MIGARO. TECHNICAL REPORT

ミガロ.テクニカルレポート

No.2
2009年秋

株式会社ミガロ.

CONTENTS 目次

ごあいさつ 01

Migaro.Technical Award 2009 お客様受賞論文／ミガロ.テクニカルアワード

最優秀賞 **JACi400 で既存 Web サービスの内製化を実現** 04
佐々木 仁志様●株式会社ジャストオートリーシング

ゴールド賞 **.NET 環境での Delphi/400 の活用** 08
福田 祐之様●林兼コンピューター株式会社

シルバー賞 **5250 で動作する「中古車 在庫照会プログラム」の GUI 化** 13
佐久間 雄様●株式会社ケーユー

優秀賞 **Delphi による 輸入システム「MISYS」の再構築** 18
秦 榮禧様●株式会社モトックス

Delphi/400 による 物流システムの再構築 23
仲井 学様●西川リビング株式会社

Delphi/400 で開発し3台のオフコンを1台の IBM i へ統合 26
島根 英行様●シルフ

Migaro.Technical Report 2009 ミガロ.SE 論文／ミガロ.テクニカルレポート

JACi400 **JACi400 環境でマッシュアップ!** 32
岩田 真和● RAD事業部 技術支援課

Delphi/400 **Delphi/400 を利用したはじめての Web 開発** 41
福岡 浩行●システム事業部 システム２課

Delphi/400 を使用した Web サービスアプリケーション 50
尾崎 浩司●システム事業部 システム３課

Delphi/400 によるネイティブ資産の応用活用 56
吉原 泰介● RAD事業部 技術支援課 顧客サポート

RPG **RPG でパフォーマンスを制御** 63
松尾 悦郎●システム事業部 システム１課

MKS Integrity **MKS Integrity を利用したシステム開発** 67
宮坂 優大　田村 洋一郎●システム事業部 システム１課

ごあいさつ

いつもミガロ.製品をご愛用いただき誠にありがとうございます。

さて、昨年2008年10月に“ミガロ.製品をご利用のお客様に、より多くの技術情報の発信を行う”ことを目的に技術論文集『Migaro.Technical Report』を創刊いたしました。お陰さまで、この論文集は多くのお客様に非常に高いご評価をいただくことができました。たいへん感謝をしております。

そして、今回『Migaro.Technical Report』の第2号を無事に発刊する運びとなりました。創刊号と同じく2部構成になっています。

第1部は「Migaro.Technical Award 2009 お客様受賞論文」を掲載し、第2部は「ミガロ.SE論文」を掲載しています。

「Migaro.Technical Award」とは、日々アプリケーション開発・保守に携わるエンジニアの方々の努力と創意工夫の効果を顕彰することを目的とし、「Delphi/400」「JACi400」「MKS Integrity」などの弊社製品をご利用中のユーザー様を対象に実践レポート（論文）を公募し、厳正な審査・選考のうえ表彰する制度です。

また「ミガロ.SE論文」は、弊社SEによるアプリケーション開発・保守のノウハウやハウツー、Tipsなどの技術論文を掲載しております。システム開発やサポートからの実践的な技術情報が少しでも皆様の開発・保守のお役に立てれば幸いです。

最後に『Migaro.Technical Report』第2号を発刊するにあたりまして、多くのお客様・パートナー様にご支援、ご協力いただきましたことをこの場をお借りして、あらためて厚く御礼申し上げます。

2009年 秋

株式会社ミガロ.

Migaro.Technical Report

Migaro. Technical Award 2009

お客様受賞論文／ミガロ. テクニカルアワード

JACi400で 既存Webサービスの内製化を実現 ――Webサービス「J-line」の再構築

佐々木 仁志 様
株式会社ジャストオートリーシング
営業企画部 システム課

株式会社ジャストオートリーシング
http://www.justauto.co.jp/

神奈川県、東京都南西部を営業基盤とする独立系オートリース専門会社。神奈川県内で最大規模の自動車整備工場を運営し、自動車リースを中心に、車検や新車 / 中古車の販売、損害保険など、自動車に関する総合的なサービスの提供をコンセプトに日々、お客様に対応している。

ソリューション導入の経緯 ―Web サービス「J-line」の再構築

ジャストオートリーシングが提供するWeb サービス「J-line」は、2001 年 3 月より、リース車両の契約情報や整備履歴、事故履歴、保険契約情報などリースにかかわる情報を、オンラインでお客様に提供してきた。

導入後、いくどか修正案件も発生した。だが、システムが Java で開発されたものであるため、RPG しか手掛けたことのない当社の開発要員では対応ができず、大きな修正案件があった際に、まとめて外注に依頼せざるを得なかった。このため、メンテナンスの頻度も少なく、顧客の要望に迅速に対応できない状態だった。

また、導入から 7 年以上も経て、ハードの老朽化も著しくなった。しかし、その入れ替えで発生する Java プログラムの移植には、システム構築時とほぼ同額の投資が必要となることが判明。さらに、追い打ちをかけるようにリース会計制度の改定など、早急に対応しなければならない問題が次々と発生した。

そこで今回、自社開発が可能な新たなWeb 開発ツールを導入し、「J-line」の再構築を行うこととなった。【図 1】

JACi400の選定理由

まず、新たな Web 開発ツールの導入にあたって、過去の反省からいくつか条件を設けた。

①社内開発要員で開発・保守の実現
Web 開発の専任を設けず、誰でも対応が可能な体制作りを実現する。

②既存と同等か、それ以上の機能の実現
セキュリティの必須要件 SSL を導入する。ファイルのダウンロード・メール送信機能を実現する。

③開発工数の削減
シンプルな開発手順。今ある資源をできるだけ生かし、開発工数の削減を図る。

④ハードやソフトの入れ替わりに伴う、システム再開発のリスクを低減

⑤拡張性の高い商品の導入
社内システムの Web 化が可能か？ また、他社の Web サービスと連動が可能か？

⑥サポートの安心感

これらの条件を満たすことを前提に、さまざまな開発ツールを精査した結果、「JACi400」がその候補にあがった。

JACi400 は、最低限の HTML の知識と RPG の開発経験さえあれば、誰でも Web アプリケーションの開発が可能となるツールである。画面属性の定義や RPG のスケルトンプログラムを自動生成する JACi400 Designer も洗練されており、その操作も非常に容易だ。

さらに、全体のシステム構成が非常にシンプルなので、既存の Java のシステムと比べて、大幅に開発工数の削減を見込めた。【図 2】

例えば機能追加 / 拡張を例にとると、既存の Java システムの場合は、CLP で CSV 変換プログラムの開発や FTP の設定作業、スクリプトプログラムの修正、

図1　J-line

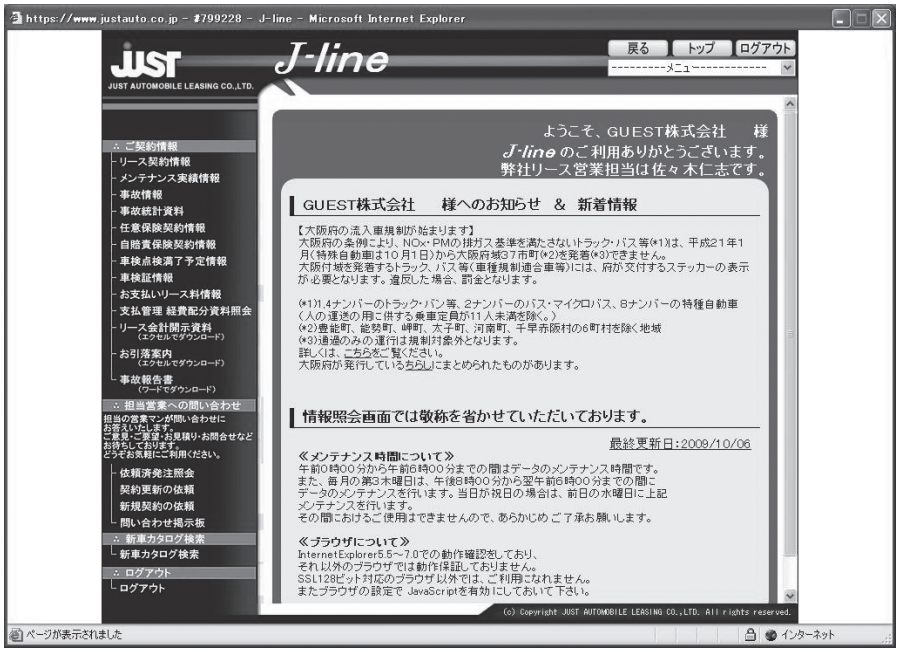


図2　JACi400のシステム図

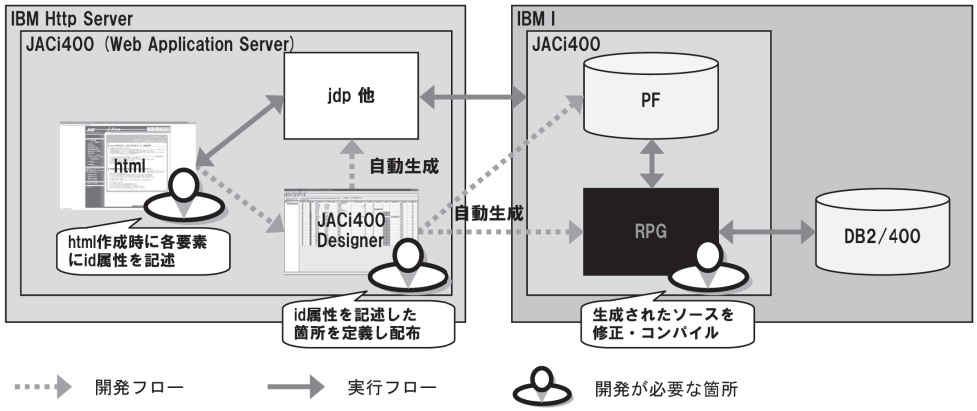
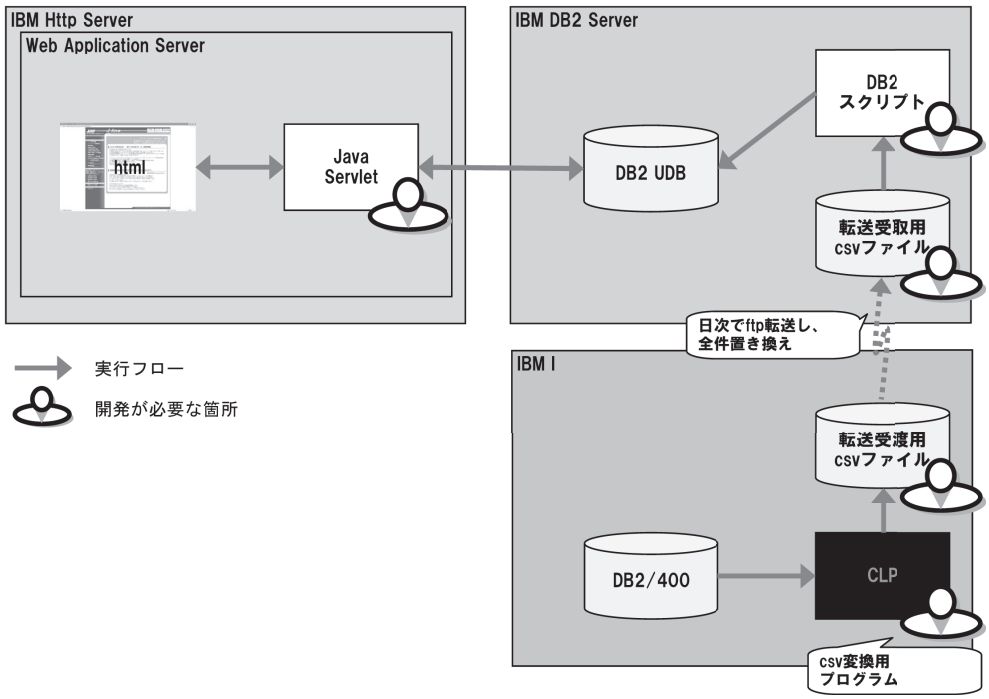


図3　旧来のシステム図



Java での開発、HTML の修正など、多くの工程を踏まなければならなかった。【図 3】

ところが JACi400 では、HTML の修正、JACi400 Designer で画面属性を定義、そして画面に表示するプログラムロジック部分を RPG で開発するだけで、同等の機能が実現可能であった。【図 2】

既存の 5250 画面を Web 化する優れた製品は数多くあると思う。だが、今回はベースとなる HTML やデータベースは完成しており、既存の Web サービスの再構築、および前述の Web 開発ツールの導入の条件を踏まえると、他の製品より優位性があると考え、JACi400 の導入を決定した。

問題点と解決手法

今回、実際の開発はミガロ．が中心となり、筆者もいくつかの画面や機能を追加した。そこで発生した問題とその解決手法は、JACi400 で開発する際の参考になると思われるので、簡単に記したい。

●ファイルのダウンロード機能・メール送信機能の実装～ Delphi/400 と VB-Report との連動で実現～

JACi400 には、ファイルのダウンロード機能が存在しない。このため、今回は JACi400 でファイルの元データまで作成。ファイルの作成～配布の処理については、Delphi/400 と VB-Report を利用し実現した。（※ 1）

● IBM i へのログオンの問題～オートログオン機能を使用し、デフォルトのログオン画面をキックする手法を採用～

JACi400 は、定型の IBM i へのログオン画面とメニュー画面を使用しなければならない。

このログオン画面は、IBM i ログオン時のユーザープロファイルを特定するための機能である。しかし、今回は、当社が独自に作った顧客判別用のログイン画面とメニューを利用したかったため、この機能は省きたかった。

そこで「J-line」起動時に、ログオン画面とメニュー画面を裏で起動。ユーザープロファイルに JavaScript 内で固定の値を入力し、自動で IBM i にログオン後、こちらが用意した独自のログイン画面を呼び出すというやり方で、この問題を回避した。（※ 2）

●ログイン履歴管理の問題～ジョブ番号と IP アドレスをお客様コードと関連づけ DB に保管し、お客様のログイン状況の監視を実現～

JACi400 は、HTML 上にフィールドを追加・変更した場合に、必ず Web Application Server（WAS）の再起動が求められる。このため、リリース時にはログイン状況を把握することが不可欠であった。

JACi400 ではログオン後、IBM i のジョブ番号がブラウザのウィンドウに表示される。この機能を利用し、顧客がログインを試みた際やログアウトの際に、ジョブ番号、IP アドレス、お客様コード、パスワード、時刻などの情報をデータベースに作成することで、ログイン状況の把握を実現した。【図 4】

●パフォーマンスの改善～ WF の作成タイミングを分割し改善～

IBM i と WAS 間の通信量が増えると、そのぶんどうしてもパフォーマンスが悪くなった。特に一覧画面はデータ量が格段に増えるため、上記問題が顕著となった。

今回の画面遷移は［メニュー画面］→［検索画面］→［一覧画面］→［照会画面］となっている。そこで検索画面に遷移時に、一覧画面の元ファイルを作成。さらに一覧の表示件数の初期値を 20 件にすることで、表示負荷を軽減させた。

●ブラウザのツールバーが表示されない問題～プルダウンメニューや［戻る］ボタンを画面に配置し改善～

ブラウザの［戻る］機能を、IBM i に持たせることはとても難しい。そのため、今回は CALL された上位画面に遷移するという機能を［戻る］ボタンに実装させた。ただ、これでは厳密な意味で［戻る］にはなっていないため、今後の課題となった。

※ 1 「i Magazine 11 号」では、ダウンロード機能が秋頃サポート予定との記述があった。

※ 2 実は、開発中に JACi400 のバージョンアップがあり、この処理が可能となった。

新「J-line」稼働後の効果

新「J-line」に関する顧客の利用件数については、まだ以前との差はさほどない。だが、画面の追加や機能修正は随時行っており、自社開発・保守の実現の部分において、着実に成果をあげている。

また日次で行っていた、DB2/400 側から DB2 UDB 側に FTP 転送する処理がなくなった。直接 IBM i とデータとやりとりをすることになったため、3 時間程度、処理時間が短縮されたことも成果といえる。

JACi400 の評価と「J-line」の今後

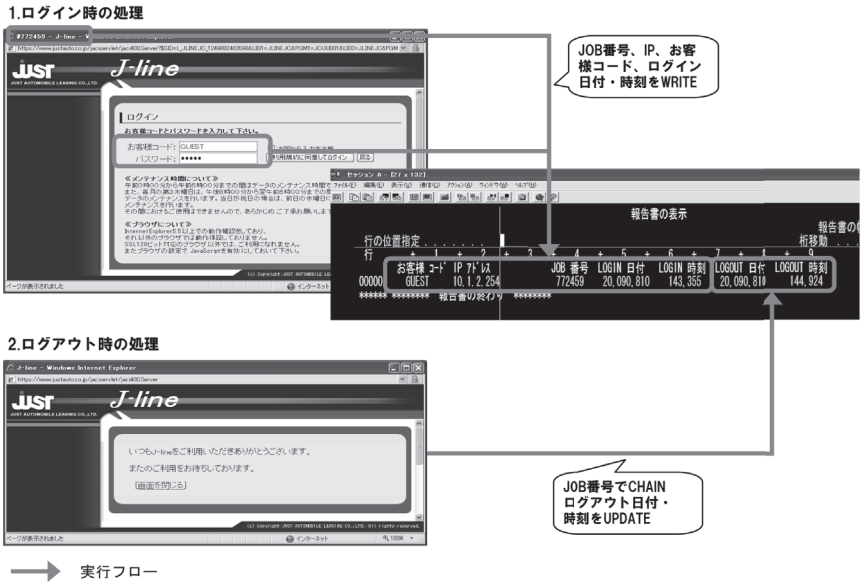
JACi400 は、Web サービスの新規開発においてはすでに述べた通り、非常に便利なツールである。今回、足りない機能は Delphi/400 に頼った感は否めないが、通常の Web サービスの開発には何ら支障はない。

また、社内システムの Web 化においても、既存の 5250 画面の制約にしばられない画面作りが可能。既存のプログラムと共通するロジックをモジュール化することで、資源の一元化や保守にかかる工数の削減も見込める。

今後当社は、今回再構築した「J-line」に所有車や他社リース車の管理機能の追加、車両運行管理システムの追加、データのダウンロード機能強化などを予定。また営業支援システムなど、社内システムの Web 化も計画している。

M

図4　ログイン履歴の管理



.NET環境でのDelphi/400の活用
――RPG開発した販売管理システムのGUI化

福田 祐之 様
林兼コンピューター株式会社
ソリューション営業部 企画課 係長

林兼コンピューター株式会社
http://www.hcc-com.co.jp/

林兼コンピューターはIBMソリューション・プロバイダーとして、山口県と福岡県を中心に業務アプリケーション構築からネットワーク構築、IT機器販売までサポート。お客様のビジネスをITの面から支援している。

Delphi/400 採用の経緯

近年、AS400ユーザーから5250画面のGUI化要望が多くなり、林兼コンピューターとしてもユーザーの要望に応えるべく検討を開始。その最中、既存の5250システムをオープン化したいという案件が確定し、開発計画を早急に確定する必要に迫られた。

【開発案】

① Delphi習得の時間が足りないため、VB.NETでGUI化を行う。
②複雑なロジックは、RPGを使用したい。
③データベース(DB)には、AS400(DB2/400)を使用したい。

AS400をDBサーバーと位置づけ、フロント画面の開発を行う言語をVB.NETとした。これは、開発スケジュールを照らし合わせてDelphiを習得する時間が足りないと判断したためであり、社内のVB技術者による開発となった。

解決すべき問題として、RPGプログラムの起動という問題があった。「VBからDelphi/400のコンポーネントを使用できないか」との質問をミガロ．に相談したところ、AS400とのコネクションをDLL化することでVBとの連携が可能、と判明した。

その結果、AS400 + Delphi/400でフロントVB.NET、という環境で開発を行うことが確定した。【図1】

Delphi/400 の DLL 開発

Delphi/400のDLL開発については、短期間での開発に迫られていたため、ミガロ．協力のもと開発を行った。

以下の④～⑥を繰り返し、ブラッシュアップを図りながら開発を行った。

【開発の流れ】

①打ち合わせの実施
②仕様作成
③DLL作成
④実装テスト
⑤改善検討
⑥DLL改修
⑦開発への実装

できあがったDLLの概要は、図2の通りである。RPG呼出DLL(Delphi/400)のポイントは、次の3つとなった。【図2】

【DLLの特長】

①パラメータを動的に変更し、汎用的にAS400側プログラムの呼出が可能【図3】
② DLLが起動したプログラムの終了を待つAS400とVBの同期呼出【図4】
③ Delphi/400の機能によりジョブID取得が可能となり、セッション管理が可能【図5】

メタフレームの利用 (Citrix XenApp)

Delphi/400 + VB.NETでの開発を行うにあたり、アプリケーションの配布と、

図1　Delphi/400を利用したシステム構成図

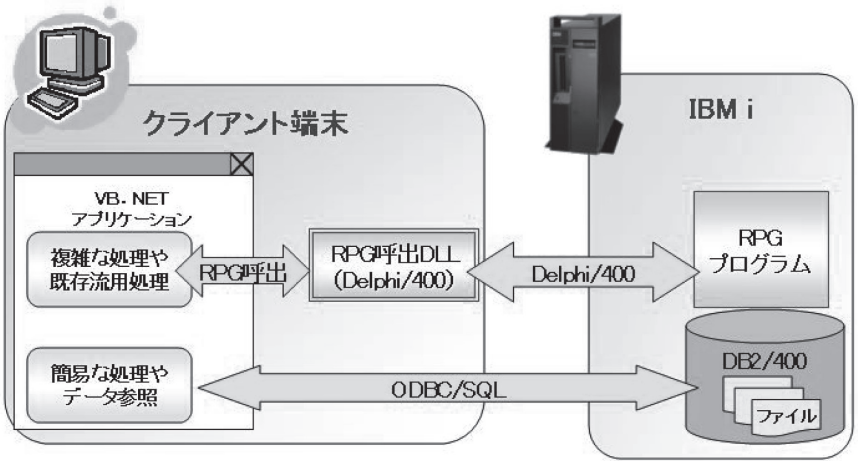


図2　DLLの機能詳細

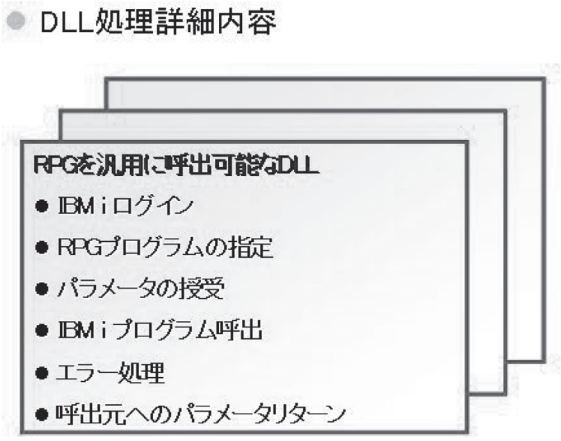
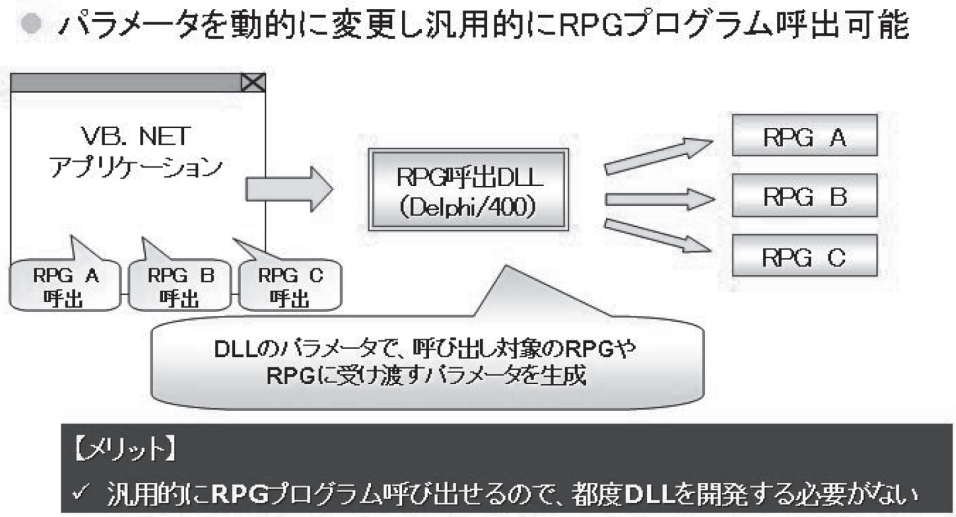


図3　RPG呼出DLL（Delphi/400）のポイント①



クライアントPCのスペックによる実行レスポンスの格差が問題となった。

【問題点】
- プログラム変更に伴う、アプリケーションの配布作業。
- クライアントPCの変更やトラブルに伴う、アプリケーションの再配布。
- 実行レスポンスは、クライアントPCのスペックに依存してしまう。

このような問題を解決するために、メタフレーム「Citrix XenApp」を採用し、サーバーによる集中管理で、クライアントPCに対する運用管理コストの削減を行った。

図6のように、アプリケーションはXenAppサーバー上で一括管理され、各クライアントはサーバー上のアプリケーションを使用する。結果の画像のみがクライアントに配信されるので、実行レスポンスが改善される。【図6】

また、アプリケーション、BDEの導入もサーバーのみでよいので導入作業にかかる負担も少なくてすむ。

【導入効果】
- 運用管理コストの削減
- アプリケーションの保全
- 実行レスポンスの改善

アプリケーション開発—RPG開発した販売管理システムのGUI化

提案活動

システム再構築の提案を行う際には、予算の関係等で全体の再構築が難しい場合がある。しかし、既存のRPGをオープン側からも使用できると、連動システムを作成することなく、既存のRPGシステムとの共存が可能となる。

これにより、お客様へのコストメリットのある提案が実現した。

開発体制

もともとオープン系の開発は、VB.NETですべてのロジックをコーディングしていたため、あまり生産性のよい開発ではなかった。しかし、Delphi/400の導入により、ストアドプロシージャの代わりにRPG、CLでコーティングすることができ、開発工数を大幅に削減することができた。

それと同時に、RPG技術者をプロジェクトに投入できるようになり、社内リソースの有効活用にもつながった。

開発事例：RPG開発した販売管理システムのGUI化

図7と図8の通り、5250画面で運用していた販売管理システムを、VB.NET＋Delphi/400でGUI化を行った。【図7】【図8】

5250画面は制限（80文字×24行）があるため、項目数の多い画面を作成する場合は複数の画面に分かれることとなる。オープン系の画面と比べると、非常に入力しづらい画面であった。【図9】

画面の切り替えを行わなくても、1画面に多くの情報を組み込めるようになったことがGUI化の大きな利点である。【図10】

【GUI化の開発方針】

①複雑なチェックロジックは、RPGで処理を行う。

②コンボBOXやラジオボタンを使用することで、各種コードの選択を容易に行えるようにする。

③キーボードだけの入力も行えるように、ファンクションキーやショートカットキーの設定を行う。

④ユーザー制御を組み込む。（色、初期表示項目など）

RPGの活用に関しては、印刷関連、更新関連は既存の資産が利用可能だったため、DLLからのパラメータ部分とテーブル名の書き換え程度で移行することができた。

既存資産を利用したことで開発工数の削減ができたこと、実績のあるロジックを使用することでテスト工数の削減もできたこと、これらが大きなメリットとなった。

導入後のユーザー評価

GUI化された販売管理システムについて、導入直後は操作性の違いに戸惑われた様子だった。

運用していく中で操作に慣れてくると、今まで複数画面に分かれていた情報が1画面で確認できることや、照会画面からの情報をExcelに出力可能な点などで、評価をいただくことができた。

Delphi/400の効果と影響

VB.NETとRPGを連携させる効果として、以下のようなことがあげられる。

①開発工数の削減

新規に開発するロジックは、処理内容に応じてVBとRPGの使い分けを行うことで工数を削減できる。再構築の場合は、既存資産を活用することで工数を削減できる。

②社内リソースの有効活用

オープン開発でもRPG技術者が参画できるので、社内リソースをフル活用することが可能となる。

③AS400のGUI化提案

RPG資産が活用できることにより、既存ユーザーに対してGUI化提案が容易になる。

また、オープン開発を行っていく中で、RPG技術者にSQLのスキルが加わったことが社内スキルの向上につながった。

今後の開発方向としては、AS400＋Delphi/400を有効活用したオープンシステムの提案をさらに行っていく。それと同時に、社内スキルの向上を図り、お客様に満足していただけるユーザーインタフェースの構築を目指す。

また、技術・ノウハウの蓄積を行うことで、生産性・品質の向上に努めていくことで顧客満足度の向上を図る。

M

図4　RPG呼出DLL（Delphi/400）のポイント②

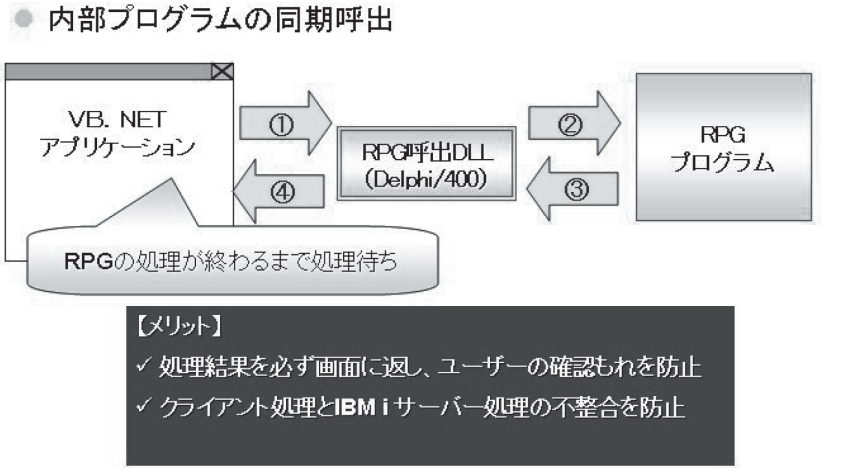


図5　RPG呼出DLL（Delphi/400）のポイント③

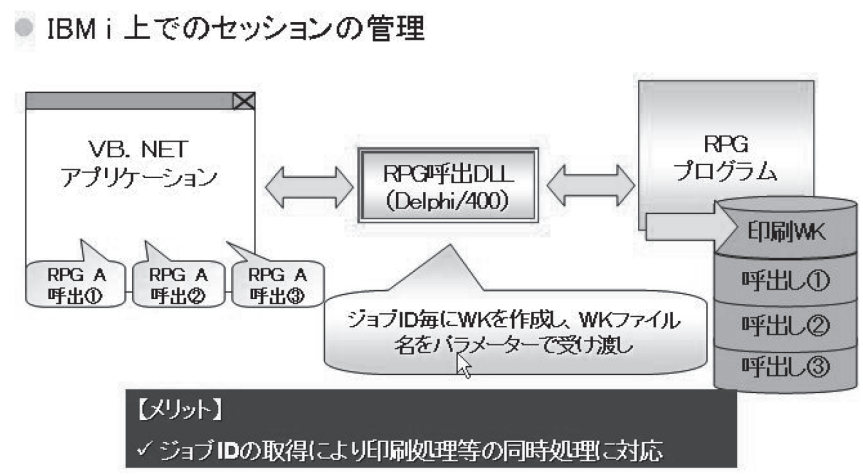


図6　メタフレーム構成図

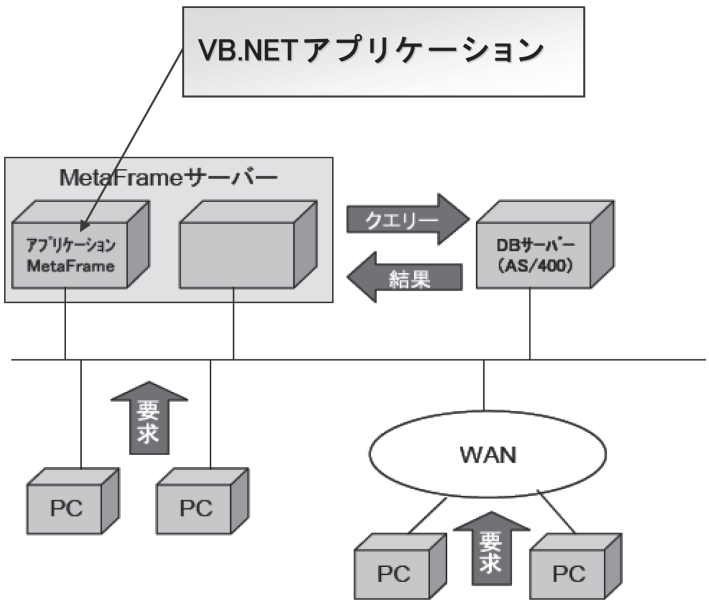


図7　販売管理システム（5250画面）

図8　販売管理システム（GUI）

図9　5250画面の制約（受注の例）

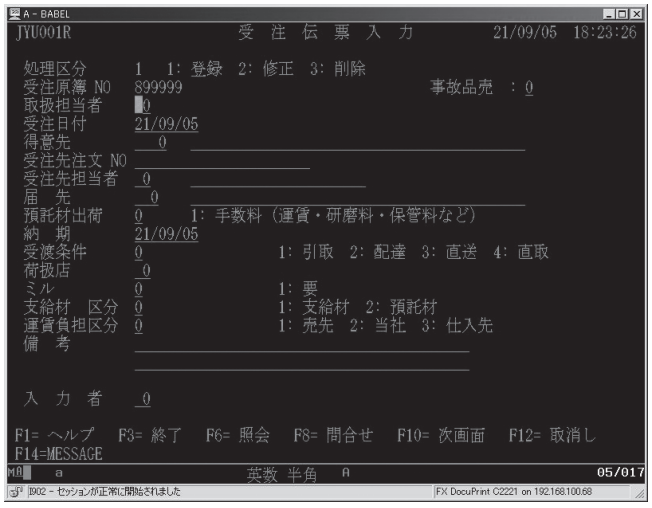
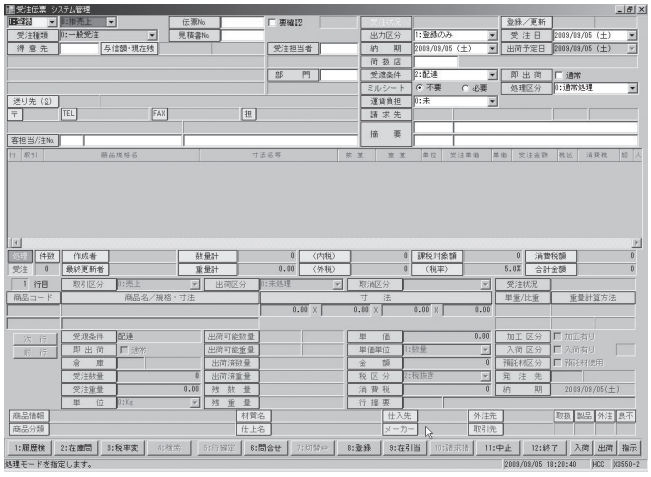
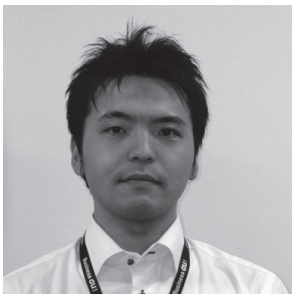


図10　GUI開発方針（受注の例）



5250で動作する「中古車 在庫照会プログラム」のGUI化 ――RPGを知らなくとも その流れをDelphi/400で再現

佐久間 雄 様
株式会社ケーユー
管理本部 システム担当



株式会社ケーユー
http://www.keiyu.co.jp/

ケーユーは、東名高速横浜町田インターそばの本店を中心に東京、神奈川、千葉、埼玉、栃木へとネットワークを拡大している。トータルディーラーとしての強みは、お客様の多彩なニーズに応えられるということ。

1.開発経緯 ―もっと使えるアプリケーションを

ケーユーは、ケーユーホールディングスのグループ会社の中において核となる会社で、17 拠点で月間約 1500 台の中古車を販売している。

特に、個人のお客様へ中古車を販売するという業務形態上、お客様への迅速かつ正確な対応がたいへん重要である。

運用部門からは、顧客サービスの充実のために、お客様の来店時に即座に情報照会できるよう、もっと見やすく使いやすいアプリケーションを提供してほしいとの要望がきていた。

2.照会プログラムのGUI化

Delphi/400 を使用する前は、簡単な HP を作成したことがある程度で、プログラムはまったく未経験であった。

最初に、導入支援プログラムで基礎的な手法のレクチャーを受け、その知識をもとに、従来 5250 で動作している照会プログラムの GUI 化を行った。

在庫検索の GUI 化

販売可能な車両の在庫情報に関する検索である。5250 画面では、在庫検索から詳細に至るまでは 3 画面を必要とし、ユーザーからは「コードがわからない」「見づらい」「抽出条件の追加」等の意見や要望が多かった。

そこで、検索から詳細までを 2 画面とし、マウスのクリック操作でコード一覧の表示を行えるようにして、ユーザーの手間を省いた。

また、検索結果の並び順等も指定できるようにし、詳細はユーザーが見慣れている車検証のレイアウトに合わせた画面とした。【図 1-1】【図 1-2】

開発上苦労した点は、1 画面で数多くの情報を公開すること。現場との打ち合わせを繰り返しながら、必要な情報だけをピックアップし、車検証と同じ並びのレイアウトを実現した。

3.30 本のプログラム開発

その後、社内開発の利便性を生かし、ユーザーの要望に合わせ、約 30 本のプログラムを開発している。主なプログラムは、以下の通りである。

①中古車情報誌「GOO」への掲載情報の作成（EXCEL 出力）

従来は、EXCEL のフォーマットを紙に出力し、手書きしていた。これを IBM i のデータを、EXCEL OLE オートメーションを利用し、所定レイアウトの EXCEL に出力できるように開発した。ユーザーの作業工数の削減、手書きのため FAX 送信すると見づらいなどの問題も解決された。

金額や日付の書式を Delphi で計算させ、EXCEL に書き込ませるようにした。そのため、所定のレイアウト書式を忠実に再現する、という点に一番苦労した。【図 2】

②営業目標（EXCEL データ）のアップロード

従来、EXCEL で作成した営業目標は、IBM i へは 5250 で手入力していた。その際、1 レコードごとに入力していくため、入力時間が 30 分程度かかってしまっていた。エクセルにあるデータを一括登録できないかとの依頼を受け、開発に踏み切った。

プログラムを起動し、GRID にカーソルを合わせると、EXCEL で作成した目標が読み込まれる。ボタンをクリック後、IBM i のデータを新規作成、更新する機能を持たせている。入力時間を必要とせず、処理時間が 1 分以内で完了できるようにした。

開発上苦労した点は、エクセルのデータを何に読み込ませて、IBM i にアップロードするかということであった。Memo 等ではセルの区切りがなくなってしまうので、StringGrid を使用することにした。SelectCell イベントで EXCEL データを読み込ませる方法を見つけ、対応するフィールドにアップロードさせることができた。【図 3-1】【図 3-2】

③外部 Web サイト掲載用データの自動更新

「Goo-NET」など中古車情報サイトの在庫車両の情報を、日次で自動更新するための機能である。

IBM i に登録されている現在在庫を抽出し、CSV ファイルに出力。現在在庫で画像が登録されているものは、画像を抽出。CSV ファイルと画像データを FTP サーバーにアップロードする。処理が完了すると、管理者へ処理結果がメールで配信される。なお、この処理は画像があるため、大量のデータを処理することになる。

従来の VB で開発していたプログラムを、Delphi/400 で再作成した。これにより、サーバーの負荷が減少し、メモリ不足によるアプリケーションの停止が発生しなくなった。【図 4-1】【図 4-2】

開発上苦労した点としては、対応する画像枚数の指定や Mail 送信、FTP 情報の変更を可能にする方法などであった。これ関しては ini ファイルに情報を持たせることによって対応可能にした。また、CSV ファイルの保存方法に関しては、「ミガロ．テクニカルサポート」へ相談して解決した。

④中古受注済未引当データメンテナンス

受注入力後、在庫データがうまく反映されないと、在庫が引き当てられない。従来までは、未引当となってしまったデータを Query で抽出し、毎朝手作業で修正を行っていた。

これらに対して、Delphi/400 で、未引当データの抽出・更新処理・処理結果のメール送信という解決策を選択し、自動化させることに成功した。作業の自動化による工数の削減、手作業による間違いの防止などに役立っている。【図 5-1】【図 5-2】

開発上苦労した点としては、Mail 送信の際にログファイルを添付して送信しようとしたこと。しかし、インストール済みの Indy10 では、メールに添付させるコンポーネントが見つからず、メールに記載して送信するよう修正した。

4.開発しながら習得

これらのプログラムを開発するにあたり、最初はわからないことだらけであった。しかし、Delphi/400 のユーザーは多く、Web 上にも多くの情報が載っているので、これを十分に活用することを心がけている。

また、ミガロ．のテクニカルサポートにいつも的確なサポートをしてもらうことで、徐々に開発のペースも向上してきた。

5.結果と今後への期待

開発を続けていると、IBM i にある基幹システムのデータの流れや、集計方法等が少しずつわかってくるようになってきた。RPG がまったくわからなくても、その流れを Delphi/400 で再現できるように、もっと腕を磨いていきたい。

現在、ユーザーのニーズにそった小回りの効く体制が確立されてきた。今後は、Intraweb を使用したアプリケーション開発を試み、クライアント環境のセットアップに手間がかからないようにしていきたい。

M

株式会社ケーユー（詳細）
ケーユーは、東名高速横浜町田インターそばの本店を中心に東京、神奈川、千葉、埼玉、栃木へとネットワークを拡大している。トータルディーラーとしての強みは、お客様の多彩なニーズに応えられるということ。
例えば新車から中古車、国産車から輸入車への乗り換え、あるいはその逆ケースもだが、ケーユーはメーカーを問わず「こういうクルマに乗りたい」という希望にスムーズに対応できる。下取りから、クルマの選択、さらに引き渡し後のアフターメンテナンスや保険手続きまで、お客様にとって満足度の高いカーライフのサポートを展開している。
（設立：平成 19 年 10 月 1 日。事業内容：各種自動車販売および修理業、損害保険代理業など）
http://www.keiyu.co.jp/

図1-1　在庫検索画面

図1-2　在庫詳細画面

図2　中古車情報誌GOOへの送信EXCEL

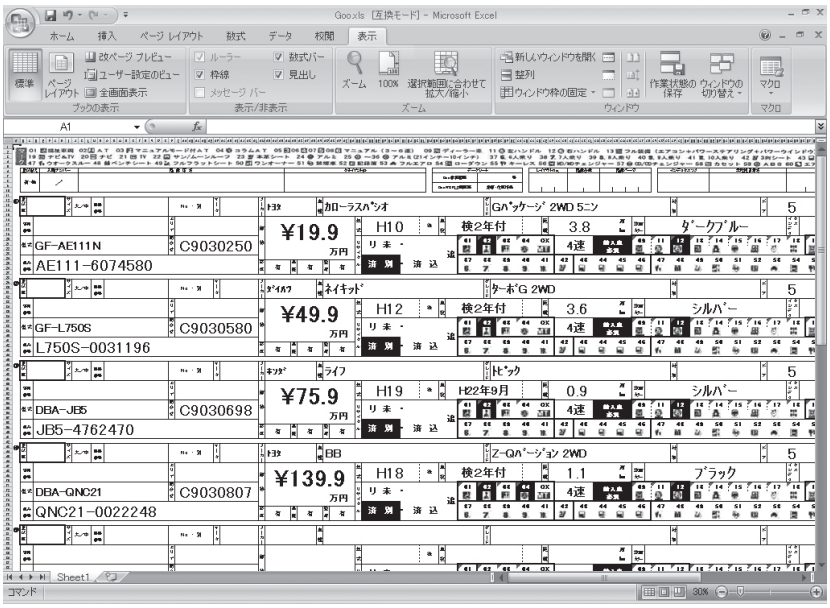


図3-1　目標入力5250画面

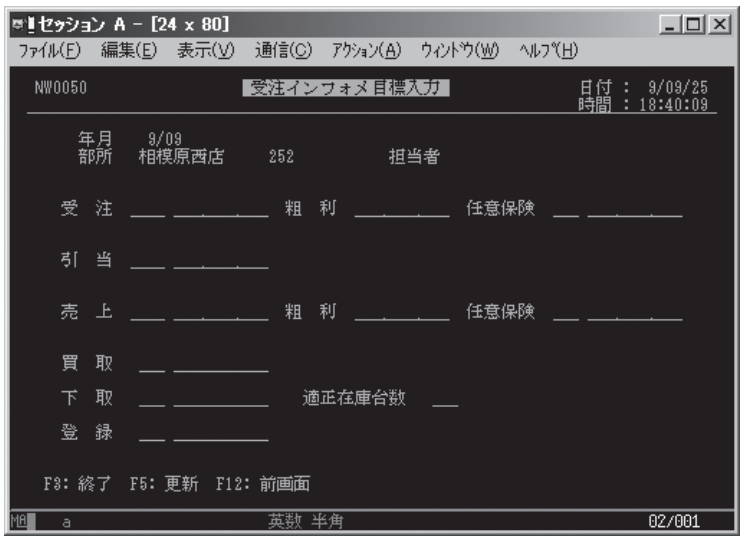
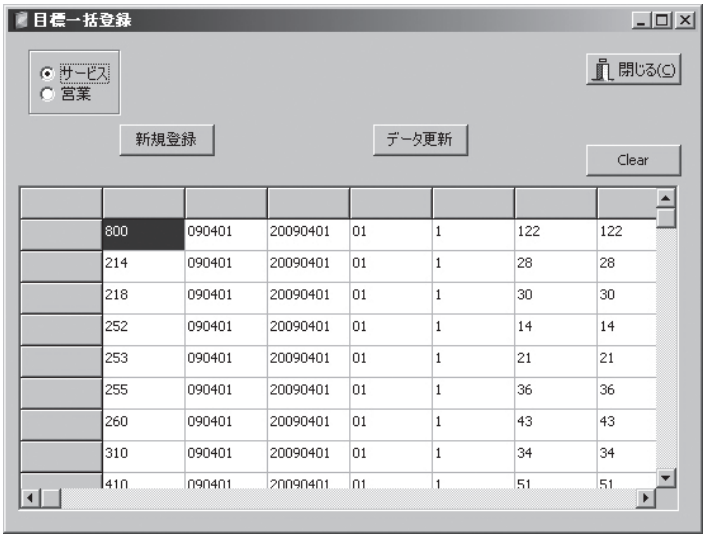
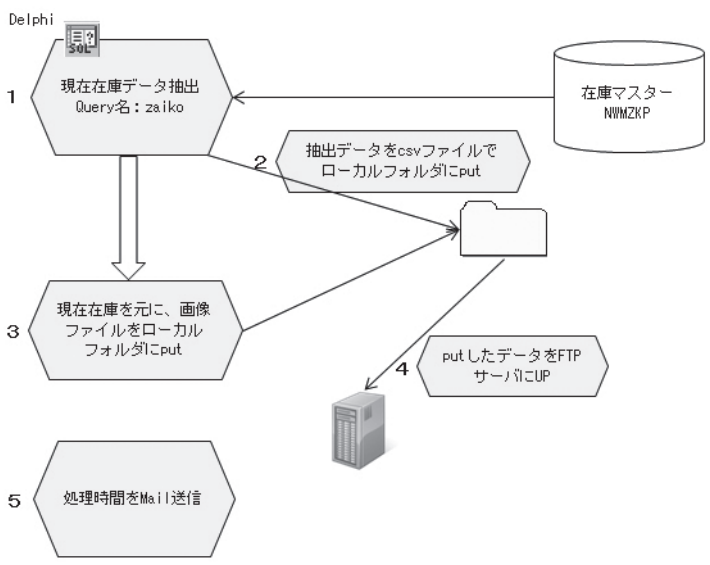


図3-2　目標一括登録Delphi/400



	800	090401	20090401	01	1	122	122
	214	090401	20090401	01	1	28	28
	218	090401	20090401	01	1	30	30
	252	090401	20090401	01	1	14	14
	253	090401	20090401	01	1	21	21
	255	090401	20090401	01	1	36	36
	260	090401	20090401	01	1	43	43
	310	090401	20090401	01	1	34	34
	410	090401	20090401	01	1	51	51

図4-1　Goonetデータアップ仕様



1．在庫マスター（NWMZKP）より現在在庫データを抽出する【抽出条件①】

2．抽出した現在在庫データをCSVファイルでローカルに保存

3．1で抽出した現在在庫データを元に画像データをローカルにコピーする

4．ローカルの特定フォルダにあるファイルを全てFTPサーバに転送

5．処理時間をメール送信

図4-2　Goonet結果Mail

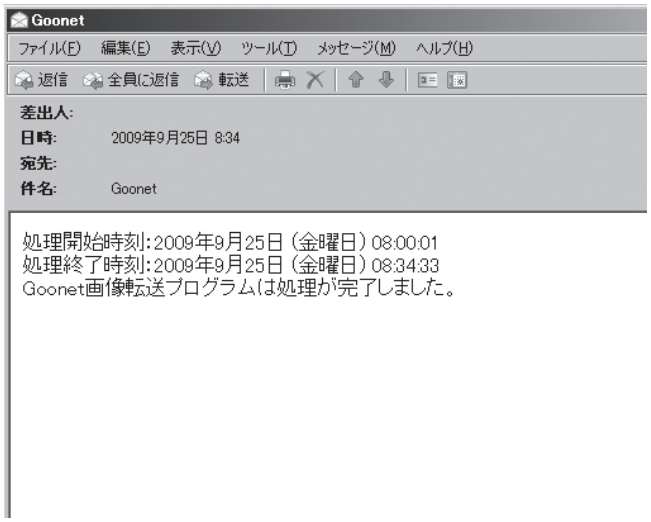
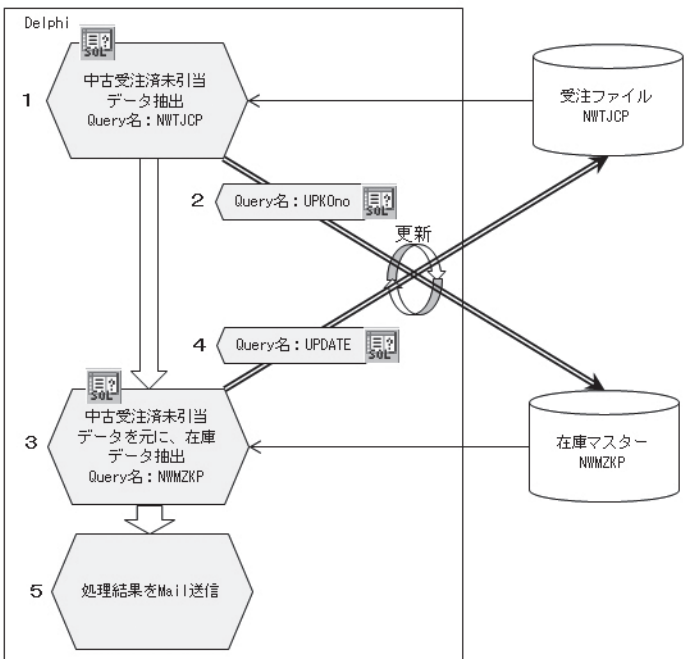


図5-1　中古受注未引当メンテナンス5250画面

図5-2　中古受注未引当メンテナンスプログラム



1．受注マスター（NWTJCP）より、中古受注済未引当データを抽出する【抽出条件①】
※1

2．抽出した受注データを元に在庫マスター(NWTJCP)を更新【更新条件①】

3．1で抽出した受注データを元に在庫データを抽出【抽出条件②】

4．3で抽出した在庫データを1で抽出した受注データに更新【更新条件②】

5．2と4で更新したレコードのkeyをメールに記載してメール送信【メール送信】

※1.抽出データが無ければその旨をメールに記載してメール送信

Delphiによる 輸入システム「MISYS」の再構築 ――システム再構築の4つの大きな狙い

秦 榮禧 様
株式会社モトックス
管理部 情報システムグループ グループ長

株式会社モトックス
http://www.mottox.co.jp/
http://www.mottox-wine.jp/

1915年「正真正銘、元手なし」という意味の「元なしや」の屋号で創業。小売業・卸売業を経て「モトックス」に社名変更。現在は、世界各国のワインと日本の北から南までの地酒・焼酎・泡盛を企画、開発、提案を行っている。

再構築の経緯

2000年にFilemakerにより「輸入システム」を構築、2008年のFilemaker保守サポートの期限切れにより、バージョンアップが必要となった。さらに、バージョンアップではプログラムは移行できず、再構築を行うことが前提となった。

これまでの輸入システムの問題点として、基幹システムとの連携はCSV出力による手動連携（双方向連携）であり、情報がリアルに把握できなかった。また、予約管理等はEXCELによる管理であり、柔軟ではあるが処理が煩雑であり、輸入システムを統合して構築することになった。さらに、業務の統合とともにDelphiで開発するにあたり、Filemakerの操作性を実装する要望も加わった。

【輸入システム「MISYS」の再構築】

2008.02 ～ 05　要件定義 / 基本設計
2008.05 ～ 10　詳細設計 / 開発開始
2008.10 ～ 12　テスト
2009.01　本稼働

システム再構築の4つの狙い

システム再構築にあたっては、当社は業務面で4つの大きな狙いがあり、それを今回の再構築で実施した。

①予約管理

- 運用
  予約金管理、予約残管理（発注管理）はExcelによる管理であった。
- 問題点
  予約管理がシステム化されていない。
- 解決策
  予約残の一元管理、予約金の管理、期中残高管理と予定金把握の導入。

②支払管理

- 運用
  予約残高管理、買掛勘定振替管理、輸入手続管理はExcelによる管理であった。
- 問題点
  予約管理がシステム化されていない。支払（商品・諸費用）管理がシステム化されていない。
- 解決策
  支払の一元管理、各種レート管理、諸費用の一元管理の導入。

③入港管理

- 運用
  発注・発注明細管理、検品期日管理、倉庫運輸業者連携はExcelおよびFilemakerによる管理であった。
- 問題点
  必要となるデータの一部のみの入力であった。
- 解決策
  輸入書類の到着管理、スケジューリング管理（検品期日）、仕入管理の導入。

④システム管理

- 運用
  マスタ・データ連携が日次になっている。在庫の把握が別システム（販売管理）になっている。
- 問題点
  輸入システムと販売管理システムが別々に稼働している。
- 解決策
  商品マスタの統一管理、入荷予定デー

ソース1　商品マスタ抽出SQL構築

```
// 商品マスタの SQL を構築します
procedure TfrmMIM020.BuildSQL(qry: TQuery);
var
  //1 画面内の条件数、条件画面数
  CCnt, LCnt: Integer;
  // 条件保存用構造体
  CondRec: array [1..120] of RCond;

  // 条件保存用構造体のセット
  procedure SetParameter(Target: String; DataType: TFieldType; Value: String; cm: TCompareMode); overload;
  begin
    Inc(CCnt);
    with CondRec[CCnt] do
    begin
      case cm of
        cmEQ: RSQL := Format('(A.%s = :%0:s%d)',  [Target, LCnt]);
        cmNE: RSQL := Format('(A.%s <> :%0:s%d)', [Target, LCnt]);
        cmGT: RSQL := Format('(A.%s > :%0:s%d)',  [Target, LCnt]);
        cmGE: RSQL := Format('(A.%s >= :%0:s%d)', [Target, LCnt]);
        cmLT: RSQL := Format('(A.%s < :%0:s%d)',  [Target, LCnt]);
        cmLE: RSQL := Format('(A.%s <= :%0:s%d)', [Target, LCnt]);
        cmLike:
          if AnsiPos('%', Value) > 0 then
            RSQL := Format('(A.%s LIKE :%0:s%d)', [Target, LCnt])
          else
            RSQL := Format('(A.%s = :%0:s%d)',    [Target, LCnt]);
      end;
      RParam    := Format('%s%d', [Target, LCnt]);
      RDataType := DataType;
      RValue    := Value;
    end;
  end;
  // 条件保存用構造体のセット - 汎用タイプ ( 等号 )
  procedure SetParameter(Target: String; DataType: TFieldType; Value: String); overload;
  begin
    SetParameter(Target, DataType, Value, cmEQ);
  end;
  // 条件保存用構造体のセット - 文字列タイプ (Like 検索 )
  procedure SetParameter(Target: String; Value: String; isLike: Boolean = False); overload;
  begin
    if Value <> '' then
      if isLike then
        SetParameter(Target, ftString, Value, cmLike)
      else
        SetParameter(Target, ftString, Value);
  end;

  // その他、整数タイプ、チェックボックスタイプ、実数タイプ等を定義しています。

var
  i, idx, j: Integer;
  sWhereFr, sWhereTo: String;
  stWhere: TStringList;
begin
```

タと仕入連携、在庫状況のリアル把握の導入。

アプリケーション開発：課題とソリューション

考慮すべき課題

●**機能面**：Filemakerによる既存機能をDelphiで実現し、操作性向上を図る。
●**業務面**：予約管理、発注管理、支払管理、入港管理、システム管理の輸入業務をシステム化する。それらによる作業効率アップと精度向上を図る。

選択した解決策

【機能面】
●商品マスタ検索を実現する。項目はマスタ項目すべてを条件内容とし、全項目 Or 条件指定最大数10までを可能にする。
●商品マスタでは、ボトルや裏ラベル画像の表示に拡大・縮小の画面を準備する。ユーザーによりわかりやすい機能を実装。【図1】
既存のFilemakerでは、画像はデータに保存し、変更があればすべて貼りなおしを行い、表示も拡大ができなかった。
●照会・検索画面では、グリッドのラベルをクリックすることによる、ソート順（降順・昇順）の変更を可能にし、ユーザーの操作性向上を図る。
●入力画面では、明細の各項目で矢印キーにより同一項目への遷移を可能にする。入力操作性の向上を図る。
● VB-Report ExcelCreatorなどの他のツール活用により、発注書の複数言語対応を実現する。

【業務面】
●予約管理機能では、予約残、予約支払（各国の通貨単位）、支払予定の管理などを構築する。従来のExcel管理からDelphi/400の予約管理に変更することにより、発注管理に連動させる。

実現した施策

【機能面】
●**商品マスタ検索の実現**
条件項目をローカルDB化し、同一レイアウトの複数条件入力を実現している。また、数値項目については、ポップアップメニューで大小等の比較演算子を指定することを可能にした。【図2】
「商品マスタ検索」のDelphi記述については、ソース1を参照いただきたい。【ソース1】
●**画像の拡大・縮小を実現**
Stretchプロパティを Trueにして TImageコンポーネントの縦横比を保ったまま、指定サイズに収まるように調整する機能を実現した。
また画像詳細画面では、元画像サイズの20%～200%の10段階の拡大・縮小表示を実現した。
●**一覧項目の動的な並び替え**
問い合わせや一覧等の機能フォームの継承元に、データセットとTDBGridを配置し、そのタイトル部をクリックすることで並び替え機能が実行されるようにした。そのため、機能フォームごとに実装する必要のない、並び替え機能が実現できた。
また、データセットにTClientDataSetを用いることで、あらゆる項目で動的な並び替えを実現した。
● **TDBCtrlGridの上下カーソルキー対応**
TDBEditのKeyDown処理で上下カーソルキーが押下されており、かつTDBCtrlGrid上に配置されている時は、TDBCtrlGridのKeyDown処理を呼び出すように改良し実現した。

【業務面】
● Excelで管理していた予約管理をデータベース化。これにより予約発注指示、予約からの発注引当（予約残管理）、予約による予約金支払、通貨別支払予定業務とこれまでの手作業による業務を迅速に処理し、把握できるようになった。
●予約からの引当に関しては、予約データから多種多様な抽出条件を指定することにより、商品の特定が簡易になった。とともに複数指定が可能となり、操作性も向上した。
●支払管理では、予約による予約金、発注による都度払いを一元化。これにより管理部門による支払業務の迅速化と操作性向上、および支払チェックが容易になった。とともにリアルな支払状況の把握ができるようになった。（通貨単位での外貨支払い予定が行える）
●発注情報に対する入港管理では、輸入書類の到着管理、スケジューリング管理（検品期日）倉庫輸入業者への通関処理指示連携を迅速化。通関後の仕入管理のリアルな基幹システム連携により在庫計上が行えるようになった。
●基幹業務とデータ連携（商品マスタ、入荷予定、通関処理（仕入））を導入。リアルに行うことにより、在庫・入荷予定の迅速な把握が行えるようになった。

輸入システム「MISYS」のメリットと今後

ノウハウの獲得

①予約→支払→発注→仕入までを一元管理
②過去データの活用（前回入荷時の仕様確認など）
③ VisualQueryを活用したさまざまなデータ抽出＆分析
④外部業者との自動連携（マスタ、入港スケジュールなど）

今後の予定・計画

- ワインの買掛金管理システムは、開発・本稼働済み。
  買掛残高、明細の一元管理。
  過払い金、為替差損益の管理。
- 本年末には2次改修を実施する。
  使い勝手にかかわる部分、実際に運用して出てきた機能追加の要望への対応。

エンドユーザーの評価

- 発注から支払、入港、仕入の全サイクルの経緯が一目でわかるようになった。
- 1つのシステムで情報共有できるため、意思疎通が図りやすい。
- 検索時間の短縮。

今後への期待感

- 外貨予約の管理と仕入支払業務との連動。
- クレジット管理。

M

```
//SELECT ～ FROM までを準備
qry.SQL.Text := cSelect;

// 条件レコードが存在すれば WHERE 句を構築します。
if cdsCond.RecordCount > 0 then
begin
  sWhereFr := 'WHERE (';
  sWhereTo := '';
  LCnt := 0;
  stWhere := TStringList.Create;
  try
    cdsCond.DisableControls;
    try
      idx := cdsCond.RecNo;
      // 条件画面数分の繰返処理
      for i := 0 to cdsCond.RecordCount - 1 do
      begin
        CCnt := 0;
        cdsCond.RecNo := i + 1;

        // 抽出条件チェック
        SetParameter('ISSYCD', cdsCondISSYCD.Value);  // 文字列・等号検索

        // 項目数だけ SetParameter を使って条件の確認を行います。

        SetParameter('ISRRY1', cdsCondISRRY1.Value);
        SetParameter('ISRRY2', cdsCondISRRY2.Value);
        SetParameter('ISSYSA', cdsCondISSYSA.Value, True);  // 文字列・Like 検索
        SetParameter('ISSYSB', cdsCondISSYSB.Value, True);

        // 抽出条件が存在すれば
        if CCnt > 0 then
        begin
          //1 条件画面の情報を (Field = Value AND Field = Value ...) の形式で構築します。
          stWhere.Text := '(';
          for j := 1 to CCnt - 1 do
            stWhere.Append(CondRec[j].RSQL + ' AND');
          stWhere.Append(CondRec[CCnt].RSQL + ')');

          qry.SQL.Append(sWhereFr);
          //2 画面目以降は ) OR (Field = Value ...) となるように変数セット
          sWhereFr := ') OR (';
          sWhereTo := ')';
          qry.SQL.AddStrings(stWhere);
          //Query の Parameter( バインド変数 ) に値をセット
          for j := 1 to CCnt do
          begin
            qry.ParamByName(CondRec[j].RParam).DataType := CondRec[j].RDataType;
            qry.ParamByName(CondRec[j].RParam).AsString := CondRec[j].RValue;
          end;
          Inc(LCnt)
        end;
      end;
      qry.SQL.Append(sWhereTo);
```

```
        cdsCond.RecNo := idx;
      finally
        cdsCond.EnableControls;
      end;
    finally
      stWhere.Free;
    end;
  end;
  //ORDER BY 句を設定して SQL 完成
  qry.SQL.Append('ORDER BY A.ISISTT, D.KUKUKJ, C.THTHKJ, ERCORK, TMCDN, ' + #13#10
        + 'A.ISCARA, A.ISWSYJ, A.ISBINT, A.ISYORY DESC');
end;
```

図1

図2

株式会社モトックス（詳細）
1915 年「正真正銘、元手なし」という意味の「元なしや」の屋号で創業。小売業・卸売業を経て「モトックス」に社名変更。現在は、世界各国のワインと日本の北から南までの地酒・焼酎・泡盛を企画、開発、提案を行っている。
「Value & Quality」な商品を求め「Serviceable Company」を目指している当社は、雑誌の投票でまっとうなインポーター、信頼の輸入元ナンバー 1 に選ばれるなど、業界でも注目を浴びている
http://www.mottox-wine.jp/

Delphi/400による 物流システムの再構築 ——実績のあるRPGプログラムを再利用する

仲井 学 様
西川リビング株式会社
経営システム室 課長代理

西川リビング株式会社
http://www.nishikawa-living.co.jp/

「眠り」から「健康」を創造し、より快適な暮らしを提案する西川リビング株式会社。時代のニーズに合わせた健康機能商品や新商品の開発を行っている。創業1566年の寝具・寝装品の製造卸業。

「物流システム」リニューアル

西川リビングでは、九州流通センターで使用されている「物流システム」が、主に処理レスポンスにおいて運用上深刻な支障をきたしていた。使用しているサーバーマシンのハードウェア保守が切れるということもあり、継続か、リニューアルかの判断が迫られていた。

そこで、Delphi/400導入を機に、システムリニューアルを行う決定がなされた。

「物流システム」の業務内容

物流システムの業務としては、引当済みの受注データに対する「出荷指図」「出荷報告処理」「荷札/内容明細」、さらに「入荷・入庫処理」をカバーしている。

追加機能として「PDラベル発行」にも対応。ラベル関連は、テキストファイルをDelphi/400によって書き出し、サトー社のMultiLabelistを自動起動させることでMT410eから出力している。

新・旧システムの構成

「旧システム」および「新システム」の機器・ソフトウェア構成は、図1と図2を参照していただきたい。【図1】【図2】

旧システムは、AS/400とWindowsサーバーで構成されており、DB2/400とOracleの間をHulftでデータ転送を行う仕組みである。クライアントは、VisualBasicアプリケーション。一部、5250で運用する処理も含んでいる。

新システムは、Windowsサーバーと Oracleを使用せず、シンプルな構成となった。メンテナンス上、シンプル化も非常に効果的である。

開発ポイント：RPGプログラムの再利用

今回は、システムリニューアルである。そのため「いかに工数を少なく」「いかに品質を保つか」という点について、「実績のあるプログラムを再利用する」をテーマに設計を進めた。

そのためには「Call400」の利用が必要不可欠であった。この「Call400」は、AS/400上のRPGやCLのオブジェクトを呼び出すことができ、もちろん、パラメータのやりとりも可能である。画面はDelphiで開発し、複雑なロジックはRPGで行うという手法である。つまり「Call400」を利用することで、シームレスなプログラムが開発可能となる。

具体的には「出荷指図」「荷札・内容明細・PDラベル」については、RPGでの新規開発とした。「出荷報告処理」「入庫処理」については、旧システムで使用されていたRPGプログラムを改造して再利用した。

特に、それら処理の部分は、IOも多く、ロジックも複雑な箇所である。そのため、実績のあるRPGプログラムを再利用することで、開発工数とともにテスト工数も大幅に削減することができた。また、品質についても初歩的バグもなく、使用方法さえ間違えなければ不具合もでず、非常に高品質なプログラムを短期間で装備することができた。さらに、AS/400

への IO をできるだけ減らし、RPG 側に行わせることによって、処理速度も満足のいく水準となった。

現在、当システムは運用開始から 2 年が経過し、順調に稼働している。

今後の展望

リニューアルした「新・物流システム」については、RPG を起動できる機能に非常に満足している。Delphi/400 に関しては、今回の“再利用”を含め、うまく使えば非常に効率的なシステム開発が行えると感じている。

まだまだ、使いこなせていない機能があると思われるので、これからの開発にどんどん取り入れていきたい。

M

図1

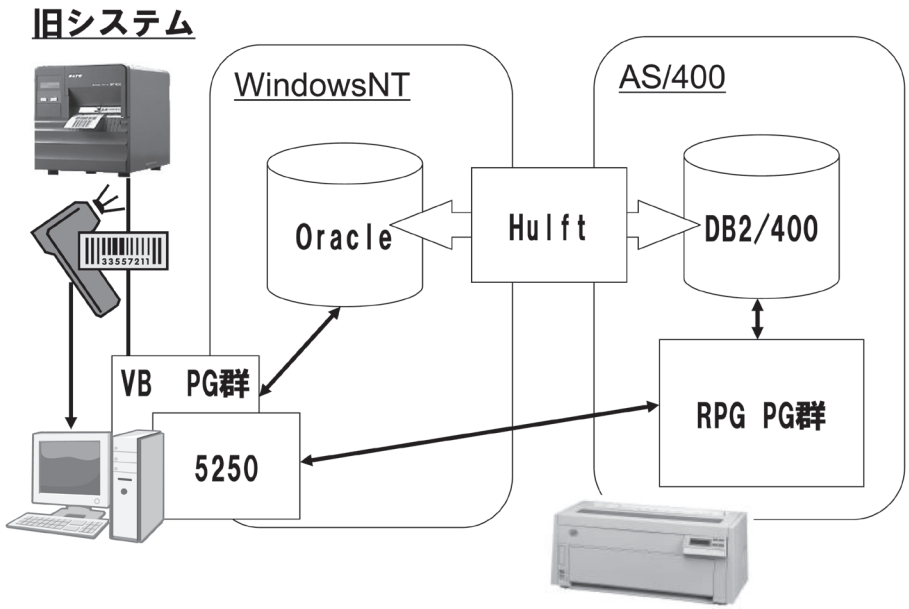
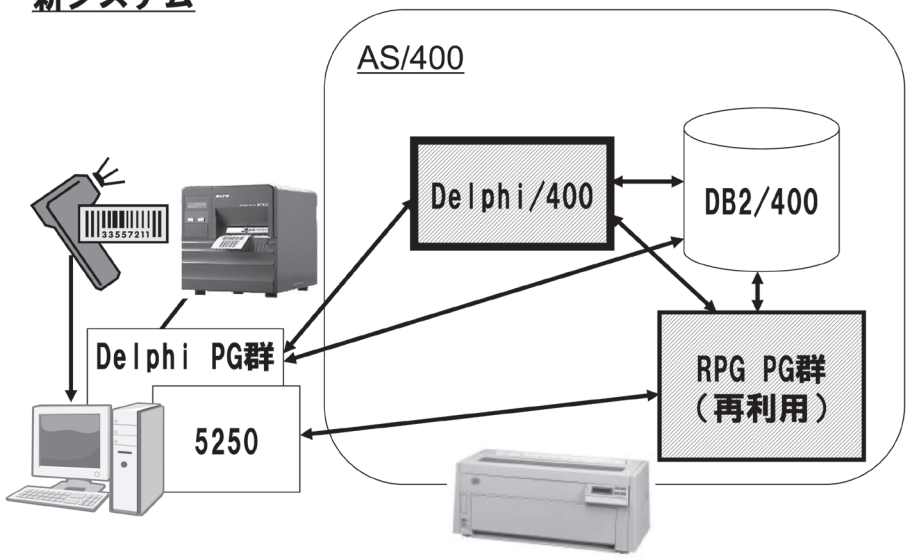

旧システム
WindowsNT
AS/400
Oracle
Hulft
DB2/400
VB PG群
5250
RPG PG群

図2


新システム
AS/400
Delphi/400
DB2/400
Delphi PG群
5250
RPG PG群
（再利用）

優秀賞

Delphi/400で開発し3台のオフコンを1台のIBM i へ統合 ――新「販売・仕入れ・在庫管理システム」

島根 英行 様
シルフ

シルフ
業務内容は IBM i を中心としたシステム開発。販売管理をメインに、オフコンや PC サーバーからのリプレース提案を得意としている。

今回は、二輪車パーツ販売の「株式会社山城」の事例を紹介。
株式会社山城
http://www.kkyamasiro.co.jp/

1. 開発事例：株式会社山城―3台のオフコンから1台のIBM i へ

株式会社山城では従来、本社・物流・大規模な営業の各拠点に3台のオフコン（富士通 GRANPOWER）を設置し、アプリケーションは COBOL を使用して自社開発を行っていた。だが、3台のオフコンを用いた運用は高コストであり、COBOL で開発したアプリケーションの機能には限界があった。そのため、同社は、新たに Delphi/400 でアプリケーション開発を行い、1台の IBM i への統合を実現した。

2.「販売・在庫管理システム」の新開発

新システムでは、顧客からの受注後、ピッキング指示・出荷データの登録・納品書発行・出荷・売上計上までの流れを1つのシステムで実現している。【図 1】

現在、基本システムの構築は完了し、運用を開始している。また、システムをより使いやすくかつ業務効率を向上させるために、以下の機能を順次追加しているという過程にある。

①セット品のマスタ化

山城では、販売する商品の特質から1つの製品を単品で販売したり、組み合わせてセット販売を行っている。

システムでは、セット品の作成は、画面上でセット内容の明細をドラッグ＆ドロップすることで簡単に作成できる。また、明細と数量を設定すると、セット商品の在庫を作成できるとともに、個別の商品の在庫も自動でマイナスするよう工夫している。【図 2】

なお従来、マスタにない商品を、個人の担当者が商品に作成し販売することが可能であったが、この機能を廃止した。

今回のセット品のマスタ化機能の実現により、すべての商品を効率的にマスタ化することが可能となり、在庫データ、販売データの分析精度の向上に役立っている。

②受注時の在庫不足品の処理

受注時に在庫が不足している場合、発注データを自動作成する機能もある。

また、在庫不足品が入荷処理されると、受注データに対し自動的に出荷引当を行い、ピッキングの指示が発行され、通常の出荷処理の流れと同様の処理扱いになる。

これらにより、従来は引当に3人で2時間かかっていた工数を、大幅に削減することができるようになった。【図 3】

③見積システムの追加

現在は、受注からの入力になっているが、見積システムの追加を予定している。【図 4】

④外部からの在庫照会機能

携帯電話から、部品番号を入力したメールを規定アドレスに送信すると、Delphi/400 で作成したアプリケーションが IBM i に在庫数量を取得に行き、携帯電話へ在庫数量が自動で返信される。

このモバイル対応の在庫照会機能を実

図1　メニュー

図2　セット品作成画面

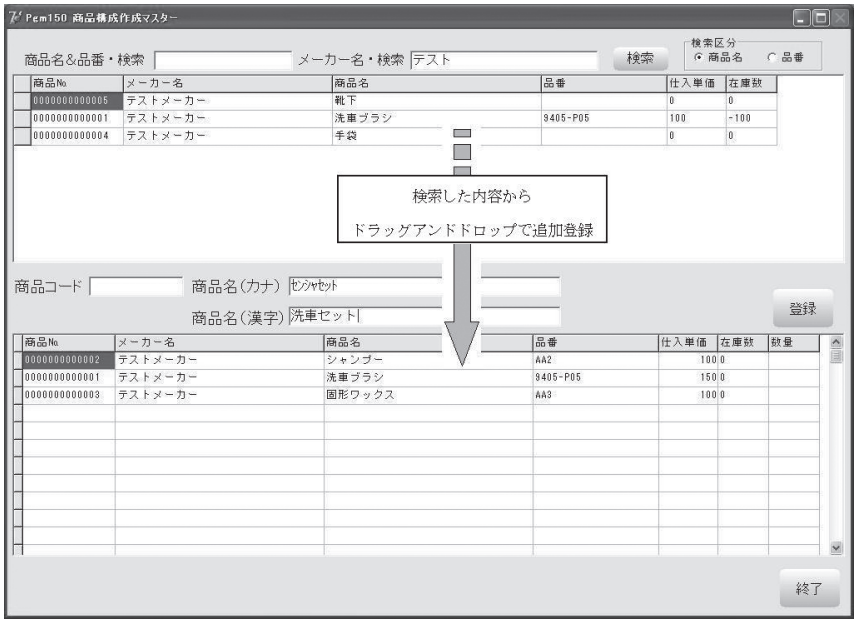
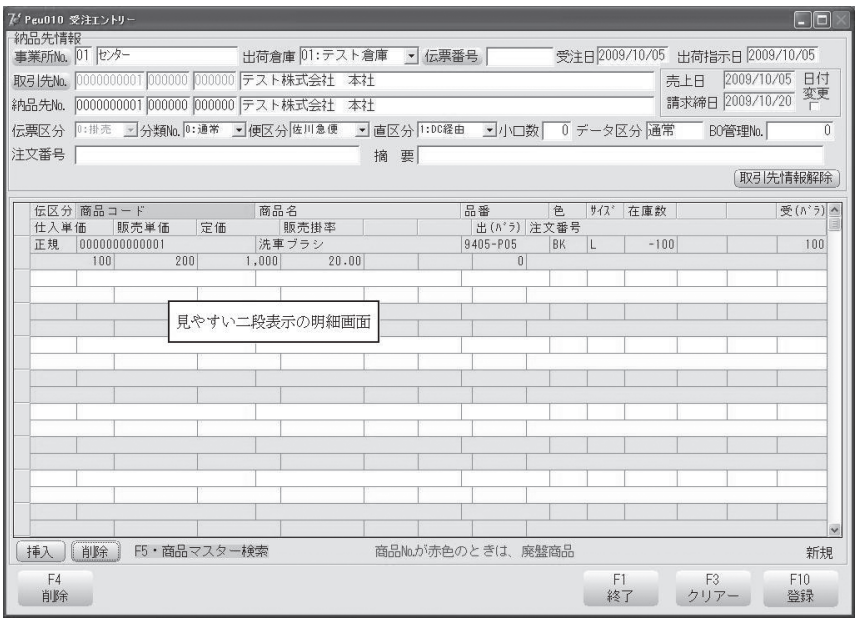


図3　受注入力画面(明細2段表示)



現した。もちろん、サーバー側でアドレスのチェック､暗号化などを行っている。

⑤在庫棚移動時の処理

季節によって売れ筋商品はまったく異なる。そのため、ピッキング作業の効率アップに向けて、売れ筋商品が取りやすい位置にくるよう商品の在庫棚を大幅に変更する。

上記に対応して、システムの在庫マスタ上の棚位置も変更しなければならない。これをまとめて変更できる処理の追加を予定している。また、棚のレイアウトを作成する機能や、棚のダブルクリックで行える保管在庫の照会機能も予定している。【図 5-1】【図 5-2】

3. 業務効率とコスト削減

受注から出荷、売上まで、一連の流れを 1 つのシステムで処理をすることにより、業務効率を大幅に向上させることができた。

また、3 台のサーバーを 1 台にまとめたことにより、インフラや運用管理のコストを大幅に削減することもできた。

将来的には、図 6 のような範囲をカバーするアプリケーションを実現することを目指している。【図 6】

M

図4　見積画面案

図5-1　在庫棚登録画面

図5-2　在庫棚明細画面

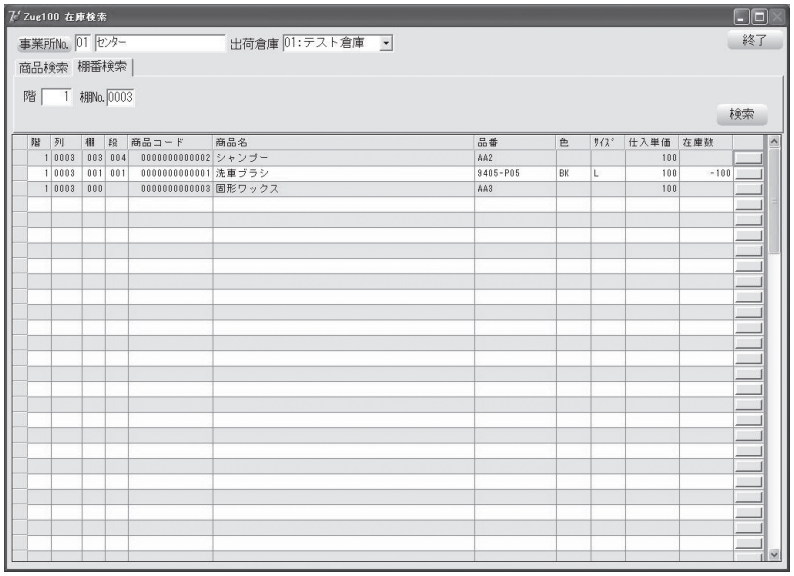
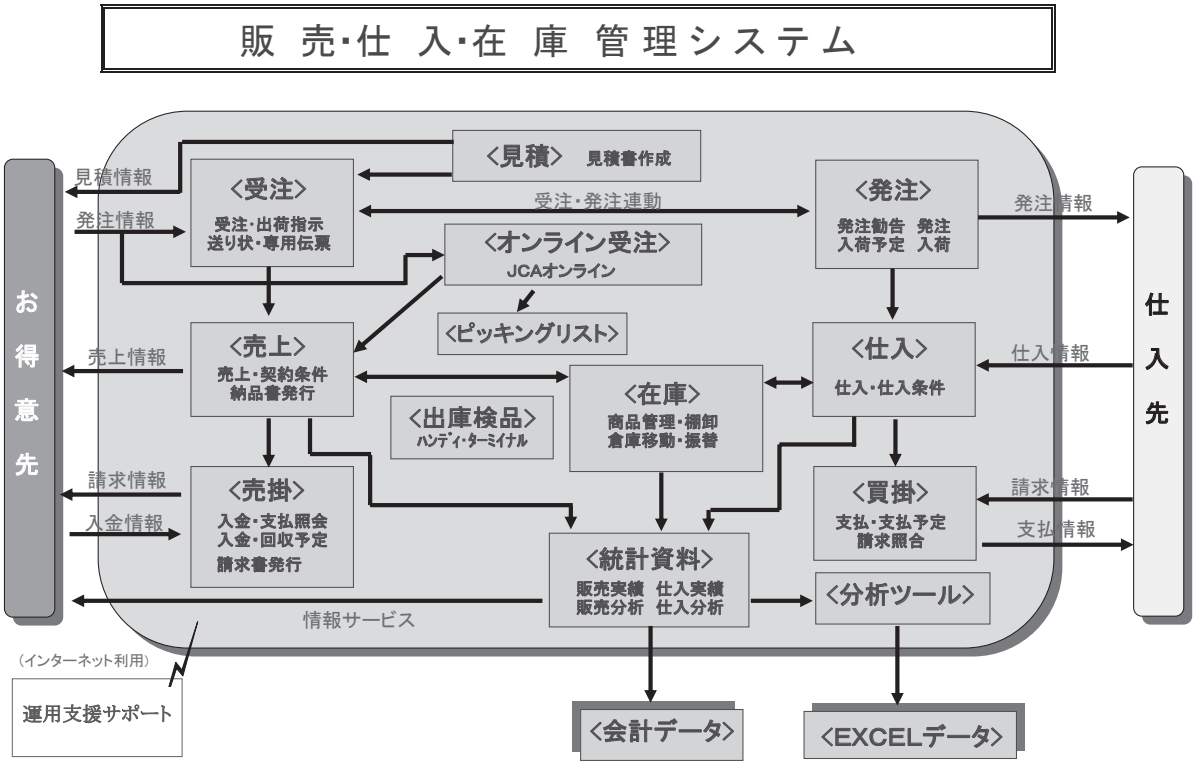


図6　システム概念図


販 売･仕 入･在 庫 管 理 システム
お得意先
仕入先
〈見積〉 見積書作成
〈受注〉
受注･出荷指示
送り状･専用伝票
〈オンライン受注〉
JCAオンライン
〈ピッキングリスト〉
〈発注〉
発注勧告 発注
入荷予定 入荷
〈売上〉
売上･契約条件
納品書発行
〈出庫検品〉
ハンディ･ターミナル
〈在庫〉
商品管理･棚卸
倉庫移動･振替
〈仕入〉
仕入･仕入条件
〈売掛〉
入金･支払照会
入金･回収予定
請求書発行
〈買掛〉
支払･支払予定
請求照合
〈統計資料〉
販売実績 仕入実績
販売分析 仕入分析
〈分析ツール〉
見積情報
発注情報
受注･発注連動
発注情報
売上情報
仕入情報
請求情報
入金情報
請求情報
支払情報
情報サービス
（インターネット利用）
運用支援サポート
〈会計データ〉
〈EXCELデータ〉

Migaro.
Technical Report
2009

ミガロ.SE 論文／ミガロ. テクニカルレポート

岩田 真和

株式会社ミガロ.

RAD事業部 技術支援課

JACi400環境でマッシュアップ!
――本格的なWeb2.0をJACi400で実現しよう!

これからのWebユーザビリティを考えると
どうしてもWeb2.0の技術が外せない。その際、JACi400により
どの程度要望に応えることができるか
その回答を考察する。

略歴
1984年7月7日生れ
2007年京都学園大学経営学部卒
2007年04月株式会社ミガロ. 入社
2007年04月システム事業部配属
2009年09月RAD事業部配属

現在の仕事内容
JACi400やDelphi/400などの開発経験を経て、現在はJACi400のサポート業務を担当。

●はじめに
●JavaScriptの活用
●Web APIの活用
●JACi400とのマッシュアップ
●JACi400アプリケーションの可能性

はじめに

ミガロ.のWebアプリケーション開発ツールの1つである「JACi400」。これはRPG/COBOLといった現有プログラムスキルやDBなどの既存資産を生かして、スムーズなWebアプリケーション開発を可能にしてくれる製品である。

2008年のテクニカルレポートをご覧いただければ、比較的少ない工数で開発が可能なことや、開発自体の容易性など、そのメリットをご理解いただけると思う。

しかしながら、実際JACi400を手にしてWeb開発をスタートされたお客様の要望は、その“容易性”という視点にとどまらない場合が多い。BtoC、BtoBのやり取りに使用されるWebアプリケーションの場合はもちろんだが、社内・事業所間などで使用される場合にも、“Webとしてのユーザビリティ”というものが追求される必要がある。それはお客様が求められている「Web化」の中に、このユーザビリティが前提となっているという意味である。

さらに今後は、一般的に「Web2.0」と呼ばれる、新しいWeb技術を取り入れたいという要望も増えるであろう。それは、これからのWebユーザビリティを考えると、どうしてもWeb2.0の技術が外せなくなってきているからである。その際、JACi400により、どの程度その要望に応えることができるのか、というのは気になるところかと思う。

本稿は上記のような、現実にある要望と、これから多くなるであろう要望に対する回答になればと考えており、考察を行っていく。

内容は、参考ソースやサンプル画面の紹介も交えて「JavaScriptの活用」「Web APIの活用」「JACi400とのマッシュアップ」「JACi400アプリケーションの可能性」という4つの段階に分けて解説する。

結論的には、JACi400というIBM i専用のツールを利用しても、一般的なWeb技術がへだたりなく利用可能ということをこのレポートで強くお伝えしたい。すでにご使用の方々は、あらためてJACi400の可能性にご興味を持っていただければ幸いである。もちろんWebに精通されている方々なら、JACi400環境でも自由なWebアプリケーション開発が実現可能だということをさらに確信されるだろう。

※本レポートの内容はどちらかというと、JACi400の応用的な使い方が中心になっている。JACi400の基本的な使用方法(Web画面とIBM iとの連携)については、ここでは詳細に紹介していない。基本的な使用方法は、2008年発行のテクニカルレポートや、ミガロ.のホームページをぜひご覧いただきたい。

※本稿で紹介しているコードは、実行確認をしているものであり、できるだけそのまま使ってもらえるように心がけた。環境によっては、カスタマイズが必要な場合があることをご了承いただきたい。

ソース1

```
<body>↓
<form method="POST" name="frm">↓
<script type="text/javascript">↓
function migarodsply(){↓
var adhtml='http://www.migaro.co.jp/';↓
   var aWin=window.open(adhtml,null);↓
   aWin.focus();↓
}↓
</script>↓
```

ソース2

```
<!---// メッセージ --->↓
<INPUT type="text" name="MSG" id="MSG" style="display:none;">
```

図1

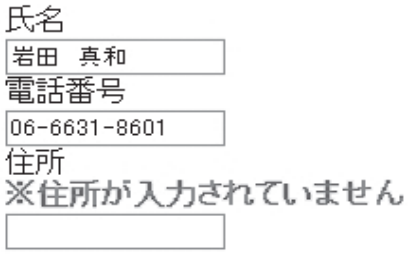
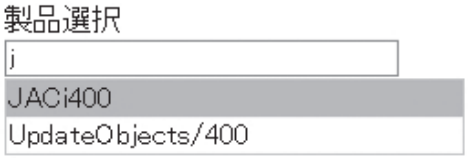


図2



ソース3

```
 (HTML部) ↓
<!---// メッセージ --->↓
<INPUT type="text" name="MSG" id="MSG" style="display:none;">↓
<!---// メッセージ位置 --->↓
<INPUT type="text" name="MSGR" id="MSGR" style="display:none;">↓
↓
 (JavaScript部) ↓
function error(){↓
        // message変数を定義する↓
        var message = '';↓
        // メッセージ内容をmessageにセット↓
        message = document.frm.MSG.value;↓
        // メッセージ位置にmessageを出力↓
        document.getElementById(document.frm.MSGR.value).innerHTML = message;↓
}↓
↓
 (表示先HTML) ↓
氏名<br>↓
<div id="NAMEM" name="NAMEM" style="color:red;font-weight:bold;"></div>↓
<input type="text" id="NAME" name="NAME"><br>↓
電話番号<br>↓
<div id="TELM" name="TELM" style="color:red;font-weight:bold;"></div>↓
<input type="text" id="TEL" name="TEL"><br>↓
住所<br>↓
<div id="ADDM" name="ADDM" style="color:red;font-weight:bold;"></div>↓
<input type="text" id="ADD" name="ADD"><br>↓
```

JavaScriptの活用

JavaScriptとは

まず取り上げたいのが「JavaScript」である。JavaScriptは、ブラウザ上で動く簡易言語と考えていただければよい。インターネットが誕生してまもなく登場し、以前から個人のWebサイトでも多く使用されているほど、敷居の低い言語として知られている。

なぜ、JavaScriptを最初に取り上げるのかというと、それだけWebの世界になくてはならない存在だからである。役目としては、画面定義するHTMLとデータベースに書き込むメインプログラム（RPG/COBOL）の仲介役と言えるだろう。今では「Ajax（エイジャックス）」という技術が話題になっているが、JavaScriptはその柱になっている。

さらに心強いことに、JACi400では入力文字数の制限・カンマ編集といったWebで必要とされている基本的なJavaScriptの機能が自動的に提供される。そのため、開発者が付け加えるのは、プラスアルファで組み込みたい機能のみとなる。

今回は、実際のWebアプリケーションで使えそうな、JavaScriptの活用例をいくつか紹介する。一般的なWebサイトに接続していることを想像しながらご確認いただきたい。

JACi400環境でのJavaScriptの使用

さて、例をご紹介する前に、JACi400上でJavaScriptを使用するにあたり、いくつかあるコツをお伝えしておきたい。

通常のWeb開発であれば、“HEADタグ”内にJavaScriptを記述するのが普通である。JACi400では、HEAD部分は、JACi400アプリケーションが起動する際に、自動的にJACi400の動作上必要なソースで上書きされる。そのため、JavaScriptを“BODYタグ”内に記述することにより、上書きを回避する必要がある。逆に考えれば、そこだけ気をつけていれば、JACi400アプリケーションでは、通常のWebと同様にJavaScriptを使用することができる。【ソース1】

また、よく使うテクニックとして、JACi400で使用するフィールドを「隠しフィールド」にしてしまう手法がある。テキストフィールドのstyle属性に"display:none;"を指定することで、そのフィールドを見えなくしてしまう。こうすると、画面からは見えなくても、内部的には値を持っていることになり、JavaScriptから自由に入出力が可能になる。【ソース2】

わかりやすいメッセージ出力

簡単な例として、JavaScriptを使用して、“わかりやすいメッセージ出力”というものについて考え、表現してみたい。

例えば、入力画面で入力不備があり、入力不備のエラーを発生させたいケースでは、通常は、あらかじめ設計者が決めたメッセージ出力用の場所にメッセージが出力される。その際、どんなメッセージを出力しても、常に同じ位置に表示される。この方法だと、ユーザーとしては、どの項目が原因でエラーとなっているのか把握しにくい。

そこで、JavaScriptを使用し、エラーの原因となっているフィールドの真上にメッセージが出てくるようにする。この場合は、隠しフィールドとしてメッセージ内容とメッセージ位置を用意し、エラー時にIBM iからセットするようにする。【図1】【ソース3】

【ソース解説】

JavaScriptにより、JACi400から取得した値を、表示先HTML部のSPANタグにセットさせる。ここでは、メッセージ位置がNAMEMの場合は氏名のSPANタグ、TELMの場合は電話番号のSPANタグにメッセージがセットされる。

なお、SPANタグにはあらかじめ赤色・太字のフォントが指定してあるので、通常よりも目立ってわかりやすいと思う。

suggest.jsによる自動補完（オートコンプリート）

最近、検索サイトなどでよく見かける、文字列の自動補完の機能を導入しよう。自作しようとすると高度な知識が必要になるが、「ライブラリ」と呼ばれる関数群をうまく使用することにより、シンプルなJavaScriptで機能の実現が可能になる。

多くのライブラリはオープンソースになっており、ライセンス規約を守れば無償での利用が可能である。suggest.jsはMITライセンスであり、再配布をする際にはライセンスの表記が必要になる。（※）【図2】【ソース4】

【ソース解説】

掲載したのは、ライブラリ部分から変更したところのみにした。今回はミガロ.の製品を選択する形である。

なお、このぐらいの項目であれば、コンボボックスのほうが使いやすいかもしれない。

このライブラリは、外部ファイルの読み込みも可能であるため、自動補完を有意義に使用できる場面で利用していただきたい。

※ライブラリ提供元：Enjoy*Study
http://www.enjoyxstudy.com/javascript/suggest/

Flashとの連携

「Flash」を用いると柔軟な表現が可能になるため、Webカタログなどで演出に最適である。

では、Flashで選択したデータをJACi400に送りたい場合、どのようにすればよいのだろうか。これも、JavaScriptによって可能になる。Flashの場合は、Flash内に「ActionScript」と呼ばれる（JavaScriptによく似ている）コードを記述することで、FlashとJavaScriptを連携させる必要がある。【図3】【ソース5】

【ソース解説】

ActionScriptは、Flash.externalパッケージのクラスを利用することで、JavaScriptの呼び出しを可能にしている。Flashで入力された文字列は、ActionScript内でtextと定義された変数として取得できる。これをパラメータとして、JavaScriptを呼び出すのである。JavaScriptに渡されたパラメータは、alert関数によりメッセージを出力している。

ちなみに、ここでは掲載しないが、JavaScriptからActionScriptを操作することもできる。そのため、IBM iから取得した初期値を、Flashに渡すとい

ソース4

```
（項目設定部分）↓
// 補完候補の配列作成↓
var list = [ 'JACi400', 'Delphi/400', 'UpdateObjects/400', '*noMAX', 'MKS Integrity' ];↓
↓
（表示先HTML）↓
<!-- 入力エリア --> ↓
<input id="text" type="text" name="text" value="" autocomplete="off" size="40" style="display: block"/> ↓
<!-- 補完候補を表示するエリア --> ↓
<div id="suggest"></div> ↓
←
```

図3

ソース5

```
（ActionScript部分）↓
import flash.external.*;↓
import mx.controls.*;↓
↓
btn.onRelease = function(){↓
        retuneText.text = ExternalInterface.call("FlashToJS",text);↓
}↓
↓
（JavaScript部分）↓
function FlashToJS(message){↓
        alert(message);↓
}↓
←
```

ソース6

```
|（HTML）↓
<form action="upload.php" method="post" enctype="multipart/form-data" name="upfrm">↓
  <input type="file" name="upfile" size="30"><br>↓
  <input type="submit" value="Webサーバーにファイルをアップロード"↓
         onmousedown="parent.document.frm.PATH.value = document.upfrm.upfile.value;">↓
</form>←
```

図4

表1

代表的なWeb API 一覧

サービス名	取得できる主な情報
Yahoo!デベロッパーネットワーク	ニュース、オークション
Amazon Web サービス	商品情報
価格.com	商品情報
お天気Webサービス	天気情報
できるじゃらんWebサービス	宿情報
ホットペッパーWebサービス	飲食店情報
Wikipedia API	Wikipedia情報
SOBA Web API	カメラ画像の共有、デスクトップ共有
Force.com WebService API	SalesForce情報
SimpleAPI:ウェブサイトサムネイル作成API	ウェブページのサムネイル作成

う仕組みも可能となる。

PHPプログラムとの連携（ファイルアップロード機能の付加）

画像ファイルをJACi400上で扱う場合、ポイントが2つある。まずは、画像ファイルをWebサーバーに置く。それから、IBM i 側でその表示したい画像のパスを指定する。このような連携を用いて、JACi400は画像を表示している。

照会のみであれば、上記で問題はない。だが、ユーザーサイドから画像を送りたい場合は、別の手段を用いて、画像ファイルをWebサーバーにアップロードする必要がある。このような仕組みは、サーバーサイドで動作するプログラムによって実現できる。

いろいろな手段があるが、今回は一般的なWebにおいて一番敷居が低いと思われる、オープンソースのPHPを使用した実現方法を解説する。

ここでは、PHP Laboが提供するファイルのアップロード機能を参考に、画像ファイルをWebサーバーに転送する。JACi400上で使用する際のコツは、この画面をインラインフレームに置くという点である。そうすれば、制御がIBM i 側に戻らずにファイルをアップロードできる。【図4】【ソース6】

なお、PHPやPHP Laboに関しては、以下のURLを参照のこと。（※）

【ソース解説】

まず、Webサーバー上にPHPをインストールし、使用できる環境を用意する。

ここでは、JACi400との連携に向けて、アップロードしたファイル名を取得するためにJavaScriptを組み込み、親フレームに配置してあるJACi400のフィールドへ値を渡している。

※ PHP Labo
http://www.php-labo.net/tutorial/php/upload.html

以上、JACi400環境において、Webアプリケーション作成時に使えそうなJavaScriptの活用例を4点挙げた。

ここでご紹介した内容は応用的ではあるが、特別複雑なことを行っているわけではなく、一般的なWebサイトでは多く導入されている技術である。しかし、このようなことでも使用感が全く違ってくる。JACi400アプリケーションを作成する際にぜひ参考にしてほしい。

Web APIの活用

Web APIとは

次に取り上げたいのは「Web API」である。

「API」とは、Application programming interfaceの略であり、特定の機能を持ったプログラムの部品と解説される場合が多い。利用者はその都度プログラミングすることなく、その機能を利用できる。APIというとOSの機能をプログラムから利用するための窓口というイメージがあると思うが、Web APIはOS上ではなくWeb上に散在しているAPIである。一般的に「Webサービス」とも呼ばれる。

OSやフレームワークが提供するサービスではなく、Web上のサービスとして提供されており、その機能を取り込める。つまり、今までローカルのパソコンではできなかったさまざまな機能を簡単に利用できるようになる。

Web APIの代表例としてよく出される「Google Maps API」を題材に、簡単な活用例を説明する。

Google Mapsは、言わずと知れたGoogleの地図検索サービスであり、衛星写真やストリートビューといったサービスで話題になっている。しかし最近は、他のWebサイト上でもGoogle Mapsを見かける機会がないだろうか？

具体的には例えば、グルメサイトから、出張先でランチに食べるラーメン屋を探すとする。そこで表示された地図を見て、縮尺などを操作して印刷する。いつもの使用感と変わらないなと思っていたら、地図の左下に「POWERED BY Google」のロゴが入っていたことに気づく――実はこれが、Google Maps APIを使用したサイトの例だ。Google以外のサイト上でも、多くのユーザーが使い慣れているGoogle Mapsを利用できるのである。

このようにWeb APIを利用することで、いままで地図データを自社で用意しなければいけなかったものが不要になるだけでなく、その構築工数を省くことが可能になる。しかも地図の更新等は、Webサービスの提供元により自動的に行われる。

このグルメサイトの地図サービスの例は、既存の「飲食店検索システム」とのマッシュアップ例とも言える。“マッシュアップ（Mashup）”とはマッシュポテトをイメージしていただければわかりやすい。Web APIなどを掛け合わせて新しい価値を生み出す、というWeb2.0の用語である。

なお、マッシュアップ、JACi400でのマッシュアップの実践については、後半にくわしく説明する。

Web APIが提供する機能と必要な知識

JACi400との関係や連携方法を解説する前に、Web APIについて、どのような機能が提供されているのかを確認しておこう。また、それを使用する際に必要な知識を解説する。

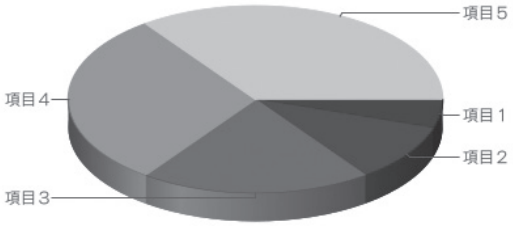
Web APIの実例の中で、有名でわかりやすい例として挙げたいのが「Google AJAX API」である。前述した「Google Maps API」もこの一部である。またGoogle AJAX APIは、簡単にグラフ化を行う「Google Chart API」や任意のサーバーからRSSフィードを取得する「Google AJAX Feed API」など、多くのAPIを提供している。

Google以外にもニュースや天気予報など、切りがないほどのWeb APIが公開されている。しかも、一部有償のものもあるが、ほとんどは無償で提供されている。【表1】

1つ1つの情報は、ネット検索でその都度調べれば不便ではない情報ばかりだが、これらWeb APIを組み合わせる（マッシュアップ）ことにより、新しい価値を生み出すことができる。つまり、JACi400を使用してマッシュアップすれば、IBM i の資産とつながることが可能になる。

これらを実装する際には多くの知識は必要なく、ハードルはそれほど高くない。前述したようなJavaScriptの基礎を理解していれば、ある程度は扱えるようになる。もちろん、使用方法は各デベロッパーサイトにも載っている。なにより、これらWebの世界は技術情報が多くWeb上に存在しているため、信用できるサンプルソースが入手できれば、それ

図5



ソース7

```
 (HTML) ↓
<input type="text" id="JS1A" name="JS1A" style="display:none;"><!--項目１-->↓
<input type="text" id="JS2A" name="JS2A" style="display:none;"><!--項目２-->↓
<input type="text" id="JS3A" name="JS3A" style="display:none;"><!--項目３-->↓
<input type="text" id="JS4A" name="JS4A" style="display:none;"><!--項目４-->↓
<input type="text" id="JS5A" name="JS5A" style="display:none;"><!--項目５-->↓
<input type="text" id="JS1B" name="JS1B" style="display:none;"><!--データ１-->↓
<input type="text" id="JS2B" name="JS2B" style="display:none;"><!--データ２-->↓
<input type="text" id="JS3B" name="JS3B" style="display:none;"><!--データ３-->↓
<input type="text" id="JS4B" name="JS4B" style="display:none;"><!--データ４-->↓
<input type="text" id="JS5B" name="JS5B" style="display:none;"><!--データ５-->↓
 (JavaScript) ↓
function dsp()[↓
        var str = '';↓
        str = str + '<img src="http://chart.apis.google.com/chart?';↓
        str = str + 'chs=500x200';↓
        str = str + '&amp;chd=t:';↓
        str = str + document.frm.JS1B.value;↓
        str = str + ',';↓
        str = str + document.frm.JS2B.value;↓
        str = str + ',';↓
        str = str + document.frm.JS3B.value;↓
        str = str + ',';↓
        str = str + document.frm.JS4B.value;↓
        str = str + ',';↓
        str = str + document.frm.JS5B.value;↓
        str = str + '&amp;cht=p3&chco=0000ff';↓
        str = str + '&amp;chl=';↓
        str = str + document.frm.JS1A.value;↓
        str = str + '|';↓
        str = str + document.frm.JS2A.value;↓
        str = str + '|';↓
        str = str + document.frm.JS3A.value;↓
        str = str + '|';↓
        str = str + document.frm.JS4A.value;↓
        str = str + '|';↓
        str = str + document.frm.JS5A.value;↓
        str = str + '" alt="売上種別" />';↓
↓
        document.getElementById('TEST1').innerHTML = str;↓
]↓
 (出力用HTML) ↓
<div id="TEST1"></div>↓
```

を参考にして組まれていってもいいだろう。一般的なWebの技術情報をそのままJACi400に応用できることも、HTMLを自由に作成できるというJACi400の魅力である。連携も非常にシームレスである。

JACi400環境でのWeb APIの使用 ―JACi400でグラフを表示させる

JACi400環境での、Web APIのシンプルな例を1つだけ紹介する。活用については、マッシュアップという形で説明したいと思うので、ここでは、Web APIとJACi400との連携方法を中心に述べる。

例えばIBM i上にある売上情報をビジュアルに把握したい場合、どうしてもグラフを活用したくなる。しかし、Web上でグラフを表現する場合、テーブルを駆使したり、複雑なJavaScriptを実装したり、そのためだけにFlashを導入したり、ハードルの高いイメージが少なからずある。

そこで、前述のGoogle Chart APIを用いてグラフ化を導入してみる。Google Chart APIは非常にシンプルな構造になっていて、「データ」と「フィールド名」をURLのパラメータとして受け渡せば、グラフを画像として表示してくれるものである。

なお、この方式を一般的に「REST方式」と呼び、導入がシンプルでわかりやすいため、多くのWeb APIがこの方法を採用している。

呼び出しURLの例

```
http://chart.apis.google.com/chart?cht=p3&chd=t:60,40&chs=250x100&chl=Hello|World
```

例えば、このようなURLを実行すると、自動的に2つの項目があるグラフとして画像を表示させることができる。

このURLを分解すると以下のようになる。

```
① http://chart.apis.google.com/chart?
② cht=p3&
③ chd=t:60,40&
④ chs=250x100&
⑤ chl=Hello|World
```

まず、①とそれ以降に大きく分かれる。①がグラフを表示する関数（もしくはAPI名）で、それ以降はパラメータと思っていただくとわかりやすいかと思う。

ちなみに、②はグラフの種類、③は項目の値、④はグラフの大きさ、⑤は項目の表題にあたる。このようなシンプルな構造により、動的にURLを作成するだけで、動的なグラフ表示が可能になる。

図5は、JACi400と連携を行ったグラフサンプルだが、ここで記述しているソース7についてはそれほど多くないことに気づくであろう。このソースでは5項目固定になっており、動的に増やそうと思えば、それに応じたJavaScriptを記述してURLを完成させればいい。【図5】【ソース7】

また、ここで1つ気をつけていただきたいことがある。このソース7をそのまま実行すると、項目名に、例えば全角文字を指定した際にその文字が欠けてしまう。これは、多くのWeb APIがWeb上で主流になりつつあるUTF-8の文字コードを採用しているためであり、うまくパラメータの受け渡しができていないのが原因である。

Shift_JISで開発を行っている場合は、何らかの処置が必要である。一般には、JavaScriptであらかじめ用意されているencodeURIComponent関数を利用することによって、対処が可能だ。

```
encodeURIComponent(document.frm.JS1A.value);
```

セットする際に、このencodeURIComponent関数を挟んで受け渡すことで、文字コードの問題を回避できる。図5のような形で、JACi400上でのグラフ表示が可能になる。

Web APIを利用する場合の注意点

今回、非常に便利なサービスとして、Web APIの活用を取り上げた。ここで念のため、Web APIの利用において配慮が必要な点を2つ挙げておきたい。

●利用許諾条件

利用用途によっては、サービス対象外のケースとなる。例えば、Google Mapsは一般のWebユーザーが観覧できないサイトでなければ使ってはいけない、という規約が存在する。注意してほしい。

●サービスの停止リスク

第三者から提供されるサービスであるため、サービスの停止などには留意していただきたい。複数のWeb APIを使用する場合は1つのサービス停止が、システム全体の停止につながらないように工夫が必要だ。また、これらのサービスを使用する場合、事前に代替サービスを探しておくことも必要な対策と言える。

JACi400とのマッシュアップ

例：ミガロ. オリジナルTシャツ注文システム

今回は、JavaScriptやWeb APIなど、Webアプリケーションのさまざまな仕組みをご紹介した。せっかくなのでこれらの技術を使用し、JACi400でのマッシュアップを実践してみたい。

今回作成するのは、在庫照会注文システムだ。わかりやすいように「ミガロ. オリジナルTシャツ注文システム」と名称をつけ、ユーザーからオリジナルTシャツの注文をもらうシンプルな仕組みをWebアプリケーションに作り上げる。【図6】【図7】【図8】

これまで紹介した技術、JavaScript、Web APIなどをできるだけ多く使用するものとする。

①タブ「商品を選ぶ」

まず、Flashでビジュアルなカラー選択ボタンを作成した。もし、すでにWebカタログとしてFlashがある場合は、それをカスタマイズしてもよい。自社で作成しなくても、この部分が得意なWebデザイン会社等に依頼するなどと想定していただければと思う。

Flashを使用しているので滑らかな表現が可能になる。ここでカラー選択、個数指定なども可能になる。人気カラーの内訳はGoogle Chart APIによるものである。【図6】

②タブ「画像を合成する」

Flash指定が終われば、ページ自体は変わらずタブを移動する。この機能はJavaScriptによって実現している。

ユーザーは、PHPの機能によってファ

図6


MOTOS
Migaro. Original Tshirts Order System
ミガロ.オリジナルTシャツ注文システム
1. 商品を選ぶ → 2. 画像を合成する → 3. ご注文内容の確認をする
お好きなカラーと個数を選択してください。
Blackを選択しています。
人気カラー
20枚/月
40枚/月
30枚/月
Black
Blue
Yellow
現在の在庫：5個
Black
Blue
Yellow
3
個
商品確定

図7


MOTOS
Migaro. Original Tshirts Order System
ミガロ.オリジナルTシャツ注文システム
1. 商品を選ぶ → 2. 画像を合成する → 3. ご注文内容の確認をする
イメージをアップロードしてください。
C:¥Documents and Settings¥m
参照...
UPLOAD
ミガロマラソンクラブ
ななうまさがえ
画像確定

図8


MOTOS
Migaro. Original Tshirts Order System
ミガロ.オリジナルTシャツ注文システム
1. 商品を選ぶ → 2. 画像を合成する → 3. ご注文内容の確認をする
ご注文内容にお間違いがなければ、注文確定ボタンを押してください。
Color:1
数量:3
画像:logo.jpg
Web ページからのメッセージ
注文を確定してよろしいですか？
OK
キャンセル
注文確定

イル(ロゴデザインイメージなど)をアップロードし、JavaScript により T シャツとアップロードした画像を組み合わせたイメージ（T シャツの完成イメージ）を確認することができる。【図 7】

③タブ「ご注文内容の確認をする」

最後に、確認画面に移る。この時点ではまだ IBM i には情報を渡していない。今まで入力された情報はすべて隠しフィールドに存在しているので、その値を確認情報として出力している。注文確定の OK ボタンを押して注文を完了させる。

応用したいツールが Flash であっても Web API であっても Ajax であっても、結局はその隠しフィールドに値が入力される。そこからの処理は JACi400 の通常機能であり、JACi400 だから複雑なロジックが必要になる、といったことは発生しない。

なお、ブラウザとしての画面遷移は最後の 1 回だけになる。そのほかは、タブ形式で徐々に右に移っていくという視覚的にわかりやすい流れになっている。【図 8】

実際のアプリケーションでは、このように各種の技術を必要以上に盛り込むことは実用的とは言えないだろう。とはいえ、今回のプログラムにより、JACi400 環境において、Web2.0 と呼ばれるような先進の技術との連携の可能性が確かめられた。

JACi400アプリケーションの可能性

今後は、今まで以上に基幹システムと Web システムが統合していく社会が訪れる。しかも、それは金融 / 銀行業やネット販売業、ネットサービス業といった、Web システム自体が基幹システムとなっている業界だけの話ではない、と言われている。

これは、多くの企業がこれまで見てきた「Web ＝情報発信」の常識が変わろうとしているということである。情報発信を超える可能性が Web にはある。

そんな中、JACi400 というツールに注目していただくメリットとしては、次のようなことが考えられる。

- **IBM i のメリットを最大限に生かした Web アプリケーションの実現**
- **基幹システムとのシームレスな連動**
- **開発コスト、運用コストの削減**

そして、本稿で取り上げ、ここまで考察を加えてきた以下のポイントにより、JACi400 の利用方法も今後、大幅に広がっていくと思われる。

- **あらゆる Web ツールとの連携による可能性**

さいごに、本レポートを執筆して、あらためて JACi400 が初級者から上級者まで扱えるツールだということがわかった。さらに Web 初心者だからこそ、JACi400 によって、IBM i 上から直接データを配信し、今までにない新しい価値を創出することもできる。そういったユーザーの方々が増え続けることを願っている。

M

現在の仕事内容（詳細）
JACi400 や Delphi/400 などの開発経験を経て、現在は JACi400 のサポート業務を担当。
幅広いバックボーンを備えた SE を目指し、日々業務に邁進中である。今後はさらに CSS や JavaScript、Ajax、その他 Web に関する知識を高めたい。

福岡 浩行

株式会社ミガロ.

システム事業部 システム2課

Delphi/400を利用したはじめてのWeb開発 ―VCL for the Webによる簡単な開発方法!

開発者であれば、Web アプリケーションを
開発してみたいと思うだろう。
ここでは
Delphi/400 の VCL for the Web で可能になった
Web アプリケーション開発方法を紹介する。

略歴
1984 年 11 月 11 日生れ
2007 年関西学院大学理工学部卒
2007 年 04 月株式会社ミガロ. 入社
2007 年 04 月システム事業部配属

現在の仕事内容
Delphi/400 や RPG の開発業務を担当。Web 開発技術に関する知識（JavaScript、CGI、PHP 等）を習得し、オールマイティな提案や開発が担える SE を目指している。

●はじめに
●VCL for the Webでの開発用途/形態
●VCL for the Webでの画面設計
●VCL for the Webでのプログラムロジック開発
●VCL for the Webでの応用開発
●開発端末でのWebサーバー環境構築
●おわりに

はじめに

近年、ユビキタス社会が確立されていく中、開発者であれば一度は Web アプリケーションを開発してみたいと思ったことはないだろうか。私もその中の一人で、企業にある蓄積された情報を Web 化することで、より情報を有効に扱えるものと考えている。

しかし、簡単に Web アプリケーションを作ることについてちょっと考えてみても、Java や PHP 等の新しい技術を習得する必要があったり、既存のシステムとの調整も必要になったりする。さらに、これらのさまざまな不安要素を解決するには大幅な時間を費やすことが求められる。

Delphi/400 の Web 開発機能である「VCL for the Web（旧 IntraWeb）」では、従来のネイティブ Windows アプリケーション開発と同じような手法で、Web アプリケーション開発を行うことができる。

本稿では、Delphi/400 の VCL for the Web を利用した基本的な開発方法と各種テクニックの紹介を目的とする。

そこで今回は、簡単な商品検索の Web 照会アプリケーション作成を例に挙げて、VCL for the Web の基本的な手順を、以下の流れにそって説明していく。（Web 照会アプリケーションの完成画面は、手順最後の図 5 を参照）【図 5】

開発形態の決定→画面の設計→プログラムロジック開発

そして、応用テクニックと開発端末での Web サーバー検証環境の作成についても簡単にふれていきたい。

VCL for the Web での開発用途 / 形態

アプリケーションの種類

VCL for the Web（以下 IntraWeb）で Web アプリケーションを動作させるには、以下の 3 種類の方法がある。

A スタンドアローンモード
プログラミング作業と実行を手軽に行う場合、このモードを選択する。このモードでは Web サーバーも IntraWeb が提供するため、Windows アプリケーションと同じ実行形式が作られる。

B アプリケーションモード
既存の Web サーバー（IIS）を利用し、簡単にプログラミングを行いたい場合はこのモードを選択する。メモリや DB コネクションに余裕がある場合は、このモードが一番利用しやすい。

C ページモード
Web アプリケーションに WebSnap または WebBroker を利用する。リソースを再利用して大量の要求を受け付ける場合は、このモードを利用する。

用途によって、さまざまなモードを選択し、開発を行っていく必要がある。

本稿では初心者の方でも一番手軽に行うことができる、アプリケーションモードを利用した開発方法を記述する。

アプリケーションモードのWeb開発

それでは、アプリケーションモードを用いてのWebプログラミングの開発方法を順に説明していく。

まず、新規プロジェクトを作成する。この時、新規プロジェクトを作成するには［ファイル］→［その他］→［Delphiプロジェクト/VCL for the Web］→［VCL for the Web Application Wizard］を選択する。【図1a】【図1b】

すると、Webアプリケーションのウィザードが開始される。ここで表示されるApplication Typeというのが、上記で説明した3種のモードに該当する。ここでは、アプリケーションモードのISAPI Extensionを選択する。

Optionは特に個別に設定する必要がないため、初期値の状態で設定し、Project Name、Project Directoryは任意の値を設定する。すべての設定が完了した後に、［OK］ボタンを押下すれば、アプリケーションモードでの開発画面に遷移する。【図2】

開発画面には、ServerController.pas、Unit.pas、UserSessionUnit.pasの3種が自動生成される。ServerControllerはその名の通り、サーバーのコントロールを司るファイルで、ブラウザの戻るボタンやIntraWeb終了時の制御を行う等のことができる。Unitファイルは実際に開発を行う画面で、このファイルにコンポーネントを貼り付けて開発を行う。UserSessionUnitは、アプリケーション開発を行う時のDataModuleと同じ扱いになる。

今回は、条件を指定して検索ボタンを押下すると商品の検索を行うことができる、という簡単な照会システムの作成手順を説明する。

VCL for the Webでの画面設計

IntraWebで使用できるコンポーネントは、C/Sアプリケーションを開発するコンポーネントとは違ってくる。ツールパレットの中で、頭にIWがつくコンポーネント群が使用できるコンポーネントになる。簡単な開発であればIW Standard群が主に使用される。

今回の開発で使用するコンポーネントとしては、TIWEdit、TIWGrid、TIWButton、TIWImageFile、TIWLabel、TIWRegion、TIWLink、TIWRadioButtonがある。コンポーネントの配置方法は従来のアプリケーション開発と変わらず、表示を行う箇所にコンポーネントの配置を行い、必要に応じてプロパティの設定を行う。【図3】

簡単に各コンポーネントと主なプロパティの説明を、以下に記述する。

- **TIWEdit**
  ブラウザ画面で文字等の入力制御を行う。プロパティのMaxlengthに数値を入れることで、最大入力文字数が指定できる。
- **TIWGrid**
  表形式の画面を出力する。HTMLで言えばTABLEに相当する。プロパティのUseFrameをTrueにすることで、開発画面上で指定した幅を固定にできる。Falseの場合は、出力するデータによってGridの幅が変化する。また、Lineの設定や列の間隔を設定することも可能。
- **TIWButton**
  画面上でクリックすることで、イベントを呼び出す。
- **TIWImageFile**
  画像を画面上に表示するコンポーネント。プロパティのImageFileのFile Nameまたは画像保管場所のURLを指定することで、画像を表示できる。
- **TIWLabel**
  テキストを表示するコンポーネント。
- **TIWRegion**
  TPanelと同じ働きをするコンポーネント。
- **TIWLink**
  IntraWeb内の画面に遷移するコンポーネント。似たコンポーネントにTIWURLがあり、こちらのコンポーネントはIntraWeb外の画面に遷移する時に使用する。
- **TIWRadioButton**
  複数の選択肢から条件を指定できるコンポーネント。プロパティのGroupで同じ名前を指定したものが同一選択対象になる。

画面の構成が決定した後に、IBM iとの連携を行うための非ビジュアルコンポーネントの配置を行う。よくWebアプリケーションで課題となるのが、サーバーからの応答時間の問題である。データベースとの接続方法はさまざまな方法があるが、Delphi/400で最もレスポンスが早い単一方向データセットを採用しているDBExpressを使用することが多い。

そのため、今回の説明もDBExpress接続方式を採用する。DBExpressのTSQLConnection、TSQLQueryをUserSessionUnitのフォームに配置し、TSQLConnectionを右クリックして接続設定を図のようにする。【図4】

ここで設定するDataBaseNameとHostNameには、Configurationで扱う接続名を設定する。これらの設定はプログラム内部で指定することが可能だが、必要に応じてプロパティで設定をする。

VCL for the Webでのプログラムロジック開発

次にプログラム内部の設計に入る。プログラム内部の設計は従来のアプリケーション開発と変わらず、各コンポーネントにあるイベントにプログラムを記述していけばよい。

例えば、メッセージのみの出力だけを考えると、Windowsアプリケーションでは ShowMessage関数があるが、同様にWebアプリケーションでもShow Message関数が存在する。

```
<Windows アプリケーション>
Application.ShowMessage（'Hellow World !'）;
<Web アプリケーション>
WebApplication.ShowMessage
('Hellow World !'）;
```

OnClickイベント

今回のプログラムでは、TIWButtonにあるOnClickイベントに検索を実行するプログラムを記述する。ボタンを押下することで、Queryを利用してIBM iに登録されているデータベースにアクセスを行い、検索条件に一致したレコードを取得する。【ソース1】

お気づきかもしれないが、OnClick

図1a

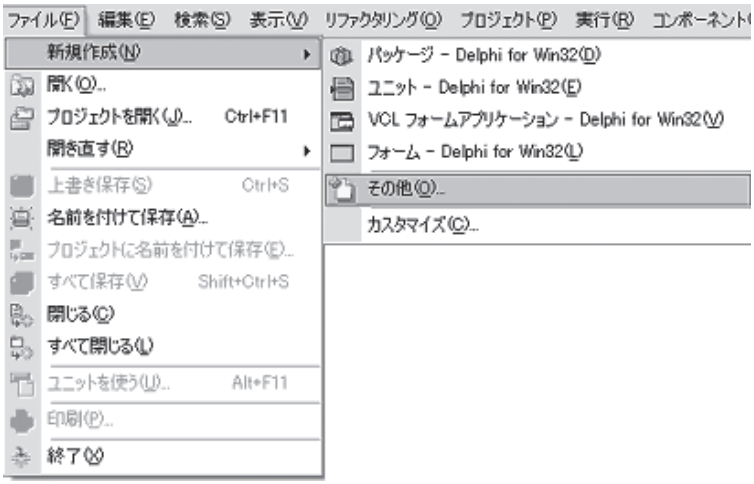

ファイル(F)
編集(E)
検索(S)
表示(V)
リファクタリング(O)
プロジェクト(P)
実行(R)
新規作成(N)
開く(O)...
プロジェクトを開く(J)... Ctrl+F11
開き直す(R)
上書き保存(S) Ctrl+S
名前を付けて保存(A)...
プロジェクトに名前を付けて保存(E)...
すべて保存(V) Shift+Ctrl+S
閉じる(C)
すべて閉じる(L)
ユニットを使う(U)... Alt+F11
印刷(P)...
終了(X)
パッケージ - Delphi for Win32(D)
ユニット - Delphi for Win32(E)
VCL フォームアプリケーション - Delphi for Win32(V)
フォーム - Delphi for Win32(L)
その他(O)...
カスタマイズ(C)...

図1b

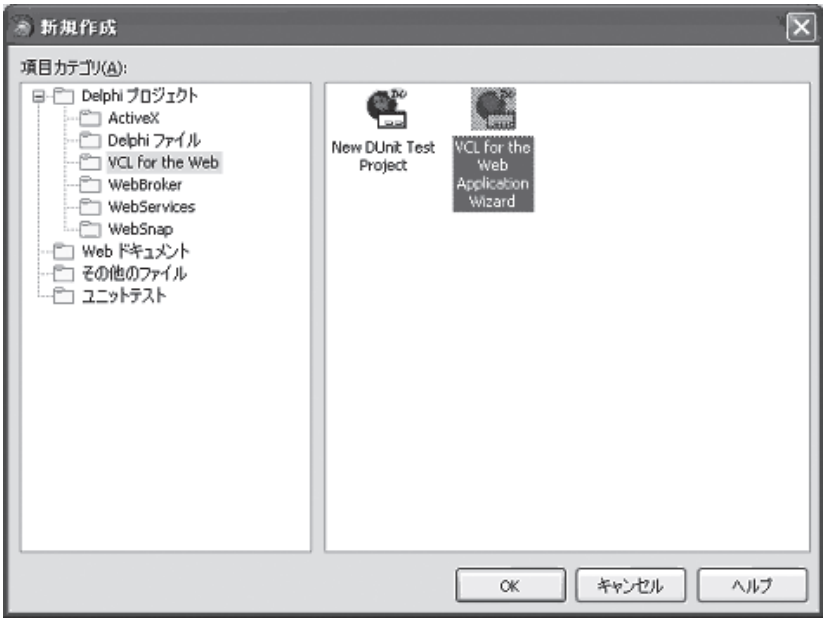
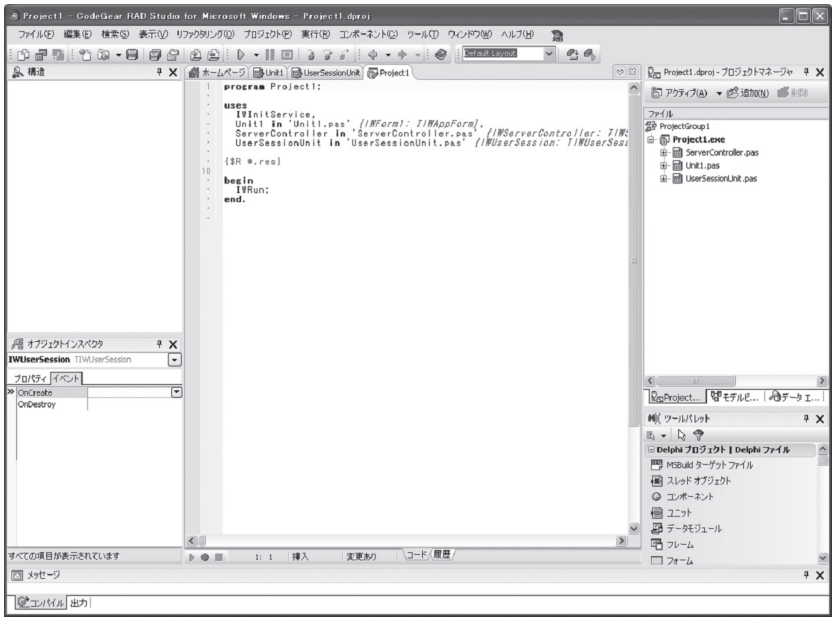

新規作成
項目カテゴリ(A):
Delphi プロジェクト
ActiveX
Delphi ファイル
VCL for the Web
WebBroker
WebServices
WebSnap
Web ドキュメント
その他のファイル
ユニットテスト
New DUnit Test Project
VCL for the Web Application Wizard
OK
キャンセル
ヘルプ

図2



イベントは通常の OnClick イベントと OnAsyncClick イベントの 2 種類がある。これら 2 つのイベントはボタンを押下時に呼び出されるイベントである。

まず OnClick イベントでは、押下時にデータをサーバーに通知して、処理結果を得た後にページ全体をロードする（Submit を行う）というイベントになっている。

一方、OnAsyncClick イベントは、指定した URL から XML ドキュメントを読み込む機能を使い、ユーザーの操作や画面描画等と並行する Ajax のようなイベントになる。

そのため、画面上のみで制御を行う場合は OnAsyncClick にイベントを記述するほうがよいが、逆のパターンでは OnClick イベントに記述するほうがよい。また、OnAsyncClick イベントでは、Submit が行われないためコンポーネントの画面表示切替処理などの制御が行えない制約がつく。

明細部への画像表示

また、今回は明細部の中に画像を表示する仕様にする。画像ファイルはサーバー側の特定のフォルダに保存されていることを前提に、IBM i のデータベースにはそのパスを保持するようにする。

画像を特定のパスで読み込むためには、前述した TIWImageFile を使用する。このコンポーネントをプログラム内部で生成するようにし、URL プロパティに画像を保存しているパスを指定することで、明細部の中に画像の表示を行うことができる。

DLLの作成

一通りの制御開発が終了した時点で、［プロジェクト］→［再構築］を選択し、プログラムの DLL を作成する。［プロジェクト］→［コンパイル］を選択しても問題はないが、再構築を行うことで中間ファイル等を再生成してくれるため、再構築を勧める。

この作成した DLL を外部からアクセスできる Web サーバーに設置し（それに関連する画像とのファイルも設置する）、ブラウザの URL に DLL が保管されているパスを指定する。

すると、Web アプリケーションが起動する。【図 5】

VCL for the Web での応用開発

さらに、一歩進んだことを行ってみよう。よく他のサイトをみるとクリックした画像が大きく表示されたり、文字のフォントを統一していたりするサイトが多く目立つ。IntraWeb でも JavaScript やスタイルシートが利用できる。

今回の開発では、明細に表示した画像をクリックすることで新しい画面を作り、画像を表示する JavaScript と、明細に表示する画像の大きさを統一するスタイルシートを組み込むことにする。

スタイルシートとの連携

スタイルシートとは Web ページのレイアウトを定義する技術であり、一般的にスタイルシートと言われると、スタイルシート言語の 1 つである CSS を指す。本文でもスタイルシートのことを CSS として明記する。

HTML を少しでも知っている方は、CSS を読み込む際に、その記述を HTML のヘッダにすることをご存知だろう。IntraWeb では、Form のプロパティに ExtraHeader というのがある。ここに CSS を読み込む処理を記述するか、プログラム内部で記述するかの 2 種類の方法がある。今回の開発では、動的に使用できるようにプログラム内部に記述する。【ソース 2】

次に、CSS を利用するコンポーネントに設定を行う。CSS の設定できるコンポーネントには、プロパティに CSS というのがある。ここに、CSS に記述されているセレクタを記述すれば、そのコンポーネントはスタイルシートに従った表記になる。【ソース 3】

JavaScriptとの連携

JavaScript とは Web ページに動きを追加する簡易言語のことで、画像をクリックした時に新しいページを表示させたり、TAB の移動といった制御等を行うことができる。

通常の Web 開発では、HEAD タグ内に JavaScript を記述する。IntraWeb でも同様に HEAD タグ内に JavaScript の記述が行えるが、今回は明細の画像をクリックすることで詳細な画像を表示させる仕様にすることから、コンポーネントに JavaScript を記述する方針をとる。

コンポーネントに JavaScript を記述するには、プロパティの ScriptEvents に記述するか、プログラム内部で記述するかの 2 種類がある。静的なコンポーネントであれば ScriptEvents に記述するほうがよいが、今回は動的に画像ファイルを作成しているので、プログラム内部で JavaScript を記述する。【ソース 3】

ソース 3 のように記述を行えば、実行し、画像をクリックした時に仕様どおりの動きを行うことができる。他にも JavaScript を駆使すれば、Enter キーでフォーカスの移動が行えたり、コンポーネントのイベントと連携を行ったりすることが可能になる。

開発端末での Web サーバー環境構築

ここまでは Web アプリケーション作成を説明してきたが、作成したアプリケーションを Web サーバー上で動作検証するためのテクニックを説明する。

ここでは開発端末に Web サーバーを構築し、作成した DLL が正常に動作するか検証を行えるようにする。

今回の動作環境は、Windows XP の IIS（バージョン 5.1）を使用する。Windows のコントロールパネルの中に［管理ツール］→［インターネットインフォメーションサービス］（以下 IIS）というのがある。（ない場合は［プログラムの追加と削除］→［Windows コンポーネントの追加と削除］からインストールが可能）。この IIS は、Microsoft 社のインターネットサーバーソフトウェアで、この IIS を構築することで Web サーバーを作ることができる。

構築するやり方として、IIS を起動すると、Web サイトという項目の下に既存の Web サイトという項目がある。ここの項目に指定したフォルダが Web サーバーの仮想ディレクトリとなり、外部からのアクセスが可能となる。既存の Web サイトの項目で、右クリックを行うと［新規作成］という項目がある。ここの項目を選択し、ウィザードに従って仮想ディレクトリを作成する。

ここで注意しなければならないのが、DLL を実行するためのアクセス許可であり、ISAPI アプリケーションや CGI

図3

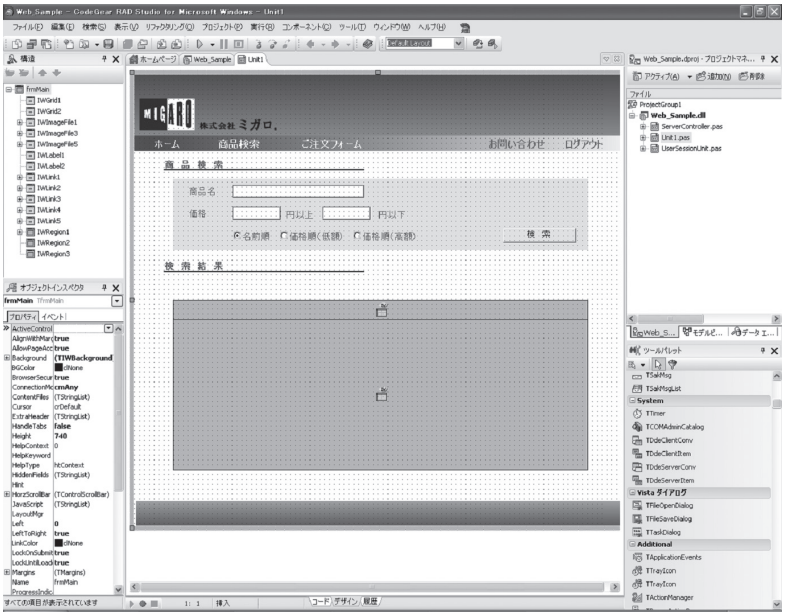

株式会社 ミガロ.
ホーム
商品検索
ご注文フォーム
お問い合わせ
ログアウト
商品検索
検索結果
検索

図4

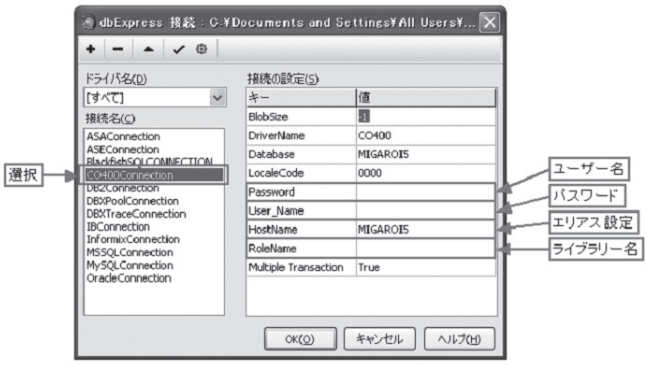
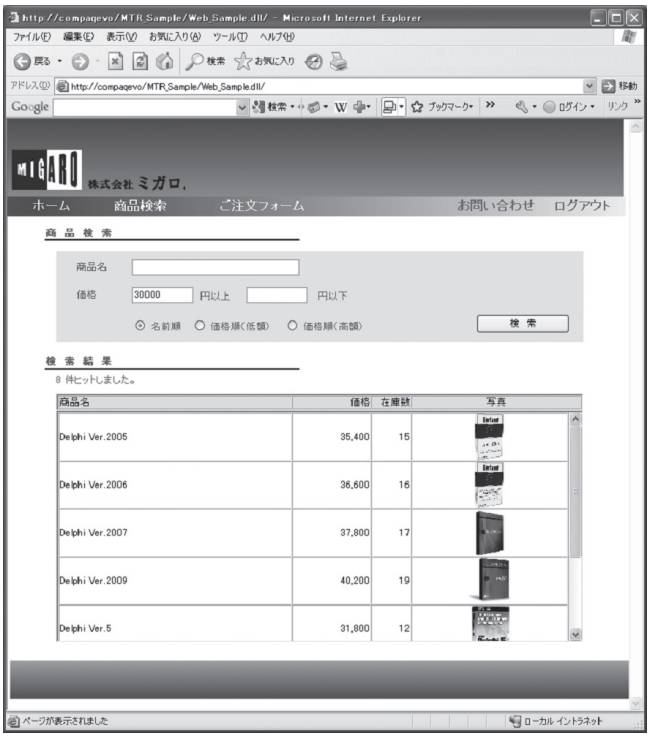

dbExpress 接続 : C:¥Documents and Settings¥All Users¥...
ドライバ名(D)
[すべて]
接続名(C)
ASAConnection
ASEConnection
BlackfishSQLCONNECTION
CO400Connection
DB2Connection
DBXPoolConnection
DBXTraceConnection
IBConnection
InformixConnection
MSSQLConnection
MySQLConnection
OracleConnection
接続の設定(S)
キー
値
BlobSize
DriverName
CO400
Database
MIGARO15
LocaleCode
0000
Password
User_Name
HostName
MIGARO15
RoleName
Multiple Transaction
True
OK(O)
キャンセル
ヘルプ(H)
選択
ユーザー名
パスワード
エリアス設定
ライブラリー名

図5


http://compagevo/MTRSample/Web_Sample.dll/ - Microsoft Internet Explorer
株式会社 ミガロ.
ホーム
商品検索
ご注文フォーム
お問い合わせ
ログアウト
商品検索
商品名
価格
30000
円以上
円以下
名前順
価格順(低額)
価格順(高額)
検索
検索結果
8 件ヒットしました。
商品名
価格
在庫数
写真
Delphi Ver.2005
35,400
15
Delphi Ver.2006
36,600
16
Delphi Ver.2007
37,800
17
Delphi Ver.2009
40,200
19
Delphi Ver.5
31,800
12

などを実行するための許可を追加しなければならない。【図6】

作られたディレクトリにDLL等のファイルを設置すれば、ブラウザからアクセスが可能となる。

ブラウザからアクセスする際は、以下のようにアクセスを行う。

http:// サーバー名（コンピュータ名）/ 設定した仮想ディレクトリ /DLL ファイル

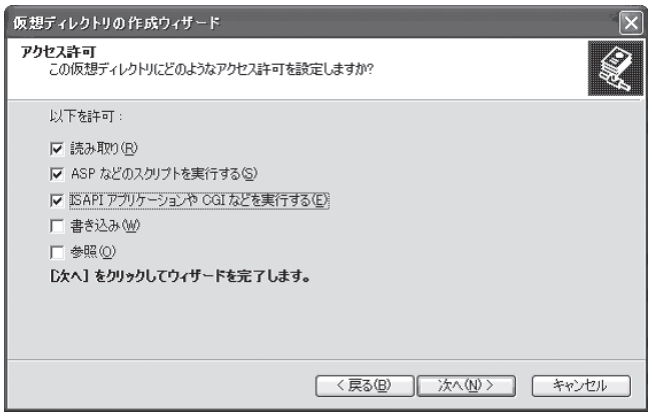
おわりに

はじめに記述したように、私の中ではDelphiのC/Sアプリケーション開発と同様にWebアプリケーションが作成できることがおおいに嬉しかった。これまでのIntraWebは、主に社内ネットワークアプリケーションとして使用されてきたが、機能が追加されバージョンが上がったことで、Webアプリケーションとしての活用が見込める存在となった。

今回の事例は照会のみのアプリケーションであったが、登録やメール送信、CGIなどもこのアプリケーションに組み込むことが可能である。今後のWeb開発にぜひとも役立てていただきたい。

M

図6



ソース1

Unit1.pas - TeraPad

ファイル(F) 編集(E) 検索(S) 表示(V) ウィンドウ(W) ツール(T) ヘルプ(H)

```
55
56  {*****************************************************************************************
57  □目的：「検索」ボタン押下時処理
58  □引数：なし
59    戻値：なし
60  *****************************************************************************************}
61  procedure TfrmMain.btnSearchClick(Sender: TObject);
62  const
63    //SQLメイン
64    sSQLMain = 'SELECT * FROM SHMAS WHERE SHDLFG IS NOT NULL ';
65
66    //SQL条件
67    sSQLStr1 = 'AND SHNAME LIKE :SHNAME '; //商品名
68    sSQLStr2 = 'AND SHKKKU >= :SHKKKU1 ';   //価格
69    sSQLStr3 = 'AND SHKKKU <= :SHKKKU2 ';
70
71    //並び替え
72    sSQLOrder1 = 'ORDER BY SHNAME ASC';
73    sSQLOrder2 = 'ORDER BY SHKKKU ASC ,SHNAME ASC';
74    sSQLOrder3 = 'ORDER BY SHKKKU DESC,SHNAME ASC';
75  begin
76
77    with UserSession do
78    begin
79      if qryMain.Active then qryMain.Close;
80      qryMain.SQL.Clear;
81      qryMain.SQL.Add(sSQLMain);
82
83      //条件
84      //商品名
85      if Trim(edtSHNAME.Text)<>'' then
86      begin
87        qryMain.SQL.Add(sSQLStr1);
88        qryMain.ParamByName('SHNAME').AsString := '%'+edtSHNAME.Text+'%';
89      end;
90      //価格（最低）
91      if Trim(edtSHKKKUF.Text)<>'' then
92      begin
93        qryMain.SQL.Add(sSQLStr2);
94        qryMain.ParamByName('SHKKKU1').AsInteger := StrToInt(edtSHKKKUF.Text);
95      end;
96      //価格（最高）
97      if Trim(edtSHKKKUT.Text)<>'' then
98      begin
99        qryMain.SQL.Add(sSQLStr2);
100       qryMain.ParamByName('SHKKKU2').AsInteger := StrToInt(edtSHKKKUT.Text);
101     end;
102
103     //並び替え
104         if rbtnG1.Checked then qryMain.SQL.Add(sSQLOrder1)
105     else if rbtnG2.Checked then qryMain.SQL.Add(sSQLOrder2)
106     else qryMain.SQL.Add(sSQLOrder3);
107
108     //SQL実行
109     try
110       qryMain.Open;
111
112       //Gridに描画する
113       GridDraw;
114     finally
115       if qryMain.Active then qryMain.Close;
116     end;
117   end;
118 end;
119
```

ソース2

Unit1.pas - TeraPad

ファイル(F)　編集(E)　検索(S)　表示(V)　ウィンドウ(W)　ツール(T)　ヘルプ(H)

```
212
213 {*****************************************************************************************
214 □目的：画面読み込み時
215 □引数：なし
216   戻値：なし
217 *****************************************************************************************}
218 procedure TfrmMain.IWAppFormRender(Sender: TObject);
219 const
220   Row0 = '300';
221   Row1 = '100';
222   Row2 = '50';
223 begin
224   //Imageファイル HOST取得
225   imgHostPath := 'http://'+ WebApplication.Request.Host + '/MTR_Sample/Files/image';
226   //CSSファイル HOST取得
227   cssHostPath := 'http://'+ WebApplication.Request.Host + '/MTR_Sample/Files/css';
228   //CSS読み込み
229   ExtraHeader.Add(
230     '<link rel="stylesheet" type="text/css" href="'+cssHostPath+'/System.css" />');
231
232   //タイトル部
233   wgTitle.Cell[0,0].Width := Row0;
234   wgTitle.Cell[0,1].Width := Row1;
235   wgTitle.Cell[0,2].Width := Row2;
236   wgTitle.Cell[0,0].BGColor := clWebKHAKI;
237   wgTitle.Cell[0,1].BGColor := clWebKHAKI;
238   wgTitle.Cell[0,2].BGColor := clWebKHAKI;
239   wgTitle.Cell[0,3].BGColor := clWebKHAKI;
240   wgTitle.Cell[0,0].Alignment := taLeftJustify;
241   wgTitle.Cell[0,1].Alignment := taRightJustify;
242   wgTitle.Cell[0,2].Alignment := taRightJustify;
243   wgTitle.Cell[0,3].Alignment := taCenter;
244   wgTitle.Cell[0,0].Text := '商品名';
245   wgTitle.Cell[0,1].Text := '価格';
246   wgTitle.Cell[0,2].Text := '在庫数';
247   wgTitle.Cell[0,3].Text := '写真';
248   //名細部
249   wgMain.Cell[0,0].Width := Row0;
250   wgMain.Cell[0,1].Width := Row1;
251   wgMain.Cell[0,2].Width := Row2;
252 end;
253
254 initialization
255   TfrmMain.SetAsMainForm;
256
257 end.
```

ソース3

Unit1.pas - TeraPad

ファイル(F)　編集(E)　検索(S)　表示(V)　ウィンドウ(W)　ツール(T)　ヘルプ(H)

```
119
120 {*****************************************************************************************
121 □目的：Grid描画処理
122 □引数：なし
123   戻値：なし
124 *****************************************************************************************}
125 procedure TfrmMain.GridDraw;
126 const
127   Row0 = '300';
128   Row1 = '100';
129   Row2 = '50';
130 var
131   iRow : Integer; //変数
132   str  : string;
133 begin
134   //初期化
135   wgMain.RowCount := 1;
136   iRow := 0;
137   str := '';
138
139   with UserSession.qryMain do
140   begin
141     //明細部
142     wgMain.Refresh;
143     while not(EOF) do
144     begin
145       //商品名
146       wgMain.Cell[iRow,0].Width := Row0;
147       wgMain.Cell[iRow,0].Alignment := taLeftJustify;
148       wgMain.Cell[iRow,0].Text := FieldByName('SHNAME').AsString;
149       //価格
150       wgMain.Cell[iRow,1].Width := Row1;
151       wgMain.Cell[iRow,1].Alignment := taRightJustify;
152       wgMain.Cell[iRow,1].Text := FormatFloat('#,#',FieldByName('SHKKKU').AsInteger);
153       //在庫数
154       wgMain.Cell[iRow,2].Width := Row2;
155       wgMain.Cell[iRow,2].Alignment := taRightJustify;
156       wgMain.Cell[iRow,2].Text := FormatFloat('#,#',FieldByName('SHZIKO').AsInteger);
157       //写真
158       if FieldByName('SHPHTO').AsString <> '' then
159       begin
160         with wgMain.Cell[iRow, 3] do
161         begin
162           Alignment := taCenter;
163           Control := TIWImageFile.Create(Self);
164
165           //画像ファイルを作成する
166           with TIWImageFile(Control) do
167           begin
168             ImageFile.URL :=  imgHostPath+'/'+FieldByName('SHPHTO').AsString;
169             StyleRenderOptions.RenderSize := False;
170             AutoSize := True;
171             Css := 'image';
172             str := imgHostPath + '/' +FieldByName('SHPHTO').AsString;
173             str := StringReplace(str,'¥','/',[rfReplaceAll]);
174
175             //JavaScriptイベント
176             ScriptEvents.Values['onClick'] :=
177             'newWin = window.open("","new","width=480,height=360,resizable=1,menubar=1,sta
178             'newWin.location = "'+str+'";'+
179              'newWin.focus()';
180
181           end;
182         end;
183       end;
184
185       //行追加
186       wgMain.RowCount := wgMain.RowCount+1;
187
188       //変数追加
189       Inc(iRow);
190
191       Next;
192     end;
193
194     //該当レコードが0件の処理
195     if (iRow=0) then
196     begin
197       lblComment.Caption := '該当するレコードが存在しませんでした。';
198       wgTitle.Visible := False;
199       wgMain.Visible := False;
200     end
201     else
202     begin
203       //最終行削除
204       wgMain.RowCount := wgMain.RowCount-1;
205       lblComment.Caption := IntTostr(wgMain.RowCount)+' 件ヒットしました。';
206       wgTitle.Visible := True;
207       wgMain.Visible := True;
208     end;
209     lblComment.Visible := True;
210   end;
211 end;
212
```

尾崎 浩司

株式会社ミガロ.

システム事業部 システム3課

Delphi/400を使用したWebサービスアプリケーション

インターネット技術を応用し、XML 処理を行うというとたいへん敷居が高く感じる。実は、Delphi/400 を用いるとそれらは容易に使用可能である。

略歴
1973 年 8 月 16 日生れ
1996 年三重大学工学部卒
1999 年 10 月株式会社ミガロ. 入社
1999 年 10 月システム事業部配属

現在の仕事内容
ミガロ. 入社以来、主に Delphi/400 を利用した受託開発を担当している。

- Webサービスとは
- SOAPとREST
- SOAPの使用方法
- RESTの使用方法
- 最後に

Web サービスとは

Web サービスとは、インターネットの技術を活用し、遠隔サイトにあるアプリケーションの機能をネットワークを通じて、自社アプリケーションから利用できるようにしたものである。

インターネットが発達したおかげで、現在では有用な情報が容易に取得できるようになった。だが、従来の Web アプリケーションでは、取得した情報をそのまま自社のアプリケーションに取り込むことができなかったため、自社アプリケーション自体が有用な情報を直接活用するということは難しかった。しかし、Web サービスの普及により、企業の枠を超えた情報の連携および活用が可能になった。

一例を挙げると、従来であれば「乗換案内」のような路線情報を提供するサイトを使って、使用した路線の交通費をブラウザで確認し、その結果を自社の出張精算システムに手で入力するといった手順が必要であった。しかし、路線情報サイトの Web サービスを使用することで、直接、出張精算システムと連携することが可能になったのである。

つまり、使用した路線を指定すると、自動的に交通費を出張精算システムに登録するといったことができるわけだ。しかも、路線情報を提供するサイトは常に最新の運賃情報を提供しているため、自社アプリケーションでは、運賃マスターといった情報をまったく管理する必要がないというメリットもある。

このように Web サービスを使用することで、自社のアプリケーションの利便性を大きく向上させることができるのである。

では、Web サービスは、どのようにして、システム間の連携を可能にしているのだろうか？ 従来の Web システムでは、人がサイトにアクセスして処理結果をブラウザで見るのに対し、Web サービスでは、プログラムがサイトにアクセスして処理結果を XML 形式で受け取るというのが特徴である。

インターネットの技術を応用して、XML で処理を行うというとすごく敷居が高いような話にも思えるが、実は Delphi/400 を用いると、これらを容易に使用することが可能である。本稿では、具体例を挙げながら、Delphi/400 を使用した Web サービスの活用方法について触れていくこととする。

SOAPとREST

Web サービスを使用するにあたって、Web サービスを提供しているサイトを調査すると、おそらく使用方式に「SOAP」あるいは「REST」といった言葉がでてくるであろう。

SOAP とは、Simple Object Access Protocol の略で、SOAP メッセージという XML によってメッセージ交換を行う方法である。

対して、REST とは、Representational State Transfer の略で、HTTP の GET メソッドを使って指定された URL にアクセスすると、XML が返ってくるというものである。

図1

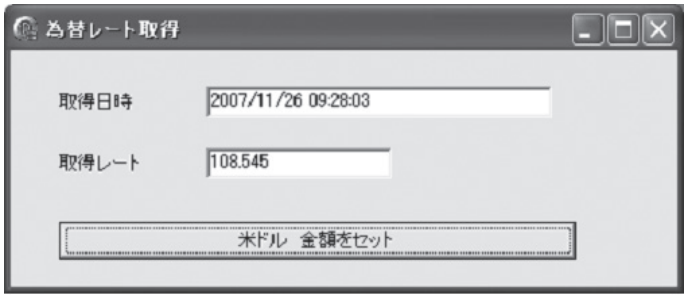

為替レート取得
取得日時
2007/11/26 09:28:03
取得レート
108.545
米ドル 金額をセット

図2

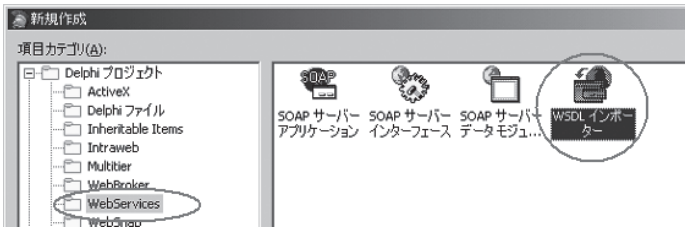
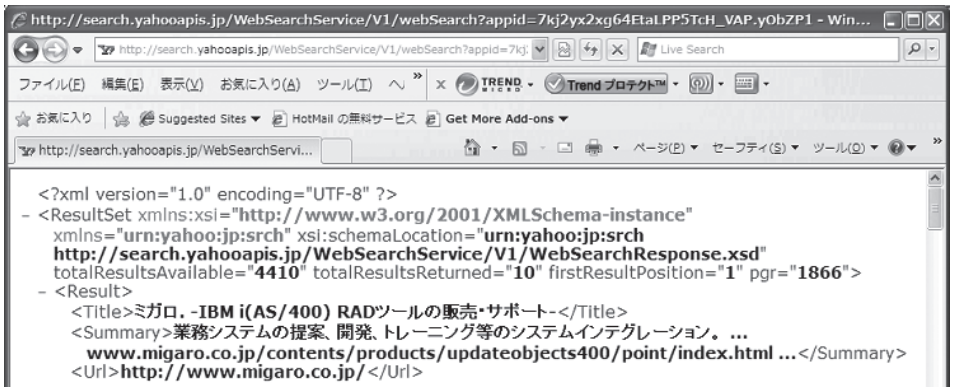

新規作成
項目カテゴリ(A):
Delphi プロジェクト
ActiveX
Delphi ファイル
Inheritable Items
Intraweb
Multitier
WebBroker
WebServices
SOAP サーバー アプリケーション
SOAP サーバー インターフェース
SOAP サーバー データモジュ...
WSDL インポーター

図3


WSDL インポートウィザード
WSDL ソース(W)
WSDL ファイルの位置または URL:
http://www.webservicex.net/CurrencyConvertor.as
UDDIを検索(U)..
オプション(O)
< 戻る(P
完了(F)
キャンセル
ヘルプ

図4


http://search.yahooapis.jp/WebSearchService/V1/webSearch?appid=7kj2yx2xg64EtaLPP5TcH_VAP.yObZP1 - Win...
<?xml version="1.0" encoding="UTF-8" ?>
<ResultSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:yahoo:jp:srch" xsi:schemaLocation="urn:yahoo:jp:srch
http://search.yahooapis.jp/WebSearchService/V1/WebSearchResponse.xsd"
totalResultsAvailable="4410" totalResultsReturned="10" firstResultPosition="1" pgr="1866">
<Result>
<Title>ミガロ. -IBM i(AS/400) RADツールの販売・サポート-</Title>
<Summary>業務システムの提案、開発、トレーニング等のシステムインテグレーション。...
www.migaro.co.jp/contents/products/updateobjects400/point/index.html ...</Summary>
<Url>http://www.migaro.co.jp/</Url>

一般的には、SOAP は、XML メッセージのやり取りを定義する必要があるため難易度が高く、REST は、URL を指定するだけで情報が XML で受け取れるため難易度が低いといわれる。

しかし、どちらの手法も Delphi/400 では容易に使用可能である。それでは、それぞれの使用方法について具体例を挙げながら説明を進めていこう。

SOAPの使用方法

SOAP を使用するのに必要なのが「WSDL」と呼ばれるものだ。WSDL というのは、Web Services Description Language の略で、XML でメッセージ交換を行うためのルールが定義されたものである。（WSDL 自体が XML で記述されている）。

この WSDL を使用することで、Delphi/400 アプリケーションから容易に Web サービスが利用可能になるのだ。

今回、SOAP の具体的な使用例として「Webservicex.Net」（http://www.webservicex.net/WCF/webServices.aspx）と呼ばれるサイトで公開されている、「Currency Convertor」という為替レート情報取得 Web サービスを使用してみよう。

この Web サービスを使用したサンプルプログラムを実行したアプリケーションが、図 1 である。このアプリケーションは、ボタンをクリックすると、米ドルに対する日本円のレートを表示するというものだ。【図 1】

Currency Convertor のサイトを確認すると、WSDL ロケーションとして、以下のように記載されていることがわかる。この情報をもとに、Delphi/400 で使用可能なクラスを生成することが可能なのである。

```
http://www.webservicex.net/CurrencyConvertor.asmx?WSDL
```

作成手順は次の通りである。

①新規プロジェクトを作成
②メニューより［ファイル］→［新規作成］→［その他］を選択
③［WebServices］から［WSDL インポーター］を選択【図 2】
④ WSDL ファイル URL を入力【図 3】
⑤［完了］ボタンを押下

上記ウィザードを実行すると、WSDL を解析し、Delphi/400 で使用可能なクラスユニットが自動生成されるので、これをアプリケーションから利用すればよい。

ソース 1 は、先程の図 1 のサンプルアプリケーションのソースコードである。【ソース 1】

自動生成された CurrencyConvertor ユニットを uses 節に含めることにより、為替レート取得のために定義されたクラスやメソッドが使用可能になったことがわかるであろう。

RESTの使用方法

REST は、先程も述べた通り、Web サービスを提供するサーバーに対して URL を与えると結果として XML を受け取れるというものだ。

REST の具体例としては、「Yahoo! デベロッパーネットワーク」（http://developer.yahoo.co.jp/）で提供されている Web サービスを紹介しよう。それらの Web サービスは、Yahoo! JAPAN で提供されているさまざまな機能を Web サービス化したものとなる。（サービスを使用するには、アプリケーション ID が必要なので、事前に取得しておこう）。

例えば Yahoo! 検索 Web サービスを使用し、キーワード ”migaro” で問い合わせを行うには、以下のような URL を使用する。

```
http://search.yahooapis.jp/WebSearchService/V1/webSearch?appid=<アプリケーション ID>&query=migaro
```

これをブラウザに入力し実行すると、図 4 のように XML が取得できることがわかる。【図 4】

Delphi/400 から REST の使用

このように REST とは、HTTP リクエストに対して XML をレスポンスとして返すものなのである。では、これを Delphi/400 から使用するにはどうすればよいだろうか？

Delphi/400 で HTTP クライアントを実装するのは、TIdHTTP と呼ばれるコンポーネントである。URL を指定して結果を取得するには、Get メソッドを使用すればよい。

ここで具体的なサンプルアプリケーションを紹介しよう。図 5 のように TIdHTTP コンポーネントを貼り付けたフォームを用意し、ソース 2 のようにボタンクリックイベントを記述する。【図 5】【ソース 2】

本サンプルは、Yahoo! ニュースのトピック用 Web サービスであるが、実行すると、メモコンポーネント内に、先程のブラウザの場合と同じように XML が表示されることがわかるだろう。

Delphi/400 から XML の扱い

これで Delphi/400 で XML が取得できることはご理解いただけたと思う。では、どうすればこの XML をプログラムからアクセスしやすくできるのだろうか？

XML 文書というのは、HTML と同じマークアップ言語と呼ばれるものだが、HTML とは違い使用するタグ自体の定義を自由に行えるのが特徴である。その定義にあたるのが XML スキーマと呼ばれるものだ。実はこの XML スキーマを使用すると、Delphi/400 から XML の扱いが容易になるのである。

Yahoo! ニュースのトピック用 Web サービスのサイトを確認すると、レスポンス用の XML スキーマの定義（拡張子 xsd）が記載されているのがわかる。これを Delphi/400 に取り込めばよいのだ。

その手順は次の通りである。

①［新規作成］→［その他］を選択
② XML フォルダの［XML データバインディング］を選択
③ソースコード欄に XSD ファイルを指定【図 6】
④ウィザードを進み、データバインディングの［設定を保存しない］に設定。【図 7】
⑤［完了］ボタンを押下

上記ウィザードを実行すると、XML スキーマを解析してできたクラスユニットが自動生成される。このユニットを使

図5


Form1
ニュース取得

図8

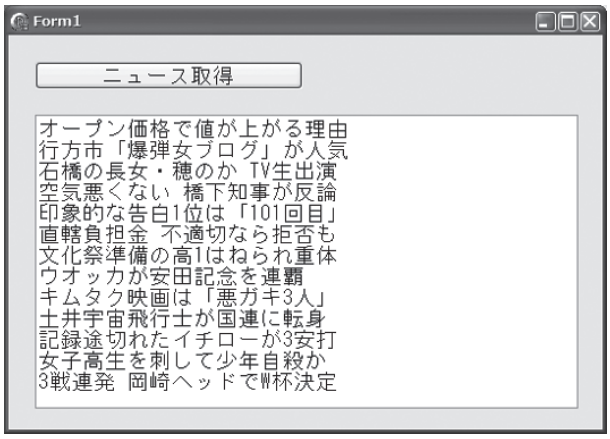

Form1
ニュース取得
オープン価格で値が上がる理由
行方市「爆弾女ブログ」が人気
石橋の長女・穂のか TV生出演
空気悪くない 橋下知事が反論
印象的な告白1位は「101回目」
直轄負担金 不適切なら拒否も
文化祭準備の高1はねられ重体
ウオッカが安田記念を連覇
キムタク映画は「悪ガキ3人」
土井宇宙飛行士が国連に転身
記録途切れたイチローが3安打
女子高生を刺して少年自殺か
3戦連発 岡崎ヘッドで㈽杯決定

図6


XML データバインディングウィザード
ソースコード
スキーマあるいは XML データファイル(S):
tp://news.yahooapis.jp/NewsWebService/V1/Topics.xsd
XDB 設定ファイルを使う(X):
オプション(O)
< 戻る(P)
次へ(N) >
キャンセル
ヘルプ(H)

図7


XML データバインディングウィザード
バインドの一覧(S)
生成されたインターフェイス
IXMLResultSet
IXMLResult
IXMLRelatedtopics
IXMLRelatedsite
生成コードのプレビュー(C):
{ IXMLResultSet }
IXMLResultSet = interface(IXMLNodeCollection)
['{6C72813D-FFDC-432A-96D6-C24A2B042FDA}']
{ プロパティ参照関数 }
function Get_TotalResultsReturned: Integer;
function Get_Result(Index: Integer): IXMLResult;
procedure Set_TotalResultsReturned(Value: Integer);
{ メソッドとプロパティ }
function Add: IXMLResult;
function Insert(const Index: Integer): IXMLResult;
property TotalResultsReturned: Integer read Get_TotalResultsRetu
property Result[Index: Integer]: IXMLResult read Get_Result; defa
end;
データバインディングの設定(S)
設定を保存しない(N)
スキーマファイルに保存(X)
ファイルに保存(I):
.jp/NewsWebService/V1/Topics.xdb
オプション(O)
< 戻る(P)
完了(F)
キャンセル
ヘルプ(H)

用することで、容易にXMLの各要素にアクセス可能になるのである。

なお、XML文書をDelphi/400から使用するには、TXMLDocumentコンポーネントを使用するのであわせて覚えておこう。

では、先程のサンプルアプリケーションを改良してみよう。TXMLDocumentコンポーネントを1つ追加した後に、XMLスキーマをデータバインディングウィザードでユニットを作成し、ソース3のようにプログラムを修正する。【ソース3】

XMLスキーマの定義にもとづき、ニューストピックの結果セット集合体や各要素がプログラムから使用可能になっているのがわかるだろう。

このサンプルプログラムを実行すると、図8のようにメモコンポーネントにトピックが一覧表示されるのである。また、このサンプルプログラムでは、ニュースの件数やトピック名等がクラス変数として取得できているのもわかる。【図8】

最後に

Webサービスを Delphi/400から使用するのは、SOAPであってもRESTであっても容易であることがおわかりいただけたであろう。

今回紹介した「Webservicex.Net」や「Yahoo! デベロッパーネットワーク」だけでなく、他にも多様なWebサービスが提供されているので、いろいろ試してみてほしい。そして、自社のアプリケーションとWebサービスをどう組み合わせるか、いろいろ検討してほしい。

自社アプリケーションにWebサービスを活用することで、これまで以上の利便性をユーザーに提供できると思われるので、ぜひともチャレンジしていただきたい。

M

ソース1

```
var
  Form1: TForm1;

implementation

uses CurrencyConvertor;

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
var
  rw: CurrencyConvertorSoap;
  cRate: System.Currency;     // ---- 通常のCurrency型はSystemユニットに定義
  dDate: TDateTime;
begin
  // 通貨情報を取得
  rw := GetCurrencyConvertorSoap();
  cRate := rw.ConversionRate(USD, JPY);
  dDate := Now;

  // 取得結果を画面表示
  edtDate.Text := FormatDateTime('YYYY/MM/DD HH:NN:SS', dDate);
  edtRate.Text := CurrToStr(cRate);
end;

end.
```

ソース2

```
implementation

{$R *.dfm}

procedure TForm1.btnGetClick(Sender: TObject);
const
  cCMD = 'http://news.yahooapis.jp/NewsWebService/V1/Topics'
    + '?appid=<実際はアプリケーションIDを記述>';
var
  sData: String;
begin
  //URLを指定して実行結果を取得
  sData := IdHTTP1.Get(cCMD);
  //文字コードを変換 (UTF8->Ansi)
  sData := Utf8ToAnsi(sData);
  //取得結果をメモコンポーネントに表示
  Memo1.Lines.Text := sData;
end;

end.
```

ソース3

```
uses newsyahooapisjpNewsWebServiceV1Topics;

{$R *.dfm}

procedure TForm1.btnGetClick(Sender: TObject);
const
  cCMD = 'http://news.yahooapis.jp/NewsWebService/V1/Topics'
    + '?appid=<実際はアプリケーションIDを記述>';
var
  i: Integer;
  sData: String;
  AXMLResultSet: IXMLResultSet; //結果セット
  AXMLResult: IXMLResult;       //1件の結果
begin
  //メモコンポーネントをクリアする
  Memo1.Clear;

  //URLを指定して実行結果を取得
  sData := IdHTTP1.Get(cCMD);
  //取得結果をXMLコンポーネントにセット
  XMLDocument1.LoadFromXML(sData);

  //XML結果セットの取得
  AXMLResultSet := GetResultSet(XMLDocument1);

  //結果セットより取得した件数分ループで情報を取得する
  for i := 0 to AXMLResultSet.TotalResultsReturned - 1 do
  begin
    //1件の情報を取得する
    AXMLResult := AXMLResultSet.Result[i];
    //メモコンポーネントに件名をセットする。
    Memo1.Lines.Add(AXMLResult.Title);
  end;
end;
```

吉原 泰介

株式会社ミガロ.

RAD事業部 技術支援課 顧客サポート

Delphi/400によるネイティブ資産の応用活用

ネイティブ資産を有効活用するための実践的なテクニックを紹介する。
SQL では実現が難しいが
ネイティブ資産やコマンドが利用できる Delphi/400 では
簡単に実現できる。

略歴
1978 年 3 月 26 日生れ
2001 年龍谷大学法学部卒
2005 年 07 月株式会社ミガロ. 入社
2005 年 07 月システム事業部配属
2007 年 04 月 RAD 事業部配属

現在の仕事内容
Delphi/400 や JACi400 の製品試験、および月 100 件に及ぶ問い合わせサポートとセミナー講師などを担当している。

●ネイティブ資産・コマンド
●ネイティブコマンドの活用
●scdtoolsユニットの活用
●まとめ

1.ネイティブ資産･コマンド

Delphi/400 でアプリケーション開発を行う利点として、IBM i の特有のネイティブ資産やコマンドを活用できるということが挙げられる。例えば、RPG や COBOL プログラムを Delphi/400 から利用できることが、Delphi/400 の大きな特長（機能）である。

もちろん、Delphi/400 では SQL も自由に扱えるため、これらネイティブ資産を使わずともアプリケーションを開発することは可能だ。だが、ネイティブ資産を有効に活用すると、さらにアプリケーション開発の幅を広げることができるのである。

次のようなことを考えたことはないだろうか。

●ライブラリ環境を自由に切り替えられたら・・・
● SQL でメンバが扱えたら・・・
● QUERY 資産を利用できたら・・・
●ライブラリやファイルのリストが取得できたら・・・
●スプールファイルを利用できたら・・・

これらは、IBM i 上では簡単にできることだが、SQL などでは単純に実現できない。逆にいうと、ネイティブ資産やコマンドを利用できる Delphi/400 では、簡単に実現することができるのである。

本稿では、こうした Delphi/400 からネイティブ資産を有効に活用するための実践的なテクニックを紹介していきたい。

2.ネイティブコマンドの活用

2-1.Delphi/400からのコマンド実行

Delphi/400 では、IBM i 上のコマンドを直接実行する機能がある。

具体的には、TAS400 コンポーネントの RemoteCmd メソッド、あるいは TCMD400 コンポーネントからコマンドを実行できる。

この 2 つのコンポーネントの用途の違いは、次の通りである。

● TAS400 コンポーネント RemoteCmd メソッド：パラメータなしのコマンド
● TCMD400 コンポーネント：パラメータを扱うコマンド

ここでは、TAS400 コンポーネントを使って説明する。例えば TAS400 コンポーネントでは、Name を AS4001 とすると、以下のようなコーディングだけで実行できる。

```
AS4001.RemoteCmd（' ネイティブコマンド ');
```

ここまで、Delphi/400 からの、IBM i 上のコマンド実行を紹介した。以降からは、ネイティブコマンドが有効となるような活用実例をいくつか紹介しよう。

2-2.ライブラリ環境に対するコマンド活用例

Delphi/400 で接続しているセッショ

図1

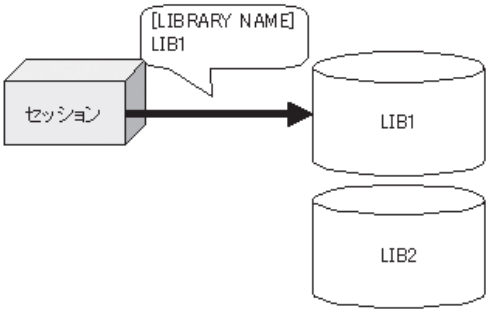

[LIBRARY NAME]
LIB1
セッション
LIB1
LIB2

図2

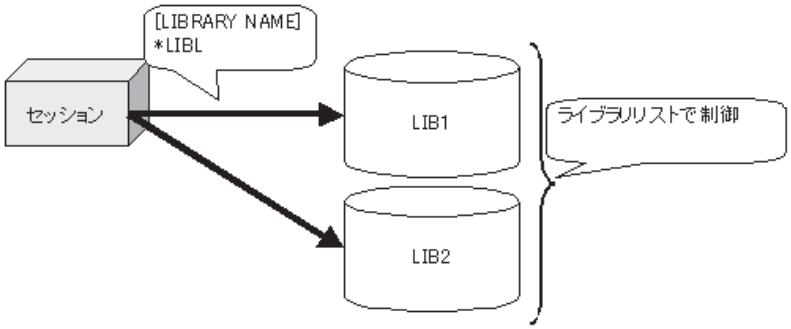

[LIBRARY NAME]
*LIBL
セッション
LIB1
LIB2
ライブラリリストで制御

図3

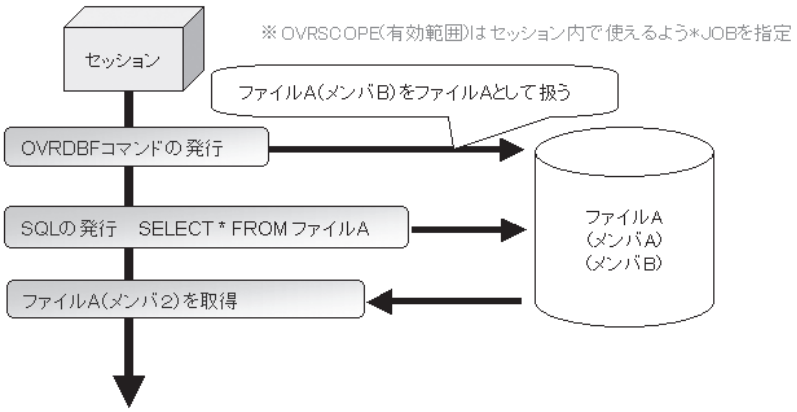

※OVRSCOPE(有効範囲)はセッション内で使えるよう*JOBを指定
セッション
ファイルA(メンバB)をファイルAとして扱う
OVRDBFコマンドの発行
SQLの発行　SELECT * FROM ファイルA
ファイルA(メンバ2)を取得
ファイルA
(メンバA)
(メンバB)

図4


セッション A - [24 x 80]
ファイル(F) 編集(E) 表示(V) 通信(C) アクション(A) ウィンドウ(W) ヘルプ(H)
報告書の表示
QUERY. . . : D4TECLIB/QRYTEST 報告書の幅 . . . . . : 261
行の位置指定 . . . . . 桁移動 . . . . . .
行 ...+...1...+...2...+...3...+...4...+...5...+...6...+...7..
部課 CD 受注No. 営業担当者 受注日 納期日 得意先 CD
000001 101 100,001 1 20,090,815 20,090,907 10,000,000
000002 101 100,002 2 20,090,815 20,090,906 20,000,000
000003 101 100,003 1 20,090,816 20,090,918 10,000,000
000004 101 100,004 1 20,090,817 20,090,901 10,000,000
000005 101 100,005 1 20,090,818 20,090,908 10,000,000
000006 102 100,006 3 20,090,820 20,090,910 30,000,000
000007 102 100,007 3 20,090,821 20,090,908 50,000,000
000008 101 100,008 1 20,090,821 20,090,908 10,000,000
000009 101 100,009 1 20,090,821 20,090,908 10,000,000
000010 102 100,010 3 20,090,822 20,090,924 40,000,000
000011 102 100,011 3 20,090,822 20,090,924 40,000,000
000012 102 100,012 3 20,090,822 20,090,906 50,000,000
000013 101 100,013 2 20,090,825 20,090,926 20,000,000
000014 102 100,014 3 20,090,826 20,090,925 30,000,000
****** ******** 報告書の終わり ********
終わり
F3= 終了 F12= 取り消し F19= 左 F20= 右 F21= 分割
英数 半角
03/032
D02 - セッションが正常に開始されました
FinePrint on FPR5:

ンでは、デフォルトのライブラリが設定されている。

例えばBDEの場合、BDEのエリアスまたはTDatabaseコンポーネントの［LIBRARY NAME］設定である。

通常はここに、アプリケーション上で使用したい参照ライブラリをデフォルトとして設定する。これを設定しておけば、例えばTTableコンポーネントなどでファイルの指定を行う場合、ライブラリ名の指定を省くことができる。【図1】

では、複数のライブラリを使用したい場合はどうすればよいか。もちろんTTableコンポーネントなどで、ファイルを’ライブラリ名/ファイル名’で直接指定することができる。だが、複数ライブラリ間でライブラリ指定を省略したい場合には、［LIBRARY NAME］に’*LIBL’と設定しておくとよいのである。

’*LIBL’と設定した場合、デフォルトの参照ライブラリはどこになるかというと、その接続セッションのライブラリリストが対象となる。【図2】

ここで、ネイティブコマンドを非常に有効に使うことができる。

例えば、LIB1, LIB2というライブラリを参照ライブラリにしたい場合、以下のようなコマンド実行を行うことで、接続中のセッションのライブラリリストにLIB1, LIB2というライブラリを追加することができ、これをデフォルトライブラリとして参照できるようになる。

```
AS4001.RemoteCmd('ADDLIBLE LIB1');
AS4001.RemoteCmd('ADDLIBLE LIB2');
```

つまり、このセッションのライブラリリストをデフォルトライブラリとして使用するので、ネイティブコマンドによって、例えば、本番環境のライブラリと試験環境のライブラリを簡単に切り替えるといったことができるようになるのである。

もちろん、これらのライブラリリストの設定を行うCLプログラムを用意しておき、TCall400コンポーネントを使用して制御することも可能である。

また、ライブラリリストを編集する際にCHGLIBコマンドを使用する場合は、すでに設定されているライブラリがリストから外されてしまう可能性があるので、注意が必要である。

2-3.SQLからメンバを扱うためのコマンド活用例

ファイルのメンバを利用したシステムの場合、SQLでの制約が問題となる。

TTableコンポーネントでファイルを指定する場合、TableNameプロパティに、次のように指定することができる。

```
ライブラリ名/ファイル名(メンバ名)
```

しかし、TQueryコンポーネントなど、SQL上では次のような指定になる。

```
ライブラリ名/ファイル名
```

この場合、メンバを指定することができないため、扱われるメンバは必ずファーストメンバがデフォルトになってしまう。つまり、メンバを利用したシステムにおいて、SQLは使える範囲が限定されてしまう。これは開発効率上、非常に問題がある。

ではSQLで、メンバを扱うためにはどうしたらよいだろうか。方法としては、SQL上ではメンバが指定できないのであれば、その指定をセッション上で事前に設定を行うことで可能にするというのはどうだろう。

ここで、ネイティブコマンドを非常に有効に使うことができる。データベース・ファイル一時変更（OVRDBF）コマンドを使うことで、セッション上でのファイル名の認識をメンバを含めて制御できるのである。

例えば、次のようなOVRDBFコマンドを実行する。

```
AS4001.RemoteCmd(‘OVRDBF
 FILE(ファイルA)
  TOFILE(ライブラリA/ファイルA)
  MBR(メンバB) OVRSCOPE(*JOB)’);
```

こうすると、セッション上でファイルAを扱うと、ファイルA（メンバB）を扱うことができるようになる。これによって、SQL内でメンバが指定できなくとも、実際には特定のメンバに対して処理を行うことができる。【図3】

なお、ここではコマンドでOVRSCOPEを指定しているが、これは、セッション上でOVRDBFを有効にするためである。これを指定しておかないと、実際にSQLで処理をする際に有効とならないので注意が必要である。

2-4.Queryを扱うためのコマンド活用例

IBM iのユーザーは、Query（ここではDelphi/400のTQueryコンポーネントではなくIBM i上のオブジェクト）を、データ抽出や集計といった業務で使用していることが非常に多い。【図4】

もちろんDelphi/400から、同じようなSQLなどを実行すれば、データを取得・集計することはできる。だが、Queryと同じ内容のものを、SQLとしてプログラムを新規に作成する必要がある。しかし、同じ内容のプログラムであれば、すでにIBM i上に存在しているQueryをそのまま利用できるほうがよいだろう。

実は、これもネイティブコマンドを応用活用することで実現することができる。

ただし、Delphi/400から直接Queryを利用するには、1つの課題がある。それは、Queryが対話型ジョブで利用する機能であるという点だ。Delphi/400は対話型CPW値を使用しないバッチ型のジョブであるため、対話型の処理を行うことができない。

では、どうやってDelphi/400からQueryを使用するか。RUNQRYというネイティブコマンドが利用できるのである。

通常、Queryは5250画面上においてRUNQRYを実行すると、結果が表示される。これは対話型で画面情報が返されているからである。しかしこの画面情報を、Delphi/400側では受け取ることができない。そのため、情報のアウトプットをファイルで出力できるように、RUNQRYコマンドのオプションで指定する必要がある。

例えば、Queryを実行した結果をQtemp上に結果ファイルとして出力する場合、次のようなRUNQRYコマンドを実行する。

```
AS4001.RemoteCmd(‘RUNQRY
 QRY(ライブラリ名/Query名)
  OUTTYPE(*OUTFILE)
  OUTFILE(QTEMP/出力ファイル名
  *FIRST *RPLFILE)’);
```

この後に、Delphi/400のTTableコ

図5

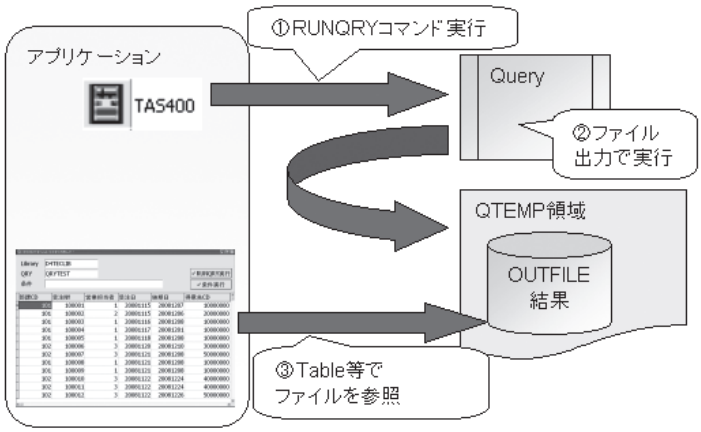


図6

```
procedure TfrmT1.Button1Click(Sender: TObject);
var
  LibraryName, QryName, RUNQRY : String; //Library名, QRY名, RUNQRY実行文
begin
  tblQRY.Close;                  // 使用ファイルをClose
  LibraryName  := EdtLIB.Text;   // Library名
  QryName      := EdtQRY.Text;   // QRY名
  //RUNQRY実行文の編集
  //RUNQRYを実行してQTEMPにQRY名の結果ファイルを作成
  RUNQRY :=  ('RUNQRY QRY(' + LibraryName + '/' + QryName + ') ' +
             'OUTTYPE(*OUTFILE) ' +  'OUTFILE(QTEMP/' + QryName + ' *FIRST *RPLFILE)');
  //RUNQRYの実行
  DMmain.As400.RemoteCmd(RUNQRY);
  //RUNQRYの実行結果ファイルをTableで取得
  tblQRY.TableName := 'QTEMP/' +  QryName;
  tblQRY.Open;
end;
```

図7

関数	機能
TcGetListLib	ライブラリのリストを取得
TcGetListFile	ファイルのリストを取得
TcGetListMbr	メンバのリストを取得
TcGetListDataArea	データエリアのリストを取得
TcGetListDataQueue	データキューのリストを取得
TcGetListOutQueue	アウトキューのリストを取得
TcGetListProg	プログラムのリストを取得

図8

```
procedure TForm1.FormCreate(Sender: TObject);
var
   List1:TStringList;            //Description用
begin
  AS4001.Active := true;         //ASへ接続
  List1 := TStringList.Create;   //Description用リストを作成
  ComboBox1.Items.Clear;         //コンボボックスクリア
  //関数を利用してライブラリのリストをコンボボックスへ設定
  TcGetListLib(AS4001.GetHandle, '*ALL', ComboBox1.Items, List1, 32000);
  List1.Free; //StringListの破棄
end;
```

ンポーネントなどで出力したファイルを読み込めば、Query の実行結果を画面に表示することができる。

なお、出力ファイルを Qtemp 上で扱うことで、出力ファイルの削除などの後処理を不要としている。【図 5】

この仕組みのコーディングは、図 6 のように非常に簡単に実現することができる。【図 6】

ポイントとしては、2 回目以降の実行のことを考慮して、RUNQRY コマンドで *RPLFILE を指定しておく。これにより、同じ Query を実行した場合、結果ファイルを上書きするようにできる。また、RUNQRY コマンド実行時にファイルがつかまれているとエラーの原因となる。そのため、処理の最初に、TTable コンポーネント（このサンプルコードでは tblQRY）は Close しておく必要がある。

3.scdtoolsユニットの活用

3-1.scdtoolsとは

Delphi/400 には、TFile400 コンポーネントの LibraryName プロパティで、ライブラリのリストを検索するダイアログが表示されて選択できる機能がある。

この機能は設計画面上の動作だが、Delphi/400 で実現できる。こうした機能を利用するために、Delphi/400 が提供しているのが「scdtools」である。

3-2.scdtoolsの使い方

「scdtools」はコンポーネントではなく、共通関数を提供するユニットとして存在する。「scdtools」に用意されている主な関数を図 7 に示す。パラメータなど詳しい使い方は、HELP の scdtools にも記載されている。【図 7】

ここでは、ライブラリのリストを取得する例で基本的な活用方法を説明する。「scdtools」の TcGetListLib という関数を活用することで、簡単にライブラリのリストを取得することができる。

次のようなプログラムを作ってみよう。

① Uses 節に scdtools を追記。
② TAS400、TComboBox を画面に配置。
③ FormCreate のイベントにプログラムを記述。【図 8】

画面を起動すると、コンボボックスにライブラリのリストが表示できる。これによって、画面からユーザーが使用するライブラリを選択して、指定することが可能になる。【図 9】

また上記はライブラリのリスト取得の例であるが、同様の使い方でファイルやメンバ、データエリア、データキュー、アウトキュー、プログラムなどのオブジェクトのリストを取得することもできる。動的なプログラムを作成する場合に非常に便利である。

3-3.scdtoolsの応用活用例

この「scdtools」のオブジェクトリストの取得を応用して、スプールファイルの照会画面を作成しよう。完成画面は図 10 に示す。【図 10】

このスプールファイルを照会する画面を作るためには、次の機能の実装が必要となる。

①ライブラリのリスト取得
②アウトキューのリスト取得
③スプールのリスト取得
④スプールの取得

以下順番に、仕組みとコーディングのサンプルを提示する。

①ライブラリのリスト取得
（TComboBox のリストに設定）

これは「scdtools」の TcGetListLib 関数で取得することができる。前述(3-2)をそのまま参考にして実装が可能である。

②アウトキューのリスト取得
（TComboBox のリストに設定）

これは「scdtools」の TcGetListOutqueue 関数で取得することができる。関数の使用方法は TcGetListLib 関数とほぼ同じ。パラメータに、アウトキューを取得する対象のライブラリが増えているだけである。【図 11】

③スプールのリスト取得

スプールリストの取得には、Delphi/400 で専用の TListSpool400 というコンポーネントが用意されているので、これが利用できる。

TListSpool400 コンポーネントのプロパティで、ライブラリとアウトキューを設定する必要がある。①②のコンボボックスで選択されている値を設定して、Active プロパティを True にして接続すれば、TTable のようなデータセットの形でスプールのリストデータを取得することができる。【図 12】

これは TDataSource、TDBGrid コンポーネントでそのまま画面に表示することができる。

④スプールの取得

最後にスプール情報の取得には、これも Delphi/400 で専用の TSpool400 というコンポーネントが用意されているので、これを利用する。

TSpool400 コンポーネントのプロパティで、スプール名、スプールナンバー、ジョブ名、ジョブナンバー、ユーザー名を設定する必要がある。これらの情報は③で取得している TListSpool400 コンポーネントですべて項目として持っているので、その値をプロパティに転送するだけである。【図 13】

そして、TListSpool400 も同様に、Active プロパティを True にして接続すれば、Spool のデータを取得できる。これも TDataSource、TDBGrid コンポーネントでそのまま画面に表示する。

以上で、スプールファイルの照会画面が完成である。【図 10】

4.まとめ

このように Delphi/400 を使用する際にひと工夫すると、IBM i のネイティブ資産の活用範囲をさらに広げることができる。どれも SQL などでは実現が難しい内容だが、Delphi/400 のネイティブコマンドやコンポーネントを使用することで簡単に実現することができる。

本稿では、よく使われる実践的な応用テクニックを紹介した。Delphi/400 でのアプリケーション開発時に、IBM i の資産をより有効に活用する参考にしていただきたい。

M

図9

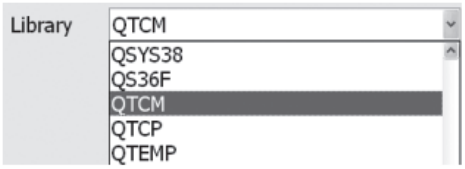


図10

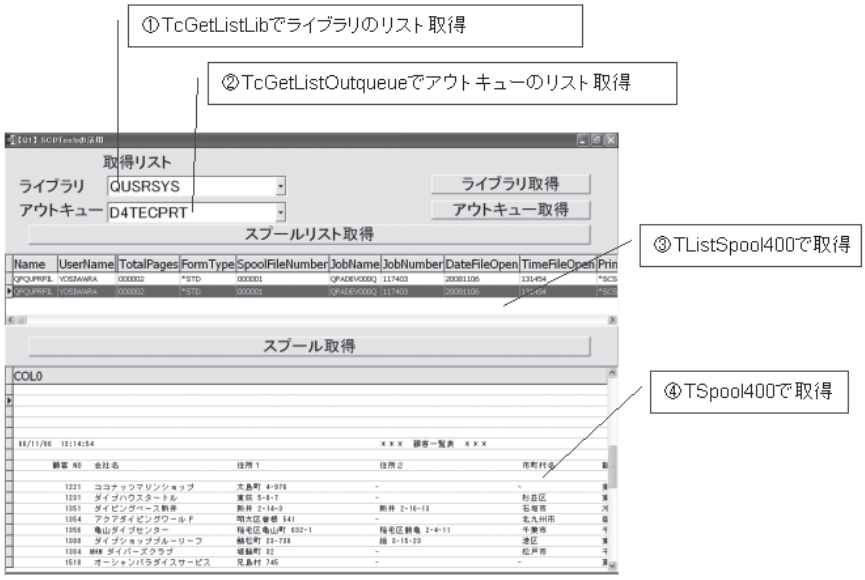


図11

使用コンポーネント
cbLIB:　TcomboboxコンポーネントcbOUTQ:　TcomboboxコンポーネントAS4001:　TAS400コンポーネント

```
procedure TForm1.btnOUTQClick(Sender: TObject);
var
  List1 : TStringList;   //Discription格納用StringList
begin
  //Discription格納用StringListの生成
  List1 := TStringList.Create;
  //アウトキュー用コンボボックスの初期化
  cbOUTQ.Items.Clear;
  //アウトキューリストの取得
  TcGetListOutqueue(As4001.GetHandle,          //接続ハンドル
                    '*ALL'                     //絞込み文字列
                     Trim(cbLIB.Text),         //ライブラリ名
                    cbOUTQ.Items,              //アウトキューリスト（戻り）
                    List1,                     //アウトキュー記述リスト（戻り）
                    32000);                    //バッファサイズ
  List1.Free; //StringListの破棄
end;
```

図12

```
procedure TForm1.btnListSpoolClick(Sender: TObject);
begin
  //TListSpool400のプロパティを設定してリストを取得
  with ListSpool4001 do
  begin
    Active      := false;              //切断
    LibraryName := Trim(cbLIB.Text);   //ライブラリ名
    OutQName    := Trim(cbOUTQ.Text);  //アウトキュー名
    Active      := true;               //接続
  end;
end;
```

使用コンポーネント
ListSpool4001：TListSpool400コンポーネント

図13

```
procedure TForm1.btnSpoolClick(Sender: TObject);
begin
  with Spool4001 do
  begin
    Active      := false;  //切断
    //※ワーク名をクリアしておかないと2回目同じワークとなります。
    WorkFile    := '';
    //スプール名
    SpoolName   := ListSpool4001.FieldByName('Name').AsString;
    //スプールナンバー
    SpoolNumber := ListSpool4001.FieldByName('SpoolFileNumber').AsString;
    //ジョブ名
    JobName     := ListSpool4001.FieldByName('JobName').AsString;
    //ジョブナンバー
    JobNumber   := ListSpool4001.FieldByName('JobNumber').AsString;
    //ユーザー名
    User        := ListSpool4001.FieldByName('UserName').AsString;
    Active      := true;   //接続
  end;
end;
```

使用コンポーネント
Spool4001：TSpool400コンポーネント

松尾 悦郎

株式会社ミガロ.

システム事業部 システム1課

RPGでパフォーマンスを制御
―順次読み込みの方法とDelphi/400やJACi400との連携

どのようなシステムにも存在する一覧照会について
RPGプログラムで処理速度を制御するテクニックと
それをDelphiやJACi400で連携する方法を紹介する。

略歴
1979年6月16日生れ
2002年広島大学理学部卒
2006年株式会社ミガロ. 入社
2006年6月システム事業部配属

現在の仕事内容
主にJACi400を使ったWebアプリケーションの開発を担当しており、システムの要件定義から納品・フォローまでを行っている。

●はじめに
●基本的な一覧照会
●実践的な一覧照会
●順次読み込みの手法
●Delphi/400プログラムとの連携
●JACi400プログラムとの連携
●最後に

はじめに

今日、私たちが生活する社会にはたくさんの業種があり、いろいろなシステムが稼働している。同じ業種でも、企業によっては、全く違うアプリケーションが使われていることもある。それは、それぞれの企業の業務や取り組みが異なるので、仕方のないことであり当然である。1つとして同じシステムはない、と言っても過言ではないかもしれない。

このように多種多様なシステムが存在しているわけであるが、システム開発者が共通して考慮し、頭を抱えるポイントがある。それは「処理速度（パフォーマンス）」だ。「人間がコンピュータの応答時間に我慢できるのは3秒以内」という説を聞いたことがあるが、今では高速化が進み、検索ボタンを押して、表示されるまで10秒かかる照会プログラムはユーザーに使われない。一方で、ユーザーからの要望は難易度を増し、便利な機能を次々と求められるのである。

そこで今回は、どのようなシステムにも存在する一覧照会の機能について、IBMのRPGプログラムで処理速度を制御するテクニックと、それをDelphiやJACi400（Webシステム）で連携する方法を紹介したいと思う。

基本的な一覧照会

詳細に入る前に、一覧照会プログラムについて、確認の意味で簡単にパターン別の仕組みを説明する。

A　5250のアプリケーション

サブファイルを使用し、対象のレコードをこのサブファイルに書き込む。画面の制約があるため、1画面で表示できない内容はPAGEDOWNやPAGEUPで表示する。

B　Delphi/400のGUIアプリケーション

Delphi/400を使用したアプリケーションであれば、ワークファイル（中間ファイル）を使用し、対象レコードをいったんRPGでこのワークファイルに書き出す。それをDelphi/400で画面に表示し、スクロールさせて全件を表示させる。

簡単な仕様であれば、RPGを使わずに、SQLで抽出することもある。

C　Webアプリケーション

JACi400を使用したアプリケーションであれば、内部テーブルを使用し、対象レコードをいったんRPGでこの内部テーブルに保持する。その後、Webサーバーに対象のデータを送信し、JACi400を介してHTMLで作成された画面に表示させる。これも、スクロールさせて全件を表示させる。

以上が基本的な仕組みで、いずれもRPGで対象レコードを抽出するものである。

実践的な一覧照会

基本的な一覧照会プログラムの仕組みを確認したところで、いよいよ詳細な内容に入っていく。当然のことだが、実際

の場面ではいろいろと考慮しなければならない。代表的なものが対象レコード数である。

対象となるレコードが100件や200件であれば気にしなくてもよいかもしれないが、企業によっては、対象レコードが1万件や10万件、さらには100万件と膨大になることがほとんどで、この点を無視すると実際に使える（ユーザーが使用する）ものにはならない。

この場合、制御するポイントをまとめると、以下の2点となる。

●処理速度に関する、パフォーマンスの制御

検索を開始してから画面に表示し終えるまでの時間

●表示件数に関する、パフォーマンスの制御

1回の処理で画面に表示できる件数

表示件数での制御は、特にWebアプリケーションで考慮する必要がある。それはWebアプリケーションの場合、一度に通信できるデータの量を考慮する必要が出てくるからである。
（Webアプリケーションであれば一度に表示する件数を決め、ページを分けて次のレコードを参照するのが一般的）

いずれも一度に全件処理をするのではなく、順次に処理をする考え方である。それでは、これらのポイントをどのようにRPGで実装するか、その方法を以降で具体的に説明していく。

順次読み込みの手法

これまでの5250アプリケーションの開発で工夫できる順次読み込みの仕組みを用いて、Delphi/400やJACi400（Web）で活用するための実装方法を紹介していこう。

処理速度のパフォーマンス制御

前述した通り、100万件のデータを一度に抽出して画面に表示すると、処理に時間がかかってしまうのは簡単に想像できると思う。この場合、1回の検索処理の最大時間を設定することで対応する。

つまり、全体の抽出処理を複数回に分割してしまうのだ。分割することでユーザーのストレスを軽減する。と同時に、必要なデータを確認したら、以降の処理をせずに終了することもできる。業務効率が上がるはずだ。

さっそく、実際のプログラミングのポイントを説明しよう。

①設定時間の計算

処理の開始ポイントを決め、その時点でのタイムスタンプを内部で保持しておく。この際、開始ポイントは、プログラム起動時でも、ファイルのREAD開始時でもかまわないので、開発しやすいようにルールを決めておく。

そして、繰り返しでREADするたびにタイムスタンプを取得して、開始ポイントと比較するのである。比較した結果、設定時間を超えていなければ処理を継続し、時間を超えていれば処理をいったん中止すればよい。非常に簡単だ。

②読み込み処理を終了するタイミング

設定時間を過ぎたら処理を中止するのだが、ここで次回のために、続きの開始ポイントを保持しておかなければならない。また、その開始ポイントは、読み込んだ最後のキーを持っても、次回開始時のキーを持ってもいい。ただし、保持するタイミングに気をつけなければならない。

読み込んでいるファイルが1つで、キーがUNIQUEであれば、そのまま終了しても問題はない。しかし、複数ファイルを参照し、見出しと明細のような組み合わせで処理している場合は、設定時間を過ぎたからといって即時処理を中止すると、本来対象として表示するべきレコードが漏れてしまったりする。

このような場合は、見出しファイルのキーに紐付く明細ファイルの内容をすべて処理し終えてから、読み込み処理を終了しなければならない。

以上のことを考慮して、処理終了のタイムスタンプを取得する場所が、適切なタイミングになるようにうまく設定する。

③次回開始キーの保持

最後に、次回開始キーを保持する方法である。キーテーブルを作成し、そこに書き込む。

読み込み処理終了時にキーテーブルに対して更新し、続きの読み込み処理を行う時に、キーテーブルを参照し、読み込み開始ポイントをセットするのである。

この場合、一覧照会画面を閉じる時にキーテーブルをクリアするか、画面起動時にクリアする処理を忘れないようにしてほしい。

以上が、処理速度のパフォーマンス制御のためのポイントである。【図1】

その他の細かいこと、例えば未処理のレコードがあるのか、全件処理を終えているのかを表示するなど、便利な機能も実装していただきたい。

表示件数のパフォーマンス制御

次に、表示件数（データ量）において、パフォーマンスを制御する方法について説明しよう。

これは、Webアプリケーションの一般的な仕様で、インターネットの検索ページを思い出してもらうとイメージしやすいと思う。【図2】

よく見かけるのは、1ページに20件の検索結果が表示され、画面の一番下に残りの検索結果がページ番号として表示されており、そのページ番号をクリックすると画面が変わり、別の20件の検索結果が表示されるというものである。これは、一度に通信できるデータ量を考慮しており、さらには安定したレスポンスを提供することができる。

ここでは、JACi400を使ったWebアプリケーションを例に、表示件数のパフォーマンス制御のためのプログラミングポイントを説明する。

①レコードの抽出

JACi400の場合、プログラムはすべてRPGで組まれることになる。一覧照会は、内部テーブルを使って処理をする。

そのため、あらかじめ一覧表示用の外部記述ファイルを作成しておき、その記述を参照するように内部テーブルを宣言する。そして、その内部テーブルに演算命令の「OCUR」を使用して、抽出したレコードをすべて保管する。

②内部テーブルから、表示するレコードを取り出す

対象レコードをすべて内部で保管してしまうと、次は、画面に表示するレコードのみを抽出する。

図1

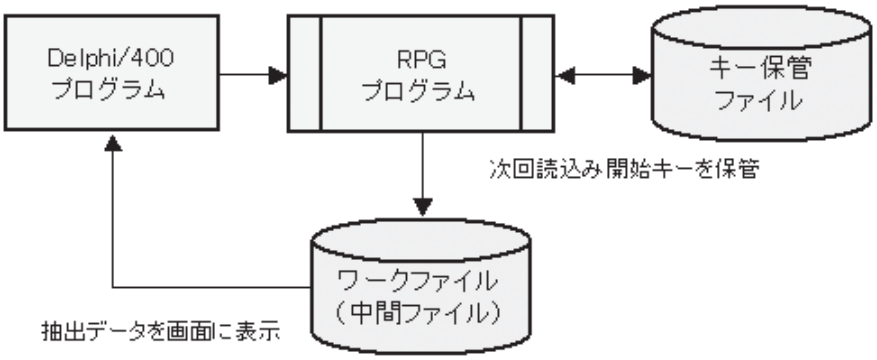

Delphi/400
プログラム
RPG
プログラム
キー保管
ファイル
次回読込み開始キーを保管
ワークファイル
（中間ファイル）
抽出データを画面に表示

図2

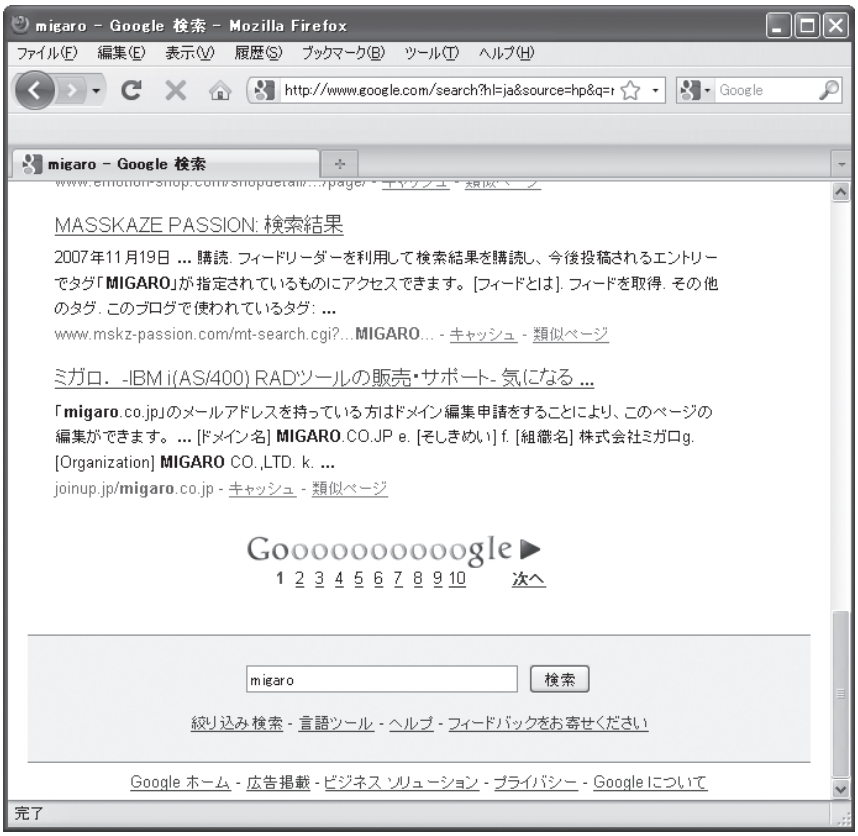
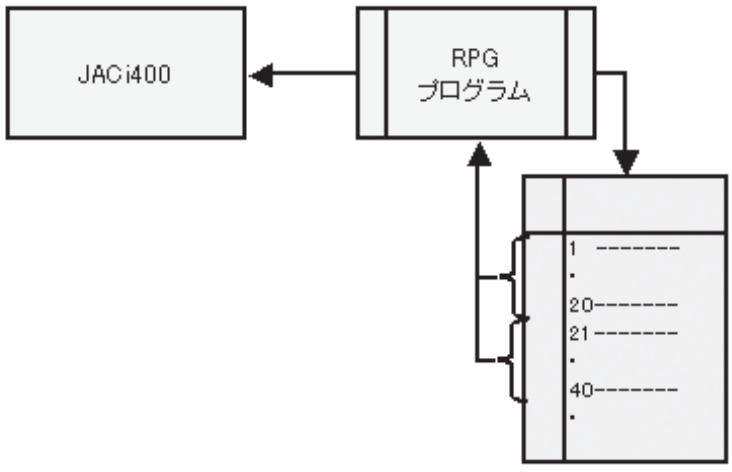

migaro - Google 検索 - Mozilla Firefox
ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
http://www.google.com/search?hl=ja&source=hp&q=r
Google
migaro - Google 検索
MASSKAZE PASSION: 検索結果
2007年11月19日 ... 購読. フィードリーダーを利用して検索結果を購読し、今後投稿されるエントリーでタグ「MIGARO」が指定されているものにアクセスできます。[フィードとは]. フィードを取得. その他のタグ. このブログで使われているタグ: ...
www.mskz-passion.com/mt-search.cgi?...MIGARO... - キャッシュ - 類似ページ
ミガロ. -IBM i(AS/400) RADツールの販売・サポート- 気になる ...
「migaro.co.jp」のメールアドレスを持っている方はドメイン編集申請をすることにより、このページの編集ができます。... [ドメイン名] MIGARO.CO.JP e. [そしきめい] f. [組織名] 株式会社ミガロg. [Organization] MIGARO CO.,LTD. k. ...
joinup.jp/migaro.co.jp - キャッシュ - 類似ページ
Goooooooooogle ▶
1 2 3 4 5 6 7 8 9 10 次へ
migaro
検索
絞り込み検索 - 言語ツール - ヘルプ - フィードバックをお寄せください
Google ホーム - 広告掲載 - ビジネス ソリューション - プライバシー - Google について
完了

図3


JACi400
RPG
プログラム
1 -------
20-------
21-------
40-------

1画面に20件を表示する場合、内部テーブルの1番目から20番目までを最初に抽出する。その方法も、テーブルに保管した時と同様に「OCUR」命令を使用し、演算項目1に1から20までを繰り返しセットして、画面に表示する内容として取り出す。

JACi400はデータを取り出してしまえば、画面への送信はあらかじめ準備されているので、新たにロジックを組み込む必要はなく簡単である。

また、画面へデータを送信した後に、ページ番号を選択し他ページに遷移する場合も、内部テーブルの何番目から何番目と指定して取り出すだけなので、やはり簡単である。【図3】

以上が、表示件数のパフォーマンス制御を実装する際のポイントである。

なお､内部テーブルを使用することで、次のようなメリットがあることも強調しておきたい。

●ファイルの読み込みや書き込み処理をしないので、ワークファイル（中間ファイル）に対象レコードを保管するよりも早い。
●簡単に早く対象レコードを抽出することができるので、ページの画面遷移が早い。

しかし、内部テーブルを使用するとレコードの上限が決まってしまうということもあるので、選択は実際の状況に合わせて判断しなければならない。

Delphi/400プログラムとの連携

RPGでパフォーマンスの制御ができれば、Delphi/400のプログラムと連携する方法を紹介しておく。

基本は前述したとおり、RPGで作成されたワークファイルからDelphi/400でデータを取得して画面に表示するもので、順次読み込みをする場合は、Delphi/400の画面で「続きの検索」処理を組み込む必要がある。

まず画面に「検索」ボタンと「続きの検索」ボタンを用意する。画面起動時は「検索」ボタンを有効にし､「続きの検索」は使用不可の状態にしておく。検索を実行するとRPGプログラムを呼び出すのだが、その時にRPGから続きの処理の有無をパラメータでもらうようにする。

例えば､”1”：続き有り､”2”：処理完了､”9”：対象データなしと設定し、この値によって、用意したボタンを制御するのである。続きがあれば「続きの検索」ボタンを有効にし、続きがなく完了であればそのまま使用不可とし、処理完了の内容を表示するといった具合だ。

とてもシンプルな内容なので、ぜひ試していただきたい。

JACi400プログラムとの連携

次に、JACi400での実装方法を紹介する。

JACi400では前述したとおり、表示件数で制御することが一般的となる。画面で表示する件数を固定するか、画面でユーザーが選択できるようにする。

ユーザーが選択する場合も、プルダウンリストで20件、50件とリストを決めておくほうがよいだろう。その画面の表示件数をもとに、対象の全レコード数からページ数を計算し画面に表示する。表示件数が20件、対象レコードが80件であれば、画面に1から4ページを選択できるようにするのである。

そしてページ番号を選択すると、JACi400ではアクションコード（設定したコード）がRPGに渡されるので、そのコードで内部テーブルから抽出するレコードを判断し、必要なぶんだけを画面に送信するのである。

画面に表示する1レコードの項目数が多い場合は、同じ20レコードでも通信するデータ量が増え、処理に時間がかかってしまうので注意が必要であること補足しておく。

最後に

以上、一覧照会のテクニックとして紹介したが、処理速度（パフォーマンス）に悩む開発者の方々に少しでもヒントになれば幸いである。RPGとDelphi/400、JACi400をうまく連携させて、今後もより使いやすいシステムを提供し紹介していきたいと思う。

M

現在の仕事内容（詳細）
主にJACi400を使ったWebアプリケーションの開発を担当しており、システムの要件定義から納品・フォローまでを行っている。
以前はRPGプログラムの開発を担当していたこともあり、JACi400の開発フェーズでは、RPGプログラム開発の管理を行っている。また、HAツールである*noMAXの技術サポートも担当している。

宮坂 優大　田村 洋一郎

株式会社ミガロ.

システム事業部 システム1課

MKS Integrityを利用したシステム開発

**統合管理ツール「MKS Integrity」を用いて
進捗管理やドキュメントのコントロールを行い
「生産性の向上・品質向上・コスト削減」に結びつける。**

略歴　宮坂 優大
1982年11月19日生れ
2006年近畿大学理工学部卒
2006年04月株式会社ミガロ. 入社
2006年04月システム事業部配属

現在の仕事内容
主にDelphi/400を利用したシステムの受託開発とMKSサポートを担当。Delphi および Delphi/400のスペシャリストを目指している。

略歴　田村 洋一郎
1983年9月27日生れ
2006年近畿大学理工工学部卒
2006年04月株式会社ミガロ. 入社
2006年04月システム事業部配属

現在の仕事内容
RPGやDelphi/400などの開発経験を経て、現在は主にDelphi/400を利用した受託開発を担う。また、MKSを利用した開発手法を社内に広めたり、MKSの技術サポートも担当している。

●はじめに
●MKS Integrityのメリット
●MKS Integrityの機能紹介
●有効機能の紹介～レポート機能
●最後に

1.はじめに

アプリケーション開発を効率よく品質の高いものにするためには、開発プロセスと関連ドキュメントを1つのツールで管理をすることが望ましい。1つのツールで統合管理をすることができれば、進捗管理やドキュメント等のコントロールを高めることができ、結果として「生産性の向上・品質向上・コスト削減」に結びつけることができる。

「MKS Integrity」はアプリケーションの資産管理、開発管理、変更管理、要求管理、保守管理を行い、この問題を解決する統合管理ツールである。

MKS Integrityは、以下の5つの基本機能で構成される。【図1】

1. MKS Integrity
2. MKS Requirements
3. MKS Source
4. MKS Test
5. MKS Deploy

2.MKS Integrityのメリット

MKS Integrityは、さまざまな人々へ、多彩なメリットを提供する。

● CIOや経営者のメリット

MKS Integrityは、企業の複雑で多様化したIT環境において、最大の生産性とビジネス価値を導きだせるようIT環境全体の統制管理手段を提供してくれる。このIT統制管理により、社内の情報をビジネスゴール（会社の目標）に迅速に結びつけ、生産性とビジネス価値を改善させることができる。

また、各々のプロジェクトのリアルタイムなデータが一覧表示された管理画面（ダッシュボード）を通して、今後の見通しや進捗状況、コスト、管理統制状況を把握して判断をすることができる。

このプロジェクトのデータはすべて開発現場のものであり、そこでプロジェクトメンバーが入力しているデータをそのまま使用しているので、正確かつリアルタイムに状況を確認することができる。

●プロジェクト管理者のメリット

プロジェクト管理者は、承認された要件や変更に基づいて「WBS（*1)」を作成し、管理することができる。もし仮に変更要件が発生し、設計モデルの変更が生じても、進行中の各種仕事に影響する場合は、画面に警告を表示することができる。

大規模なプロジェクトでは、プロジェクト計画ツールやリソース管理ツールからWBSやスケジュール表、見積表をMKS Integrityへインポートしたり、他ツールへエクスポートしたりすることもできる。

さらに、見積をWBS上に取り込むこともできる。見積活動の結果から手作業で作成された資料はときに古いものとなってしまうことが起こるが、MKS Integrityを導入すれば、プロジェクト運営のために資料を作成しなおす必要はなく、本来の管理そのものに時間を割くことができる。

●プロジェクトリーダーのメリット

プロジェクトリーダーはMKS Integrityを利用することにより、現在の作業内容とその作業コスト、現在誰が作業のない状態なのかをリアルタイムで確認し、プロジェクトへ技術者を割当てることができる。

MKS Integrityにより開発プログラムのリビジョン管理が自動化されるので、品質保証に対してメンバー間の協調をとることができる。その結果、全体の効率と生産性の向上につながる。

また、ロールバック、リリース再現も含め、ソフトウェア構成に関する優良な情報を取得することができる。

●開発者のメリット

開発者ももちろん、リアルタイムで現在進行している作業に影響する要件や、設計の変更等の告知を画面から確認することが可能である。

メンバー全員が作業中の画面にアクセスして要件を確認したり、設計内容やテスト仕様についてドキュメントを閲覧したりすることができ、プロジェクト全体の生産性の向上を図ることができる。

プラットフォーム（*2）に依存せず、あらゆる開発者がMKS Integrityを通して作業にアクセスでき、マルチプラットフォーム開発に向けたプロセスのメリットを享受することができる。

その他、以下のようなメリットがある。

・開発者が個人ごとに利用できるワークスペース
・パッケージの変更
・リリースと履歴管理
・並列開発をサポート
・互いに競合するプロセスの解決（ワークファイルの結合）
・さまざまな統合開発環境の統合

以上が、プロジェクトのさまざまな立場の人々から見た、MKS Integrityを導入した時のメリットである。

次に、MKS Integrityの機能を少し詳しく紹介する。

*1 WBS（Work Breakdown Structure）
WBSは、プロジェクトマネジメントで計画を立てる際に用いられる手法の1つで、プロジェクト全体を細かい作業に分割した構成図。「作業分割構成」「作業分解図」等とも呼ばれる。

*2 プラットフォーム
プラットフォームとは、アプリケーションソフトを動作させる際の基盤となるOSの種類や環境、設定等のこと。WindowsやUNIX、Mac OSは、それぞれ異なるプラットフォームである。
また、OSにとっては、自らを動作させる基盤となるPC/AT互換機、Macintoshなどのハードウェアの種類がプラットフォームである。

3.MKS Integrityの機能紹介

MKS Integrityの基本機能の中から、MKS IntegrityとMKS Sourceの機能について紹介をする。

● MKS Integrity（プロセス管理）

アプリケーションライフサイクルの全領域にまたがって存在する各種のプロセスを、自社のスタイルに合わせて設定できる。つまり、独自のワークフローを作成することができる。

この設定されたワークフローを開発チーム全員が利用することにより、ソフトウェア開発の全プロセスにおいて標準化を図ることができ、より完成度の高い開発体制を作り上げることができる。

● MKS Source（ソース管理）

ソース管理、ソフトウェア構成管理、リソース管理の各機能を管理する。これまでと同じ開発作業をそのまま行いながら、作業内容の登録や履歴管理ができる。

この章では、前述したようにシステム開発でさまざまなメリットを提供しているMKS Integrityについて、主要な基本機能であるMKS IntegrityとMKS Sourceを利用して、開発プロジェクトの情報や関連ファイルを登録する設定方法を紹介したいと思う。

3-1.MKS Integrityの機能について

MKS Integrityの設定は、主にMKS Integrity Administrationを使って設定する。

MKS Integrity Administrationのメイン画面は、設定項目がツリー状になっており、設定したい項目をクリックすることで、画面右側にその情報が表示され、さらにその項目をクリックすることで詳細を設定することができる。【図2】

①MKSドメインの設定

まずは、MKS Integrityを使用するユーザーを、MKSドメインに設定する。

MKSドメインにユーザーを追加する場合は、［MKSドメイン］→［ユーザー］のツリー部分で右クリックし、ポップアップメニューから［作成］を選択する。【図2①】

そうすると、ドメインユーザー作成画面が表示され、ログインID、パスワード、フルネーム（MKS Integrity内で表示される名称）、メールアドレスを登録することができる。【図3】

グループも同様に作成することになる。グループ作成後、そのグループに割当てるユーザーを登録する。また、グループに権限を付与することも可能である。

②ビューセット分布の設定

前述の、その他メリットに「開発者が個人ごとに利用できるワークスペース」を作成できると記述した。このワークスペースを利用することにより、個人ごとにMKS Integrityの画面レイアウトを自由にカスタマイズすることができる。

ビューセットの項目で、個人ごとあるいはグループごとに基本的な画面項目配置を設定し、項目の表示・非表示設定等のレイアウトの設定をする。

③ワークフローの設定

図4をご覧いただきたい。これはプロジェクトフェーズ例のワークフローである。「ステート」と呼ばれるステータス（「外部設計」や「プログラム開発」などがステータスにあたる）をあらかじめ設定しておき、GUI画面上でステートのつながりをドラッグ＆ドロップで設定する。【図4】

あとは、主担当者や作業予定時間、作業実績時間などを登録するフィールドを作成し、ステートごとに表示フィールドから使用したい項目だけを設定すれば、ワークフローは完成する。【図5】

この作成したワークフローを開発者全員で使用すれば、ソフトウェア開発プロセスの標準化を図ることができる。

ステートは、名称やフィールド、権限、ステート同士のつながりや分岐も自由に設定することができる。すなわち、自社のスタイルに合わせて設定できるため、いろいろな部門を持った会社の管理にも力を発揮する。

さらに、ステートを移行させる場合に

図1　MKS 構成図

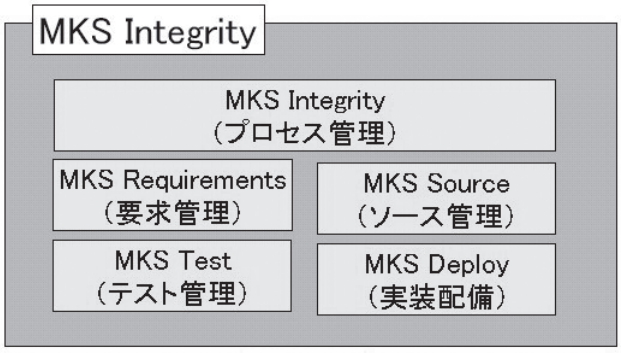


図2　MKSAdminstrator 画面図

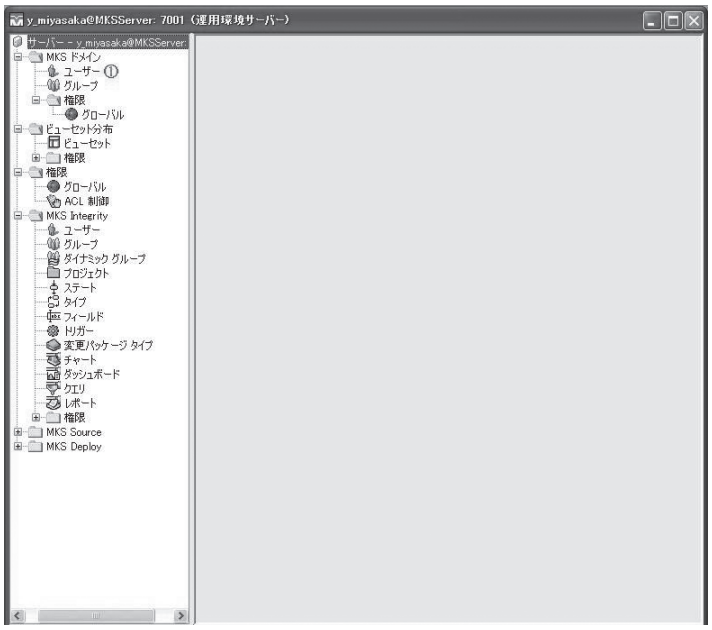
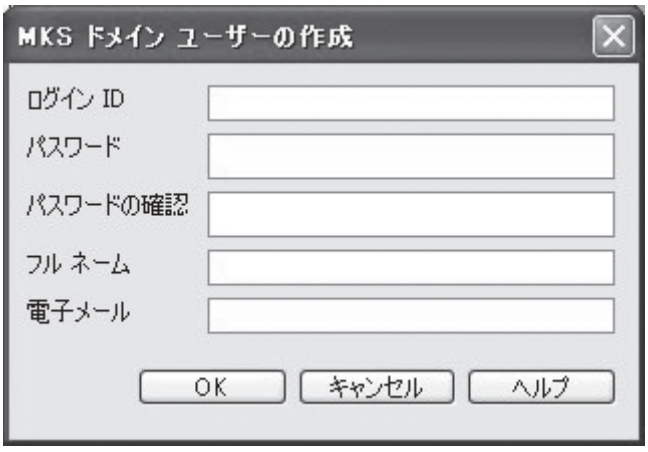


図3　ユーザー設定画面図



権限を設定することで、各担当者の役割を厳しく制限することも可能である。例えば「プログラム開発→結合テスト」の移行はプログラマー、「結合テスト→テスト完了」はテスト担当者とし、それ以外の者がステートの変更を行えなくするような管理が可能である。

また、ステートの変化に伴い、管理者にメールで通知することもできる。その結果、管理者が担当者にステートの変更をわざわざ確認するといったこともなくなり、管理者はプロジェクトの進捗状況を把握できるようになる。

④アイテムの作成

個々の作業を［アイテム］として設定し、アイテム単位で進捗管理を行うことができる。

プロジェクトリーダーがプログラマーに対して、プログラムの開発依頼を行う場合、開発要求を１つのアイテムとして設定する。

具体的には、プロジェクト名、ステート、担当ユーザー、サマリー（要件名）、開発担当者、開発完了希望日、開発予定工数等の情報をアイテムに設定する。（［アイテム］に登録する情報をあらかじめ設定しておくことが必要）。【図 6】

アイテムの設定は自由に行えるため、プログラム開発といったプログラム単位で設定することも、外部設計・内部設計といったフェーズ単位で設定することも可能である。プロジェクトにアイテムを登録することで、進捗管理を行っていくのである。

また、アイテムの情報を確認することで、リアルタイムで稼働状況や進捗状況を把握することができ、プロジェクトの現状が非常に把握しやすくなっている。

3-2.MKS Sourceの機能について

MKS Source の基本的な登録内容や機能を記述する。

①プロジェクト＆サンドボックスの登録・関連性について

複数のソース管理が必要な場合、それぞれに［プロジェクト］を登録することで、ソースを任意の単位で管理することができる。

また、開発環境でソースがあるフォルダ（ディレクトリ）に［サンドボックス］を設定し、サーバーに登録した［プロジェクト］の最新ソースと関連付けて登録する。

そうすることで、サーバーでは常に最新環境（プロジェクト）を管理し、開発環境（サンドボックス）ごとに最新環境と同期をとりながら開発を行うことができる。

②チェックアウト・チェックイン

プログラムの修正を行う場合には、チェックアウトを行う。

チェックアウト中は、他の開発者が同じプログラムを編集できないようにロックをかけることができる。【図 7】

プログラム修正完了後にチェックインを行うことで、サーバーに最新環境を適用し、ロックも解除する。また、チェックインの際には、チェックインを行ったプログラムに関しての特記事項等を情報として登録することもできる。【図 8】

③凍結・解凍

リリースされたプログラムや修正を行わないプログラムは、凍結を行うことで修正ができない状態に設定できる。【図 9】

また、解凍で凍結状態を解除できる。

④ソースの履歴管理

MKS Source で登録を行った履歴は、すべて記録されており、確認することができる。

また、ソースの差分情報を単語単位で認識したり、確認することができる。【図 10】

3-3.MKS IntegrityとMKS Sourceの連携について

前述した MKS Integrity と MKS Source の機能を連携して、プロジェクト管理を行うことができる。

MKS Integrity はプロジェクトの計画･実績等を管理する機能として設定し、MKS Source はプログラムのソース（ドキュメント）を管理する機能として設定する。

この２つを連携することで、プログラムが何の計画・実績により作成されたものなのかを関連付けることができ、結果としてすべてのソース（ドキュメント）の変更履歴と変更理由を確認することが可能である。

4.有効機能の紹介 ～レポート機能

MKS Integrity の機能であるレポート機能について紹介する。

4-1.レポートについて

アイテムで登録したプロジェクトの予定・実績情報を参照して、レポートに出力することができる。設定次第では、プロジェクトの進捗状況を把握するためのレポートを作成することもできる。

では、続いて、進捗状況を把握するためのレポートの設定方法と、設定したレポートの確認方法を紹介する。

4-2.レポートの設定例（進捗状況確認レポートの設定）

① MKS Integrity Administration から、MKS Integrity のレポートを選択。【図 11】

②次に、メニューの［レポート］→［管理の作成］を選択。【図 12】
すると管理レポートの作成ウィザードが表示されるので、順に設定を行っていく。

③タイプの設定：任意のレイアウトのタイプを指定する。【図 13】
ここで設定したタイプのレイアウトに、レポートが出力される。

④属性の設定：レポートの名前と（データの参照を行う）クエリを指定する。【図 14】

⑤スタイルの設定：ブラウザや印刷時の色や文字サイズ、罫線のスタイルを指定する。【図 15】

⑥ロゴの設定：レポートに出力するロゴを任意で指定する。（任意指定項目）。【図 16】

⑦パラメータの設定：レポートに出力するタイトルやヘッダー、フッター等を指定する。【図 17】

⑧項目フィールドの設定：レポートで表示するフィールドを指定する。【図 18】（※）

※指定したクエリが持っているカラムのみ指定可能なので、表示するフィールドを追加する場合は、クエリにカラムを追加する必要がある。

以上で、レポートの設定は完了である。

図4　ワークフロー図

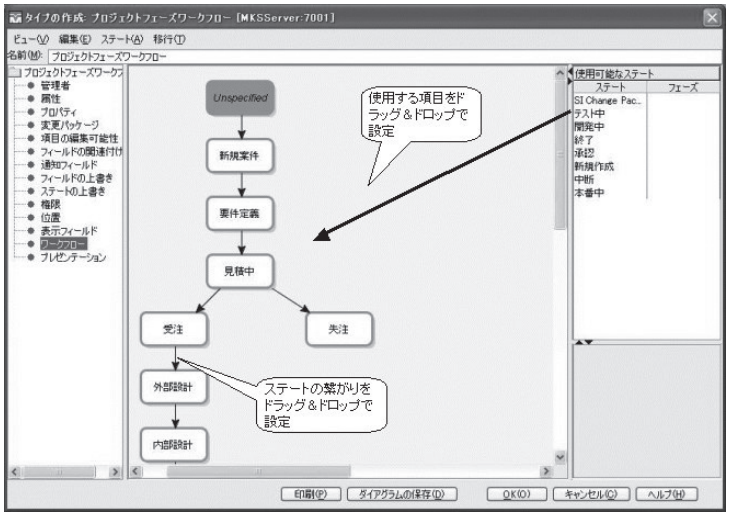


図5　表示フィールドの設定図

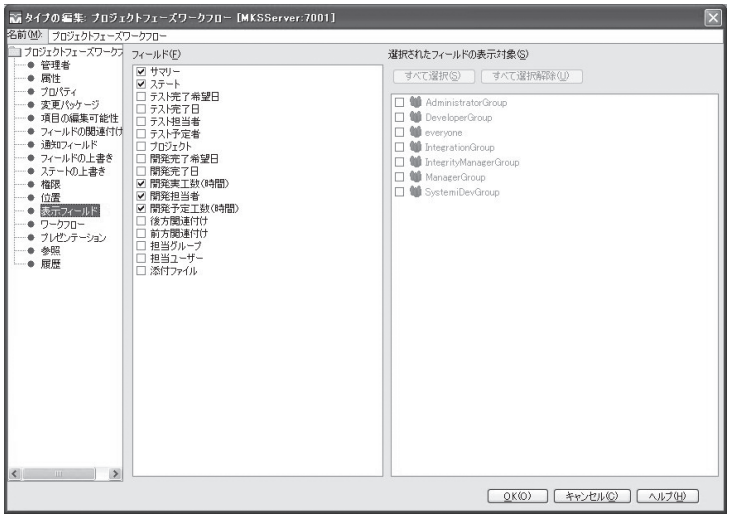
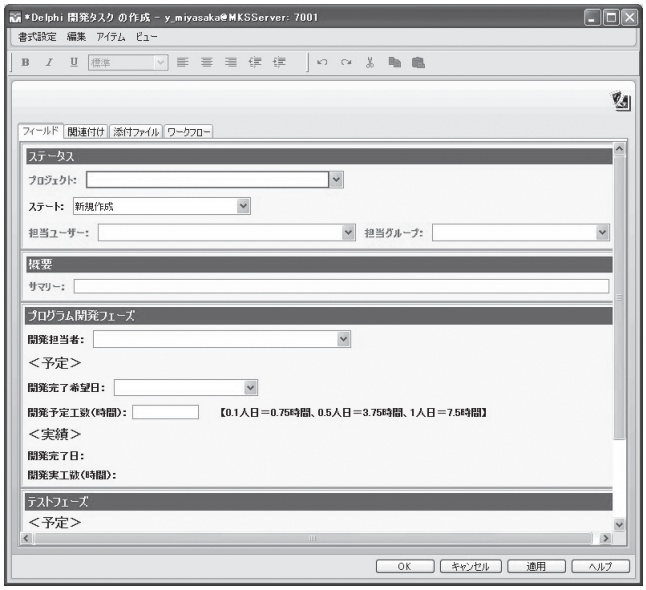


図6　アイテム登録図



上記のように、レポート機能を利用することによって、図19のようなプロジェクトの状況を把握できるレポートを自由に作成することができる。加えて、レポートはブラウザで確認することができる。【図19】

また、レポート機能以外にも、MKS Integrityにはシステム開発の幅を広げ、品質を向上させるためのさまざまな機能が多数あるので、ぜひ活用していただきたい。

5.最後に

以上が、MKS IntegrityとMKS Sourceを利用した場合の、システム開発のメリットと簡単な機能の紹介である。

ポイントとしては、MKS Integrityでは、開発工程をワークフローとして設定し、要求をアイテムとして登録する。

アイテムごとに開発の予定を設定し、実績を登録することによって、開発の管理者はそれぞれの要求（アイテム）がどういった計画でどういった状況であるかをリアルタイムに確認することができる。また、MKS Integrityに登録した情報は変更不可能な情報として蓄積されていくので、すべての履歴情報を確認することもできる。内部統制に対応したツールといえる。

さらに、アイテム（MKS Integrityの機能）とプログラムソースの作成・変更情報（MKS Sourceの機能）を関連付けることができるため、どの要求（アイテム）に対してソースを作成（変更）したのかを管理･確認することもできる。

このようにMKS Integrityは、煩雑になりやすいシステムの開発を高品質なシステム開発にしていくためのツールである。今回紹介した特長や機能以外にも数多くのさまざまな機能を持っているMKS Integrityの有効な機能を活かし、より質の高いシステム開発を今後も目指していきたい。

M

図7　ロックイメージ図

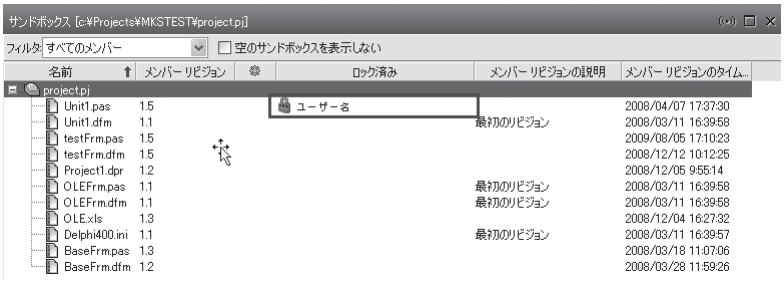


図8　特記事項イメージ図

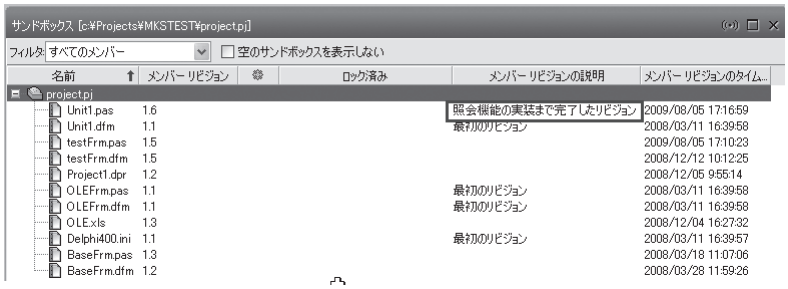


図9　凍結イメージ図

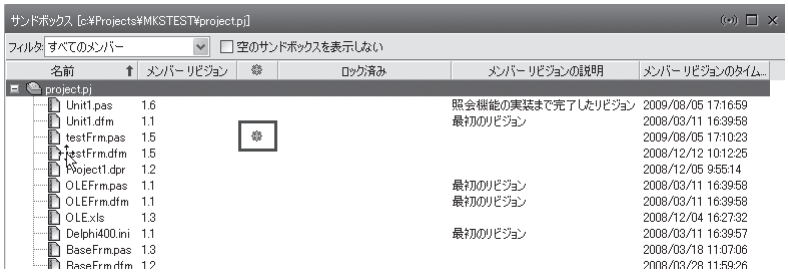


図10　ソースの差分イメージ図

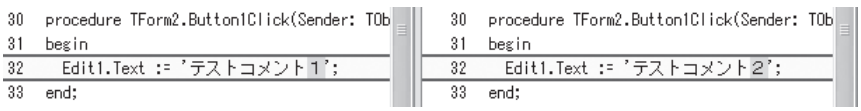


図11　レポート指定図

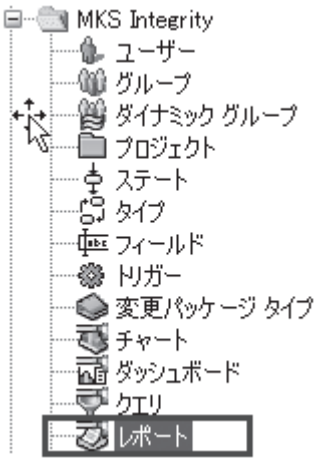


図12　管理の作成設定図

図13　タイプの設定図

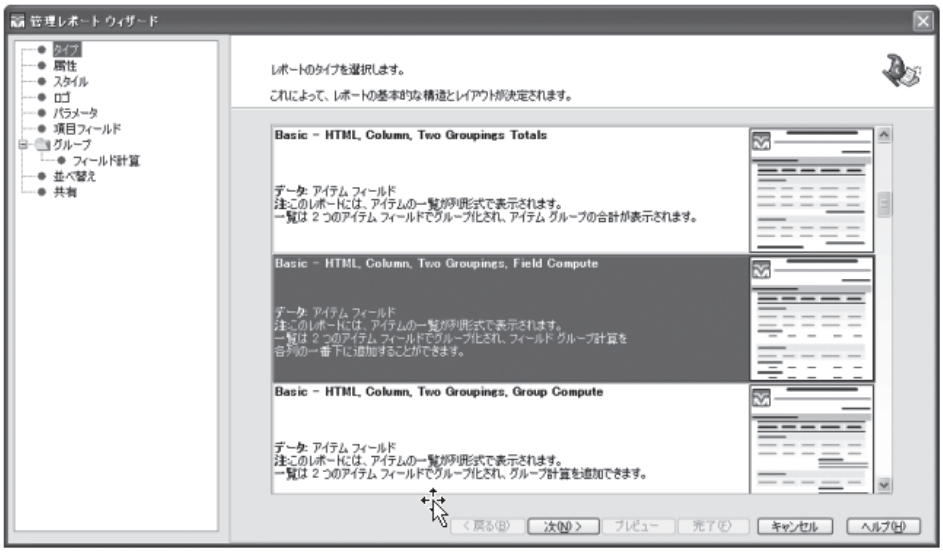


図14　属性の設定図

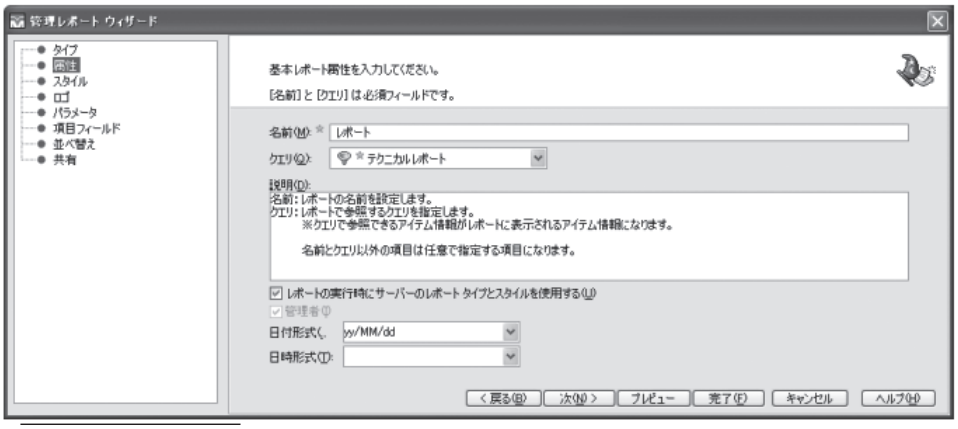


図15　スタイルの設定図

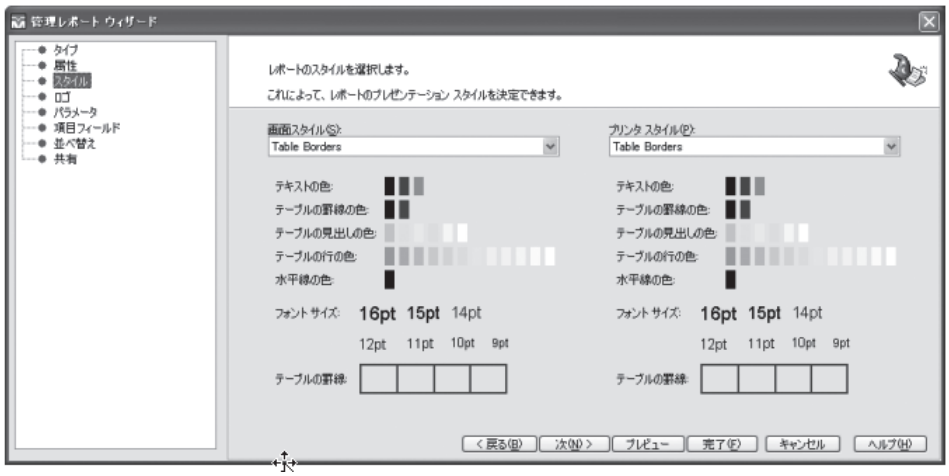


図16　ロゴの設定図

図17　パラメータの設定図

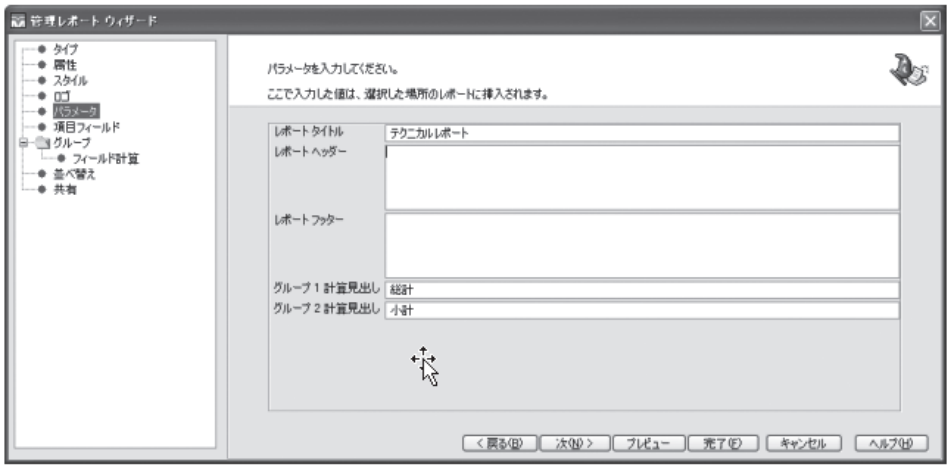


図18　項目フィールド設定図

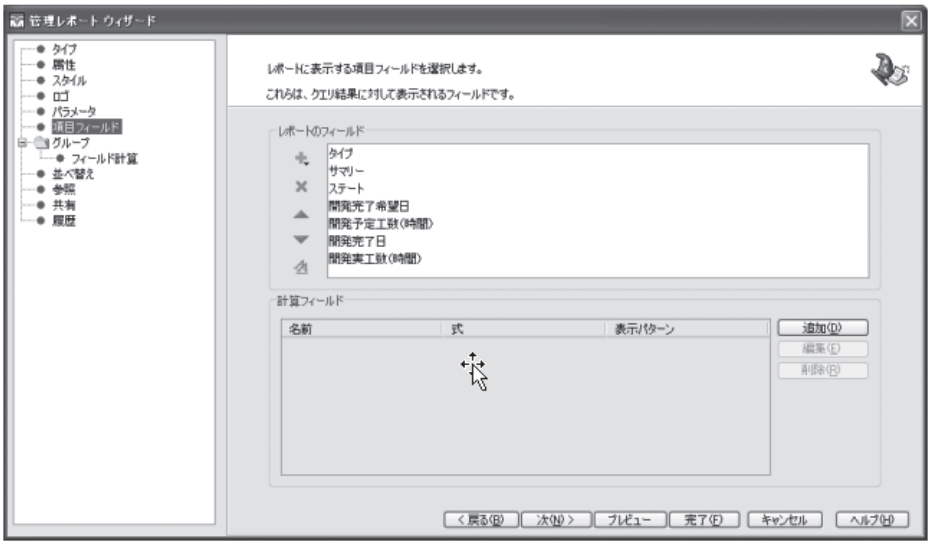
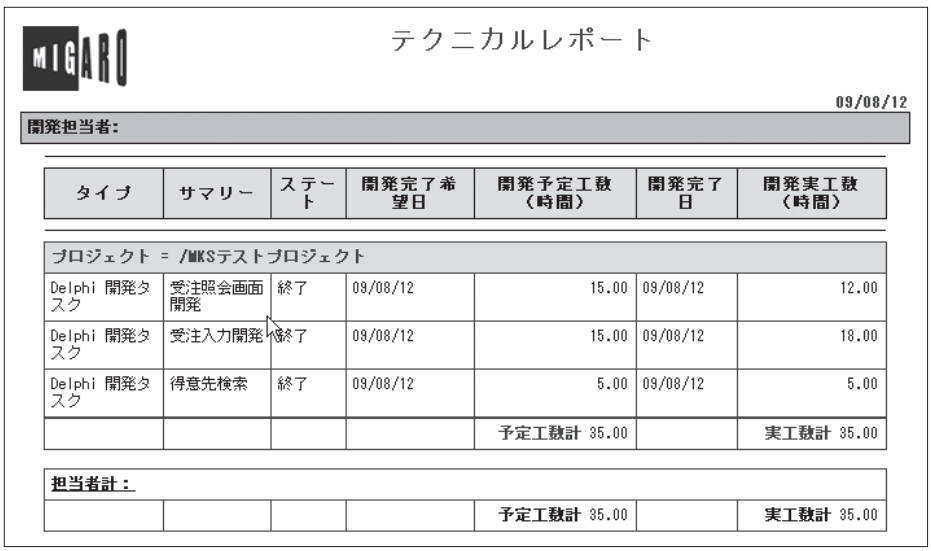


図19　レポートイメージ図



テクニカルレポート

09/08/12

開発担当者:

タイプ	サマリー	ステート	開発完了希望日	開発予定工数（時間）	開発完了日	開発実工数（時間）
プロジェクト = /MKSテストプロジェクト						
Delphi 開発タスク	受注照会画面開発	終了	09/08/12	15.00	09/08/12	12.00
Delphi 開発タスク	受注入力開発	終了	09/08/12	15.00	09/08/12	18.00
Delphi 開発タスク	得意先検索	終了	09/08/12	5.00	09/08/12	5.00
				予定工数計 35.00		実工数計 35.00
担当者計：						
				予定工数計 35.00		実工数計 35.00

MIGARO.
TECHNICAL
REPORT

Migaro.Technical Report
No.2 2009年秋
ミガロ.テクニカルレポート

2009 年 11 月 1 日 初版発行

◆発行
株式会社ミガロ.
〒 556-0017
大阪府大阪市浪速区湊町 2-1-57 難波サンケイビル 13F
TEL：06（6631）8601　FAX：06（6631）8603
http://www.migaro.co.jp/

◆発行人
上甲 將隆

◆編集協力
アイマガジン株式会社

◆デザインフォーマット
近江デザイン事務所

MIGARO. TECHNICAL REPORT

ミガロ.テクニカルレポート

株式会社ミガロ.

http://www.migaro.co.jp/

本社
〒556-0017
大阪市浪速区湊町2-1-57
難波サンケイビル 13F
TEL:06(6631)8601
FAX:06(6631)8603

東京営業所
〒106-0041
東京都港区麻布台1-4-3
エグゼクティブタワー麻布台 11F
TEL:03(5573)8601
FAX:03(5573)8602