Kiban@asaban および LUCINA を適用し構築した FEDI システムとフレームワークの紹介

FEDI System Developed by Applying LUCINA and Kiban@asaban, and of its Framework

藤 井 伸 行, 小 川 昭 彦

要 約 本稿は，統合アプリケーション・ホスティング・サービスである Kiban@asaban およびコンポーネント開発技法である LUCINA を適用して構築した FEDI[*1] システム，および FEDI システム構築時に定義したアプリケーションのフレームワークの紹介である．

Kiban@asaban のプラットフォーム，共通サービス，および運用サービスを利用することで，決済サービスを担える，安全でかつ高信頼システムの短期構築が可能となる．また，LUCINA をベースにして，拡張性をもったフレームワークをデザインし，業務アプリケーション構築に適用することで，品質のばらつきを抑え平準化したシステム開発を短期間で行うことができる．

Abstract This paper introduces the FEDI system, which is developed by applying LUCINA (method of developing components) and Kiban@asaban (hosting services of integrated application), and the application framework defined at the time of FEDI system development.

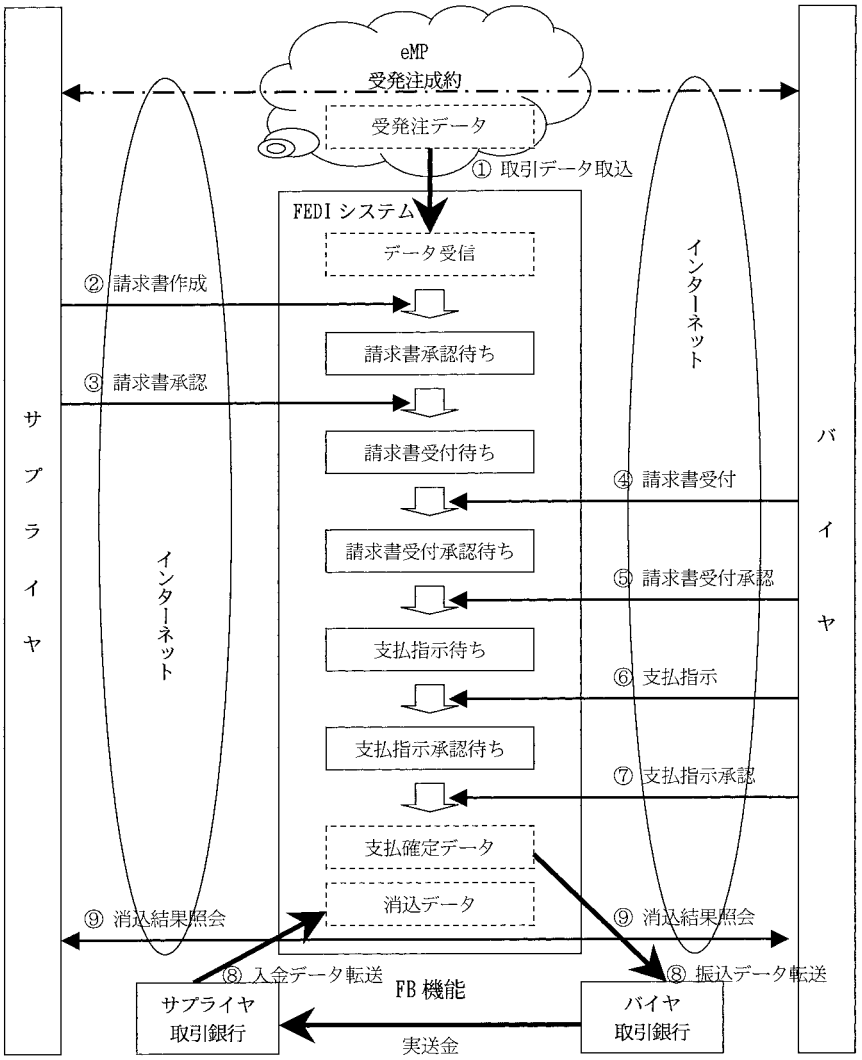
By using the platform, common services and administration services of Kiban@asaban, the secure and highly reliable settlement system can be developed for a short period of time. Also, Designing the framwork with extendibility and applying it to development of business application based on LUCINA can quickly perform system development while smoothing dispersion in quality.

1. はじめに

近年，米国では e-MarketPlace（以降 eMP）が急速に発展した．また，日本でも商品の種類や業界ごとに様々な eMP が立ち上がってきている．eMP とは，インターネット上に設けられた Web サイトを通じてサプライヤとバイヤを結び付ける電子商取引市場のことである．eMP が急速に発展した理由として，サプライヤとバイヤが eMP で直接取引をおこなうことにより，これまでの中間流通業者を経由せずに取引することができ，流通コストを削減できることなどがある．サプライヤにとっては，新規取引先の開拓や営業コストの削減，取引先の増加による在庫リスクの平準化，在庫調整などを実現できる．また，バイヤにとっては，調達コストや物流コストの削減，スポット取引による緊急時の調達手段の確保などを実現できる．

eMP の運営には，サプライヤもバイヤも安心して取引できる信用や決済，与信管理などの金融機能が必要となる．米国では専門企業が独自のソフトウェアやノウハウを武器に多くの業界向け eMP を運営しており，業績を急速に拡大している．日本では，鉄鋼や化学などでは商社主導のものが多く，電子部品や製紙関連，建設資材などではメーカ主導のものが多い．

本稿では，このような背景の中，あるユーザが BtoB ビジネスへの参入を目的に，



① 取引データ取込-- eMP で成約した受発注データ（取引データ）を，ファイル転送により本システムに取り込む．
② 請求書作成------ サプライヤが，取引データをもとにバイヤに発行する請求書を作成する．
③ 請求書承認------ サプライヤの承認権限者が，請求書を承認しバイヤに発行する．
④ 請求書受付------ バイヤが，請求書を受け付ける．
⑤ 請求書受付承認-- バイヤの承認権限者が，受け付けた請求書を承認する．
⑥ 支払指示-------- バイヤが，請求書受付承認で承認した請求書に対して支払指示をおこなう．
⑦ 支払指示承認---- バイヤの承認権限者が，支払指示の承認をおこなう．
⑧ データ転送------ 支払指示承認により，バイヤの口座からサプライヤの口座に振込を実施する．
⑨ 消込結果照会---- サプライヤおよびバイヤが請求状況，支払状況，振込状況の照会をおこなう．

図 1　本システム概要図

Kiban@asaban および開発技法の LUCINA（ユニシス「技報」通巻 65 号及び 68 号参照）を適用し開発した鋼材用 eMP における FEDI[*1] システムの事例を紹介する．

また，短期開発のために定義したフレームワークの内容を併せて紹介する．

なお，本稿に登場する ASP は，Active Server Pages を意味する．

2. システム構成

2.1 システム概要

FEDI システムは，eMP で成約した受発注に対する決済システムである（図 1　本システム概要図）．

本システムは，サプライヤから③の請求書承認で作成するデータをバイヤに配信するのではなく，バイヤがインターネットを介し本システムに対してサプライヤからの請求書情報を参照する仕組みである．

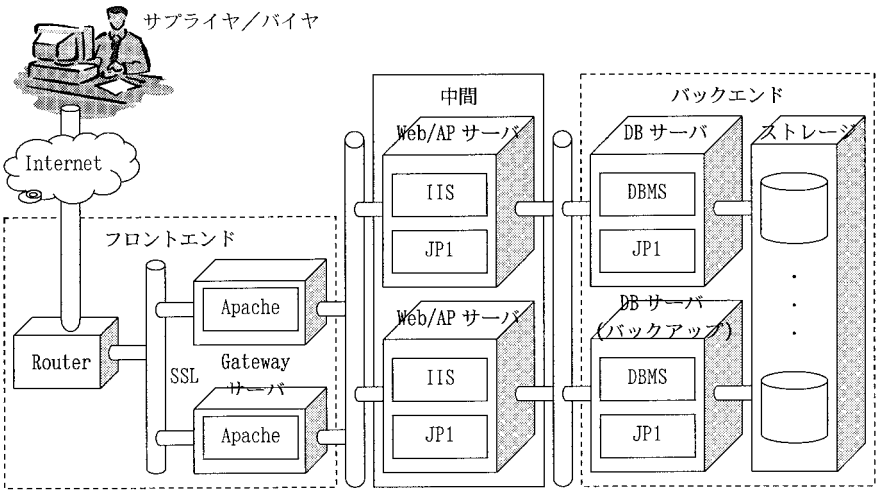


図 2　ハードウェア構成概要図

2.2 ハードウェア構成

ハードウェア構成は，Kiban@asaban のシステムアーキテクチャに則る．フロントエンドとバックエンドを利用し，中間に業務アプリケーションを挟み込む（図 2　ハードウェア構成概要図）．

2.3 ソフトウェア構成

2.3.1 アプリケーション制御構造

LUCINA に準じて次の 3 層構造とする（図 3　ソフトウェア構成図）．

1）　プレゼンテーション層

プレゼンテーション層は更にサーバ上の ASP（VBScript と HTML）とクライアント上の JavaScript に分かれる．JavaScript を使用したのは，チェックルーチンの共通化を図ることができ，また，クライアント上でチェック処理を実行でき，そうすることでサーバとの通信回数を減らすことができるからである．

・ASP : GUI

・JavaScript：クライアント上でのチェック処理

2） ビジネスロジック層

・ビジネス COM：業務処理

3） データアクセス層

・共通 COM：ビジネス COM から共通で呼ばれる DB へのアクセス処理

ビジネス COM，共通 COM は共に VisualBasic（VB）で作成する．

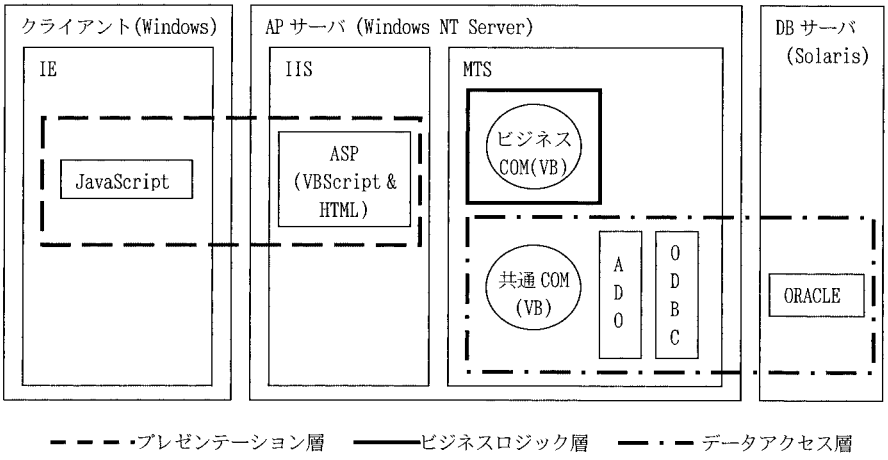


図 3 ソフトウェア構成図

3. Kiban@asaban および LUCINA と本システムの関連

新しい分野のアプリケーションを約 3 ヶ月という短期間で開発し，本番稼働を実現するため，Kiban@asaban および LUCINA を採用した．

Kiban@asaban を適用することで，フロントエンドとバックエンドの中間に挟み込む業務アプリケーションの開発に，専念することができた．また，LUCINA を適用することで，短期間でプロトタイプ開発をおこない，拡張性を持ったアプリケーションのフレームワークをデザインした．そのフレームワークを業務アプリケーションに適用することで，品質のばらつきを抑え平準化したシステム開発をおこなった（図 4 適用関連図）．

次章で，フレームワークの概要を紹介する．

4. フレームワーク概要

フレームワークとは，そのシステム開発のもととなる汎用的に利用できる枠組み，すなわち，システムの構造（アーキテクチャ），実装方法（パターン），共通部品のことである．フレームワークを定義し適用することで，要員の知識，経験の違いなどによる品質のばらつきを抑え，平準化した品質の高いシステム開発をおこなうことができる．

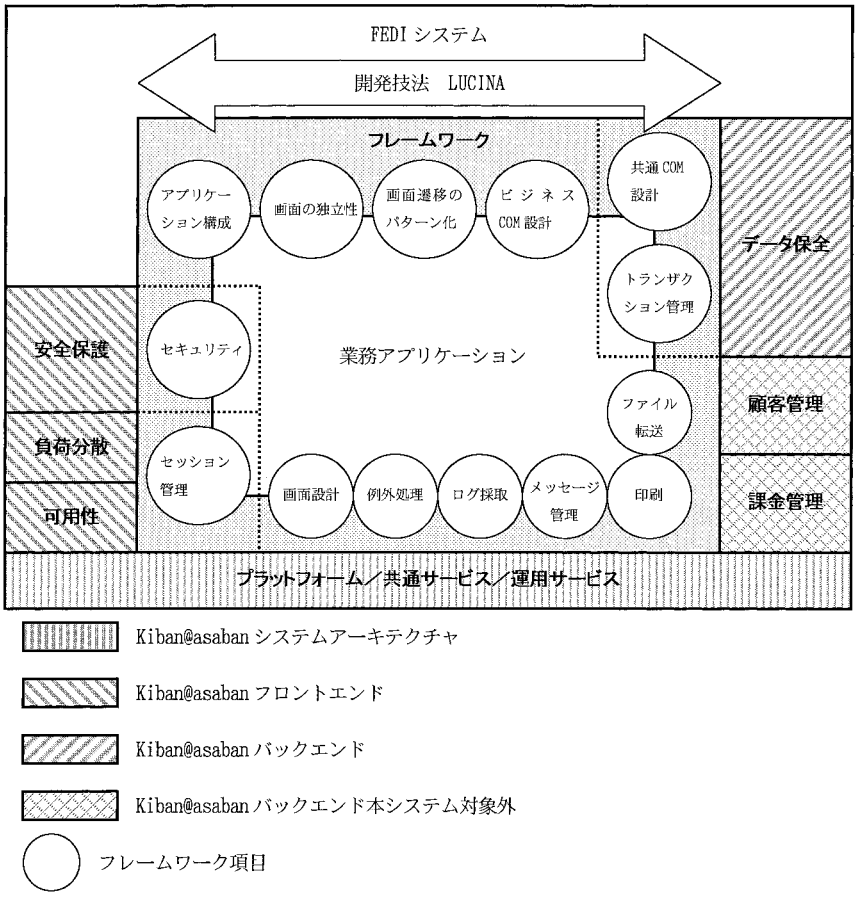


図 4　適用関連図

Web アプリケーションである本システムは，その特性や，短期開発，高信頼性および拡張性の要件から最初にフレームワークを定義した．本章では，その中で主要な 8 項目について紹介する．

1）　アプリケーション構成
2）　画面の独立性
3）　画面遷移のパターン化
4）　ビジネス COM 設計
5）　共通 COM 設計
6）　トランザクション管理
7）　セッション管理
8）　セキュリティ

4.1　アプリケーション構成

開発およびメンテナンスを容易にするために，ASP および COM の役割を明確（2.3.1　アプリケーション制御構造参照）にし，構成を単純化する（図 5　アプリケーション構成図）．

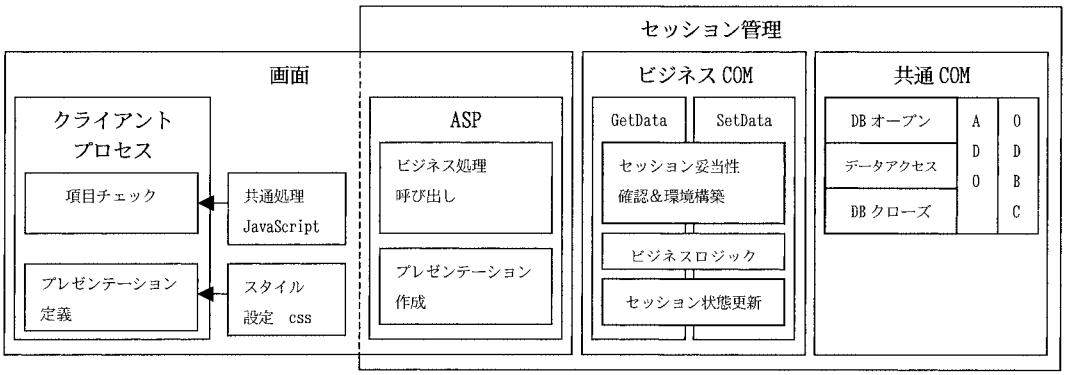


図 5　アプリケーション構成図

4.2　画面の独立性

通常，1 インタラクションでの処理の構成がアプリケーションの作成単位となるが，本システムでのアプリケーションの作成単位は，次のパターンを基本とする．

1）　画面を表示するために必要な情報をワークテーブルから取得する（次画面でアプリケーションの「戻る」ボタンが押下されて制御が戻された場合を含む）．
　COM の GetData メソッドの役割である．

2）　画面を表示する．
　ASP の役割である．

3）　画面の情報をワークテーブルに出力する．
　COM の SetData メソッドの役割である．

インタラクションとは，アクタとユースケースのやりとりを意味し，アクタとシステム間の 1 回のメッセージ送受信が 1 インタラクションとなる．通常，1 インタラクションでの処理は，該画面のデータをワークテーブルへ出力することに始まり，次画面のデータをワークテーブルから取得し，次画面を表示することで終わる構成になる．

図 6 は，1 インタラクションを一つの ASP と一つの COM で実装した場合である．検索画面を表示するには，初期表示の A 1 と，エラーで再表示する A 2 があり，二つの ASP と二つの COM の構成となる．この場合，検索画面が 1 項目でも変更になると，A 1 および A 2 のそれぞれの画面表示ロジックに変更の影響がおよぶため，保守性向上の観点からこの方法は不採用とする．

この問題を回避するために，図 7 のように，検索画面の表示は，初期表示の A 1 でおこなうこととし，エラーで再表示する場合も，A 1 に制御を戻すことで表示の処理を 1 箇所とする．結果，検索画面を表示するには，一つの ASP と一つの COM の構成となり，ASP と COM の役割が明確になる．また，その中でおこなう手続きを統一することができ，保守性が向上する．

よって，アプリケーションの作成単位は，インタラクションを跨った構成とする．

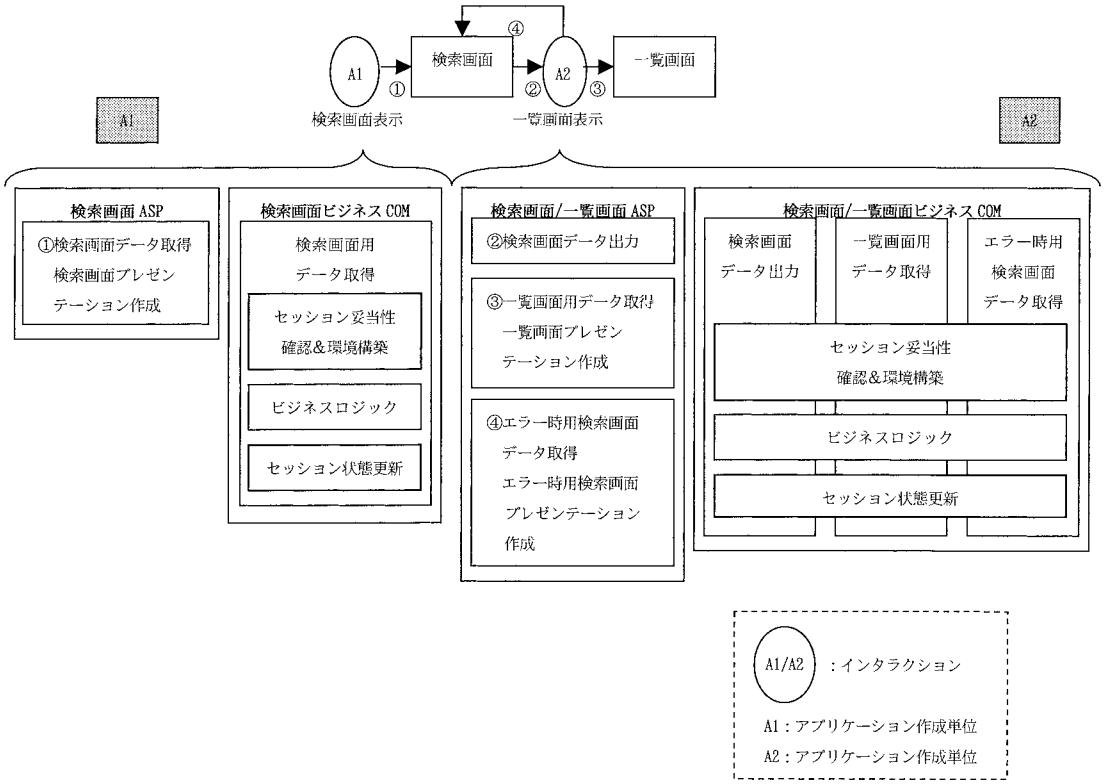


図 6　インタラクション・アプリケーション対応図（不採用）

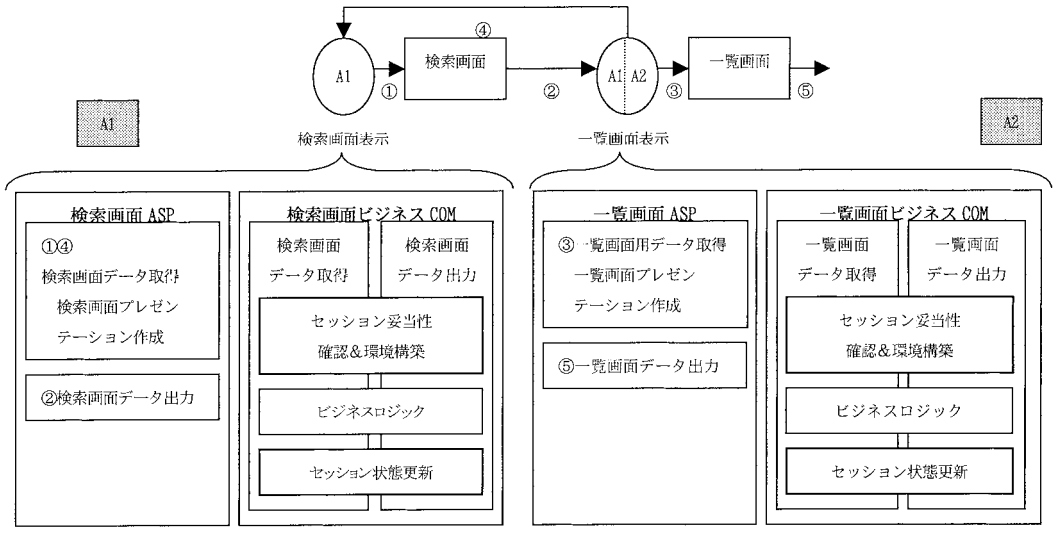


図 7　インタラクション・アプリケーション対応図（採用）

4.3 画面遷移のパターン化

1機能を完了するための画面遷移を，パターン化することにより流用性が高まる．また，操作を統一することによって，ユーザの混乱を抑えることができる．

本システムでの1機能を完了するまでの画面遷移は，次のパターンを基本とする（図8　画面遷移基本パターン図）．

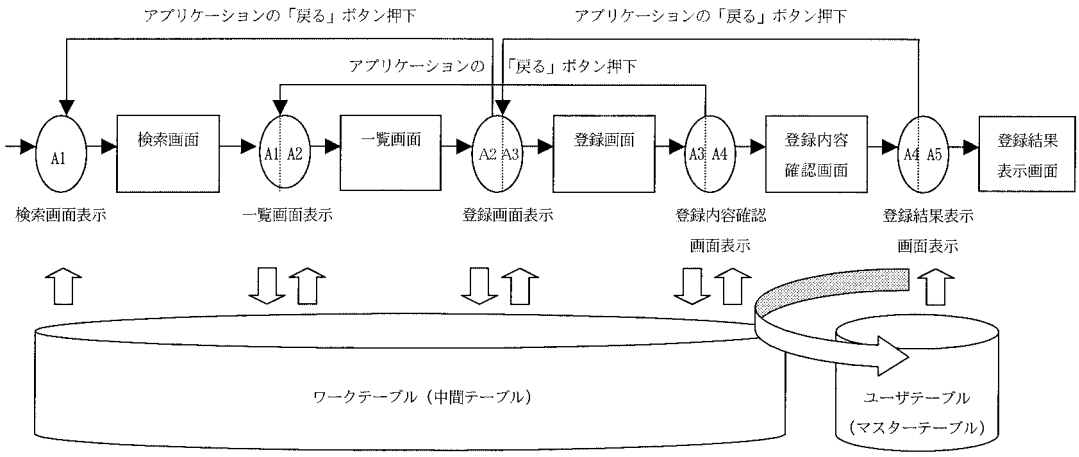


図 8　画面遷移基本パターン図

4.4 ビジネス COM 設計

ビジネス COM では，ビジネスロジックの処理をおこなう．また，DB（データベース）へアクセスする際の SQL の組み立ては，ビジネス COM の役割とする．画面を表示するために必要なデータを取得するメソッド（GetData メソッド）と，表示した画面に対してユーザが操作した状態を保存するメソッド（SetData メソッド）でビジネス COM を作成する（図9　ビジネス COM・共通 COM 関係図）．

各メソッドでは，次の順に処理をおこなう．

1）セッション管理(セッションの妥当性チェックおよびセッション管理環境構築)

2）ビジネスロジック

・データを更新する場合は，一貫性を保証するために各ユーザテーブルにある項目 update_number を更新条件に設定する．

・データを更新する要求は，共通 COM（データアクセスをおこなう）の該当するメソッドを使用し，自ら直接 DB への要求はおこなわない．

・金額計算は，VB ではなく SQL でおこなうことを基本とする．有効桁が 10 桁以上の場合は，必ず SQL でおこなう．

・アプリケーションで扱う定数は，共通ファイル（VB の bas ファイルで共通として提供）の定数宣言の値を使用する．

3）セッション管理（セッションの状態更新）

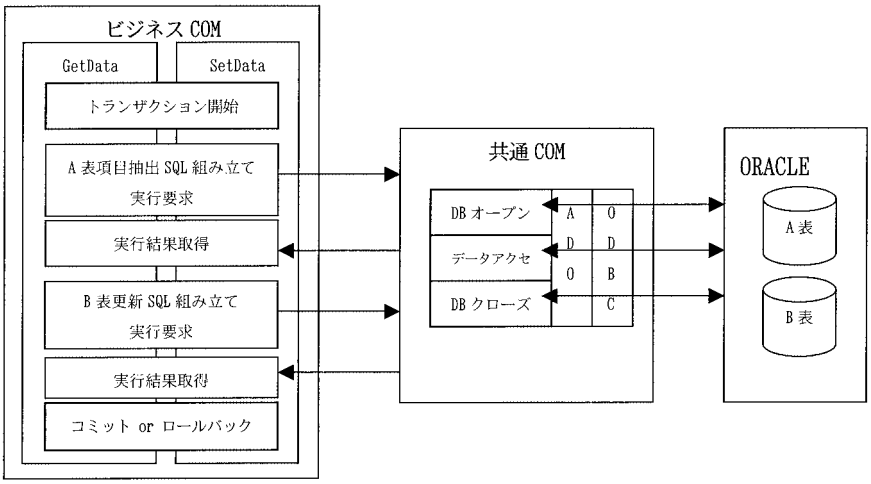


図 9　ビジネス COM・共通 COM 関係図

4.5　共通 COM 設計

DB へのアクセスはビジネス COM でおこなわず，共通 COM でおこなう．本システムでは，DB のプログラムインタフェースに ADO（ActiveX Data Objects）を採用している．ADO は十分汎用的ではあるが，システム開発者の経験や知識にばらつきがあり，ADO を使用して多くのデータアクセス層の COM を作成した場合，システムの品質を保てない可能性がある．そのため，ADO の機能を隠蔽し ADO の経験が少ないシステム開発者にも使用しやすいように DB アクセスの共通 COM を作成する．この共通 COM は，照会では，ビジネス COM から SQL を受け取り DB に問い合わせ，その結果，DB が返すレコードセットを二次元のバリアントテーブルの配列に格納しビジネス COM に返す．DML では，SQL をビジネス COM から受け取り処理結果をビジネス COM に返す（図 9　ビジネス COM・共通 COM 関係図）．

そうすることにより ADO を隠蔽でき，ビジネス COM のコーディング量を削減することができる．

ただし，共通 COM は ADO の隠蔽をおこなうのみであってデータベース構造に依存しない．また，データの確実性を高めるために，全てステートレスオブジェクト[*2]として実装する．

本システムは開発規模が小さいので，DB の変更にともなうビジネス COM への影響が少なく，また短期間での開発であるため，この方法を採用する．

4.6　トランザクション管理

本システムでは，MTS（Microsoft Transaction Server）をトランザクション制御サーバとして採用する．MTS は，オブジェクトコンテキストと呼ぶ構造をビジネスオブジェクトに割り当てて，トランザクションを管理している．MTS は，「トランザクションが必要」もしくは「新しいトランザクションが必要」と宣言したオブジェクトを呼び出すと，そのオブジェクトに関連付けたオブジェクトコンテキストにトランザクション状態のマークを付けトランザクションを開始する．これ以降の処理は，同一トランザクション内での処理として扱われコミット，ロールバックの対象となる．

また，トランザクションを終了するタイミングは，トランザクションを開始したメソッドが終了する時点である．この場合，トランザクションをコミットするかロールバックするかは，セッション中でユーザが SetAbort を一度でも呼び出したか否かで決定する．

同一オブジェクトコンテキストの中で, ユーザが SetAbort を一度でも呼ぶと, MTS はトランザクションの終了時にロールバックをおこなう．ユーザは，処理が正常終了した場合は，SetComplete を呼び出す．ユーザが SetAbort,SetComplete のどちらも呼び出さなかった場合は，MTS が SetComplete を暗黙で呼び出す．

ビジネス COM はトランザクションサポート属性を「必要」に設定し，共通 COM は「サポート」に設定する．ビジネス COM の生成は ASP がおこなうが，ASP ではトランザクション付きのコンテキストを作成できないため，自動的にトランザクション付きのコンテキストを作成する．これに対し，共通 COM はビジネス COM でインスタンスを生成するため，上位のオブジェクトであるビジネス COM からトランザクションを継承する．共通 COM を ASP から直接呼び出した場合，前述のとおり ASP ではトランザクション付きのコンテキストを作成できないため，トランザクション管理がおこなわれなくなる．そのため，ASP から直接共通 COM を呼び出すことは禁止する．

4.7 セッション管理

セッション管理はユーザ単位でおこない，1 セッション 1 ユーザとする．あるユーザ ID でログインしていて， 後から同一ユーザ ID でシステムにログインした場合は，前者のセッションを破棄し後者のセッションを有効とする．

本システムは，ロードバランサを用いて AP サーバ 2 台で HTTP トランザクションの負荷分散をおこなう．セッション管理では，IIS のセッション ID と AP サーバ固有のサーバ ID とを組み合わせた値をキーとし，ワークテーブルに保存する．セキュリティの観点から SSL を利用し，またセッションの HTML への埋め込みはおこなわず，受け渡しは引数によりおこなう．

4.8 セキュリティ

本システムは FEDI システムであり，ユーザの口座情報などの機密情報を管理，操作している．また，銀行に対して直接資金移動の依頼をおこなうため，誤操作によるデータ不整合を起こさないようにしなければならない．そのため，ハードウェアやネットワークでは，Kiban@asaban のセキュリティポリシーに準拠し FireWall の設置や SSL の利用，また，Gateway サーバ上での使用プロトコルの制限 （HTTPS のみ）をおこなう．ソフトウェア面では，アプリケーションで次のセキュリティ要件を実現する．

1） ユーザ管理

・認証

システムログイン時に認証をおこなう．パスワードに関してはマスク処理をおこなう．

・操作権限

階層的にユーザを管理し，ユーザ単位に操作権限を付与する．システムに

ログインしたユーザ ID から操作権限を取得し，操作可能なメニューのみ表示する．

・実行権限

ユーザには，操作権限の他に実行権限を付与する．ユーザに操作権限を付与していても，最終的な更新は，実行権限を付与していなければおこなえない．

2） 操作ログ

ユーザがおこなった操作を管理し，企業の管理者やシステム管理者が閲覧できる．

3） ASP 間のデータ維持

本システムでは，異なる ASP 間の情報の維持はデータベースのワークテーブル上で保持する．セッション変数や Cookie を用いたデータの維持はおこなわない．

セッション変数を使用した場合は，サーバのメモリ上で管理するため，大量のデータを処理した場合などに信頼性の問題がある．また，Cookie の場合だと Cookie に関する知識があるユーザは，その内容を見ることができ，クライアントの設定で Cookie を無効にすると，クライアントデータの管理がおこなえない．ゆえに，データの取り出しや格納に関してはビジネス COM でおこない，ASP から直接取り出すことはおこなわない．

4）ツールバー，ロケーションバーの非表示

ブラウザの「戻る」ボタンでの操作や URL を直接入力されることを避けるため，オペレーションをおこなうウィンドウではツールバーおよびロケーションバーを非表示にする．デフォルトページの home.htm で「メンバー入り口」のハイパーリンクをクリックすると新しいウィンドウを開き，フルスクリーンで表示する．その際に，ツールバーとロケーションバーを表示しないようにパラメータを設定する．また，すでにオペレーションをおこなうウィンドウを開いている場合，二重起動を避けるために JavaScript でアラートダイアログを表示する．

使用するブラウザが将来的にも Internet Explorer に限定している場合は，ActiveX を使用しツールバー，ロケーションバーを非表示にすることも可能である．

5）フレーム表示制御

本システムでは，画面を三つの区画（フレーム）に分ける．当日日付などシステムの基本的な情報を表示するフレームと，常に別のメニューを選択できるようにメニューを表示するフレームと，画面が遷移するメインのフレームになる．メインのフレームに表示する画面に関しては，不正アクセスを防止するためにフレーム名を取得し，不正な画面表示をおこなっていないかのチェックをおこなう．適切なフレーム名でない場合は，画面フローエラー画面に遷移する．

6）DB のデータ隠蔽

ユーザのパスワードなど機密性のある情報に関しては，Crypt モジュールで暗号化をおこない DB へ格納する．

5. おわりに

本システムに求められた要件は，短期間の開発および本番稼働，決済サービスを担える安全かつ高信頼性システムの構築，変化が激しいEビジネスに対応可能な拡張性や汎用性の高いシステム構築である．基盤設計および構築・運用では，Kiban@asabanを適用することで，また，業務アプリケーションの開発では，LUCINAにもとづきフレームワークを定義し業務アプリケーションに適用することで，予定どおり本番を迎えることができた．特に，実績があり評価済みのKiban@asabanを適用することで，早期に基盤を構築できたことが，非常に有効であった．

本稿で紹介したKiban@asabanおよびLUCINAとLUCINAにもとづいたフレームワークを利用することで，同様の基盤構成およびアプリケーション構成の他システムで，再利用が可能である．

最後に，本稿が同様の基盤構成およびアプリケーション構成でのシステム開発で，広く利用されることを期待する．また，本稿の執筆にあたり，Eサービスシステム部の方々に感謝する．

＊1　金融決済に関する情報を企業間で電子的に交換する仕組み（Financial Electronic Data Interchange）
＊2　内部にデータを保持しないオブジェクト

参考文献　［1］　保科剛，ASPサービス基盤「Kiban@asaban」，ユニシス技報，通巻68　Vol. 20　No.4，2001年3月，pp.135−143
［2］　LUCINA for Windows Type I
［3］　高橋令子，IT最前線システム開発技術の動向(15)，短期・高品質システム開発に向けてアプリケーション・フレームワークをデザイン・適用した「企業間電子資金決済システム」開発事例，ユニシスニュース，第486号，2001年10月，pp.10

執筆者紹介　藤　井　伸　行（Nobuyuki Fujii）
1966年生．1988年東京理科大学理学部数学科卒業．同年日本ユニシス㈱入社．証券業務アプリケーション開発，金融機関営業店端末システム開発，Webアプリケーション開発に従事．現在，Eサービスシステム部インテグレーションサービス室に所属．

小　川　昭　彦（Akihiko Ogawa）
1968年生．1991年北星学園大学経済学部経営情報学科卒業．同年日本ユニシス㈱入社．プロダクト開発，業務基盤開発，ミドルウェア開発，業務アプリケーション開発に従事．現在，Eサービスシステム部インテグレーションサービス室に所属．