オープン勘定系システムにおけるバッチ処理を支える仕組み

Mechanism that Batch Processing is Supported in Banking System based on Windows' Open Architecture

宮 澤 貴 之，鈴 木 俊 也，福 丸 昌 宏

要　約　BankVision では，バッチ処理環境にバッチ専用データベースを保有しており，大多数のバッチ処理はこのデータベースから必要な情報を抽出して処理を行っている．このバッチ専用データベースには，2 時間程度で作成を終了させるという性能的要件や，24 時間稼働を前提としているオンラインデータベースの日付繰越の時点で未確定の勘定を反映して作成しなければならないという業務要件があった．

バッチ専用データベースを基に処理を行う部分，特に帳票作成処理部分においては，オープンプロダクトを採用し，開発の効率化，保守性向上によるコスト削減を目指すとともに，時流となっているペーパレス化を実現する一方，旧システムで実現されていながらオープンプロダクトで対応しきれていない部分については，新たに仕組みを作ることによって対応した．

Abstract　BankVision, Banking System based on Windows' open architecture concept, has a unique database for batch processing. Almost all the batch processing's extract necessary information from this database.

Regarding to the generation of this database, we had to meet the requirement that the generation should be completed within two hours, reflecting unfixed accounts at the timing of date carryover included in the online database, continuously operated for 24 hours.

Designing the batch processes using this database, especially report writing process, we adopted open products to improve the cost, the efficiency and maintainability, achieving trendy paperless operation. At the same time, we developed a new mechanism on top of the open products to compensate insufficient functions, already achieved in the legacy system.

1. は　じ　め　に

オープン IT 技術が一般に広く浸透し，企業の業務活動でも利用されてきている．勘定系システムにおいても，システム技術者以外の一般行員が IT 技術を駆使して，その効果を享受し，効率の良い業務を行いたいという要望が強くなってくるのも当然である．

オープン勘定系システムである BankVision は，フルバンキングシステムであり，大量の還元帳票が存在し，それを処理するバッチ処理の本数も大量である．オープン環境上で，バッチ処理を開発するにあたっては，電子帳票システムを採用しペーパレス化を進めることで，帳票管理面，セキュリティ面の向上を図ることと，大量の帳票作成処理についてもオープンプロダクトを積極的に活用することで，保守性の向上，ひいてはコスト削減の効果を引き出すことが要件であった．

さらに，BankVision は汎用機の勘定系システムの後継システムと位置づけており，旧システムで実現されている機能は，オープン環境にて同じ機能を実現しなければならないというのも必須の要件であった．

また，旧システムにおけるバッチ処理の保守性を改善するため，新システムではバッチ専用データベースを保有している．このバッチ専用データベースを保有することとなった背景については，ユニシス技報通巻 77 号の論文「オープン勘定系システム開発の事例紹介」[1]を参照して頂きたいが，24 時間 365 日稼働する「止まらない」オンラインデータベースから，バッチ処理に必要な静的な状態のバッチ専用データベースを作成するために，大規模データを効率良く処理することで時間的な要件を満足することに加えて，作成処理のタイミングを考慮することでテーブルの特性固有の問題を解決する必要があった．

本稿では，これら BankVision のバッチ処理を支える仕組みと，考慮した点について記述する．

2. BankVision におけるバッチ処理の概要と要件

勘定系システムにおけるバッチ処理は，リアルタイムのデータ，つまりオンラインデータベースを基にする処理と，前日末確定のデータまたは前月末確定のデータといった静止したデータを基にする処理の二つに大別できる．いずれの場合においても，入力となるデータベースを作成し，それを参照して各種還元帳票，他システムへ連携するファイルといった最終出力を作成する，という流れになる（図 1 参照）．

BankVision では，静止したデータを入力とするバッチ処理環境を確保することを目的として，従来の勘定系システムとは異なり，オンライン環境から分離した「基幹系 DWH」と呼ぶバッチ専用環境を構築した．本章では，バッチ処理の大部分を占める基幹系 DWH における処理において，①入力となるバッチ専用データベースの作成，②作成したデータベース等を基に，最終出力としての各種還元帳票を作成する部分について記述する．特にこれらをオープン勘定系として構築していくにあたり，汎用機上に構築された従来の勘定系システムで抱えていた課題に対する解決策，従来のシステムから継承することが求められている機能の実現方法について記述する．

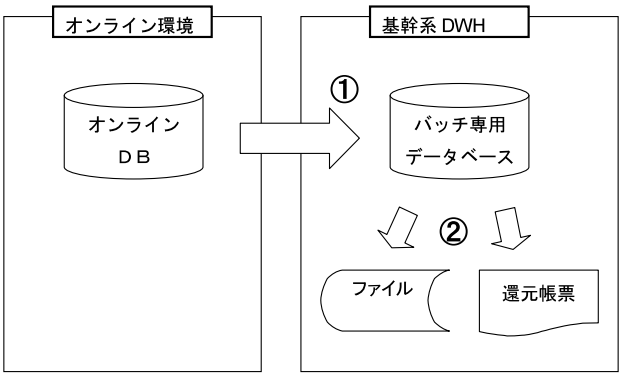

図 1　バッチ処理の流れ

2.1　バッチ専用データベース作成処理の要件

勘定系のバッチ処理には，バッチ専用データベースを入力とする数多くの後続処理が存在している．バッチ専用データベースを作成するのに時間がかかると，バッチ処理全体の処理が遅れてしまい，翌朝の営業店への帳票の配布遅延で，営業店業務に支障をきたす．このため，オンライン環境のデータベースからバッチ環境のデータベースを作成し終わるまで「2 時間」という目標が設定され，大規模のデータを効率良く処理しなければならないという性能要件となった．

BankVision はフルバンキングのシステムであるがゆえに，大規模データを短時間で処理できなくてはならず，この課題に対しては，データベーステーブルに保有されるレコードのデータ特性に応じて，処理方法・稼働環境を選択することで，解決を図っている．さらにオンライン取引から出力される取引ログ（以降，「業務トランザクションデータ」と記す）については，最大で日に 300 万以上のデータ件数を処理することを想定する必要があったため，夜間の一括処理とせず，日中からオンラインデータベースにて更新されたレコードを随時バッチ専用データベースに反映しておくことにより，大量一括処理とならないよう考慮している．

またバッチ専用データベースは，オンラインデータベースのオンライン日付繰越時点での断面から作成しており，オンライン日付繰越以降に確定する前日勘定分のデータが含まれておらず，バッチ専用の静的な状態のデータベースとしては不完全なものとなる．この課題に対しては，前日勘定の処理が終了した時点でバッチ専用データベースへ遅れて更新する形で実現している．

これらのバッチ専用データベース作成処理段階での要件をどのように解決していったかを 3 章にて記述する．

2.2　最終出力作成処理の要件

バッチ処理の最終出力としての各種還元帳票や他システムへの配布データは，旧システムでも存在しており，当然オープン環境での新システムでも同じ機能を継承して実現することが必須であった．それに加えて，開発・保守の効率化，それによるコスト削減，また時流である ISO14000 などの環境配慮に伴う紙の削減や，セキュリティ面・管理面でのレベルアップが要件として求められていた．

これらの要件に対する解決策として，電子帳票を採用し，大部分の帳票についてペーパレス化を実現した．また帳票を印書出力する仕組みにもオープンプロダクトを採用することで，プログラミングレス化を図り，これにより開発・保守の効率化，ひいてはコスト削減を図っている．

しかし全ての機能の中には，オープンプロダクトで実現するためには機能が不足している，またはソフトウエア間の連携が簡単にできないというものもあった．この部分については，独自に仕組みを作りこむことで対応した．

この最終出力作成処理の中で，特徴的であった帳票印書システムについて，詳細を 4 章にて記述する．

3. バッチ専用テーブル

3.1 バッチ専用テーブルの必要性

オンライン処理とバッチ処理では，表1のように，処理を最適化するために求められるテーブルの特性が異なる．しかしながら従来の勘定系システムにおいては，H/W の制約等により同じオンラインデータベースを入力として処理が行われることが多く，そのことが理由でオンライン処理，バッチ処理の修正が難解になり保守性が低下するという弊害が発生することがあった．

表1　オンライン処理とバッチ処理で求められるテーブルの特性

	オンライン処理	バッチ処理
望ましいテーブルレイアウト	業務単位の個々のオンライン処理に対して最適化されている	各種業務を横断的に、一括して扱うことが容易になるようになっている
データの基準	24時間のオンラインサービス提供を前提としたデータベース	・前日の勘定終了時点 ・前月末の勘定終了時点

BankVision においては，オンライン処理に影響を与えることなく，バッチ処理をシンプルなものにし，従来のシステムで顕在化した保守性の低下を防ぐことを目的として，オンラインデータベースとは別にバッチ処理専用のテーブルを作成した．オンラインデータベースと分離した環境を作成し，バッチ処理に必要なデータを抽出することにより，次のようにバッチ処理を最適化することが可能となった．

① 静止したデータベースの確保

24時間稼働し続けることを前提としたオンラインデータベースと分離することにより，バッチ処理で必要としている前日末確定データ，前月末確定データを確保することが可能となり，データをシンプルに扱うことが可能になった．

② オンライン処理，またはバッチ処理の変更による影響の極小化

オンライン処理，バッチ処理それぞれが専用データベースを入力とすることで，オンラインデータベースの修正が，バッチ処理用のデータベースに関係がない限り影響を受けず，保守性の低下を避けることが可能になった．

バッチ専用テーブルの構成と，バッチ専用テーブル作成処理において考慮した点について，次節以降で記述する．

3.2 バッチ専用テーブルの構成

3.2.1 オンライン環境とバッチ専用環境の関係

BankVision では，オンライン環境とは別に，バッチ処理専用の環境である「基幹系 DWH」を構築した．「基幹系 DWH」に保有するデータベースには次のものがあり，バッチ処理の多くは「共用明細 DB」と呼ばれるバッチ処理専用テーブルを使用して処理を行っている．

① 静的 DB　　：オンラインデータベースのオンライン日付繰越処理時点の確定データ

② 共用明細 DB　：静的 DB を入力として作成した各種バッチ処理専用の入力データ

図2に，オンライン環境と「基幹系 DWH」の静的 DB と共用明細 DB の関係について示す．

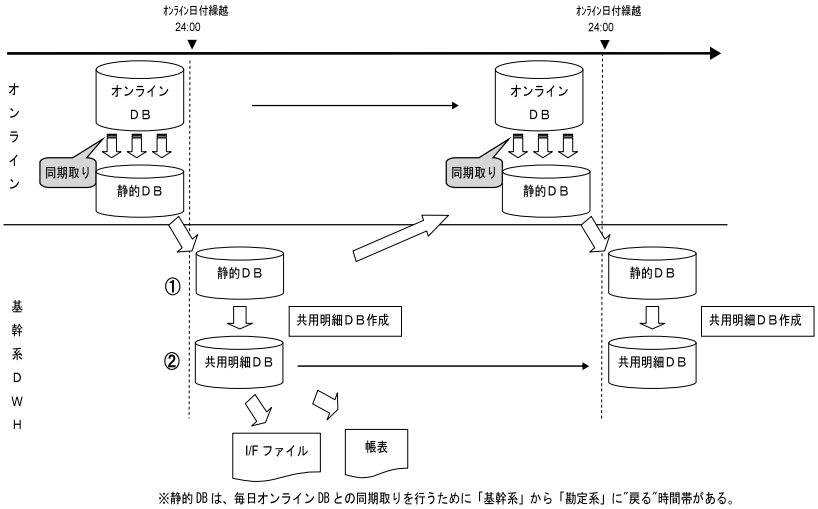

図 2　オンライン DB と静的 DB，共用明細 DB の関係

3. 2. 2　共用明細 DB の概要

共用明細 DB は，バッチ処理に適した入力データとするために設計したものであり，次のようなデータの種類，属性を保有している．

1)　データの種類

日次処理，月次処理で使用する顧客情報，口座情報，業務トランザクションデータ等の情報を保有する．口座情報については，バッチ処理での使用方法を考慮して流動性，固定性，融資のカテゴリで複数科目を統合して情報を保有する．

2)　データの属性

顧客情報，口座情報については，オンラインデータベースの保有項目に加え，①バッチ処理で必要な項目，②バッチ処理での使用頻度が高い加工項目を必要に応じて保有する．

①　バッチ処理で必要な項目

オンラインデータベースでは，オンライン処理の内容，テーブルの正規化の観点から，例えば「取引先区分」,「業種コード」といったような顧客情報を口座情報のテーブルには保有していない．一方バッチ専用テーブルでは，統計帳票作成時などバッチ処理での集計キーとなる「取引先区分」,「業種コード」を口座情報のテーブルにあらかじめ保有しておくことにより，処理構造をシンプルなものにすることができ，処理効率の向上につながる．

②　バッチ処理での使用頻度が高い加工項目

オンライン処理では不要のため保有しないが，バッチ処理では使用頻度が高く，テーブルに保有しておくことが望ましい項目がある．例えば各種契約の有無，口座保有の有無などをバッチ専用テーブルに保有することにより，各々の処理で複雑な編集処理を実装する必要が無くなる．これを参照することで，一括処理を行いやすくなり，SQL の有意性を活かした処理ができるようになる．

3.3　バッチ専用テーブル作成の要件と対応

3.2.2 項で述べたようなバッチ処理の特性からくる要件に加え，銀行業務で求められるフルバンキングシステムとして，1）大規模データに対する処理効率，2）24 時間運用に伴う日付繰越時点での未確定勘定の考慮のような性能要件を満たすために，データの量，編集内容，作成タイミングによりバッチ専用テーブルの作成方法に個別の配慮をする必要があった．

1）　大規模データに対する処理効率

翌朝に営業店に帳票を還元するため，当日業務の開始をするため，といった理由から業後に行う日次バッチには時限性がある．その時限性を受けて，日次バッチの入力となる共用明細 DB 作成に許される時間は 2 時間以内とされていた．実際の業務で扱うデータ量，またはそれを超える規模で行った実機検証により，すべての処理を従来のシステム同様 COBOL プログラムで一様に扱うことでは，この要件のクリアは非常に困難であることが明らかになっていた．そのため，データ量，データの編集内容，作成が可能になるタイミングに応じて，各々のテーブルに適した処理方法（表 2）を採用することにより，処理効率の向上を図る必要があった．

①　SQL の特性を活かした一括処理

共用明細 DB の多くが静的 DB からの一括ロード処理であり，そのような処理には SQL の特性を活かして StoredProcedure（SP）で一括処理を行い，DB サーバのリソースを十分に活用することが有効である．

共用明細 DB 作成処理においては SP での一括処理を基本とし，一括処理が困難な場合に COBOL で業務ロジックを実装した．このことは，“バッチ処理での使用頻度が高い項目”を編集する処理についても当てはまる．SQL での処理を分析して，各テーブルのインデックスの設計，編集処理時のコストを下げるようなロジックの改良などの SQL チューニングを実機検証の中で繰り返し，処理効率を時限性に対する要求に近づけていった．

②　複雑な業務ロジックでの編集処理

SQL での一括処理が困難な編集処理については，COBOL での編集を行った．COBOL がメインとなる処理においても処理効率向上のため，DB サーバの I/O 効率を高めるようなアプリケーション構造とし，参照するテーブルのインデックス設計を，必要であれば店，科目等の単位での分割処理が行えるようにしていった．

③　業務トランザクションデータの随時処理

業務トランザクションデータのように，更新方法が追加のみのデータの場合，オンライン日付繰越処理時点での確定を待たずにデータ発生の都度ロードが可能である．オンラインデータベースを入力としてデータ発生の都度（随時）処理を行う仕組みを作成し，オンライン日付繰越処理時点で大半のデータが静的 DB にロード済みの状態とすることで，速やかに処理が終了するようにした．

表 2　データの特性に合わせた処理方法の分類

適した処理方法	データ量	編集内容	データの作成タイミング
SQLでの一括処理	非常に多い	・同内容のロード ・一括処理が可能な編集 （条件Aであったら1をセットする、等）	データ確定後
随時処理	非常に多い	・同内容のロード	随時 （データ発生都度）
COBOLでの処理	―	・明細単位での編集	データ確定後

2）　24 時間運用に伴う日付繰越時点での未確定勘定の考慮

オンライン日付繰越処理時点での確定データには，資金決済など，オンライン日付繰越処理以降に確定する前日勘定分のデータが含まれていない．静止することがない 24 時間運用を考えると，前日勘定確定時のオンラインデータには既に当日勘定分のデータが含まれていることから，前日勘定が未確定のテーブルについては，確定したデータをオンラインデータベースから別途抽出して静的 DB のデータと入れ替えてから共用明細 DB を作成する仕組みとした．

4.　帳票印書システム

BankVision では複数のオープンプロダクトを採用して帳票システムを構築している．本章では，これらを採用して実現したこと及び，汎用機上で実現していた印書運用をオープン環境でどのように継承したかについて記述する．

4. 1　BankVision における帳票の概要と電子帳票

帳票とは，広義の意味では「様々な情報を一目で把握できるようにまとめたもの」を指すことが多いが，本稿ではこれに加えて「特定のレイアウトの上に，データを出力した伝票等の総称」と定義する．

BankVision では，帳票をその使用目的に応じて

（1）　顧客へ送付する DM（ダイレクトメール）

（2）　金融庁や自治体などの外部へ提出する帳票（以下，外部報告帳票）

（3）　銀行内部で業務上使用する帳票（以下，還元帳票）

の三つに分類している．DM には，主に圧着式の葉書タイプと封書タイプとの 2 種類がある．

各帳票とも AP（アプリケーション）にてデータファイルとして作成された後に，帳票を使用するユーザへ送付され，各々の用途に応じて使用される．

なお BankVision における帳票の中には，都度ユーザのオペレーションにより配信される帳票と，バッチ処理にて決められた時間までに作成される帳票とが存在するが，帳票作成の契機が異なるだけであり帳票としての機能に大差がないため，本稿では同じものとして扱う．

また，BankVision で採用した出力媒体を分類すると，主に大量印刷用プリンタ，インパクトプリンタ，電子帳票ツールの三つに分けることができる（図 3）．電子帳票は旧来のシステム仕様に，新たに追加されたものである．

帳票は元来，紙に印字されたものを使用していたが，使用目的に関わらず日々数多くの帳票が作成されるため，その送付や管理には多くの労力を必要とした．加えて，蓄積された帳票探索の手間，閲覧管理の不徹底や印刷物の紛失・外部漏洩によるセキュリティ面での不安，高い

印刷コスト，ISO14000への対応に伴う紙削減の圧力など様々な課題を抱えていた．

BankVisionでは，これらの課題の解決手段として電子帳票システムを提供しており，それまで紙に印字されていた帳票の電子化（ペーパレス化）を実現している．これにより，物理的に帳票を紙で作成せずとも，各営業店・本部の端末から必要な時にアクセスして閲覧することができる．その上，細かな閲覧権限の設定ができるようにしているため，権限のない者が閲覧することは不可能である．

ただし上述のDMや外部報告帳票のように，特定用紙へ印字して送付しなければならないものが存在するため，従来のように大量印刷用プリンタ，インパクトプリンタも備えている．

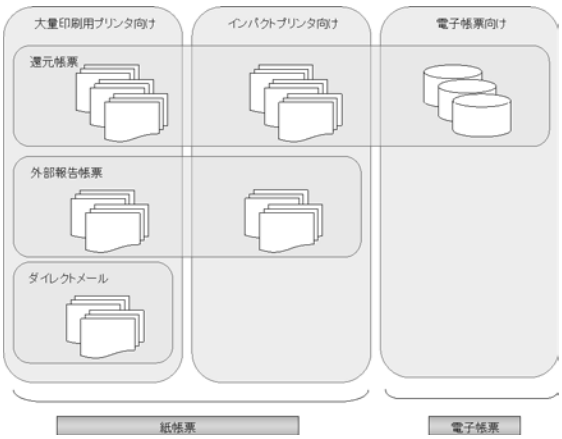

図3　帳票の分類

4.2　帳票作成とオープンプロダクトによるプログラミングレス化

各帳票を使用（閲覧）できるようになるまでには，「帳票の作成」と「帳票の送付」の大きく2プロセスに分けることができるが，このプロセスをさらに細分化すると以下のようになる．

【帳票の作成】

（1）　帳票に表示されるデータをデータベースより抽出する[*1]

（2）　抽出されたデータを帳票形式に加工する

【帳票の送付】

（3）（紙帳票）加工された帳票データをプリンタへ送る

（電子帳票）電子帳票の管理サーバ（以下，電子帳票サーバ）へ送る

（4）（紙帳票）送付されたデータをプリンタで印字する

（電子帳票）（対応するプロセスなし）

（5）（紙帳票）印書された帳票をユーザへ送付する

（電子帳票）帳票を閲覧可能状態にする

このプロセスに，BankVisionのシステム構成図を合わせると図4のようになる．

BankVisionでは，開発スピードの向上および保守コスト削減が求められており，特に帳票分野では，前節で論じた帳票の電子化に加え，これらを利用したプログラミングレス化の推進が課題であったため，各プロセスに応じたオープンプロダクトを次のように採用した（図5）．

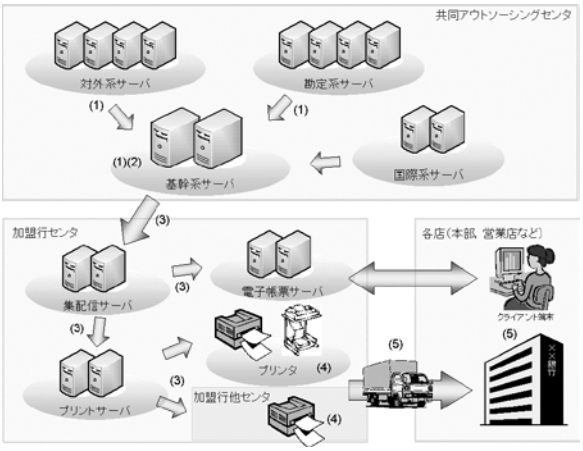

図 4　帳票作成の流れ

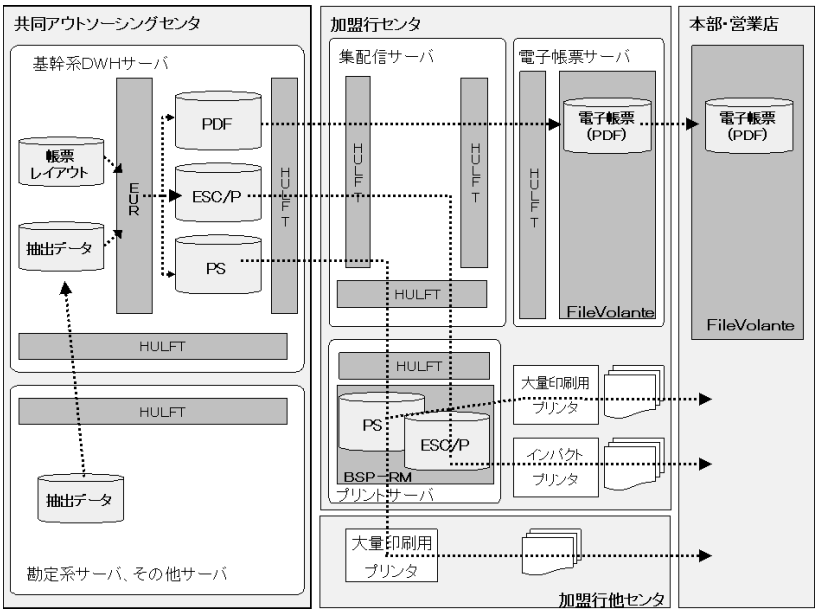

図 5　各プロセスとオープンプロダクト

【帳票の作成】

(1)　SSIS[*2]（ETL（Extract Transform Load）ツール）

データベースより指定された条件でデータを抽出し，ファイル出力するソフトウェアで，GUI（Graphical User Interface）形式で操作可能

(2)　EUR[*3]（作表ツール）

データファイルと帳票レイアウトをあわせて帳票として出力ができるソフトウエアで，帳票レイアウト（overlay）を GUI 形式で作成できる

【帳票の送付】

(3)　HULFT[*4]（ファイル転送ツール）

サーバ間のデータ連携を自動化するソフトウエア

（4） BSP-RM[*5]（印書ツール） ※紙帳票のみ
プリンタでの印書を制御・管理するソフトウエア
（5） FileVolante[*6]（電子帳票ツール） ※電子帳票のみ
電子帳票の管理・閲覧を行うソフトウエア

BankVision ではこれらのオープンプロダクトを採用したことにより，データ抽出から帳票の送付までの各部分で開発保守時の効率化を図っている．特に帳票固有の要件による対応が必要となる「帳票の作成」部分については，GUI 形式でレイアウトできるため，プログラムの経験有無にかかわらず違和感なく帳票を作成することが可能となっている．

4. 3 旧システムの機能継承に向けたオープンプロダクトの補完

オープンプロダクトを採用することで開発効率は向上できたが，旧システム仕様を継承した複雑な業務要件になればなるほど，どうしても機能不足が生じてしまう．また各プロダクト間のインタフェイスなども考慮する必要があり，これらの部分については別途仕掛けを開発する必要があった．本節では，その中でもより顕著な帳票の作成を中心に述べる．

4. 3. 1 処理フローのパターン化とファイルフォーマットの統一

帳票の作成から送付までの一連のプロセスの中で，各帳票固有の処理が必要となるのは，データベースからのデータ抽出および，次の帳票形式への加工に必要な帳票レイアウトの準備である．入力となるデータベースからどのデータをどのように採取して，どのように加工編集するかについては，各帳票固有の要件によるため明確なパターン化が難しく，編集方法も帳票の仕様にしたがって変更されていく可能性がある．

逆にデータ抽出と帳票レイアウト作成以降については，入力となるデータが異なるのみであり，印書方法や送付方法などはパターン化することができる．

以上の理由により，BankVision では帳票のデータ抽出は主として SSIS，プロダクト上実現できない処理については COBOL を使用して帳票ごとに開発することとし，以降のプロセスについては共通的な処理パターンを利用し，不足箇所については別途開発しつつ，帳票出力まで行うよう取り決めた．

パターンの種類としては，帳票の出力媒体・性質ごとに

1） DM（紙帳票）
2） 大量印刷用プリンタ（紙帳票）
3） インパクトプリンタ（紙帳票）
4） 電子帳票（電子帳票）

と大きく 4 種類に分け，帳票の送付先や印書するプリンタ，印書設定（用紙種類，両面指示等），電子帳票設定（帳票保存期間，参照権限等）などのその他付加情報については，帳票の管理マスタ（以下，最終出力一覧）を作成し，これをデータベース化して一元管理することとした．これにより，送付先や印書方法などの帳票明細に直接影響しない箇所については，管理マスタを一部更新するのみで対応することができ，保守性の向上を図っている．

データ抽出時に出力するファイルには，共通処理側でその内容を意識せずに処理できるよう，ファイル形式や出力方法に以下に例を示すような規約を設け，また処理方法や帳票レイアウトの作成にばらつきが出て保守性が落ちないようガイドを作成した．

【規約の一例】

- ファイルは項目に「”」を付加する CSV 形式
- ファイルの先頭行に，使用しなくても「基準日」「店番」項目を設ける
- ファイル名は一定の規則に則って作成する

なお，DM を除く大量印刷用プリンタで出力する帳票については，同一の入力データを使用して，電子帳票として作成することもできるため，それまで特定用紙に印刷して提出していた外部報告帳票が電子化されたとしても，プログラムの修正を経ずして，設定パラメータの変更のみで対応することができる．

パターン 1 の DM も大量印刷用プリンタを使用するため，基本的にはパターン 2 と変わりはないが，郵便発送の際に市内特別郵便（同一地区に 100 通以上送ると料金が割引される）の対象かどうか，郵便を使用せずに手渡しとするのかどうか，などを判断する必要があるため，パターンを分けている．

4. 3. 2　表紙による送付先の仕分け作業

印刷された帳票（紙帳票）は，運用員が手作業にて印書時の不備（文字かすれ，汚れなど）が発生していないかをチェックしつつ送付先ごとに仕分けしており，また印書されるべき帳票が正しく印書されているかのチェックも合わせて行っている．BankVision ではこの負荷軽減策として以下の 3 点を配慮した（図 6）．

1)　マージ出力

帳票作成が帳票単位に行われるため，通常は印書もまた帳票単位に行われることになる．各帳票が異なったタイミングで作成・印書される場合は特に問題はないが，同時間帯に作成されるような帳票についても個別に印書するのでは，仕分の効率が悪い．

そこで運用上，一度に印書する方が望ましい帳票については，あらかじめファイル自体を一つにまとめて作成するように最終出力一覧に登録しておくことで，印書を一度で行えるようにしている．これをマージ出力と呼ぶ．

これらの情報は EUR（帳票作成ツール）の入力となる印書情報ファイルに含まれており，印書情報ファイルは実行の都度，最終出力一覧から最新の情報を取得して作成している．なお運用上の何らかの理由により，マージ出力せず個別に出力する必要が生じることも考えられるため，単純なオペレーションにより一時的な変更もできる．

2)　仕分け出力

印書された帳票は，送付先ごとに仕分け作業を行い各宛先に送付される．この際に，当然ながら送付先単位に仕分けされた状態で印書されている必要があり，データを送付先単位（店番）で並べ替えてから印書している．ただし業務上，まとめずに分けて印書する必要があるものについては，この並べ替えをせずにそのまま印書することも可能としている．

3)　表紙の付加

表紙には，送付先（店番，店名），出力基準日，帳票名，出力ページ数などの情報を記載

しており，印書後のページ数確認などの作業に使用している．これに加えて，帳票本紙とは異なる色の紙を使用して，帳票仕分け作業者が送付先の切れ目を，色紙を目安に識別することができるようにした．

これらの処理は，前述の EUR を使用した帳票作成の共通処理の中で実装している．

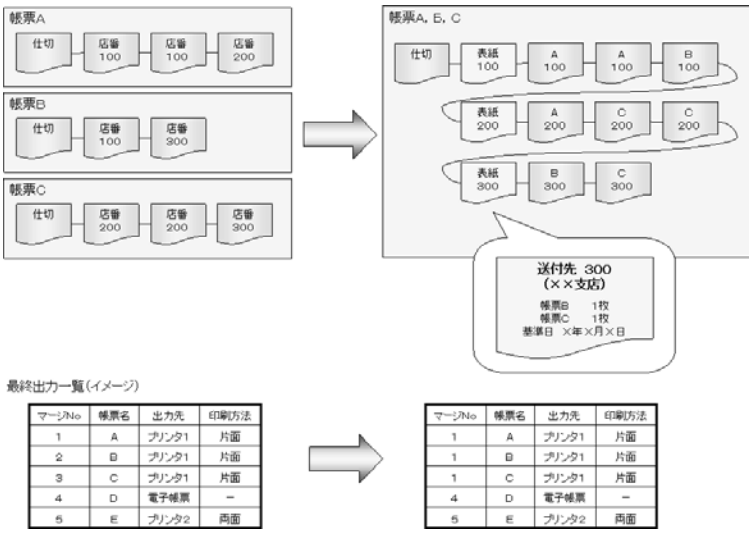

図 6　帳票仕分けと表紙出力

4. 3. 3　各形式での帳票作成と媒体への振り分け

前述のとおり，帳票の出力媒体には 3 種類存在し，それぞれ PS 形式（大量印刷用プリンタ），ESC/P 形式（インパクトプリンタ），PDF 形式（電子帳票）で出力する必要がある．

各帳票固有の処理で意識せずに実装できるよう，「帳票の作成」の最初に行うデータ抽出時点では違いを持たせていないので，どの形式で作成するかを選択し，相対する媒体に送付する必要がある．

これにはまず，帳票作成の時点で最終出力一覧の帳票種類を参照して，どの形式で出力するかを判断し，その上で EUR を利用して指定した形式で帳票を出力させている．次に，作成されたデータを HULFT により転送するが，ここでは転送対象のファイル名にその宛先情報を設定することで送付先を判断している．

以下は電子帳票の場合に使用している物理ファイルの名称である．

【転送 ID】【帳票 ID】【基準日】【優先順位】【枝番】【拡張子】
　転送 ID　：電子帳票サーバに対応する ID，ここで出力媒体を判断する
　帳票 ID　：帳票毎の ID
　基準日　：帳票の作成基準日
　優先順位：帳票取り込みの優先順位（ラッシュ時に使用する）
　枝番　　：帳票，作成基準日ごとの番号（重複登録防止）
　拡張子　：電子帳票の場合は PDF

5. お わ り に

今回この新しいシステムの本番稼働を迎えるにあたり，実現に必須の多くの助言を頂戴した，百五銀行担当者様の多大なる協力に対し，感謝の意を表したい．また開発期間中の，開発プロジェクト内の多くの開発担当者のご協力に感謝すると同時に，数々のご迷惑をお掛けしたことについては，お詫び申し上げる．

新しいシステムを作り上げる過程において，工夫しながら試行錯誤を繰り返し，最終的な実現へと導いた経験が，開発に参加した若手の SE 達にとっての財産となり，次の新たな開発に取り組む時に役立つことを願っている．

＊ 1　基幹系 DWH サーバ以外に対外系や国際系のサーバでも行っている．
＊ 2　SSIS: Microsoft SQL Server 2005 Integration Service は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標である．
＊ 3　EUR は(株)日立製作所の日本における商品名称（商標又は，登録商標）である．
＊ 4　HULFT は株式会社セゾン情報システムズの登録商標である．
＊ 5　BSP-RM は株式会社ビーエスピーの登録商標である．
＊ 6　FileVolante は JFE システムズ株式会社の登録商標または商標である．

参考文献　［1］　中原直紀，三島敏彦，「オープン勘定系システム開発の事例紹介」，ユニシス技報，日本ユニシス，Vol.23 No.1 通巻 77 号，2003 年 5 月

執筆者紹介　宮　澤　貴　之（Takayuki Miyazawa）
1989 年日本ユニシス(株)入社．金融系システムの開発を経て，2004 年より BankVision のバッチ分野の開発に従事．現在金融アウトソーシングセンター S-BITS 共同 OSC 運用 P に所属．

鈴　木　俊　也（Toshinari Suzuki）
2003 年日本ユニシス(株)入社．2004 年度より BankVision のバッチ分野の開発に従事．現在金融アウトソーシングセンター S-BITS 共同 OSC 業務開発第一 P に所属．

福　丸　昌　宏（Masahiro Fukumaru）
2005年日本ユニシス(株)入社．入社配属以来BankVisionのバッチ分野の開発に従事．現在金融アウトソーシングセンターS-BITS基盤統括P第三基盤Pに所属．