事例

流通業店舗システムへのオブジェクト指向技術の適用

―店舗システムのアプリケーションフレームワーク，ミドルウェアの観点から―

石井慎一郎　坪谷英昭　佐野建樹　小堀賢司
NECクライアントサーバソフト技術研究所

小山田正史　内田昭宏
NEC第5C&Cシステム事業本部共通技術システム本部

はじめに

1998年4月にセブン-イレブン・ジャパン「第5次総合情報システム」の「新店舗システム」は，Windows NTサーバ上に構築，全国7200以上の店舗に展開され現在に至っている．「新店舗システム」構築に当たり，コンビニエンスストア店舗システムとしての各種要件に応えるため，さまざまな施策を行ってきた．本稿ではそれらの中から，

- オブジェクト技術適用の一環としてのフレームワーク．
- 店舗システム運用を支えるミドルウェア．

について紹介する．

店舗システムの戦略性

日本におけるコンビニエンスストア（以降コンビニと呼ぶ）は，フランチャイズチェーンストア方式が主流である．これは，本部から各チェーンストアに対してさまざまな経営ノウハウを供給，共有することにより，本部，チェーンストアの共存共栄を目指す方式である．この中で，コンビニの情報システムの役割は，本部での各種経営ノウハウの整理・構築，それらの各店舗への配信，さらに店舗業務にて活用という，それぞれの局面を支えることである．

この「新店舗システム」でも，図-1のように本部，店舗間に衛星通信も取り入れて上記方式を実現し，さらに店舗内にもさまざまな情報武装化が実現されている[1]．

ここでコンビニ店舗システムが満たすべき要件を考えると以下の点が挙げられる．

(A) コンピュータについて素人レベルである利用者が，本部からの情報を徹底的に活用．

(B) 通常，数名のアルバイト店員で店舗運営する中で，24H，365日の営業．

(C) 全国数千にも及ぶ店舗への展開，ならびに運用．

まず上記（A）の背景として，コンビニでは限られたスペースにいかに売れ筋商品を取り揃え，陳列し，品切れを起こさないか...が生命線であるとの事情がある．そのため，たとえば発注・品揃業務にて発注数を決定する際に，過去の販売実績や将来予想を示す各種販促要因情報を基礎情報として活用する手順が組み込まれている．さらに，コンピュータに不慣れな利用者を想定して，ヒューマンインタフェースの容易性が強く求められる．加えて，画面イメージの変更などシステムの長期的な保守・成長も

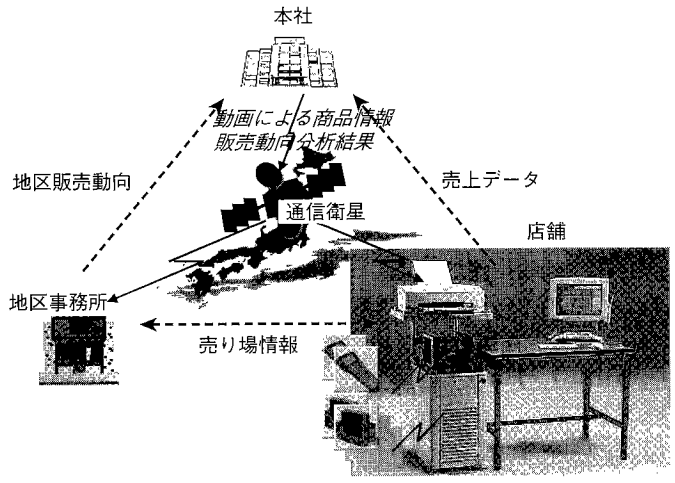

図-1　第5次総合情報システム全体概要

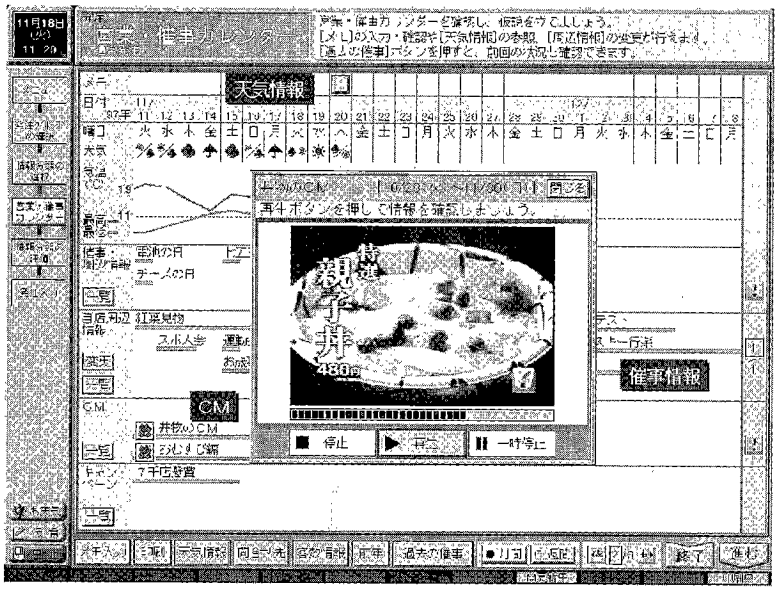

図-2　営業催事カレンダー

考慮する必要がある．そこで，これら要件に対応する仕組みを店舗システムフレームワークとして構築し，さらに店舗システムアプリケーション開発で，トータルな開発環境として適用するアプローチをとった．

また，上記（B），（C）に対応して，店舗システムには高い信頼性と運用容易性，すなわちオペレータレスな運用が要求されている．これを店舗システムのトータル・ロバストネス（信頼性，運用容易性に応える堅牢性）として捉え，ミドルウェアとして構築し，フレームワーク同様，店舗システムアプリケーション開発で，適用するというアプローチをとった．

店舗システム開発環境（フレームワーク）の構築

図-2は，「新店舗システム」の発注業務中の「営業催事カレンダー」画面例である．これには，1枚の画面内に数多くの情報が配置・表示されている．また画面の左側に配置されているボックスは発注業務手順，すなわち参照すべき画面，およびその順番を示している．これらの情報，手順はすべて本部配信情報に基づいて組み立てられる．このような店舗システムの構築を支援する環境として，店舗システムフレームワークの検討を進めたが，まず以下の3つの視点から検討を開始した．

- どのようなソフトウェア構造が最適か…システムアーキテクチャ
- 業務の流れや参照情報はどうなっているのか…業務（ドメイン）モデル
- システム構築支援CASEツールの適用性はどうか…HOLON/VP [2)]

システムアーキテクチャ

●システムアーキテクチャの課題

変更容易性の追求：1画面内に多くのGUI（グラフィカルユーザインタフェース）部品を含むプログラムを変更する際に，デグレード（プログラム変更後に出る既存部のバグ）や工数増大を避けるための仕組み．

業務手順の管理：店舗内業務にて本部発信情報を活用するための業務を誘導する仕組み．

生産性と品質の下支え：大規模プロジェクトとして，多くの人間による作業の内容を一定のレベルに規定するための仕組み．

●変更容易性

変更容易性を確保するためには，プログラム処理の相互依存関係の解消が必要である．図-2のような複雑な画面では，各GUI部品間に相互依存関係が存在する場合が多い．たとえば，ボタン押下時に表内の表示データを切り替える場合，イベント発生元はボタンだが実際に処理が反映されるのは表となる．さらに，表示するデータはデータベース（DB）から取得しなければならない．もしも，ボタンの押下に対応する1処理内でこれらをすべて記述すると，ボタン，表，DBの3者間に陰に依存関係ができてしまう．そうすると，たとえば画面表示にかかわる変更により最悪の場合，表示処理全プログラムを調べないと変更影響を予測できない．そこで，各画面表示部品に1:1に対応して，画面表示部品相互の依存関係を集中させた部品をメディエータ（デザインパターン[3)]におけるメディエータとコンポジットパターンの合成）として考える．メディエータでは，GUI部品や他のメディエータから到着したメッセージに対応して，以下のいずれかの処理を行う．

■画面表示部品の表示変更
■DBからのデータ取得
■他メディエータ宛メッセージ発行

メディエータは画面表示部品間の関連が深いものごとにグループ化し，階層化しておく．また，階層の親をフォームメディエータと呼び，親子関係を超えてメッセージを発行しないようにする．この方式により，変更が影響するのは自分にメッセージを発行するメディエータに限られる．

●業務手順

業務手順の管理のため，上記階層の根に位置するメディエータには特別な役割を持たせている．

■ルートメディエータ…画面内の全メディエータの根となる．画面の表示，消去，画面内メディエータの調整を行う．
■シナリオメディエータ…ルートメディエータを子に持ち，業務手順に従って遷移制御する．

図-3に，システムアーキテクチャ全体図を示す．

●生産性と品質の下支え

共通化：ヒューマンインタフェースガイドラインの一環として配色やフォントを統一するため，属性データとして名前を付加し，各GUI部品が即値ではなく属性データ名で値を取得する機構を用意した．これは，共通部の変更だけで全体を一貫して変更可能にする仕組みの一例である．

ジェネレータの構築：図-4は，ジェネレータを使用する場合のメディエ

ータクラス階層の模式図である．メディエータ階層の製造効率化のため，後に記すHOLON/VPの画面レイアウトエディタによる画面設計情報からメディエータ階層の自動生成を実現している．また，業務固有部品とは，店舗システムに共通汎用的なメディエータ機構に，個別ユーザごとに異なるヒューマンインタフェースガイドラインの要件に対応したメディエータを加えて実装したものであり，アーキテクチャによるガイドライン遵守を実現する．さらに，自動生成後の追加コーディングを容易にするため，自動生成の結果は，カスタマイズポイントを示したテンプレートとなっており，生産性向上に寄与している．

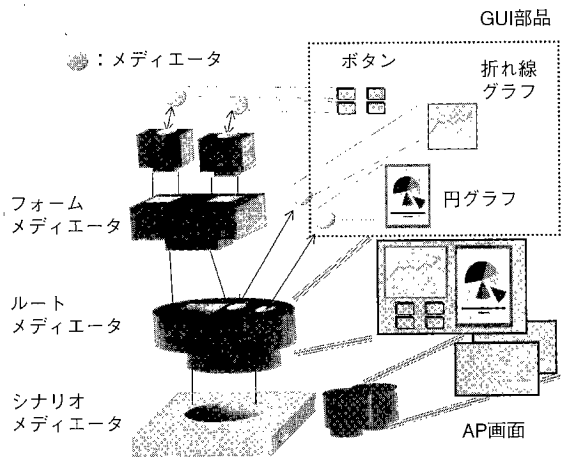

図-3　システムアーキテクチャ全体図

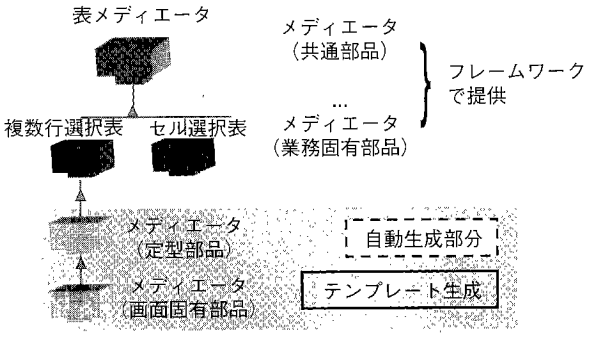

図-4　メディエータ階層とジェネレータ

業務（ドメイン）モデル

冒頭で記述したように，発注・品揃等の業務では，販売実績や各種販促要因などの多様なデータが参照される．また，同一のデータが異なる画面で表示されることも多い．このような場合，データ操作ロジックの変更修正範囲を局所化し，画面の見栄えやDBスキーマ変更の影響を最小限にする必要がある．本章では，システム開発の1工程である，上記データに対応する業務（ドメイン）モデル構築作業に対して，業務モデル構築手順書の開発経緯，概要と適用時の問題点，今後の課題をまとめる．

●業務モデル構築手順

ここでいう業務モデルとは，業務上意味のあるものをオブジェクトとして捉えた業務オブジェクトと，それらの静的，動的な関連をクラス図，シーケンス図に表したものである．業務オブジェクトは，DBとのデータの入出力や，データの加工，業務ロジックを実行する．当時，アプリケーション開発メンバにオブジェクト指向技術の経験者が少なかったことから，従来技術をベースとした開発手順が求められたため，データ中心技法を拡張して，業務オブジェクトを洗い出す手順を検討した．構築手順の概要を図-5，ならびに以下に示す．

(1) 業務の構造と流れの把握

店舗システムの業務構成と，作業の流れを整理する．整理結果は，業務階層図，業務フロー図にまとめる．

(2) 画面仕様，処理仕様の洗い出し

各業務ごとに，画面レイアウトと画面上での処理を洗い出す．処理は，画面表示前，表示中，表示終了後と正常系，異常系に分類整理し，画面処理仕様書としてまとめる．

(3) オブジェクト分析

各画面ごとに，画面レイアウトやDBのスキーマ情報から業務オブジェクト候補を洗い出す．次に画面に表示されている項目から業務オブジェクトのメンバを，画面処理仕様書からメソッドを洗い出し，クラス図，シーケンス図としてまとめる．最後に，全業務を対象とした統合整理を行う．

(4) オブジェクト設計

分析で洗い出した業務オブジェクトごとに，メンバの型，メソッド引数，リターン値の型を決定し，メソッドの処理内容を設計する．DBアクセス，データチェック，データ加工処理などの業務ロジックを詳細に定義する．

この構築手順では，(1) の業務階層図の最下位をUML（Unified Modeling Language）のシステムへの操作を表すユースケースに，(2) の画面処理仕様書をユースケース内の動きを示すシナリオに対応づけている．

●適用結果の問題点

本手順適用の結果，(3) 段階，および設計実装時に，下記の問題点が明らかになった．

■業務オブジェクトの正しさの基準が曖昧

■業務オブジェクトのメソッドの実装指針がない

これらは，各種データの参照処理にて，複数のデータをまとめる処理の実装指針がなかったことによる．画面処理から各業務オブジェクトに問い合わせてまとめるか，必要なデータをSQLのクエリやストアドプロシジャで一括してとるかなどの判定は難しい問題である．本問題については，あくまでも業務オブジェクト抽出の容易さを第1に考え，分析段階では，業務オブジェクト構造をテーブルとほぼ1:1に対応するエンティテ

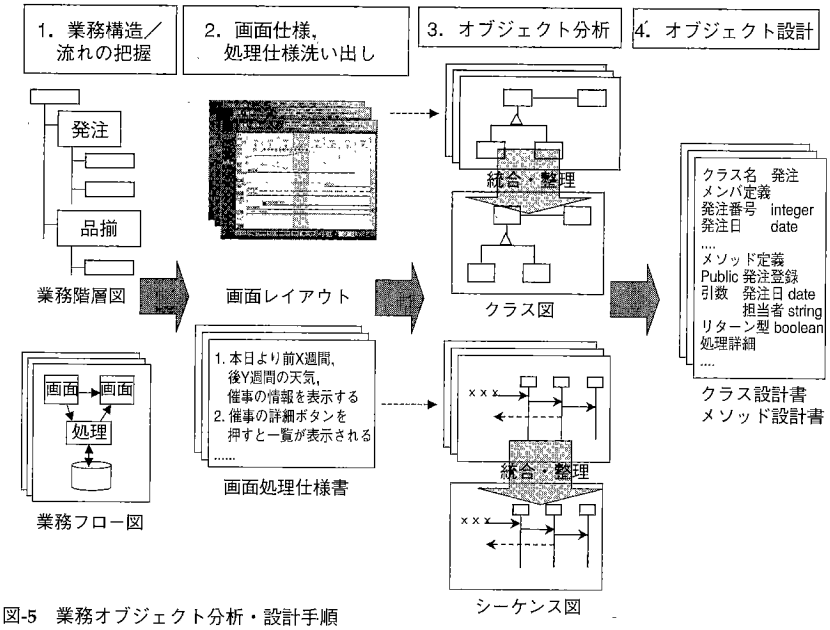

図-5　業務オブジェクト分析・設計手順

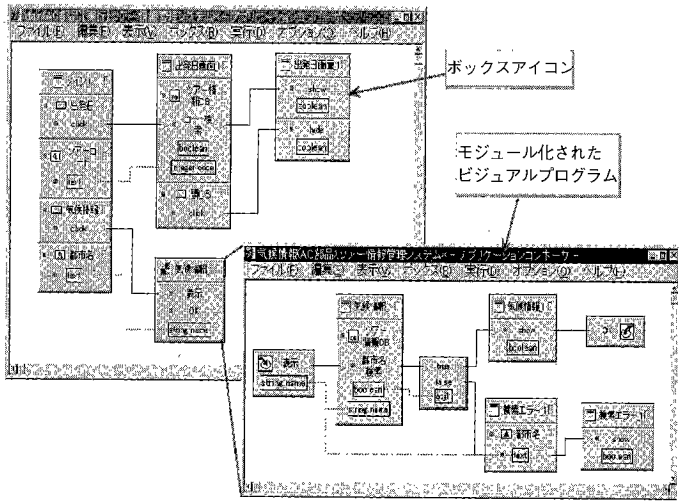

図-6　HOLON/VPビジュアルプログラミング

ィオブジェクト，複数データのマージや他エンティティオブジェクトの制御，画面のメディエータからの各種処理要求受け付けを行うコントロールオブジェクトとすることで対処した．

●一層の生産性向上に向けて

上記問題点を解決し，店舗システム開発の生産性向上に貢献するには，カスタマイズポイントをもあらかじめ提示する業務モデルが有効である．つまり，店舗システムの普遍的な部分と，個別ユーザに固有な部分とを分離して提供し，さらに個別ユーザに固有な部分を普遍な部分のカスタマイズポイントとする．たとえば，店舗業務の共通部分としては，単品ごとの商品データに基づいて発注数を検討する流れがあり，そこでのカスタマイズポイントは，個別ユーザごとに異なる，必要な商品データの選定...という具合である．これによって，業務モデルの構築作業はカスタマイズポイントに沿って進めることが可能となる．

開発ツール：HOLON/VP

●HOLON/VPとは

HOLON/VPは，店舗システムとそのフレームワークの構築に使用された，オブジェクト指向技術に基づくビジュアル開発ツールである．機能としては，画面や帳票作成用のエディタやデータベース設計支援ツールなどの他に，実行時にアプリケーションの一部として動作する部品（ランタイム部品）も提供している．HOLON/VPの大きな特長はビジュアルプログラミングである．図-6に示すように，画面やメソッドなどの部品を表す箱（ボックスアイコン）をエディタ画面上に配置し，回路図を描くように線でつなぐことによって処理ロジックを組み立てていく．

また，HOLON/VPは，Hologramという独自のオブジェクト指向言語を，メモリの自動管理やインタプリタ実行方式とともに提供している．Hologramプログラムは C++ プログラムに変換後コンパイルされる．これにより実運用プログラムの高速実行を可能としている．

●適用に際しての改善・強化点

今回の店舗システム開発プロジェクトにHOLON/VPを適用する過程で行った改善・強化について述べる．

(1) 高機能部品の提供

今回の店舗システムでは非常に複雑な画面が求められ，これをアプリケーションとして1から作っていては生産性，品質が著しく低下するとの懸念があった．そこで複雑な画面構築に対応した高機能な部品をいくつか提供したが，この例に図形部品がある．これは図形を組み合わせて画面表示部品を作るもので，図形部品エディタで初期画面を作成し，そこに配置されている図形をオブジェクトとみなし高レベルなAPI（アプリケーションインタフェース）セットを用いて操作することができる．

(2) 部品化・再利用機能の強化

大規模な開発での生産性向上のため以下のような機能強化を行った．

ビジュアルプログラムのモジュール化：結線によるプログラミングのようなビジュアル化のメリットは，構造の大局的・直感的な把握のしやすさにある．しかし，複雑，大規模な処理の場合，大きな図面になり全体が捉えにくくなってしまうため，いくつかの処理群を1つの箱にして上位部品で利用できるモジュール化機能を用意し，大規模な内容でも把握可能とした．図-6の右下の画面がモジュール化された箱の内容を表している．

画面のサブクラス化：ある画面継承した画面を作成する機能である．これにより，レイアウトが共通な部分を親画面として整理しておき，個別画面の開発時に，子画面として継承して部分的に差分開発する進め方が可能となった．

GUIカスタム部品：画面の一部を処理記述とともに部品化し再利用するものである．

(3) 性能改善

作成アプリケーションの起動時間短縮，最小必要メモリ削減を実現するとともに，ランタイム部品については，生成時間や描画時間の高速化，所要メモリ削減に取り組んだ．

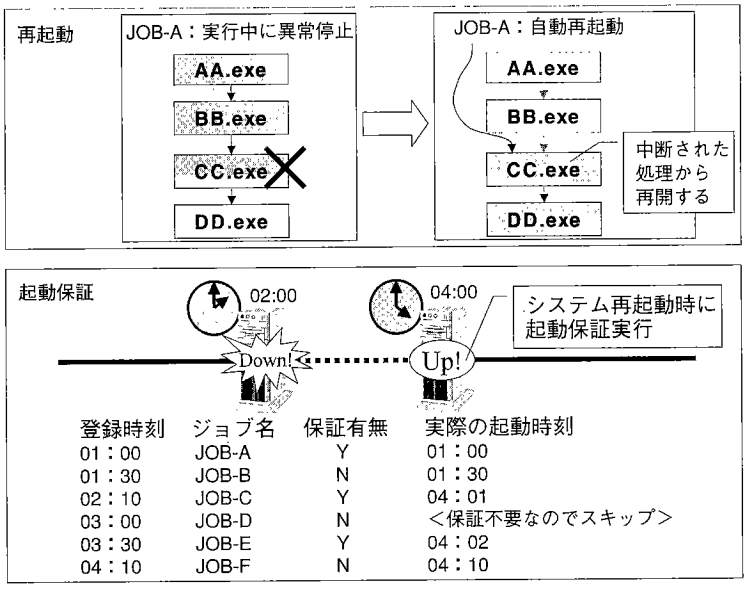

図-7　業務処理構築基盤ミドルウェア

●改善を通じて

上述のようなオブジェクト指向技術を核としたさまざまな機能拡張や性能改善によって，生産性と品質の両面において，大規模プロジェクトでのシステム開発を，ツールとして下支えすることができた．画面のサブクラス化機能やGUIカスタム部品化機能により，複雑な画面に対応する共通部品を効率的にとりそろえることが可能となった．また，性能改善の結果，ヒューマンインタフェースガイドラインに従って開発を進める限り，アプリケーションには一定レベルの性能が確保できるレベルに達した．

店舗システムのロバストネスへの対応

コンビニの特長により，店舗システムが停止すると店舗運営が継続不可能になるため，店舗システムには高い信頼性が求められる．信頼性への要件は次の2点に集約することができる．

(1) 24時間連続稼働に耐える信頼性(No Down)．数カ月連続稼働できるハード・ソフト品質．

(2) プログラムの更新や障害発生時に店舗での管理操作が不要であること（Zero Administration)．

この要件自体は多くの情報システムに共通であり，要求レベルや費用に応じて，各種サーバ管理，クライアント管理製品も利用可能である．しかしコンビニにおいては，下記に示す制約が厳しく，既存製品の適用はできなかった．

■コスト制約：数千店舗に小規模なシステムを設置するため，クラスタ構成や障害監視の付加装置などコストのかかる方式を採用することができない．

■地理的分散による制約：障害時に，保守員が出向くには時間もかかる．

■徹底したZero Administration化：一般にサーバ管理製品は，運転管理や障害管理のために多少の利用者操作が必要である．店舗ではこの多少の操作ですら期待することができない．

これらは全国多拠点を結ぶ情報システムに共通の制約ではあるが，特にコンビニでは条件が厳しいといえる．

●ロバストネス（信頼性，運用容易性に応える堅牢性）へのアプローチ（自律システム）

上記制約のもとで高信頼を実現するため，自律システム化というアプローチを採用している．自律システムには2つの側面がある．

第1に，各装置間の依存性を下げるということである．店舗システムは，本部とネットワークで結ばれて連係動作するが，発注送信など通信が必須の業務以外は，本部と切り離されても店舗システム単独で継続できる．また，通常POS端末で登録された販売トランザクションデータは逐次ストアコンピュータ（以下SC）に登録されるが，ネットワークやSC自身に障害が発生した場合，半日程度はPOS上にデータを蓄積して処理を継続可能である．

第2に，SC自身の自律システム化が必要である．SCはデータベースサーバ機能を持つ店舗システムの核であり，さらに本部との通信サーバ，画面表示端末機能を併せ持っている．SCの信頼性目標を達成するため，ユーザ操作の不要な自律管理を主眼に，ミドルウェアの構築を行った．

以下，今回開発した店舗サーバ＝SCのロバストネスを実現するミドルウェアについて説明する．

●店舗システムミドルウェアの概要

ロバストネスを実現するミドルウェアは，大きく2機能に分類できる．

(1) 業務処理構築基盤ミドル

システム異常停止後，運用可能な状態に戻すためには中断した業務処理のリカバリなどの回復操作が必要である．特にコンビニでは数千カ所に分散したシステムを運用するため，人手の介入なしに自律回復することが望まれる．しかし，個々の業務アプリケーションの中に回復のための機能を実装することは現実的でない．店舗システムでは，図-7のようにすべての業務処理をジョブとして実装し，ジョブ管理機能によってシステ

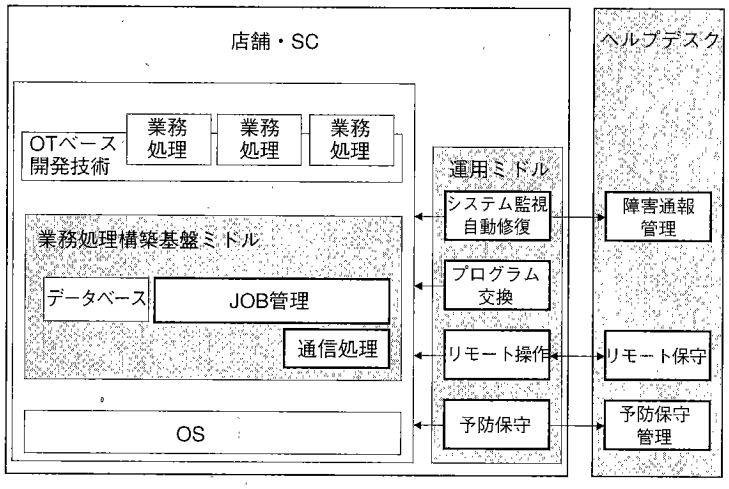

図-8　店舗システムのソフトウェア構造

ムの自動回復を行う設計とした．

■ジョブの再起動：システム異常停止からの再起動時，異常停止で中断されたジョブを再起動する．

■ジョブの起動保証：障害や保守作業などでシステムが停止すると，停止期間中にスケジュールされていたジョブがスキップされてしまう．そこで，停止時間中にスケジュールされていたジョブを探し，システム再起動時に起動保証する．

(2) 運用管理ミドル

ミドルウェアからOS・ハードウェアまでを含むシステム全体を対象に，障害の予防・早期発見と自動回復を行うために，図-8にある位置付けで運用管理ミドル群を用意した．

■システム監視と自動回復・通報

利用者が異常に気づく前に障害の兆候を検出し，可能ならば自動的に回復するための手段を講じる．また回復不可能な障害は，ヘルプデスクに対して障害通報を送る．以下の項目を監視対象とした．

• ハードウェア状態監視：温度センサやFANセンサなどをモニタし，回復不能になる前に，障害通報を送ったうえでシステムを安全に停止する．

• パフォーマンス監視：CPU負荷，メモリ使用状況，ディスク使用状況を監視する．異常が検出された場合には問題解析のためにシステム状態を記録したうえで障害通報を送る．

• プロセス稼働状態監視：常駐プロセスについての存在監視を行う．プロセスが不在となった場合には，自動的にプロセス再起動を行う．成功すれば通報は行わないが，一定回数以上繰り返して再起動に失敗した場合には障害通報を行う．

■予防保守

ハードウェアの稼働状態（稼働時間，動作回数，通信系のステータスなど）を毎日収集しヘルプデスクに送信する．ヘルプデスクでは寿命を越えたり動作が不安定になっているデバイスを抽出し，交換手配を行うことで障害発生を未然に防ぐ．

■プログラム交換

新規サービスの開始や業務の見直しなどのため，コンビニではプログラムの更新が頻繁に行われている．数千カ所でのプログラム更新を安全かつ自動的に行うため，交換時刻を指定した交換差分モジュールを店舗に配信し，指定時刻に自動更新処理を行う機能を用意した．プログラム交換には高信頼化のため次の機能強化が行われている．

• ファイル置換中に，容量不足や書き換え失敗が発生した場合，安全にかつ自動的に更新前の状態に戻す．

• プログラム交換機能を含むミドルウェアや，デバイスドライバなどほとんどのモジュールが交換可能．

■リモート操作

未知の障害に対して，最終的には店舗システム開発者がSCに接続して障害解析を行う必要がある．リモートコマンド，ファイル共有，画面共有を使って，各種障害の分析や修復を行うことができる．

●適用評価

今回，我々にとって未経験のプラットフォーム上での高信頼性確保という命題に対して，ミドルウェアによる解決を図ったが，最終的な店舗システム安定稼働までにはいくつもの困難があった．特に基本ソフトウェア（OSなど）にかかわる障害の場合，その原因の特定だけでも非常に工数がかかってしまった．ここで，積み重ねた経験を今後広範囲に適用するため，高信頼機能の部品化や各種分散処理フレームワークへの適用などを進めるつもりである．

フレームワーク，ミドルウェアの適用

最後に，店舗システム開発者への適用に言及する．フレームワークもミドルウェアも，店舗システムのプログラム構造を規定するもので，自由なコーディングを許さない以上，それぞれをよく理解してもらうことが大切である．今回はオブジェクト指向技術に不慣れなメンバも多く，適用作業には成果構築と同等の工数を必要とした．ここで，フレームワーク構築メンバと店舗システム構築メンバとの間に先行適用メンバをおき，成果評価，利用環境整備を行ったことが非常に効果的であったことを付け加えておく．

おわりに

この店舗システム構築では，変更容易性や信頼性が求められるクリティカルな部分をあらかじめ部品として用意し，トータルな開発環境として提供，展開するアプローチを採用したが，これは生産性のみならず，品質向上にも寄与できたと考えている．今後，このオブジェクト指向技術のもう1つの特長である再利用性についても，追求していく所存である．

参考文献

1) セブン-イレブン，新システムの全貌，日経コンピュータ，pp.200-208 (1997.11.24).

2) HOLON/VP関連 (1998年11月現在) http://www.nec.co.jp/japanese/product/ccsoft/vp/index.html

3) Gamma, E. et al.: デザインパターン，ソフトバンク (1995).

(平成10年11月20日受付)