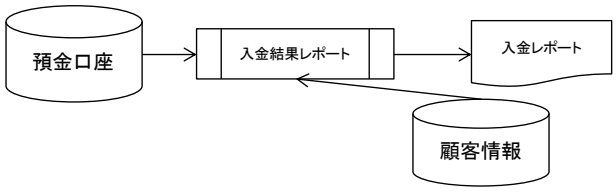


図.14 入金結果レポート

６．５追加機能解析

リバースエンジニアリングツールで解析した結果を流用する。

６．５．１データチェック機能

入金処理データについて、店番号、口座番号の設定有無、属性チェックを実施している。

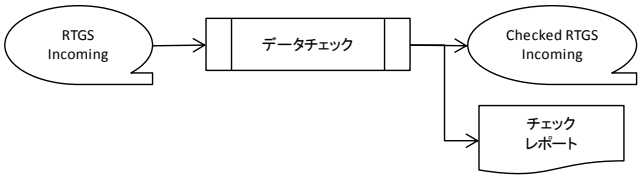


図.15 データチェック

６．５．２入金口座チェック機能

入金処理データについて、店番号、口座番号の銀行内存在有無チェックを実施している。

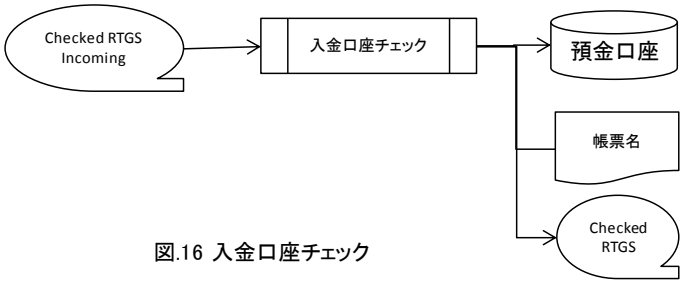


図.16 入金口座チェック

６．６業務フロー整合性チェック

各業務処理機能をシステム化業務プロセスに合わせて、一連の業務フローを作成する。各業務処理の流れに不整合がないことを確認する。この業務フローにより、送金受信業務について、パッケージの基本機能及び追加機能を合わせて、業務内容を可視化する。

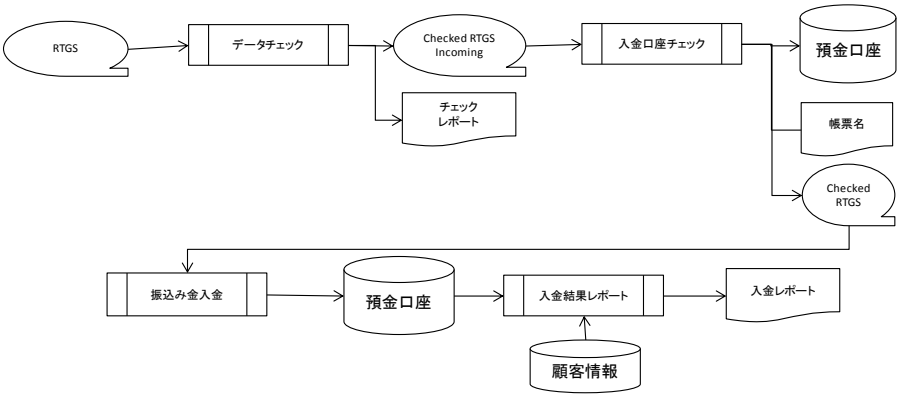


図.17 送金受信業務フロー

７．おわりに

今回の事例は、システムのバージョンアップであるため、大きな業務処理の変更があるわけではない。そのため、ユーザー部門の協力を得て大規模な業務要件定義を実施することができない状況であった。

この条件において、銀行の勘定系パッケージシステムの基本機能とカスタマイズ機能について、業務要件を効率的に明らかにする方法の開発が必要であった。その方法により、業務要件を可視化することが目的であった。

事例では、一部の業務処理機能について、開発したリバースエンジニアリング実施手順を実施した。まず、仮の業務処理フローを作成し、その次に、各業務処理がパッケージ基本機能か、追加機能であるかを識別して解析を実施した。この手順により、ユーザーの力をあまり借りずに業務要件を可視化することが可能となった。このことは、当初の目的である方法論の開発と業務要件の可視化を達成できたといえる。

現実の業務では、カスタマイズ機能の設計資料が銀行に存在しない場合は多く存在する。新システムをリプレースするために、リバースエンジニアリングによる要件定義を行うプロジェクトは、少なからずある事例と考えている。今回は銀行システムの事例であったが、他の業界の業務処理についても方法論は適用できるのではないかと考える。

今後、すべての業務要件を可視化した後、再度リバースエンジニアリングを実施する必要がないようにする必要がある。そのために、資料の最新状態の保持方法や業務要件の可視化についての方法論を研究していきたい。