MIGARO. TECHNICAL REPORT

ミガロ.テクニカルレポート

No.10
2017年秋

［創刊10周年記念号］

株式会社ミガロ.

CONTENTS 目次

Migaro.Technical Award 2017 お客様受賞論文／ミガロ.テクニカルアワード

Migaro.Technical Report 2017 SE 論文／ミガロ.テクニカルレポート

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。

「ミガロ.製品をご利用中の技術者の皆様に、日々の開発に少しでもお役に立つような技術情報をご提供したい」との思いから 2008 年に創刊した『Migaro.Technical Report』は、このたび、節目となる第 10 号を発刊するはこびとなりました。これもひとえに、ご多忙中にもかかわらず『Migaro.Technical Award（お客様論文)』にご寄稿いただいた多くのお客様、ならびに『Migaro.Technical Report』に対して貴重なご意見・ご要望をお寄せ下さった皆様のご支援の賜物と、心より感謝をしております。

今回の創刊 10 周年記念号には、過去の Migaro.Technical Award ご受賞者および弊社が日頃お世話になっている各社様より頂戴したお祝いメッセージを掲載しました。また、過去の受賞者の方々にご協力を賜り、Migaro.Technical Report についての座談会を開催し、その模様を掲載しております。ぜひこちらの記事もご覧ください。

この 10 年は本当にあっという間に過ぎたように感じていますが、現在の企業 IT での主要テーマとなっているモバイル、クラウド、AI、IoT などの本格的な普及は全てこの 10 年以内であることを思うと、やはり 10 年間の時代の変化の大きさを感じます。弊社でも、今年は Delphi/400 の新バージョン「10.2 Tokyo」のリリースや新製品「Valence」の販売開始など、変革への挑戦を続けております。

さて、今回の『Migaro.Technical Report』も従来と同様に、第 1 部は「Migaro.Technical Award 2017 お客様受賞論文」、第 2 部は「ミガロ. SE 論文」の 2 部構成としています。

第 1 部の「Migaro.Technical Award」とは、日々アプリケーションの開発・保守に携わるエンジニアの方々の努力と創意工夫の成果を顕彰することを目的とし、「Delphi/400」「SP4i」「Business4Mobile」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。昨年に引き続き、従来のお客様論文に当たる「部門 1」と「業務課題を解決した開発技術・テクニック」を簡潔にまとめていただく「部門 2」の 2 部門構成といたしました。
今回のお客様論文は、「鍵番号の刻印を文字データとして読み取り、基幹システムと統合した事例」や「Windows タブレット導入をきっかけとした製品加工現場の業務・システム改革」など、創意工夫にあふれる論文を多数ご寄稿いただきました。

第 2 部「ミガロ. SE 論文」では、弊社 SE による技術論文を掲載しております。今回は、「アプリケーションテザリングを利用した PC ＆モバイルアプリケーション連携」や「Delphi/400 バージョンアップに伴う文字コードの違いと制御」など、さまざまなテクニックを開発に活かしていただくための技術情報をご紹介しております。本レポートが少しでも皆様の開発・保守のお役に立てば幸いです。

最後に『Migaro.Technical Report』第 10 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、ご協力を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を申し上げます。

2017 年秋

株式会社ミガロ.
代表取締役社長
上甲 將隆

Migaro. Technical Report 創刊 10 周年記念

パートナー様からの祝辞
お客様からの祝辞・お客様の声

創刊10周年記念

パートナー様からの祝辞

10 年間の研鑽と共有に深い敬意

武藤 和博 様
日本アイ・ビー・エム株式会社
専務執行役員
IBM システムズ・ハードウェア事業本部長

『ミガロ.テクニカルレポート』創刊 10 周年、おめでとうございます。

この 10 年、クラウドの台頭をはじめ、高度なアナリティクスやコグニティブの浸透、セキュリティ脅威の高まりなど、IT 市場は激動の只中にあります。そういった大きな変化の中で、『ミガロ.テクニカルレポート』が 10 年間、揺らぐことなく、真摯に技術の研鑽と共有を追求されてきたことに、深く敬意を表します。

貴誌が創刊された 2008 年は、弊社でも System i と System p を統合した Power Systems が新たなスタートを切った年でもあります。

そしてその後も IBM i は、技術革新と最新のテクノロジーへの対応を続け、お客様のビジネスを支える最強のプラットフォームとして高くご評価いただいています。

私ども IBM は、『ミガロ.テクニカルレポート』に関わる皆様のご期待に応えられる魅力的な技術・機能を、引き続き IBM i で提供してまいります。

Delphi の可能性の拡大に大きく寄与

藤井 等 様
エンバカデロ・テクノロジーズ日本法人
代表

『ミガロ.テクニカルレポート』の創刊 10 周年、誠におめでとうございます。

『ミガロ .テクニカルレポート』に多数論文が掲載されている Delphi/400 の母体である Delphi は、1995 年に Version 1 がリリースされた、たいへん長い歴史をもつ統合開発環境です。その 20 年以上に及ぶ歩みの中で、20 回以上ものメジャーバージョンアップを重ね、今や同一環境で、Windows、macOS、iOS、Android 用 の 多様なアプリケーションを開発できる、世界屈指のツールへと成長しました。

その Delphi の機能を活用し、卓越したビジネスサーバーである IBM i 上の基幹アプリケーションを開発されている皆様のお取り組みの様子は、毎年の『ミガロ.テクニカルレポート』を通して存じ上げてきました。

そしてその 10 年にわたる継続的なお取り組みは、Delphi の可能性を新しい世界へ広げていくことに大きく寄与し、多くのお客様の成功を後押しされてきたものと感じており、深い感謝の気持ちを抱いてまいりました。

今後も、開発者の皆様を魅了する機能・環境をご提供していきます。Delphi/400 にご期待いただければと思います。

創刊10周年記念

パートナー様からの祝辞

世界でも類例のない素晴らしい取り組み

Serge Charbit 様

SystemObjects Corporation
CEO

I'd like to congratulate user community of Delphi/400 and SmartPad4i in Japan and MIGARO staffs for the publication of “MIGARO Technical Report” 10th anniversary volume from the bottom of my heart.

Delphi/400 is widely and globally used not only in Japan, but also in North America, South America, Europe and Asia.

The purpose of Delphi/400 and SmartPad4i is to provide two solutions for the same problem: IBM i Application Modernization.

Delphi/400 is for PC developers and SmartPad4i for RPG or Cobol developers.

However, if we look at user systems precisely, they all vary due to the business cultures, project management styles, and above all, the "innovative ideas and technology" of developers.

"Migaro Technical Report" deliberately collects such "innovative ideas and technology" of developers. I believe that there is no comparable program to the "Migaro Technical Report" even in the worldwide Delphi/400 users, and it's a very wonderful approach for sharing technical information among developers.

We will also do our best for our products, Delphi/400 and SmartPad4i to continue to motivate developer's innovative ideas and technologies.

I would also like to thank Migaro and his staff for the excellent work they have been doing for more than 10 years.

Serge Charbit 様からの祝辞 日本語訳

Delphi/400 および SmartPad4i をお使いの日本の皆様、そしてミガロ .の皆様、『ミガロ .テクニカルレポート』が創刊 10 周年を迎えられましたことを、心よりお祝い申し上げます。

Delphi/400 は PC 開発者向け、SmartPad4i は RPG または COBOL 開発者向け製品で、日本をはじめとして、北米、南米、ヨーロッパ、アジアのお客様に広くお使いいただいています。そのご利用の目的は、IBM i 上で稼働する Web/GUI アプリケーションの開発という点で共通しています。

ただし、開発されるシステムを子細に見ると、ビジネスの進め方や慣習の違い、そして何よりも開発者の方々の創意工夫の違いによって 1 つ 1 つ異なっています。『ミガロ .テクニカルレポート』は、その開発者の創意工夫を丹念に収集し、共有を目的に公開している点で、世界の Delphi/400 ユーザーの間でも類例のない、大変素晴らしいお取り組みと感じています。

私どもの Delphi/400 および SmartPad4i も、開発者の皆様の創意工夫の魂を刺激し続けるプロダクトであり続けたいと考えており、今後とも全力で取り組んでまいります。

そして、ミガロ .およびその社員が 10 年間も素晴らしい取り組みをしてきたことに感謝いたします。

発想・アイデア・スキルを共有するメディア

飯田 恭子 様

アイマガジン株式会社
取締役　i Magazine、IS magazine 編集長

『ミガロ .テクニカルレポート』の創刊 10 周年をお慶び申し上げます。

Delphi/400 は、業務システムを刷新・強化する、そして業務ノウハウを自らの手でシステムに具現化する先進的な開発ツールです。IBM i という優れたプラットフォームを利用するユーザーの方々にとって、Delphi/400、そしてミガロ .様は、常に頼もしいパートナーであり続けてきました。

『ミガロ .テクニカルレポート』は、そうした Delphi/400 ユーザーの真摯なお取り組みを紹介する優れたレポートであると同時に、ミガロ .様がユーザーの方々に最新テクノロジーや活用方法を伝え、そして Delphi/400 コミュニティの皆様が、業務アプリケーションを生み出す発想や利用アイデア、業務改革のヒント、技術的なスキルといった有益な情報を共有するための強力なメディアです。

次の 10 年に向けても、『ミガロ .テクニカルレポート』がユーザーの方々へ、その実り多き成果と未来への可能性を伝え続けていけるように願っています。

創刊10周年記念

お客様からの祝辞・お客様の声　氏名50音順

石井 裕昭 様

豊鋼材工業株式会社
監査役

『ミガロ.テクニカルレポート』創刊10周年、おめでとうございます。

テクニカルレポートの歴史と私自身のDelphi/400を活用した開発スキル向上、会社システムの充実の道のりは、まさに重ね合わせて振り返ることができます。

ミガロ.様のセミナーとレポートにより紹介されるさまざまな業種の事例やテクニックは刺激的で参考になるものが多く、弊社システムでも高い頻度で活用させていただいております。

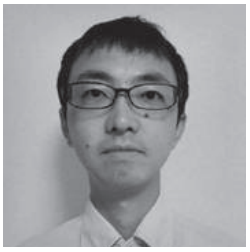
弊社の業務への適用方法が想像できない事例でも、数年たった後で見直したときにタイムリーな課題であったり、あきらめていたことへのヒントになったこともよくあります。

このような取り組みと勉強の機会を継続的に提供されているミガロ.様の努力と日頃のサポートには、本当に感謝しております。

これからもますますの質の高いテクニカルレポートとセミナーが継続し、盛り上がっていくようお祈りします。

牛嶋 信之 様

株式会社佐賀鉄工所
管理部　情報システム課
主事



ミガロ.様のDelphiテクニカルサポートにさまざまな質問をさせていただいた結果、多くの知識を習得することができました。また、ミガロ.テクニカルアワードを受賞できたのも、その賜物だと思っています。テクニカルサポートには大変感謝しております。

論文を執筆することによって、客観的に技術情報を再確認でき、自分自身、Delphi知識の理解をさらに深めることができました。受賞により、自分への自信にもなり、Delphi開発へのモチベーションにもつながっています。

今後も、さらなる知識向上ができるようにテクニカルサポートに質問をさせていただき、人様にお見せできるようなシステムを構築し、テクニカルレポート向け論文として寄稿できればと考えています。

大崎 貴昭 様

森定興商株式会社
システム運用部　係長

『ミガロ.テクニカルレポート』創刊10周年、おめでとうございます。

振り返りますと、論文執筆の依頼を受けた当時、弊社ではRPGの画面で行われていた「取引先申請」を刷新するにあたり、Delphi/400でシステム開発を進めていた状況でした。

そして論文の提出とシステム開発がほぼ終わる頃に、ミガロ.様より「テクニカルアワード」でレポートを発表して欲しいとの連絡を受け、副賞に目が眩んだ私は、軽い気持ちで引き受けました。

軽〜く引き受けたものの、あがり症の私は人前で話すことが苦手で、発表用のプレゼン資料を片手にトイレにこもり、話す練習に励んだことも、今となってはよき思い出です。

今後の『ミガロ.テクニカルレポート』、楽しみにしております。

創刊10周年記念

お客様からの祝辞・お客様の声　氏名50音順

川島 寛 様

株式会社タツミヤ
管理部　情報システム課　課長

論文を執筆した当時を振り返ると、Windows 2000 から XP に OS を切り替える時期でしたが、発注システムが Windows 2000 で作成されていたので、XP をインストールできない状況でした。そのため VB から Delphi への移行は時間との闘いで、システム開発の迅速化が要求されていたのを思い出します。

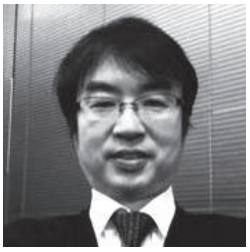
しかし、ミガロ.様の Delphi/400 のサポート支援を受けながら開発をスムーズに進められ、さらには『ミガロ.テクニカルレポート』に論文を発表できるようなことにもなりました。

それから現在までの間に Delphi/400 のバージョンアップを実施しましたが、その運用環境も新しい PC に簡単にセットアップでき、Windows のバージョンアップにも大きな影響が出ることがなく大変助かっています。

今後、32 ビット PC から 64 ビット PC への移行が進むと考えられますが、ミガロ.様の運用プログラムに当社の今後を見据えて行きたいと考えてます。これからの『ミガロ.テクニカルレポート』にも期待しています。

久保田 佳裕 様

極東産機株式会社
管理本部　システム開発室
室長



『ミガロ.テクニカルレポート』が創刊 10 周年を迎えられましたことを心からお祝い申し上げます。

私は、2012 年に「JC/400 による取引先との Web-EDI システム構築」という題名でレポートを寄稿し、幸運にも最優秀賞をいただくことができました。

ミガロ.様主催のテクニカルアワード表彰式の帰り道、社内 SNS に表彰式の写真を添えて最優秀賞受賞の報告を投稿しましたところ、社長をはじめとして上司、同僚、他部署の方からたくさんのお祝いのコメントをいただいたことを懐かしく思い出します。

テーマである Web-EDI システムは、当社のお得意様先から Web ベースで当社の在庫を確認したりご発注をいただける仕組みで、お客様にとっても当社にとっても大いに合理化に役立つものとなりました。

その後も JC400（SP4i）を使って次々に開発を行い、仕入先の Web-EDI やタブレットによる生産管理、販売管理など多くの成果を上げることができています。

日頃より当社開発にご支援いただきましたミガロ.の皆様に心より感謝を申し上げますとともに、貴社の今後のますますのご発展をご祈念申し上げます。

駒田 純也 様

ユサコ株式会社
システムグループ　マネージャー

まずは創刊 10 周年、おめでとうございます。

弊社が Delphi/400 を導入して 7 年が経ちますが、導入当時の右も左も分からない状態のときに『ミガロ.テクニカルレポート』があったことで、「こんなこともできるのか」と本当に参考になり、心強かったのを覚えておりますし、もちろん今でも毎号楽しみにしております。

わたしの論文はフレームワーク構築という地味でマニアックな内容でしたので、あの文字数でうまく伝えられるのか、テクニカルレポートとして受け入れて貰えるのかと不安な部分もありましたが、社内では理解してもらうのが難しい技術的な部分を評価していただけたことは、自分にとって自信になりました。

新しいアイディアを得る場として、そして若手が自分の実力を相対的に測る場にもなりますので、できる限り続けていって欲しく思います。

創刊10周年記念

お客様からの祝辞･お客様の声　氏名50音順

小山 祐二 様

澁谷工業株式会社
経営情報システム部
課長代理

●『ミガロ.テクニカルレポート』向け論文執筆の思い出、記憶に残っていること

澁谷工業へ入社する前は、10数年程IT業界を離れていましたが、そのRPGのスキルしか持たない私が初めてDelphi/400で構築したプログラムが「オンライン個人別メニュー」です。今からすると、「我ながらよく構築した」と思います。

●『ミガロ.テクニカルレポート』向け論文に寄稿してよかったこと

この5年間は、年に2本の論文を執筆しています。最初の頃は芳しい評価を得られませんでしたが、ここ数年は「何故このことを行うのか」「その方法は何故有効なのか」など、どうすれば読み手にわかりやすく伝えられるか、というロジカルシンキングを意識できるようになったことが、うれしい限りです。

●『ミガロ.テクニカルレポート』への期待、要望

●その他、『ミガロ.テクニカルレポート』に関連することでしたら何でも

他執筆者との交流につながればうれしく思います。また、北陸の企業様との交流も実現できればうれしく思います。

寒河江 幸喜 様

日綜産業株式会社
電算室 室長

●『ミガロ.テクニカルレポート』向け論文執筆の思い出、記憶に残っていること

7年前のことなので、あまりよくは覚えてはいませんが、社会人になってから論文を執筆した経験がなかったので、大変だったという思いと、よい勉強をさせてもらったという記憶があります。

また、論文の内容をミガロ.テクニカルセミナーで発表する前は緊張していましたが、実際に発表してみると楽しくできたことや、寝てる人がいなくてよかったことを記憶しています。

●『ミガロ.テクニカルレポート』向け論文に寄稿してよかったこと

Delphi/400の可能性を少しでも他の会社さんと共有できたこと。少しは会社のアピール（宣伝）になったかもしれません。将来にわたって、受賞者として会社名と名前が残ること。

●『ミガロ.テクニカルレポート』への期待、要望

・ユーザー（社員）さんがDelphi/400で幸せになった事例
・業務効率化・コスト削減・営業支援・売上貢献・意思決定に寄与できるアイデアの共有
・IoT AI 音声の活用　等の事例
・逆に失敗事例も聞きたい

●その他、『ミガロ.テクニカルレポート』に関連することでしたら何でも

発表もよいですが、以下のような全員参加の座談会やディスカッションも面白いかもしれません。
・難しいかと思いますが、他社ツールのネガティブ覆面座談会などは楽しそうです。
・人材不足について。どこの会社さんでも人材不足の課題があると思います。自社であれば採用・教育、他社さんであれば、開発ベンダーさん・常駐・派遣などについて他社さんとディスカッションするのも面白いかと思います。
・Delphi/400のメリット・デメリット

創刊10周年記念

お客様からの祝辞・お客様の声　氏名50音順

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　営業企画課

この度は、創刊 10 周年を迎えられまして、誠におめでとうございます！

10 年という長期間にわたり、このような取り組みを継続されることは、並々ならないご苦労もあったことと思います。

論文寄稿の思い出としては、私は大学時代、論文を書かずに卒業しましたので、論文の執筆にためらいとコンプレックスがございました。しかし、その論文が最優秀賞という高い評価をいただいたことがきっかけとなって翌年の iSUC でセッション講師を務めることになり、何とかその任を全うできたことが大きな自信になりました。

SP4i に関して、論文の数が少ないのは少々残念ですが、『ミガロ.テクニカルレポート』のようなアグレッシブな取り組みを今後も継続していただけばと存じます。

末筆になりますが、ミガロ.様のますますのご健勝を祈念いたします。

佐藤 岳 様

ライオン流通サービス株式会社
管理部　管理チーム　主務

日頃は格別のご愛顧を賜り、誠にありがたく厚く御礼申し上げます。

このたびは『ミガロ.テクニカルレポート』が創刊 10 周年を迎えられたとのこと、心からお祝いを申し上げます。

私は「Delphi/400 を利用した定型業務の PDF 化」というレポートタイトルで、第 9 回ミガロ.テクニカルアワード（2016 年秋）に参加させていただきました。私自身はシシステム関連の業務には携わっておらず、参加したときも、「こういうことがしたい」という大枠の仕様を提案して、当社のシステム担当者が設計を行った経緯があります。

現状の業務では、ミガロ.様と直接関わることがなかったかもしれませんが、ミガロ.テクニカルアワードという 1 つのきっかけでご縁ができ、うれしく思っております。

今回のきっかけで築くことができました御社との関係を、今後も継続させていただきたく思っています。

また、これから創刊 20 年、30 年を迎えられることと、御社のさらなるご活躍を祈念し、お祝いの言葉とさせていただきます。

白井 昌哉 様

太陽エコブロックス株式会社
取締役　経理部長

貴社ますますご盛栄のこととお喜び申し上げます。『ミガロ.テクニカルレポート』の創刊 10 周年、誠におめでとうございます。

また、『ミガロ.テクニカルレポート』No.6（2013 年秋）にレポートを掲載していただき、ありがとうございました。そのときに発表した仕組みは、今も社内で有効に使用しています。

当時の社内システムは、業務を 5250 画面および手書きで処理していました。それを何とかして社内システム化したいと思い、開発ツールに Delphi/400 を選択しました。

当初は「自分にこのような言語での開発できるのだろうか」と、不安もありましたが、おかげさまで社内システムの基盤も完成させることができました。これもミガロ.様のサポートがあったからこそと思い、感謝しております。

今後もテクニカルセミナーに参加し、多数のユーザーの声やミガロ.様の開発テクニックを学んでいきたいと思っています。

末筆ながら、貴社のご発展とご活躍をお祈り申し上げます。

創刊10周年記念

お客様からの祝辞・お客様の声　氏名50音順

仲井 学 様

西川リビング株式会社
システム部　課長代理

『ミガロ.テクニカルレポート』創刊10周年、おめでとうございます。

『ミガロ.テクニカルレポート』は、私にとって甘酸っぱい思い出です。

せっかくの夏休みに、頭の片隅には「レポート文字数3000字」が重く重く横たわっていました。早めに早めにと思いながら締め切りぎりぎりになり、ミガロ.様からのやさしくも、きびしい督促の電話……。それも、今となってはいい思い出です。

テクニカルレポートの執筆は、自分と向かい合うよい機会でした。自分の仕事を振り返ることは、なかなかしませんものね。その振り返りが、自分の成長にプラスになったと思います。

これからも『ミガロ.テクニカルレポート』がどんどん盛り上がりますように期待しています！

福島 利昭 様

株式会社ランドコンピュータ
代表取締役

『ミガロ.テクニカルレポート』創刊10周年、誠におめでとうございます。

このような素晴らしいプロジェクトが10年も継続できるのは、スタッフの皆様の知恵と努力の賜物と、感銘を受けているところです。

テクニカルレポートに論文を提出する、という目標に向かってがんばることで、システムの品質も上がりますし、受賞すると、次も頑張ろうという気持ちになります。

また、論文を提出することで、ほかの方の論文を読む集中力も増します。

今後もテクニカルレポートに論文を提出できるよう、Delphi/400を使って社内システムを改善していきたいと考えています。

創刊10周年記念

お客様座談会

ユーザーの経験を広く伝え他社事例の知恵を得る

Migaro.Technical Report 創刊 10 周年を記念して、今まで論文を執筆し、見事に上位入賞された 4 名の執筆者の方々にお集まりいただきました。
執筆された論文の内容やご苦労された点などをご紹介いただきながら、論文を執筆することのメリット、そしてユーザーの方々が構築事例や活用方法などの情報を相互に伝え合い、共有していくことの必要性を、この座談会で存分に語り合っていただきます。

ご出席

石井 裕昭 様
豊鋼材工業株式会社
監査役

寒河江 幸喜 様
日綜産業株式会社
電算室　室長

駒田 純也 様
ユサコ株式会社
システムグループ　マネージャー

仲井 学 様
西川リビング株式会社
システム部　課長代理

上甲 將隆
株式会社ミガロ.
代表取締役社長

司会
飯田 恭子 様
アイマガジン株式会社
取締役　i Magazine、IS Magazine 編集長

取り組みやプロジェクトを論文にまとめる

飯田　最初に、皆様がこれまでどのような論文を執筆されてきたかをご紹介ください。

石井　私は Migaro.Technical Report の No.1 で、「直感的に理解できるシステムを目指して－情報の“見える化”の取り組み」と「RPG を知らなくてもここまでできる－ユーザーサイドからのアプリケーション開発」という 2 本の論文、さらに No.8 で「iPod Touch の業務利用開発と検証」という論文を執筆しました。前者は当時、工場の操業状況や受注製品の進捗・計画状況が把握しづらかったため、それを解決する手段として Delphi/400 で情報基盤を構築した取り組みをまとめました。また後者では、製造現場で iPod Touch を使って撮影する鋼材の写真データを活用するシステムの開発経緯やその内容を紹介しています。

駒田　当社では Delphi/400 を社内の標準開発言語に位置づけています。ただ Delphi/400 は開発の自由度がきわめて高い半面、データベースやコーディング上の命名基準、画面構成や配色などが開発者ごとに自由に作成できるので、統一性に欠ける心配があります。そこでこの問題を回避するために、モデリングやコーディング規則に加え、ユーザーに同じ見た目で、同じ操作基準をもった一貫性のあるシステムを提供する「開発フレームワーク」が必要であると考えました。現行システムに不足している管理テーブルや、標準コンポーネントを拡張したオリジナルコンポーネント、キャッシュ用エンティティクラスなどの共通クラス、アプリケーションの自動配布の仕組みなど、開発生産性をさらに高めるための基盤として、自社用の開発フレームワークを構築しています。No.6 に掲載

創刊10周年記念
お客様座談会

石井 裕昭 様

論文を書くことでプロジェクトの内容を振り返り、新たな気づきが得られます。

された「自社用開発フレームワークの構築」という論文は、この取り組みをまとめたものです。

寒河江　私が執筆した論文は、「Delphi/400で写真管理ソフトとスプールファイルのPDF化ソフトを自社開発」というタイトルで、No.3に掲載されています。当社は建設用仮設機材の設計・開発・製造・販売およびレンタル事業を展開しています。論文は、こうした業務を支援するためにDelphi/400で開発した2つの業務システムを題材にしました。

1つは「検収写真くん」という写真管理ソフトです。レンタル機器が返却された際に、IBM iに検収データを入力し、修理が必要な機器や全損の商品は破損部分を撮影し、検収写真として請求を行っています。「検収写真くん」は大量の写真データを請求データにリンクさせて、業務を改善させるシステムです。そしてもう1つは、スプールファイルをPDF化する「スプールくん」というシステムです。現在は多彩なPDF化ツールが販売されていますが、自作のPDF化ツールは当時まだ珍しく、Delphi/400で意外に簡単に構築できたので、その内容を論文にまとめました。

仲井　私はこれまで、No.2、No.3、No.7、No.8に合計4本の論文を執筆しました。直近の論文は、「送状データ送信システムのWeb化について」です。送状データとは、送状を印刷するためのデータを意味し、当社では送状を各運送会社に送信し、運送会社側で印刷しています。コスト削減の観点で、この送状データの送信方法をVAN経由ではなくインターネットへ変更することに決め、流通BMSと同じようにWebに対応したシステムとして、Delphi/400で再構築しました。

運送会社がインターネット経由で専用Webサイトにログインし、当社があらかじめ用意した送状データを取得できる仕組みです。これにより通信費を大きく削減し、従来とは比較にならないレスポンスとなり、受信時間の短縮も実現しました。当社にとっては最初のWebアプリケーションであり、業務のインターネット利用を進めていくきっかけとなったシステムです。

論文を書くことで新たな気づきを得る

飯田　文章を書くのは時間がかかり、面倒だと感じることも多いかと思いますが、それでも論文を執筆することには、どのようなメリットがあると考えておられますか。

石井　論文を書くに際しては、自分のやったことを振り返ったり、確認したり、レビューしたりする必要があるので、それにより新たな気づきを得たりできます。その点が、とても有意義だと感じます。取り組みの内容をまとめる意味で、論文は自分自身に対する大きな区切りの作業になると実感しました。

駒田　私が書いたのは、自社向けの開発フレームワークについてですが、このフレームワークは当社にとって今後の開発標準、開発基盤になるものでした。論文を執筆したことで、社内に説明するための効果的なプレゼン資料、さらにコンセプトを伝える重要なドキュメントを作成するよい機会を得られたと考えています。

寒河江　論文というまとまった原稿を書いたのはこれが初めての経験でしたが、論文という形式で、どうすれば社外の方々に自分たちの取り組みをうまく伝えられるかを学ぶ貴重な機会になりました。論文内容をミガロ.テクニカルセミ

駒田 純也 様

このレポートを読んで、ほかのユーザーがどう取り組んでいるかを調べています。

寒河江 幸喜 様

他社事例を知ることは、発想を広げ、会社の枠を乗り越える力になります。

ナーでも発表しています。発表前はとても緊張していたのですが、話しているうちに、「多くの方々に自分の経験を伝えることは、とても楽しい」という喜びや高揚感が沸き上がってきたことを覚えています（笑）。

仲井　私は合計4つの論文を執筆しました。1回目は無我夢中で書いていましたが、回を重ねていくうちに、「次はもっとよいものを書こう」という野心のようなものが芽生えてくるから不思議です（笑）。1つの取り組みやプロジェクトの終了後に、論文として内容をまとめ、振り返る習慣をつけるのはとてもよい区切りになりますし、同僚や後輩に考え方や成果を伝えられる点でも貴重です。

他社の事例を知り自社の世界を広げる

飯田　Migaro.Technical Reportを1つのメディアとして考えたとき、その読者としてはどのような感想をおもちでしょうか。

石井　他社の事例を参考にできるので、とても貴重な情報共有メディアだと思います。読んだときにはとくに興味をもてなくても、あとになって何かに取り組む際に、ふと思い出してその論文を探すことがあります。実際、当社で旅費精算システムの開発を検討したとき、ネットで情報を集めていたら、Migaro.Technical Reportに掲載された論文を見つけたことがありました。また、ある論文を書かれたユーザーの方に、もっと詳しい内容を知りたくて直接コンタクトしたこともあります。自分たちだけでは実現できないことも多いので、このレポートは世界を広げる意味でとても価値があると思います。

駒田　私にも、掲載された論文を他社事例として参考にした経験があります。Delphi/400は何でも開発できるツールですが、実際に何を実現できるのかを知りたくて、普段から過去のレポートに掲載されたタイトルをすべてチェックし、ほかのユーザーの方々がどう取り組んでおられるかを調べたりしています。IBM i関連の情報は少ないので、とてもありがたい情報源だと感じています。

寒河江　同感ですね。先ほど、石井さんも指摘されたように、自分たちの会社の枠を超えて、何かを作るのはなかなかむずかしいことです。だから他社事例を知ることは、その枠を打ち破る大きな力を得られるので、とても重要です。私はテクニカルな情報ももちろんですが、業務や意思決定に貢献するアイデアや考え方、会社にとって価値のあるシステムを実現するヒントなどを得られる点が素晴らしいと思っています。レポートの形で発行されているので、手にとって過去の事例を探せる点もよいですね。

仲井　皆さんと同じように、私もネットでいろいろと検索していたら、行き着いたのはMigaro.Technical Reportだったという経験が多々あります。たとえば寒河江さんが書かれたスプールデータのPDF化に関する論文を参考に、当社もそれに挑戦したことがあります。ネットにはいろいろな情報が溢れていますが、Delphi/400というツールを利用している共同体からの情報、つまり同じDelphi/400を使っておられるユーザーの方々の取り組みが最も役に立つと感じます。

仲井 学 様

このレポートを読んで、Delphi/400を使う仲間が増えていくよう願います。

上甲 將隆

リアルでもオンラインでも、ユーザーの皆様が交流できる場を創造していくつもりです。

ユーザー同士が交流し情報を交換できる場の創造へ

飯田　今後の10年に向けて、Migaro.Technical Reportへの期待や要望、さらにミガロ.の製品やサービスについてのリクエストがあれば、ぜひお聞かせください。

石井　貴重な情報源なので、Migaro.Technical Reportは今後もぜひ継続していただきたいと思います。できれば過去のレポートに掲載された論文について、タイトルだけでもインデックス化するなどして、簡単に目的の論文を発見できるようにしていただけると助かります。

駒田　若手技術者が論文を執筆することで、自分の実力を図る場として機能する点もMigaro.Technical Reportのよさだと思います。今後はレポートやセミナーだけでなく、ユーザー同士が実際に情報を交換したり、交流を図れるような場を作っていただけるとよいですね。

寒河江　Delphi/400は、いろいろなことを実現できるツールです。でも無限の可能性を秘めるがゆえに、何ができるかのヒントやアイデアを得たり、それらの情報を共有していくことが重要だと思います。ユーザーの数だけ知恵があるはずなので、その英知を集める役割をMigaro.Technical Reportに期待したいですね。レポートが歳月を重ねることで、Delphi/400の引き出しがますます増えていくことを願います。

仲井　私はDelphi/400のユーザーにはもちろんですが、まだDelphi/400を使っていない方々にも広く、Migaro.Technical Reportを読んでいただくような機会が広がってほしいと思います。「Delphi/400ではこんなことができるのか」という理解が深まれば、Delphi/400を使う仲間が増えていくはずです。そうした仲間を増やすような役割を、Migaro.Technical Reportに期待したいですね。

上甲　検索性を高めるインデックス化については、ぜひ検討したいと思います。また東京・大阪だけでなく、全国におられるユーザーの方々が交流を図り、ヒントやアイデアを共有できる場を作ることの重要性は、当社も認識しています。

　この座談会には、皆様にオンラインで参加いただきましたが、こうした形でも十分に有益なコミュニケーションが図れることを実感しました。今後はリアルでもオンラインでも、ユーザーの皆様が交流できる最適な方法を考えていきます。さらに印刷物にしろ、ネットにしろ、Migaro.Technical Reportの内容をより多くの方々に参考にしていただけるように努力していくつもりです。

石井　IBM i市場全体が、もっと活発に情報を発信し、IBM iのよさを伝えていく必要があると思います。Migaro.Technical Reportはその先頭に立って、今後も有益な情報を積極的に発信していくメディアであるよう願っています。

M

＜受賞論文＞

石井 裕昭 様

No.1　最優秀賞
「直観的に理解できるシステムを目指して　ー情報の“見える化”の取り組み」
同 最優秀賞
「RPGを知らなくてもここまでできるーユーザーサイドからのアプリケーション開発」
No.8　最優秀賞
「iPod Touchの業務利用開発と検証」

駒田 純也 様

No.6　最優秀賞
「自社用開発フレームワークの構築」

寒河江 幸喜 様

No.3　ゴールド賞
「Delphi/400で写真管理ソフトとスプールファイルのPDF化ソフトを自社開発」

仲井 学 様

No.2　優秀賞
「Delphi/400による物流システムの再構築」
No.3　優秀賞
「イントラでのPHP-Delphi-RPG連携」
No.7　シルバー賞
「荷札発行システムリプレースについて」
No.8　シルバー賞
「送状データ送信システムのWeb化について」

Migaro. Technical Award 2017

お客様受賞論文／ミガロ. テクニカルアワード

最優秀賞

貸金庫と鍵のマッチング業務をDelphi/400で実現
―文字認識データと基幹システムデータを統合

佐藤 正 様
株式会社富士精工本社
管理部　システム課
課長

株式会社富士精工本社
http://www.fujiseiko.co.jp/

1929 年に国産初の金庫室扉を製造。以来、金融機関や原子力施設などに向けて高度な安全性・信頼性を必要とされる金庫室扉、貸金庫、防護扉などを一貫して開発・製造している。セキュリティへの社会的な重要性が高まる中で、未来に向けたトータルセキュリティの創造に取り組んでいる。

当社の業務とシステムについて

富士精工本社は金融機関や原子力施設、さらにホテルやオフィスビルなどのお客様を対象に、高度なセキュリティ機能を備えた金庫室扉、貸金庫、防護扉などの開発・製造を行っている。これらの製品はハードウェア面だけでなく、監視システム、入退室システムなどのソフトウェア面も含めたトータルシステムとして提供。また設計、製造から施工、保守まで一貫したサービスを提供している。

社内システム面では、約 30 年前の System/38 導入に始まり、その後も一貫して AS/400、IBM i を基幹システムとして使用している。ただし従来は、製造作業に直結しないシステムインフラや基幹システムにはあまりシステム投資せず、手作業を多用する事務運用を長い間続けてきた。

2009 年、「オカムラ」ブランドで知られるオフィス家具・機器メーカーの株式会社岡村製作所の出資を受け、当社が 100％子会社となったことを機に、それまで遅れていたシステム投資を積極的に行った。

まず社内に行き渡っていなかった PC の追加導入や、電話・インターネット環境の整備などのシステムインフラ整備を最優先課題として実施した。

基幹システムについては対象の業務範囲が狭く、システム改善も長い間実施していないという課題があった。本稿では、貸金庫の鍵登録システムをメインテーマとしつつ、その他のシステム開発についても簡単に紹介する。

基幹システム改善の取り組み

(1) 既存システムの改善

基幹システムについては生産手配、購買（資材発注）、営業（売上）の 3 つのシステムのみが稼働していた。しかしかなり以前に完成したシステムをそのまま利用するだけで、新しい要求へは対応していなかった。

既存システムの修正対応が難しかった理由としては、以下が挙げられる。

・既存の RPG プログラムを修正できる開発担当者が社内に存在しなかった
・設計書が未整備のため、システム仕様の理解が難しかった
・DB 項目の拡張が全プログラムに影響し、リコンパイルが必要な仕組みだった

そこで既存の基幹システムをいつでも変更可能なように、内容を理解したうえで再構築し、併せて見た目も古くなっていた 5250 画面から GUI 画面に変更すべく取り組んだ。

RPG 開発者不在という制約の中で、システム開発ツールとして考えたのが Delphi である。実は筆者は、以前担当していた業務で、Delphi（オープン系 DB）の開発経験が豊富であった（岡村製作所でシステムを担当）。最初は、IBM i で Delphi を活用する方法がわからなかったが、IBM i の DB をシーム

図1　受注一貫システム全体フロー

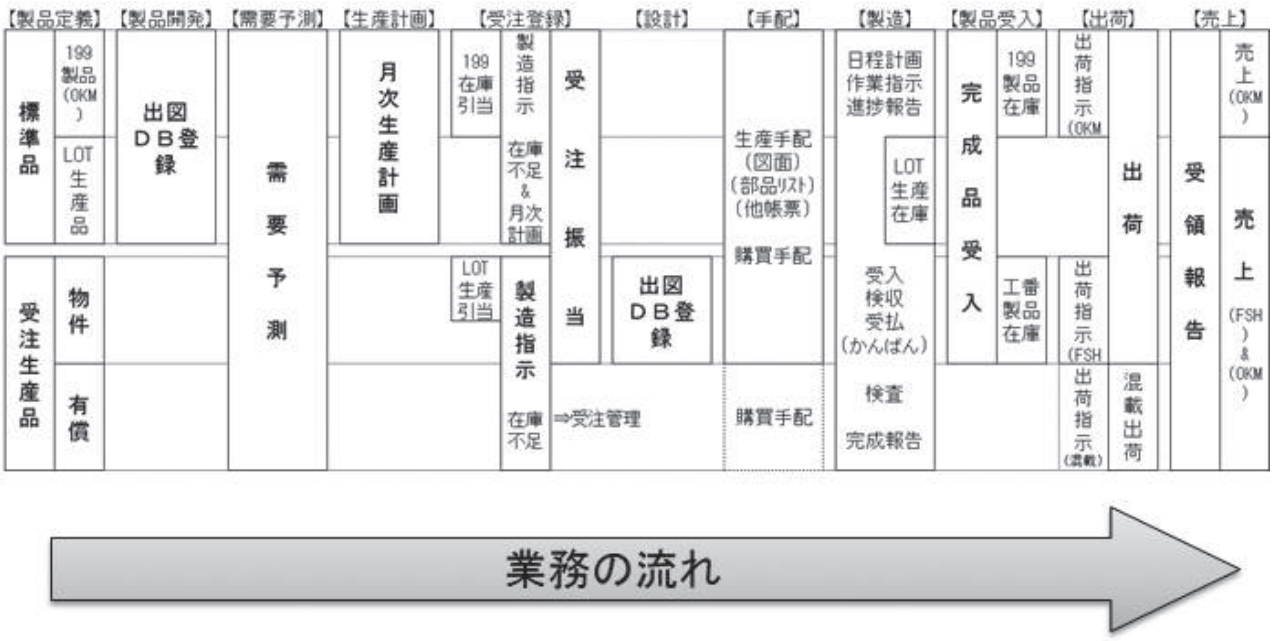


図2　手書き台帳

添付資料5

■■■銀行 ■■■支店様

検査日 ﾊﾝﾀｲ、1.10
工程責任者 望月・大橋

貸金庫錠鍵(キー)番号記録(3列用)

・工程責任者は
①デポ錠キー2個共、開閉確認を行う。
②2本共　使用キーNOを本誌に記入する。
③1BOX完成単位で、「貸金庫組立チェックシート」にて検査を行う

面板 No		
キーNo		
✓S1001	✓S1101	✓S1201
BAX09B	BAW55B	AYX49B
✓S1002	✓S1102	✓S1202
BAW56B	BAW56B	BAX00B
✓S1003	✓S1103	✓S1203
BAW51B	BAW57B	BAX01B

Box No A-1 UP

✓S2101	✓S2201	✓S2301
BAX08B	AYX42B	AYX36B
✓S2102	✓S2202	✓S2302
AYX37B	AYX43B	AYX35B
✓S2103	✓S2203	✓S2303
AYX38B	AYX44B	AYX34B

Box No A-4 UP

✓S1301	✓S1501	✓S1601
BAW52B	BAW58B	BAX02B
✓S1302	✓S1502	✓S1602
BAW53B	BAW59B	BAX03B
✓S1303	✓S1603	✓S1603
BAW54B	AYX45B	BAX04B

Box No A-2 UP

✓S2501	✓S2601	✓S2701
AYX33B	AYX27B	AYX20B
✓S2502	✓S2602	✓S2702
AYX32B	AYX25B	AYX19B
✓S2503	✓S2603	✓S2703
AYX31B	AYX24B	AYX18B

Box No A-5 UP

✓S1701	✓S1801	✓S2001
BAX05B	AYX39B	AYX45B
✓S1702	✓S1802	✓S2002
BAX06B	AYX40B	AYX46B
✓S1703	✓S1803	✓S2003
BAX07B	AYX41B	AYX47B

Box No A-3 UP

✓S2801	✓S3001	✓S3101
AYX30B	AYX23B	AYX17B
✓S2802	✓S3002	✓S3102
AYX29B	AYX22B	AYX16B
✓S2803	✓S3003	✓S3103
AYX28B	AYX21B	AYX15B

Box No A-6 UP

注)箱数は最大で書いてあるので、面板の無い部分は斜線を引くこと。

レスに活用できるDelphi/400の存在を知って即座に検討し、導入に至った。

既存の5250画面をDelphi/400でGUI化するだけの開発ではあるが、RPGをロジックから解析する苦労を乗り越えて、生産手配、購買（資材発注）、営業（売上）の既存の3システムを無事にDelphi/400で稼働することができた。

(2) 受注一貫システムの導入

製品開発から受注、製造、納品、請求にいたるまでの製造業のすべての業務工程を一貫して基幹システムの処理対象とすることで、業務効率を改善するとともに誤りを防止し、事務水準を向上させらえる。これは親会社の岡村製作所で実践していたコンセプトであり、当社でも同様の「受注一貫システム」の開発に取り組んだ。

当社では金庫という製品の特性上、お客様のニーズに合わせた特注品を受注生産するケースが多い。これを踏まえた業務の流れは、【図1】のとおりとなる。

開発はすべてDelphi/400を使用し、開発済みの生産手配、購買、営業（売上）の3つを除く、残りの全業務を基幹システムの中に組み入れる開発を行った。

貸金庫システムの開発について

次に、本稿のメインテーマであるDelphi/400による貸金庫システムについて述べる。

基幹システムの開発が一段落したところで、次の業務改善案件として目を向けたのが、貸金庫の鍵登録業務である。

ここで前提となる貸金庫について、簡単に説明する。貸金庫は金融機関（銀行）が提供するサービスの1つで、有価証券、貴金属などの貴重品を銀行で安全に管理したい顧客向けに、銀行が店舗内に所有する個別の金庫を割り当てて貸し出すものである。全自動型、半自動型などのバリエーションはあるが、最終的に顧客専用の金庫を専用の鍵で開閉する仕組みは共通している。

当社での貸金庫の製造工程では、製造した貸金庫番号に対して専用の鍵番号を対応させているが、この時点ではIBM iの基幹システム上には登録していない。また鍵番号は一定の数字を繰り返し採番しているので、理論的には長期間製造する中で、同一の鍵番号が発生することも考えられる。

ただし同じ金融機関店舗に納入する場合には、鍵番号は絶対にユニークである必要がある。鍵番号のユニーク性を確保しつつ、顧客である金融機関向けの金庫番号と鍵番号とを紐づけた台帳は、最終的には製造チームの「組み立て班」が手作業で作成・登録・確認しており、従来は基幹システムの管理対象ではなかったため、手書きの台帳であった。【図2】

この作業には多くの点で非効率性が認められたので、システム化による業務改善を考えた。

第1の課題は、鍵の刻印（鍵番号）が肉眼では即座に読み取りにくいことであった。鍵番号の読み間違いによる台帳の誤りは、貸金庫という製品の特質上絶対に許されないので、とくに慎重な作業が要求され、台帳作成に時間を要していた。【図3】

第1フェーズでは、鍵番号を読み取るための専用カメラを導入し、肉眼で簡単に読み取れるように鍵番号を表示した。ここでは、鍵番号と金庫番号・受注番号との紐づけは行わず、以降の業務は従来どおりの手作業であった。

そこで次のフェーズでは、以下のシステム対応により業務効率化を図った。

(1) 鍵番号を文字認識

画像認識装置で読み取った鍵の刻印を、文字認識ソフトで鍵番号に文字変換した。【図4】

(2) 基幹システム情報との統合

顧客からの受注情報に基づき貸金庫を製造しているので、受注番号と紐づく貸金庫番号情報はあらかじめ基幹システムに登録されている。したがって、基幹システムの貸金庫番号と文字変換後の鍵番号を結合できれば、従来は手書き作成していた台帳情報をシステムで作成可能になる。

基幹システム情報（IBM i）は、Delphi/400を使えば簡単にPCのGUI画面上に呼び出せる。一方、鍵番号情報も、画像認識装置とDelphi/400を接続することで、文字読み取りソフトの変換後ファイルをDelphi/400の同一画面で呼び出せる。技術的な補足説明となるが、画像認識装置とDelphi/400の接続には「TCP Client」を使用している。

開発したDelphi/400画面での実際の登録作業は、以下のとおりとなる。

①受注番号をもとに登録作業する貸金庫番号の範囲を選択する。実際の金庫の区画をイメージした配置図を画面に呼び出し、ドラッグ選択により、今回登録作業したい金庫番号を選択できる。【図5】

②個々の貸金庫番号を選択して、カメラ装置に鍵の現物をセットする。

③Delphi/400の鍵登録画面「刻印番号読込」により、読み取り処理を実行する。②で選択した金庫番号と、③で読み取った鍵番号がDelphi/400システム上でリンクされる。なお、鍵に刻印された鍵番号以外に、タグ番号も同時に読み取り処理を実行する。

(3) 印刷処理

上記の登録作業により、金庫番号と鍵番号が連携され、結果は基幹システムに保管された。これをもとに、懸案であった台帳の自動作成および印刷が可能となった。なお、出力帳票の承認印は、Delphiで開発した電子印鑑で押印している。【図6】

また貸金庫の鍵は、専用の鍵保管袋とともに納品しているが、この袋の印刷も基幹システムからの帳票印刷機能で実行可能となった。

以上により、読みにくい鍵の刻印を肉眼で確認する手間を省略したことに加えて、読み取った鍵番号と基幹システムの受注情報・金庫情報との連携や、その結果としての帳票印刷まで一連の業務を実行できるシステムをDelphi/400で開発できた。

その他のシステム対応

基本的に、業務システムはすべてDelphiおよびDelphi/400で開発する方針としている。前述のシステム以外に、以下のようなシステム対応も実施した。

図3　貸金庫鍵(見本)

図4　画像認識装置

(1) IBM i 基幹システム以外のシステム

社内で共通に使用するシステム（勤怠管理、ワークフロー、電力消費量確認など）も Delphi で開発し、システム統合メニューの中に組み入れた。

(2) アプリ情報検索システム

Delphi/400 開発効率化のため、システム開発者が利用するシステムを構築した。

Delphi/400 の Delphi ソースプログラムの中で使用している IBM i の DB を一覧で表示。これにより、ある DB を変更した際の Delphi プログラムへの影響範囲が一目で確認できるようになった。

今後の展望･最後に

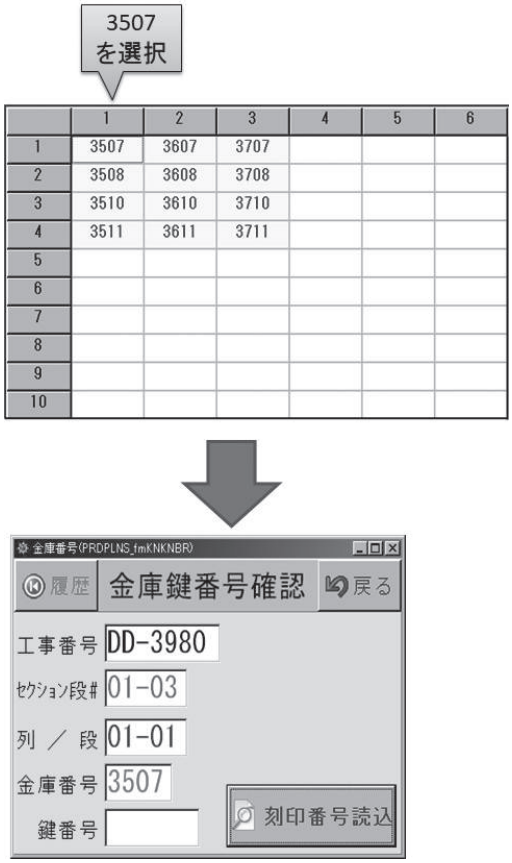
現在は設計図面のペーパーレス化に取り組んでいる。CAD で作成した設計図面（TIFF 形式）は基幹システムの受注データと紐づけられていない。これを Delphi/400 画面を使って連携させることで、設計図面を PDF で管理するとともに、部品、納期、数量などの特記事項を図面と連携する予定である。

本稿でご覧いただいたとおり、Delphi/400 は、IBM i 基幹システムとその他のシステムとの連携に非常に強みを発揮する。

当社では、すべてを Delphi/400（Delphi）で開発しているが、本稿をご覧の Delphi/400 ユーザーの皆様も、もし IBM i 基幹システムと外部システム連携の課題をお持ちであれば、Delphi/400 の活用を検討するようお勧めしたい。

M

図5　金庫選択画面から個別金庫番号を選択



	1	2	3	4	5	6
1	3507	3607	3707			
2	3508	3608	3708			
3	3510	3610	3710			
4	3511	3611	3711			
5						
6						
7						
8						
9						
10						

図6　印刷帳票

BOX錠前チェックリスト

FSH確認欄		OKM確認欄
検査日	2017/01/20	
工程責任者	承認者	施工管理者
室田		

型式：BDP増設（スライドロックⅡ取付）

凡例	鍵番号とBOX番号の一致	鍵番号の一致		アルミタグ番号とBOX番号の一致	施解錠確認		取付金具に緩みは無いか	正しい位置に格納されているか
BOX番号		一本目	二本目（※自動ボルト）		一本目	二本目（※自動ボルト）		
鍵番号								
1076								
BFH93B								
1077								
BFH92B								
1078								
BFH87B								
1079								
BFI15B								
1080								
BFH91B								
1081								
BFH86B								
1082								
BFH81B								
1083								
BFI02B								
1084								
BFI11B								
1085								
BFE70B								
1086								
BFE79B								
1087								
BFE65B								
1088								
BFE63B								
1089								
BFE78B								
1090								
BFE66B								
1091								
BFE67B								

凡例	鍵番号とBOX番号の一致	鍵番号の一致		アルミタグ番号とBOX番号の一致	施解錠確認		取付金具に緩みは無いか	正しい位置に格納されているか
BOX番号		一本目	二本目（※自動ボルト）		一本目	二本目（※自動ボルト）		
鍵番号								
1099								
BFI31B								
1100								
BFI18B								
1101								
BFH88B								
1102								
BFE62B								
1103								
BFE56B								
1104								
BFE68B								
1105								
BFE74B								
1106								
BFE43B								
1107								
BFE45B								
1108								
BFE47B								
C1109								
BFE51B								
1110								
BFI19B								
1111								
BFH90B								
1112								
BFI29B								
1113								
BFI28B								
1114								
BFI33B								

ゴールド賞

Windowsタブレット導入による工作部材料受入業務改革

小山 祐二 様
澁谷工業株式会社
経営情報システム部
課長代理

澁谷工業株式会社
http://www.shibuya.co.jp/

パッケージプラントを主力製品とする東証・名証１部上場の機械メーカー。とくに、国内外の大手飲料メーカーに採用されているボトリングシステム製造では、世界トップの地位を確立している。近年では無菌化などの技術力を活かし、再生医療事業も積極的に展開している。

はじめに

澁谷工業では、「カスタマーファースト（※1）」を貫き、お客様のニーズに合わせたパッケージングプラントを「ターンキー（※2）」で提供するビジネスを主体としている。また最近では、「再生医療」事業にも進出している。

当社のホストコンピュータの変遷は、System/32 から始まり、現在の PureFlexSystem（※3）に至る。そして新たに基幹システムを構築する場合は、主に Delphi/400 を利用し構築している。しかし現時点でも、5250 画面で稼働するシステムは少なくない。

そんな中、当社の工作部よりタブレットを利用した業務改革相談を受けた。工作部は部品加工を担当する部門である（※4）。相談された内容は、現場でタブレットを利用して既存の 5250 画面システムを使いたいというものであった。

しかし工作部の業務およびシステムを分析してみると、さらに合理化する余地があるとわかったので、新しいシステム対応を提案することにした。本稿では工作部からの依頼をトリガーとして、現場にマッチし、かつ他部門を巻き込んで構築したシステム内容を説明する。

工作部の既存システム

既存の工作部システムは、基本的に 5250 画面で構築したシステムである（※5）。【図 1】【図 2】以下に、そのシステム概要を記す。

①伝票発行時にマスターから部品工程情報を自動作成
②部品納期および予定工数から、部品工程納期を自動計算
③実績管理
④工程管理／負荷管理　など

既存システムの問題点

既存の工作部システムは、自部門内の業務をターゲットとしたため、前工程の情報がまったく見えないという課題があった。

そこでまず、前工程について説明しよう。部品加工するには、必ず材料が必要である。その材料調達は、以下の 2 つに分類できる。

①型切鋼材

加工部品の材料を外注先企業から調達する場合、「型切鋼材」と呼ぶ。この型切鋼材システムおよび業務の問題点には、下記が挙げられる。【図 3】

・前工程システムは、担当部門がスタンドアロンで管理している（※6）。そのため、工作部をはじめとする各部門とシステム情報を共有できない。
・外注先企業の事務所宛てに、伝票とともにバーコード付き荷札を郵送している（*7）。荷札は納品物とともに当社へ送付されることを想定しているが、外注先の事務所と作業現場が離れているため、作業現場に荷札が回付されず、結果的に当社に戻ってこないことが多い。

図1　既存工作部の5250画面システム

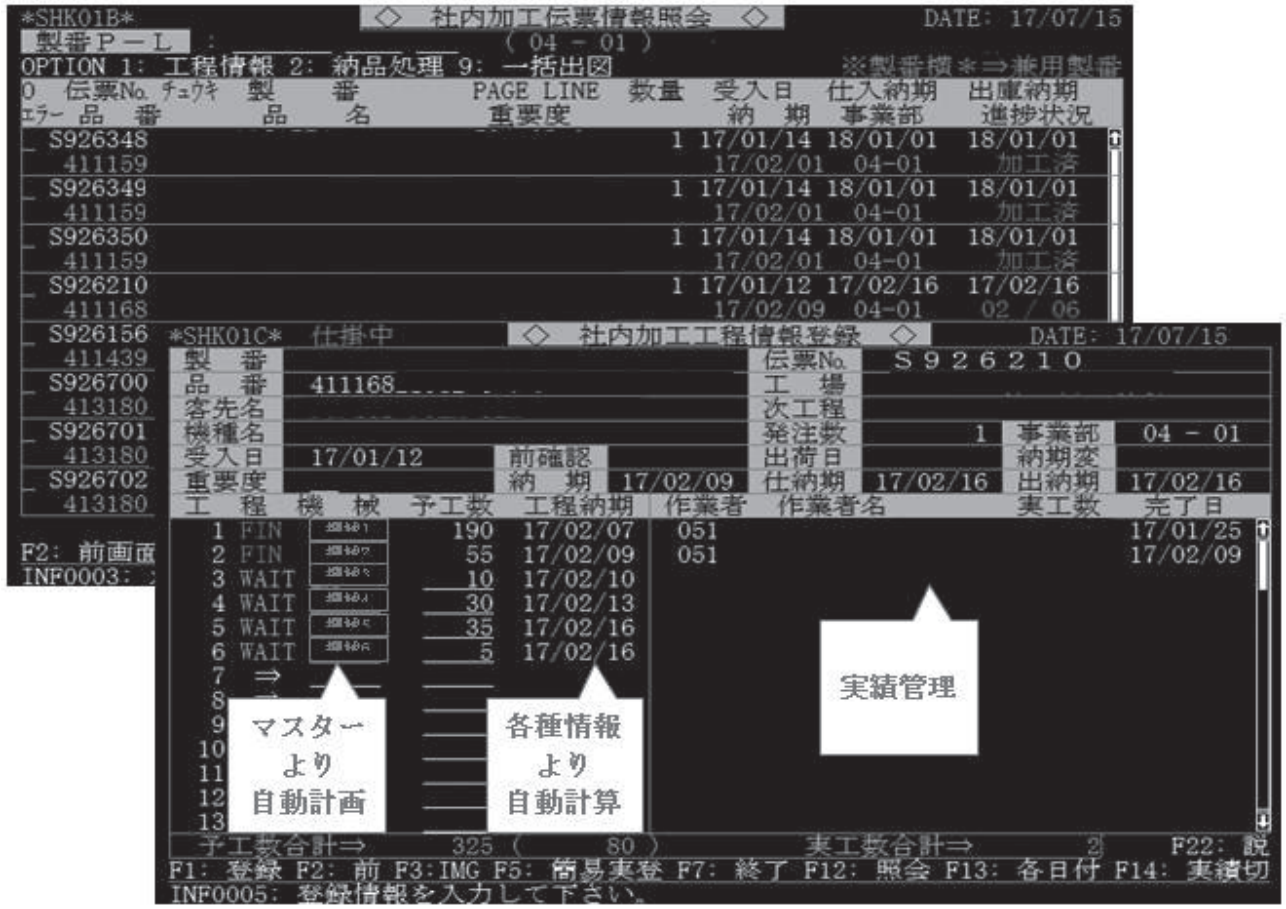
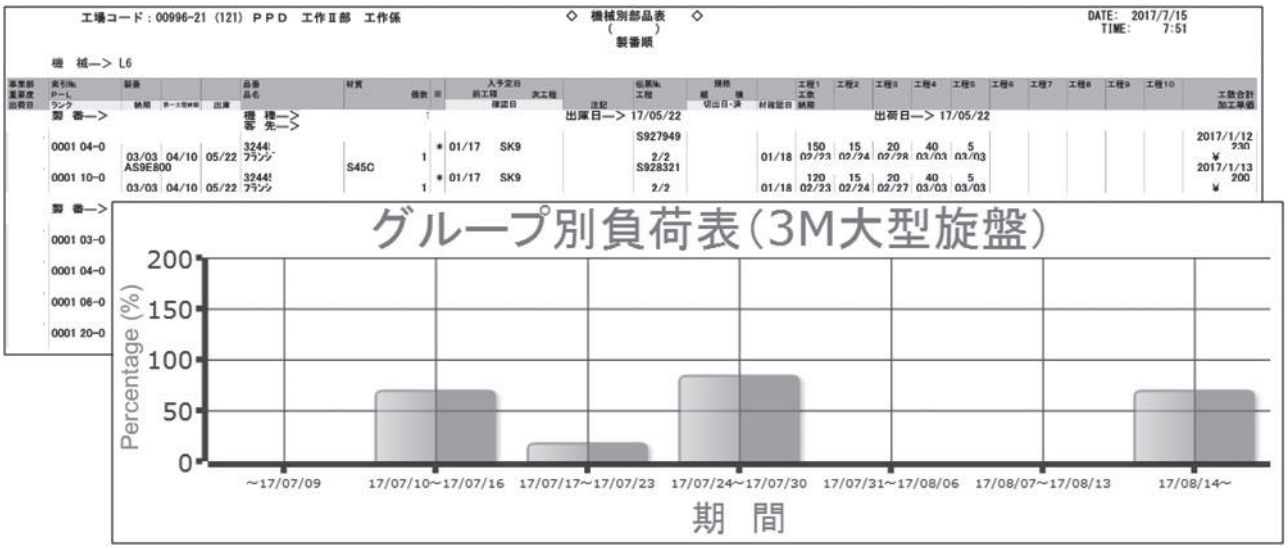


図2　既存工作部の工程管理(Excel)／負荷管理(WEB)



②切出

当社担当部門に依頼し、社内ストック素材を必要量切断する場合を「切出」と呼ぶ。切出システムおよび業務の問題点には、下記が挙げられる。【図 4】

・工作部システムとのリンクがない
・該当伝票と「切出」が 1 対多であるため、工作部が材料調達する単位の「切出」が伝票番号から一意に決定できない。

工作部システムと前工程システムが連携していないので、工作部では「型切鋼材」情報は電話、「切出」情報は切出システム画面に切り替えて確認していた。そして工作部に納入される材料は、いつの間にか各材料置き場に置かれているという運用である（以下、「材料到着」と言う）。そのため工作部メンバーが毎日、各材料置き場を回り、「この材料はどの加工に使うものか」を考えながらメモを取り、そのあと事務所で既存システムに入力していた（以下、「材料受入」と言う）。

システム設計と工夫点

これらの現状を踏まえ、システム設計に入った。まず、部品加工業務の現場でシステムを利用するので、依頼どおりにタブレットは必須と判断した。機器構成として、Windows タブレット、タッチペン、Bluetooth 接続バーコードリーダーを採用した。やはり現場で利用する場合、指での直接操作は向かないと判断した（※ 8）。

タブレット上で稼働するシステムについては、当初の依頼どおり既存の 5250 画面の使用を検討したが、文字入力ベースである 5250 画面を、選択入力との相性がよいタブレット上で利用するには非常に使い勝手が悪い。そのため方針を変更し、GUI ベースで IBM i のタブレットシステムを構築することとした。

また、工作部の業務改善に役立つ新機能のいくつかは、PC での使用を想定したが、こちらも IBM i の GUI アプリケーションとして開発した。

PC およびタブレットで稼働する IBM i の GUI アプリケーション開発ツールには、Delphi/400 を採用することとした。

これは、今までにも IBM i 対応の業務アプリケーションを数多く Delphi/400 で開発し、開発手法に慣れていること、および Windows タブレットアプリケーションであれば、従来の PC アプリケーションとまったく同じ Delphi/400 の開発手法で構築できることが理由である。

業務改善に直結するシステム改善ポイントとして、以下の項目を重点的に考えた。

1　前工程業務システム構築、および工作部システムとの連携
2　材料到着・受入業務の効率化
(1) 材料到着状況、受入状況の把握
(2) 荷札送付時のバーコード作成と送付方法
(3) 伝票 No. と荷札が 1 対多となっていることへの対応

前工程業務システム構築 工作部システムとの連携

既存の工作部システムと前工程システムの連携は必須である。

そこで型切鋼材のスタンドアロンシステムで保有している情報を IBM i の基幹システムに転送し、工作部システムで情報の共有を可能にする。スタンドアロンシステムのデータ（dBase）を CSV に変換後、IBM i に転送する仕組みが今回工夫したポイントの 1 つである。【図 5】

また IBM i への型切鋼材データ取り込み後の各種 UI 画面は、Delphi/400 で構築した。【図 6】【図 7】

材料到着･受入業務の効率化

(1) 材料到着状況、受入状況の把握

荷札バーコードをトリガーとし、材料到着および受入状況に関する情報を各拠点で共有可能とする。材料到着処理画面と材料到着監視画面（タブレット）は、Delphi/400 で構築した。【図 8】

(2) 荷札送付時のバーコード作成と送付方法

荷札送付先を外注先の事務所から材料加工場所へ変更することにより、納品物への荷札添付漏れを防止する狙いは前述したとおりである。

これを実装するには、以下が必要となる。

・荷札情報をメールで送付することにより、外注先の作業現場で受け取りやすくする。
・荷札情報を外注先の作業現場で参照可能とする。

Delphi/400 開発者なら誰しも思いつく帳票ツールは、QuickReport と Fast Report であろう。しかしこれらのツールでは、今回の荷札に求められる以下の要件をすべて満たすのは難しいと判断した。

・PDF など一般的に参照可能な電子媒体に変換可能
・PC のリソースをなるべく消費しない
・バーコードフォントを必要としない

これらを踏まえ、筆者は IBM i 側で Java を利用して Excel を作成する（プロテクト状態）方法を選択し、メール送付することとした（※ 9）。また、該当プロシージャをサブミット（※ 10）することで、PC 資源占有問題を解消した。【図 9】【図 10】

しかし問題はまだある。通常 Excel でバーコードを印字する場合、バーコード用フォントが必要となる。そのため、送付先でも同じフォントの導入が必要となる。

そこで考えたのは、無償フォントで作成したバーコードイメージ化である。今回必要な文字は、「*」「K」「0 ～ 9」となり、これらをプロシージャ側で動的にバーコードをつなぎ合わせた。これによりどこで出力しても、バーコードを表記できる。当然、印字バーコードの読み込み検証も実施済みである。【図 11】

(3) 伝票№と荷札が 1 対多の対応（材料到着および材料受入）

筆者自身、その事実に驚いたが、この状況を変えない限りシステム対応できない。しかし伝票№と荷札を 1 対 1 に紐付けると、他システムやテーブル構成に大きな影響がある。

この壁をどう考えるか。再度担当者とヒアリングを重ねた結果、たとえ 1 対多

図3 「型切鋼材」問題点

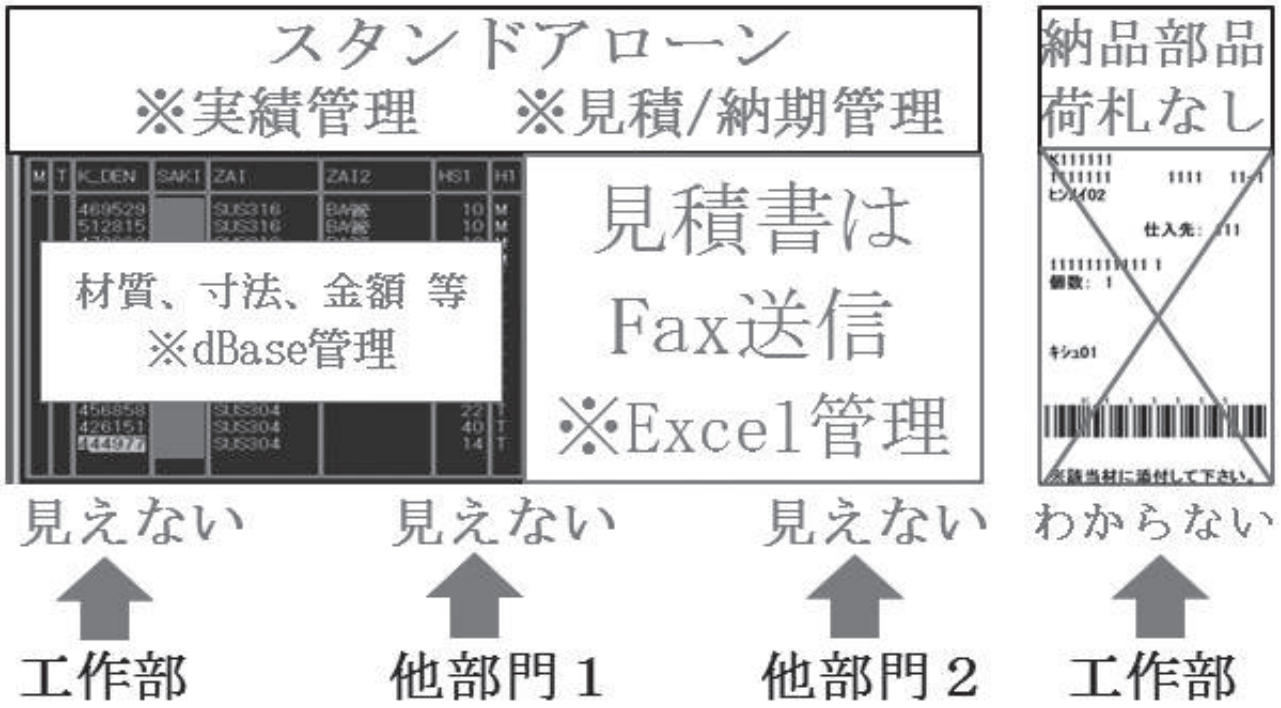
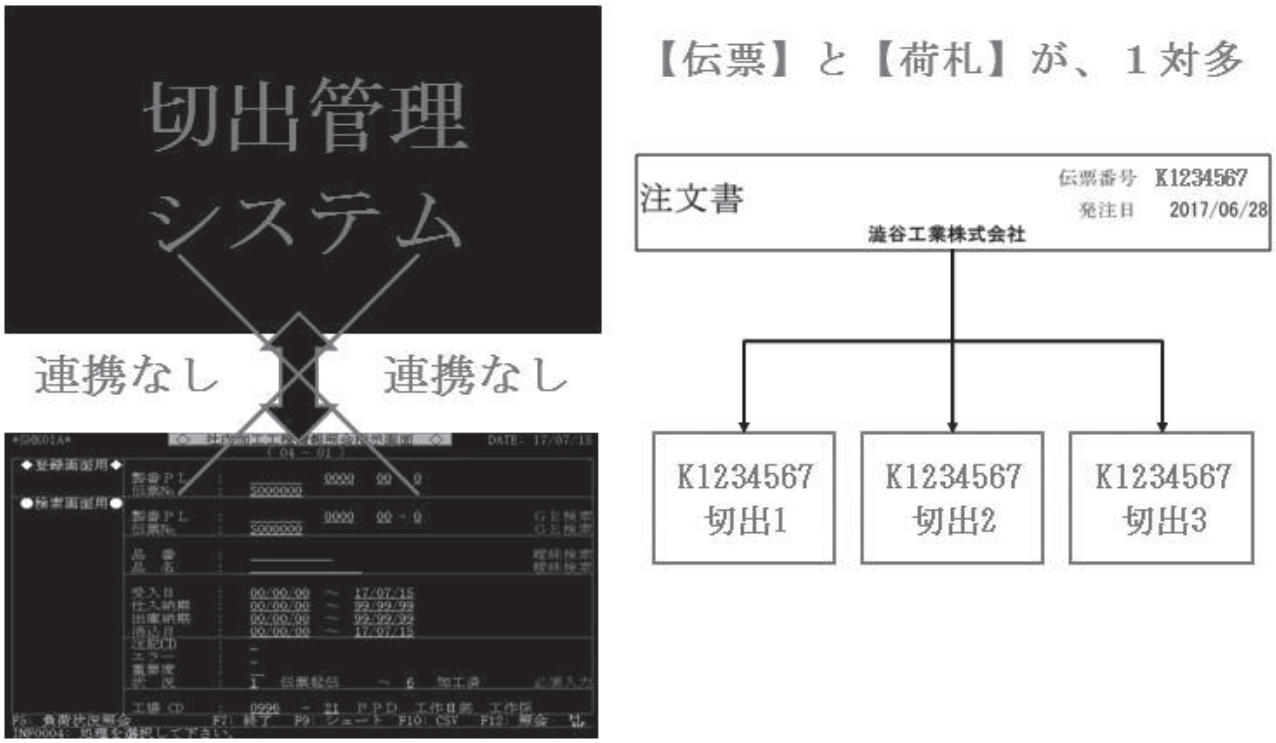


図4 「切出」問題点



であっても、その荷札がどの切出であるかは一目瞭然である旨を確認した。

そこで、Delphi/400で構築した材料受入処理画面（タブレット）の処理方法に関する考え方を改め、バーコード読み込み時に1：多の場合のみ、自動的に別ポップ画面に該当切出情報を表示させ、その中で選択する仕様とした。【図12】

システム運用後の問題点と打開策

上記の工夫点を実装して運用を開始したが､新たな問題点も浮き彫りになった。以下にその問題点を記す。

①リアルタイム参照不可

通常、Delphi/400で最新情報を取得するには、何らかのアクションが必要となる。しかし通常のWindowsアプリケーションのように、何もアクションせずに、最新状況を把握したいという意見が多くのユーザーから寄せられた。

そこで、完全リアルタイムとはいかないが、Delphiタイマーを利用し、一定間隔おきに最新状況を表示する仕様とした。【図13】

②伝票№を読み込めない場合がある

バーコードの汚れなどのため、ある程度の割合で伝票Noを読み込めない現実に直面した。そこで、タブレットで簡単に入力可能なキーボード機能を搭載した。

OSとしてもキーボード機能をサポートしているが、その表示により入力項目が隠れ、また英字・数字の切り替えなどユーザービリティが悪いと判断し、この機能を実装した。簡易キーボード機能もDelphi/400で構築している。【図14】

以上のように、構築前および運用後の問題点をクリアし、現在に至っている。工作部や前工程部門の評価も上々である。現在はさらなる業務改革に向けて、新プロジェクト進行中である。

最後に

昨今、AIやCognitive（ロボット技術を含む）技術が目まぐるしい進化をとげている。そして、それら技術の進化が話題になるたびに、人間の仕事を奪う可能性が論じられている。筆者は、これは「ある意味正しく、ある意味間違っている」と考える。

確かに、たとえ複雑でもルーチンワークなどは近い将来、それらの技術にとって代わられるのは間違いない。またそれ以外でも、多くの仕事が新技術が取って代わると予想する。しかし、新たな仕事が創出されるとも考える。

その新たな仕事とは何か。また誰もが対応できるかなど、現時点では不確定要素が多い。しかし、ひとつ断言できることがある。たとえば、マニュアル記載内容は問題なく対応できるが、そのほかに対応できない。つまりコモディティ化した技術者は、激動する時代では自然淘汰されると考える。そんな今こそ、「何ができる」ではなく「何をするか」の考えが必要である。

これまで筆者がこのMigaro.Technical Awardで発表してきた内容は、決して大きなイノベーションを起こしているわけではない。しかし今あるもの同士を、今までにない組み合わせにより、小さなイノベーションを起こし続けていると自負している。

そして、それを起こし続けることが重要であるとも考えている（※11）。そのことを深く胸に刻み、今後の激動する時代の中、Delphi/400および新技術とともに私自身も成長し続けていく所存である。

最後となるが、本稿にて今後技術者が向かうべき方向の一助となれば幸いである。

（※1）お客様第一主義

（※2）すぐに稼働できる状態

（※3）サーバー（IBM i含む）､ストレージ、ネットワーク、仮想化、管理機能が一体化したシステム。

（※4）当社で部品加工を行う部門。部品加工は社内、外注、社内＋外注など、さまざまな形態で行うケースがある。

（※5）第9回Migaro.Technical Award「Windows Like 5250への道のり」で紹介した内容にて構築

（※6）実績管理はdBase（初期マイクロコンピュータ向けに開発されたDBMS）利用。見積管理は、Excel利用（見積書はFax送信）。

（※7）当社と外注先間では、EDI（Electronic Data Interchange）を利用している場合もあるが、現時点では型切鋼材外注先でEDIはまったく利用されていない（外注先側が対応できないため）

（※8）現場では油やほこりがある中、材料を扱う。そのため、指でタブレット操作する場合、画面が汚れるのはもとより、iPad／Androidアプリケーションのようなスムーズすぎる操作性は向かないと判断した。

（※9）第6回　Migaro.Technical Award掲載の「Delphi/400とDelphiを利用したIBM i資源の有効活用」参照。メール配信は、IBM i内でORACLE社のJavaMailを利用。

（※10）Delphi/400でIBM iと接続しているジョブから、別のジョブで該当プロシージャを実行すること

（※11）岸博幸氏（慶應義塾大学大学院メディアデザイン研究科教授）講演内容「New Combination」より

M

図5　dBase→IBM i取り込み

「dBase情報をCSV変換するVBA」

```
Sub csv作成()
'
'  csv作成 Macro
'

Dim xlAPP As Application
Dim shellObject As Object
Dim batchFile As String

On Error Resume Next

With Workbooks.Open(ThisWorkbook.Path & "¥KATAGIRI.DBF")
.SaveAs Replace("d:¥型切鋼材管理用¥dbase.csv", ".dbf", ".csv"), FileFormat:=xlCSV
.Close SaveChanges:=False
End With

batchFile = "D:¥型切鋼材管理用¥FTPPUT2.bat"
Set shellObject = CreateObject("WScript.Shell")
shellObject.Run batchFile, 1, True

xlAPP.StatusBar = "集計完了！. . . . "

Set xlAPP = Nothing

Set shellObject = Nothing

End Sub
```

「CSV→IBM i及びQUOTEするbat」

```
@ECHO OFF
d:
cd ¥型切鋼材管理用¥
ECHO open 「IBM i転送先名」                                    >.¥FTPPUT.ftp
ECHO user 「IBM iユーザー」 「IBM iパスワード」                 >>.¥FTPPUT.ftp
ECHO quote site namefmt 1                                      >>.¥FTPPUT.ftp
ECHO quote type c 943                                          >>.¥FTPPUT.ftp
ECHO put  d:¥型切鋼材管理用¥dbase.csv  /KOYATEST/dbase.csv     >>.¥FTPPUT.ftp
ECHO QUOTE RCMD CALL PGM(プロシージャ名)                       >>.¥FTPPUT.ftp
ECHO quit                                                      >>.¥FTPPUT.ftp
cmd /a /c ftp -n -s:.¥FTPPUT.ftp                               >.¥FTPPUT.log
REM DEL .¥FTPPUT.ftp
REM DEL .¥FTPPUT.log
EXIT
```

図6　前工程　各種管理画面(システム情報共有可)

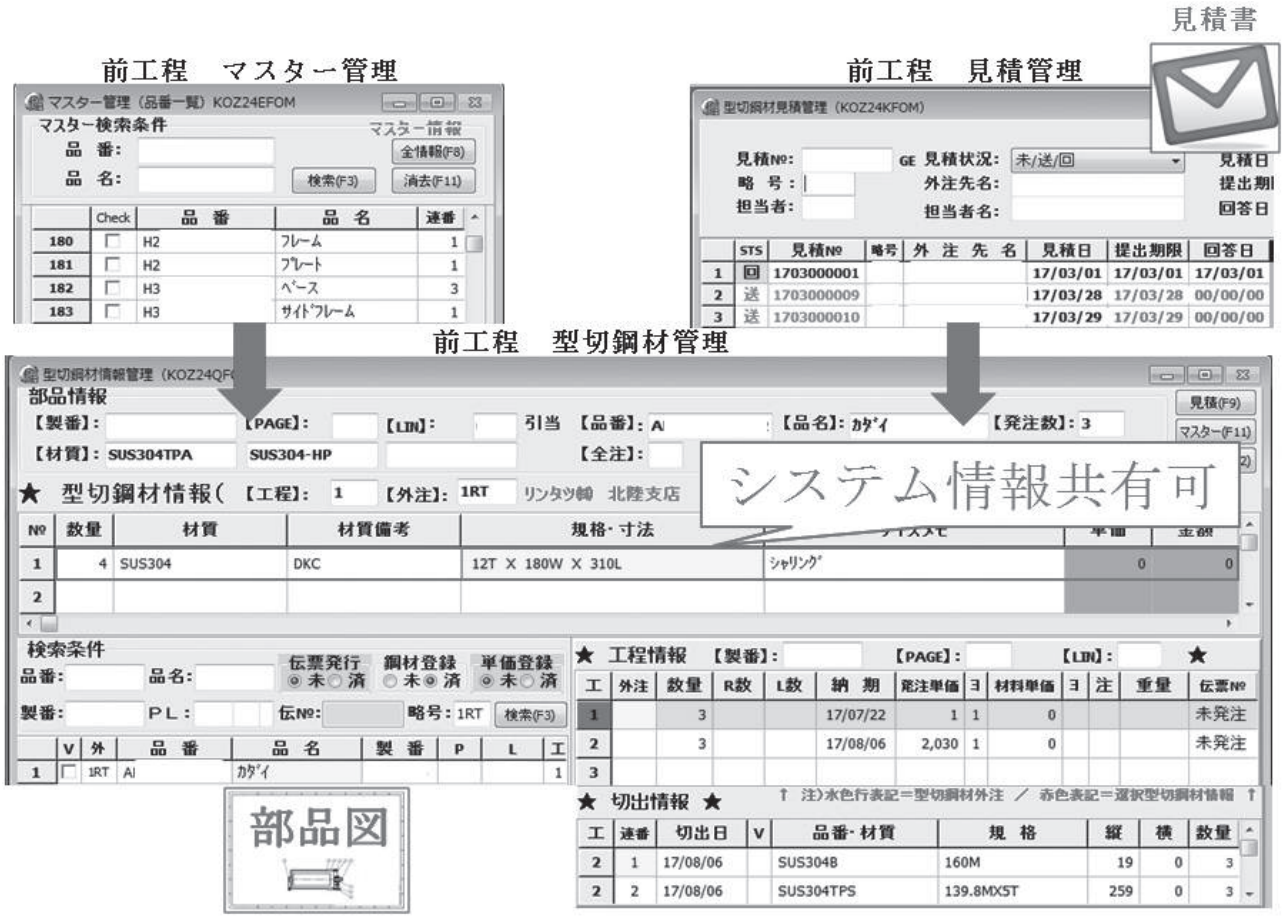


図7　メール添付見積回答→ドラック&ドロップ→情報取り込みイメージ

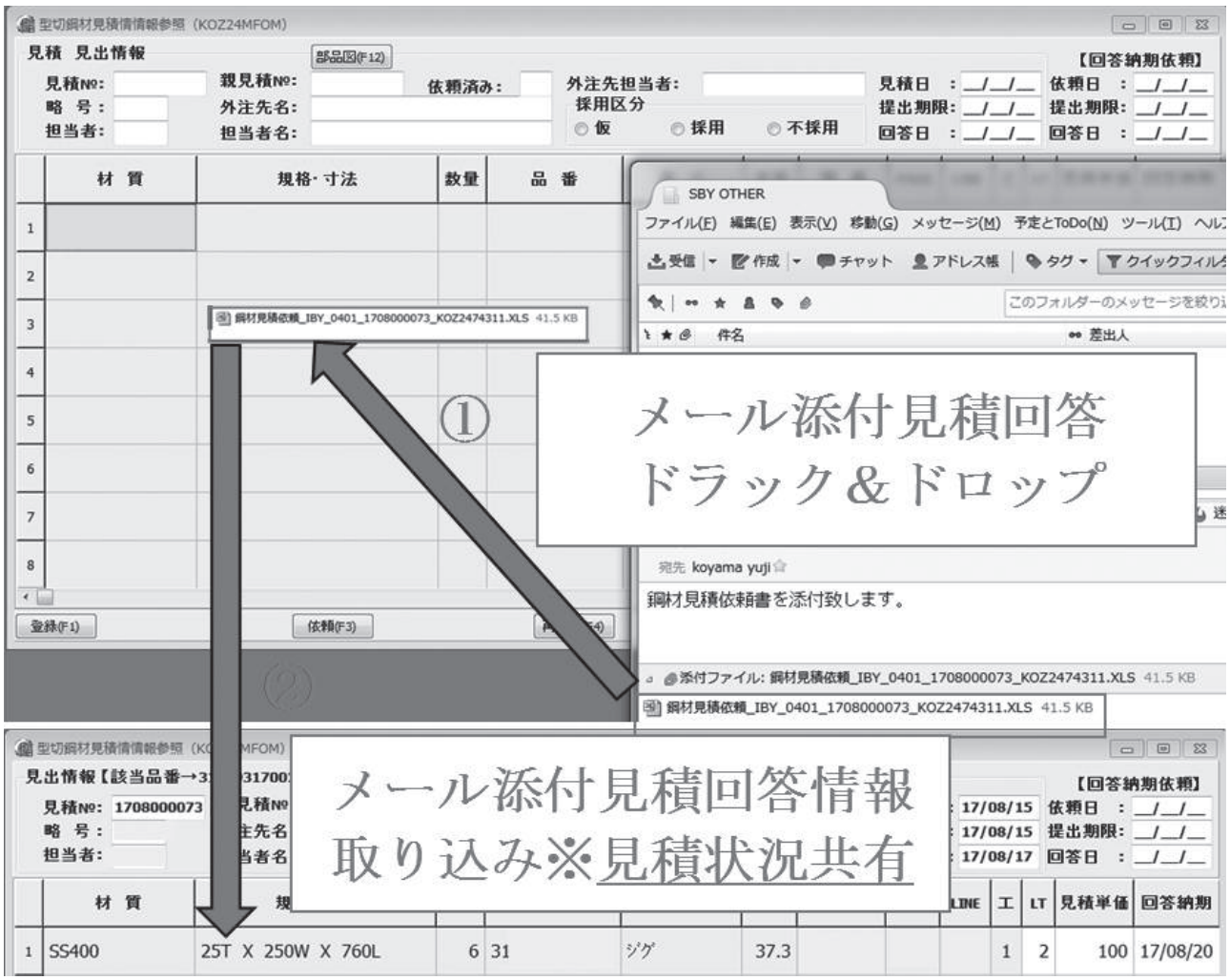


図8　材料到着処理･監視の構成イメージ

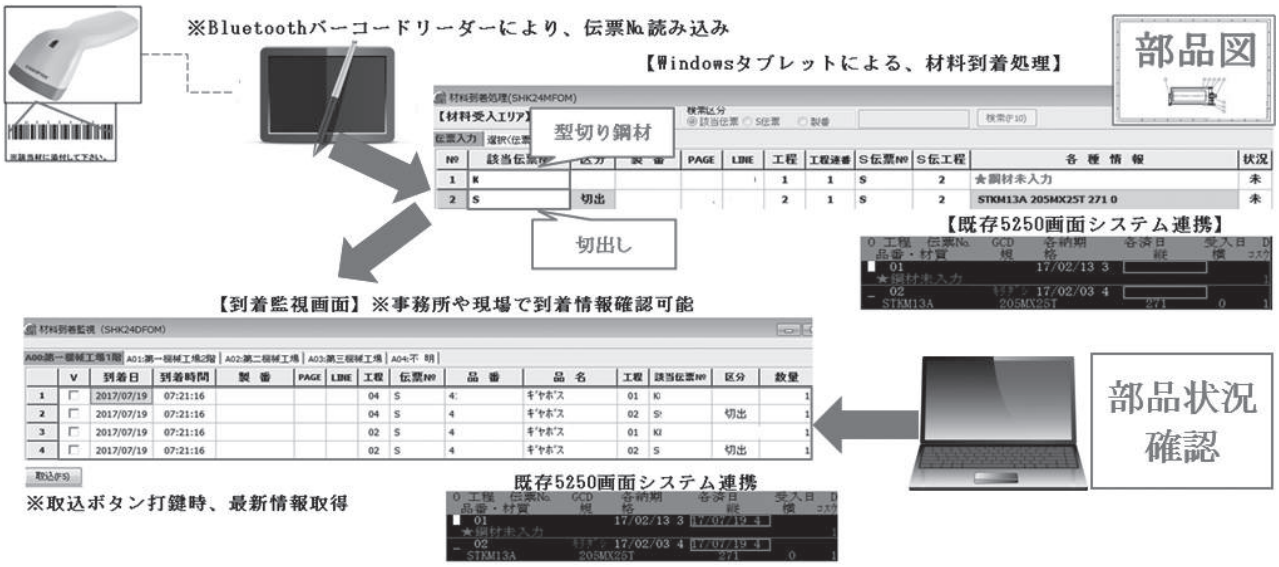


図9　PC資源解放イメージ

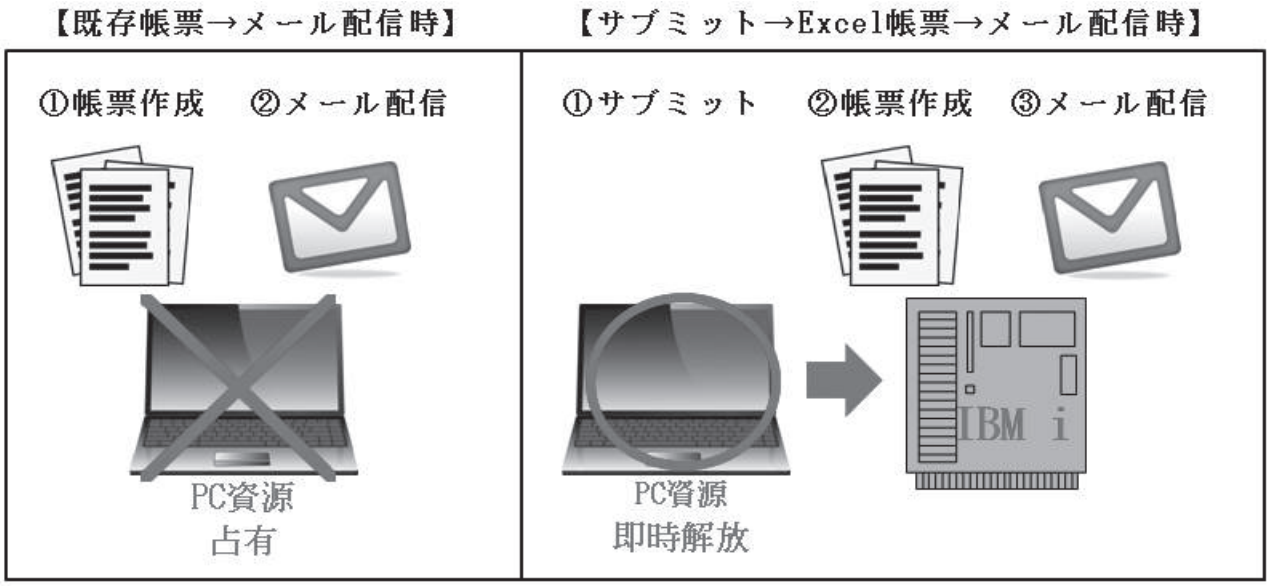


図10　CLプログラムよりJava実行イメージ

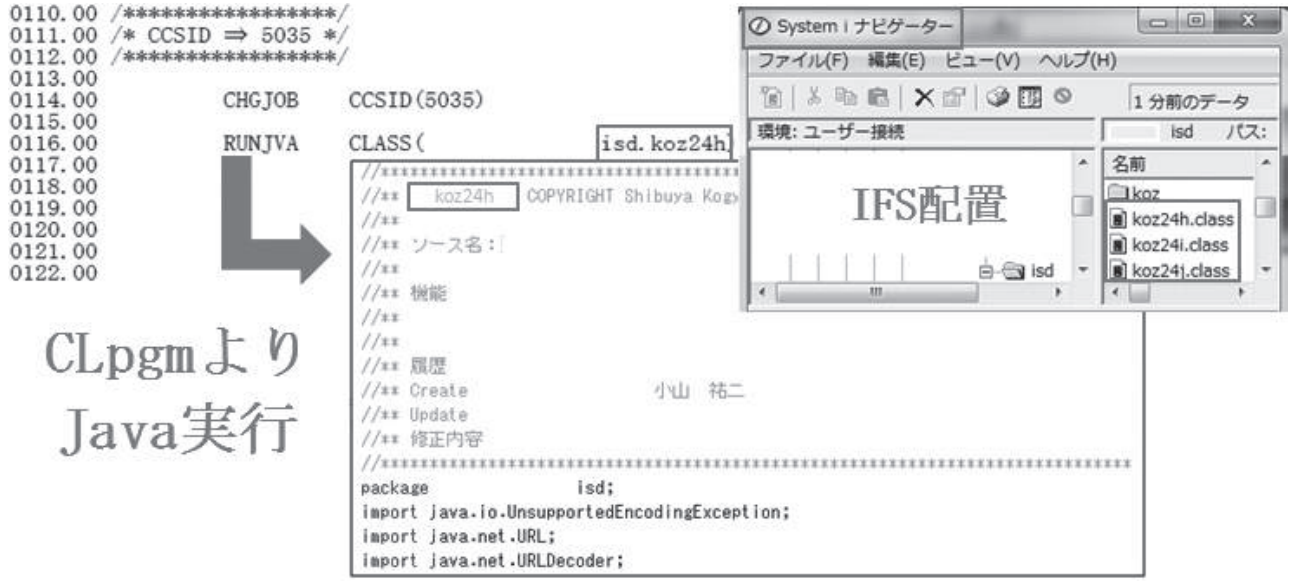


図11　バーコードの流れ

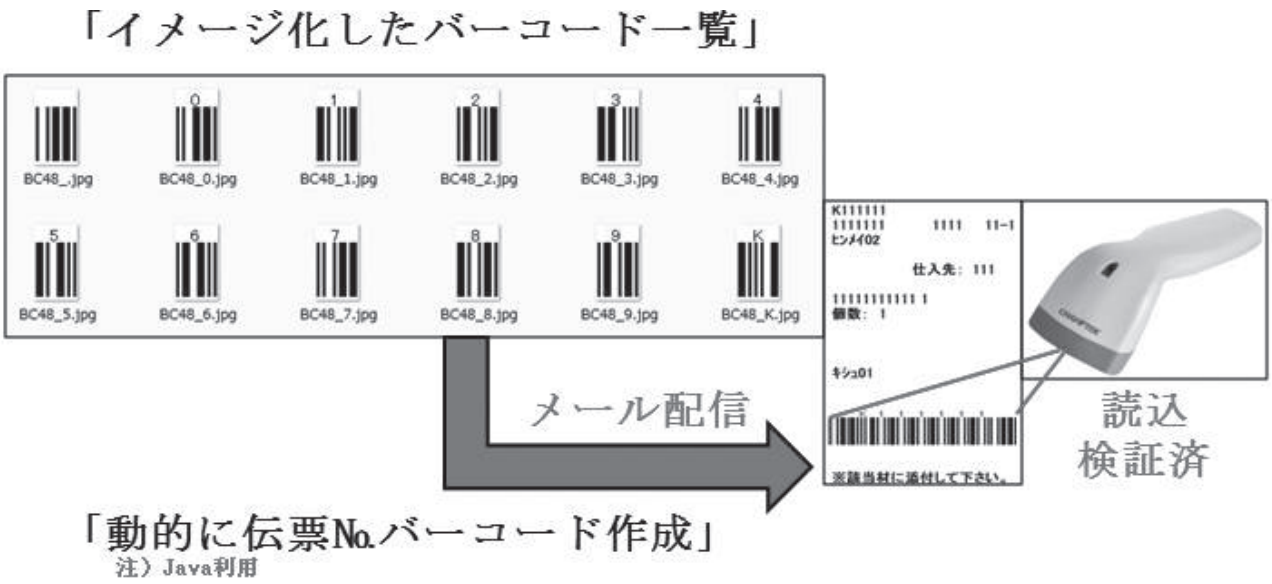


図12　タブレットベースの材料受入処置画面

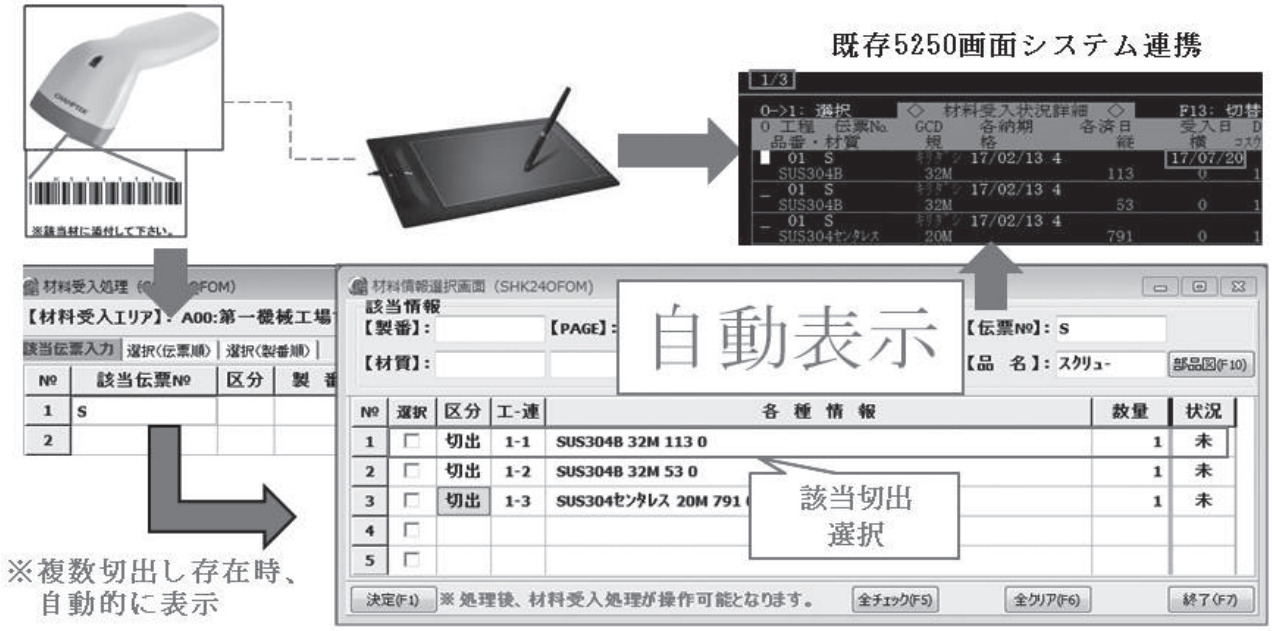


図13　Delphiタイマーコンポーネントおよび監視間隔制御

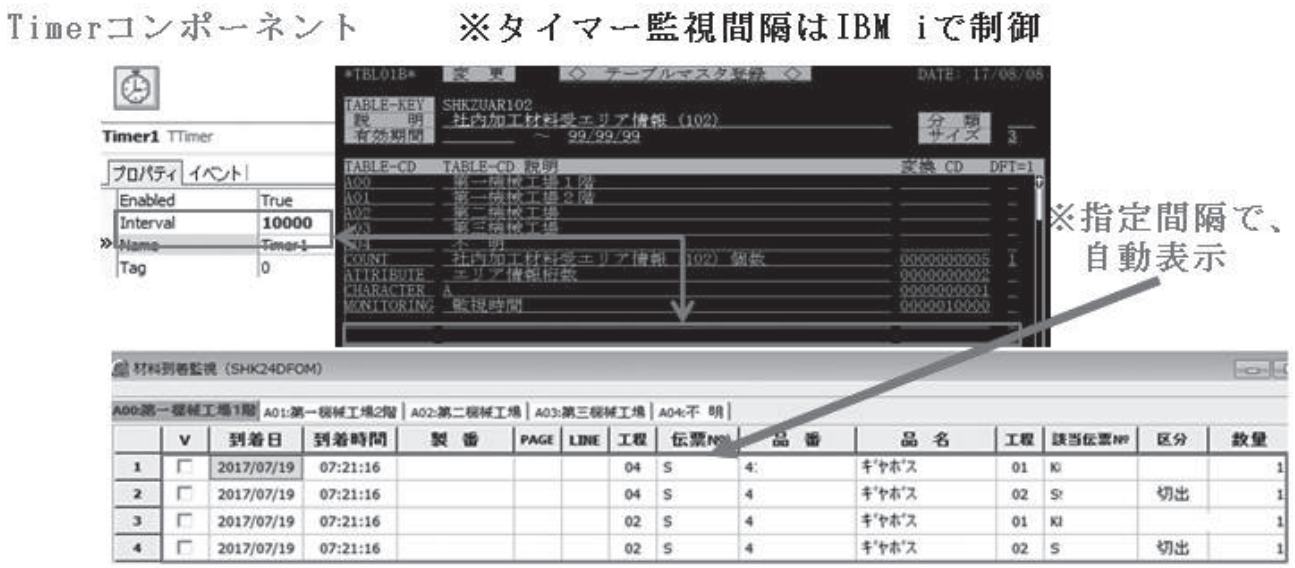


図14　簡易キーボード機能

優秀賞

Delphi/400を利用した各拠点PINGコマンド簡素化

松垣 秀昭 様

ライオン流通サービス株式会社
企画部

ライオン流通サービス株式会社
http://www.lion-logi-s.co.jp/

ライオン株式会社 100% 出資の物流子会社として、全国のグループ物流拠点、および協力物流事業者への委託業務を統括。倉庫管理・在庫管理・輸配送管理など、グループの物流業務全般を担っている。輸配送における CO2 削減など物流業務改善への積極的な取り組みを行っている。

業務課題

災害や緊急事態が発生した時、広域にわたり各拠点ごとの「システムの通信状況」を確認しなければならない。BCP時の状況確認には、迅速さと正確さが求められるが、現在は1件1件PINGにより確認しているため非効率という課題がある。

対応策として、各拠点の通信状況を一括で確認できる使いやすいGUI画面を開発したい。

技術課題

IBM i 上ならPINGコマンドは使用できる。同様に、GUI画面からパラメータ付きのプログラムを実行し、PINGが正常か異常かの判断が可能か。

技術課題の解決策

通信状況が正常か異常かを判断するために、PINGコマンドを実行し、MONMSGを取得するIBM i のCLプログラムを作成【ソース1】。Delphi/400より、IPアドレス調査対象のデータを読み込み後、CLプログラムを実行し、パラメータを取得することで各拠点の通信状況把握を実現した。

また、本機能のユーザーインターフェースとして、「IPアドレス調査」画面をDelphi/400で新規に開発した。【図1】【図2】【図3】

業務課題解決と効果

通信状況の一斉確認により、異常値の早期発見と対処が可能になった。誰でも使いやすいGUI画面により、PINGコマンドを知らない担当者でも、どこからでも通信状況を確認できる。

たとえば、仮に関東地区が被災したとしても、関西地区の従業員が容易に確認することが可能となった。

M

ソース1　PING要求のCL

【ソース1】 PING要求のCL

```
            PGM         PARM(&PURL &PFLG)

            /*-------------------------------------------------------------*/
            /*  CHKPING :  CHECK PING                                      */
            /*-------------------------------------------------------------*/

            DCL         VAR(&PURL) TYPE(*CHAR) LEN(128)
            DCL         VAR(&PFLG) TYPE(*CHAR) LEN(1) /* 1:OK 2:NG */

            DCL         VAR(&STATUS) TYPE(*CHAR) LEN(4) VALUE('UP  ')
            DCL         VAR(&MSG) TYPE(*CHAR) LEN(132)

            MONMSG      MSGID(CPF0000) EXEC(GOTO CMDLBL(ERROR))

            PING        RMTSYS(&PURL) MSGMODE(*QUIET *ESCAPE)
            MONMSG      MSGID(TCP3210) EXEC(DO)
            CHGVAR      VAR(&STATUS) VALUE('DOWN')
            ENDDO

            CHGVAR      VAR(&MSG) VALUE(&PURL *TCAT ' = ' *CAT &STATUS)
            GOTO        SNDMSG

            RETURN

ERROR:  /* RCVMSG     MSGTYPE(*LAST) RMV(*NO) MSG(&MSG) */
            GOTO        CMDLBL(ENDPRO)

SNDMSG:
        /* SNDPGMMSG  MSG(&MSG) TOMSGQ(*TOPGMQ) MSGTYPE(*DIAG) */

            IF          COND(&STATUS *EQ 'UP  ') THEN(DO)
              CHGVAR    VAR(&PFLG) VALUE('1')  /* OK */
            ENDDO

            IF          COND(&STATUS *EQ 'DOWN') THEN(DO)
              CHGVAR    VAR(&PFLG) VALUE('2')  /* NG */
            ENDDO


ENDPRO:
            ENDPGM
```

図1　初期画面

図2　正常画面

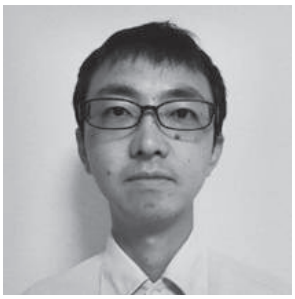
図3　異常画面


ＩＰアドレス調査
実行環境指定
本 社
東日本
西日本
調査結果表示
Excel出力
各拠点のIPアドレスを、PINGにて調査します。
閉じる(C)
№ 拠点名(NTTCOM) 拠点名(お客様) 回線番号 LAN側IPアドレス LAN側状況 WAN側IPアドレス WAN側状況
012 関西流通センター 中部流通センター A120010001 10.99.99.01 正常 10.254.99.01 正常
013 西日本営業所 西日本営業所 A120010002 10.99.99.02 正常 10.254.99.02 正常
014 上丸物流 上丸物流 A120010003 10.99.99.03 正常 10.254.99.03 正常
015 軽間急配 大阪センター A120010004 10.99.99.04 正常 10.254.99.04 正常
016 石丸倉庫 大阪センター A120010005 10.99.99.05 正常 10.254.99.05 異常
017 香川センター 香川センター A120010006 10.99.99.06 正常 10.254.99.06 正常
018 沖縄センター 沖縄センター A120010007 10.99.99.07 正常 10.254.99.07 正常
019 関西薬品返品倉庫 関西薬品返品倉庫 A120010008 10.99.99.08 正常 10.254.99.08 正常
020 関西加工センター 関西加工センター A120010009 10.99.99.09 正常 10.254.99.09 正常
021 石丸昭和物流 石丸昭和物流 A120010010 10.99.99.10 正常 10.254.99.10 正常
022 埼玉流通センター 埼玉流通センター A120010011 10.99.99.11 正常 10.254.99.11 正常
023 西濱倉庫 関東危険物管理倉庫 A120010012 10.99.99.12 正常 10.254.99.12 正常
024 広島データセンター Active A120010013 10.99.99.13 正常 10.254.99.13 正常
025 広島データセンター Standby A120010014 10.99.99.14 正常 10.254.99.14 正常
026 名古屋流通センター 名古屋流通センター A120010015 10.99.99.15 正常 10.254.99.15 正常
027 東日本陸運 新潟支店 東日本陸運 新潟支店 A120010016 10.99.99.16 正常 10.254.99.16 正常
028 千葉流通センター 千葉流通センター A120010017 10.99.99.17 異常 10.254.99.17 正常
029 千葉流通センター(A棟) 千葉流通センター(A棟) A120010018 10.99.99.18 正常 10.254.99.18 正常
030 福岡ロジスティクスセンター 福岡ロジスティクスセンター A120010019 10.99.99.19 正常 10.254.99.19 正常
031 大阪営業所 大阪営業所 A120010020 10.99.99.20 正常 10.254.99.20 正常
テストデータ
※ＲＢＪＬ７／ＢＩＭ０２Ｐに登録されている、拠点毎のＩＰアドレスを調査します。 【本社・東西の各区画毎にＢＩＭ０２Ｐをメンテナンスして下さい。】

優秀賞

汎用的な帳票出力画面

牛嶋 信之 様
株式会社佐賀鉄工所
管理部情報システム課
主事



株式会社佐賀鉄工所
http://www.satetsu.co.jp/

昭和 13 年創業。自動車用ボルトを専門領域とするリーディングカンパニーとして、日本は勿論、海外でも高い評価をている。業界でも数少ない「一貫生産方式」を採用。さらに業界屈指の開発・試験設備を保有し、世界の自動車産業を「小さなボルトで大きく」支え続けている。

業務課題

VB6 で作成した IBM i の GUI プログラムを Delphi/400 に移行するプロジェクトを実施した。

移行対象のプログラムは多数あり、その中には IBM i の元帳を印刷する機能を持つユーザー画面がいくつかあった。これらのプログラムの移行にあたり、次の課題を設定した。

課題 1

旧プログラムでは、帳票出力の選択条件は各端末内に保存していたので、PC の入れ替え時に条件ファイルを注意して移行する手間や、PC がクラッシュして条件ファイルを喪失するリスクがあった。単純な旧機能の移植でなく、この課題を解決したい。

課題 2

多くのプログラムを移行するため、可能な限りコーディングの工数を削減したい。

技術課題

課題 1 について

選択条件や出力順のパラメータは、各クライアント PC ではなく、IBM i のデータベースに保存する設計とした上で、以下の対応を行った。【図 1】

・選択条件や出力順はデータベースのパラメータを参照しセットするため、汎用的な表示が可能
・選択条件が複数あることを想定し、スクロールパネルにより、最大 20 個までの条件を指定可能
・条件設定で「と等しい、と等しくない、を含む、を含まない、から始まる、で終わる、以上、以下、より大きい、より小さい」を選択可能
・「かつ、または」で条件を組み合わせることも可能
・条件入力欄を「|」で区切って入力することで、入力値を OR 条件で使用することが可能（IBM i のクエリでいう LIST のような機能）
・出力順（昇順＆降順）の設定が可能

課題 2 について

画面内を共通で使用するコンポーネントを配置し、シンプルに作成している。

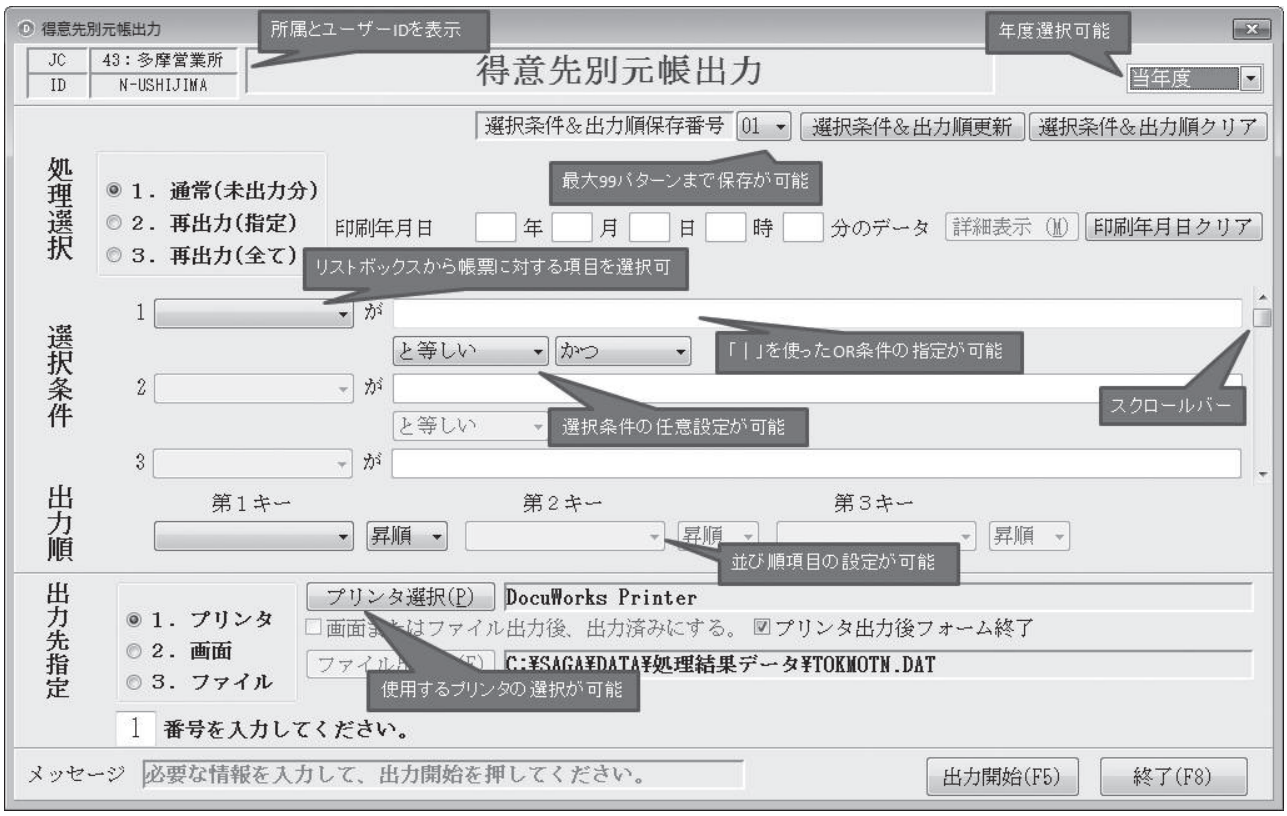
パラメータ用データベースを利用し、継承元に共通ロジックを埋め込み、継承先のコーディング量（個別ロジック）を減らすようにした。

画面による業務課題の解決

エンドユーザーが自分の好みの条件設定を維持しながら、各種帳票を印刷できるようになった。プログラム作成時の工数削減につながり、Delphi 移行への量産体制が整った。

M

図1


得意先別元帳出力
所属とユーザーIDを表示
年度選択可能
JC
43：多摩営業所
ID
N-USHIJIMA
得意先別元帳出力
当年度
選択条件＆出力順保存番号
01
選択条件＆出力順更新
選択条件＆出力順クリア
最大99パターンまで保存が可能
処理選択
１．通常(未出力分)
２．再出力(指定)
３．再出力(全て)
印刷年月日
年
月
日
時
分のデータ
詳細表示 (M)
印刷年月日クリア
リストボックスから帳票に対する項目を選択可
選択条件
が
と等しい
かつ
「｜」を使ったOR条件の指定が可能
スクロールバー
選択条件の任意設定が可能
出力順
第１キー
第２キー
第３キー
昇順
並び順項目の設定が可能
出力先指定
１．プリンタ
２．画面
３．ファイル
プリンタ選択(P)
DocuWorks Printer
画面またはファイル出力後、出力済みにする。
プリンタ出力後フォーム終了
C:¥SAGA¥DATA¥処理結果データ¥TOKMOTN.DAT
使用するプリンタの選択が可能
番号を入力してください。
メッセージ
必要な情報を入力して、出力開始を押してください。
出力開始(F5)
終了(F8)

優秀賞

バーコードリーダー読み取り後 次の入力位置にカーソルを自動遷移させる技術

上総 龍央 様

キョーラクシステムクリエート株式会社
開発部
課長

キョーラクシステムクリエート株式会社
http://www.kscnet.co.jp/

1998 年、キョーラク株式会社のシステム部門が分社し誕生。IBM i をコアとして、IBM i の販売ならびに業務アプリケーションの開発を行う。クライアント／サーバー型、Web 型などの開発にも積極的に取り組んでおり、現在は Delphi、Microsoft Visual Basic、.NET や Java などの開発も手がけている。

業務課題

タブレットを使用した SP4i の売上入力システムにおいて、以下の改善を行いたい。

1　商品コードのマスターチェック

バーコードリーダーで商品コードを読み取った時、商品コードを IBM i のマスターとチェックしたい。複数商品のすべての入力行を画面単位で一括チェックするのではなく、項目単位のチェックを行いたい。

2　カーソル遷移の制御

上記 1 を単純に実装する場合、バーコードリーダーの操作ごとに画面の先頭フィールドにフォーカスが移動してしまう。操作の利便性を考慮し、正しい商品コードの時は、次の入力位置である数量にカーソルを自動でセットしたい。また数量入力後、次行の商品コードにカーソルをセットする動作を繰り返したい。

画面機能の詳細説明

1　商品コードマスターチェックについて

バーコードリーダー側には、データ送信時に「実行キー」機能をプラス。バーコードで商品コードを読み取るたびに、RPG プログラムを呼び出して存在チェックを行うロジックを追加した。

2　カーソル遷移の制御について

① HTML には、hidden 要素でフォーカスが設定されている HD01 と行数を設定するための HD01L を追加
② JQuery でサブミット時に ID と行数をセット【ソース 1】
③ RPG では受け取った ID と行により、次のフォーカス位置を設定【ソース 2】

画面による業務課題の解決

1　商品コードマスターチェックについて

商品コードのチェックが即時に可能となったため、正しい商品コードが入力できるようになった。

2　カーソル遷移の制御について

カーソル位置の自動遷移化に伴い、カーソル位置を意識することなく、商品コードの読み取りと数量の入力が可能になったため、操作のスピードが上がり、効率化が実現された。【図 1】

M

ソース1

■JavaScriptソース

```
// フォーカス移動制御
//    Enter実行前に現カーソル位置(ID名＋行№)を取得
//

 (function($) {
  //フォーカス移動(明細)
  $.fn.dataMeisai = function() {
   var SP4iCustom2 = SP4iCustom2 || {} ;
   var $selector = this;
   //キー押下時
   this.keydown(function(e){
    if((e.keyCode >= 48 && e.keyCode <= 57) || (e.keyCode >= 96 && e.keyCode <= 105)){

     SP4i.getElementById('HD01').value = '';
     SP4i.getElementById('HD01L').value= '0';
     var e =event.target || event.srcElement;
    if(e){
     if(e.tagName.toLowerCase() == 'input' || e.tagName.toLowerCase() == 'select' ){
      SP4i.getElementById('HD01').value = 'I' + e.id;
      var table = e.parentNode.offsetParent;
      if( table.tagName.toLowerCase() == 'table'){
       var tblclass = $(this).attr('data-dataMeisai');
       if(table.className == tblclass ){
        if(e.parentNode.parentNode.tagName.toLowerCase() == 'tr' ){
         SP4i.getElementById('HD01L').value = e.parentNode.parentNode.rowIndex + 1;
        }
       }
      }
     }
    }
    }
   });

  //マウス押下時(明細)
   this.mousedown(function(e){
     SP4i.getElementById('HD01').value = '';
     SP4i.getElementById('HD01L').value= '0';
     var e =event.target || event.srcElement;
    if(e){
     if(e.tagName.toLowerCase() == 'input' || e.tagName.toLowerCase() == 'select'){
      SP4i.getElementById('HD01').value = 'I' + e.id;
      var table = e.parentNode.offsetParent;
      if( table.tagName.toLowerCase() == 'table'){
       var tblclass = $(this).attr('data-dataMeisai');
       if(table.className == tblclass ){
        if(e.parentNode.parentNode.tagName.toLowerCase() == 'tr' ){
         SP4i.getElementById('HD01L').value = e.parentNode.parentNode.rowIndex + 1;
        }
       }
      }
     }
    }
   });
  }
 })( jQuery );
```

入力が起きた要素のIDをHD01の値にセット

入力が起きたテーブル行の位置(インデックスは
ゼロ開始のためプラス1)をHD01Lの値にセット

ソース2

■RPGソース

```
＊商品
C           IF      IHD01  =  'IJ2SHC'
C           EVAL    SPCSRF   = 'IJ2SRS'
C           EVAL    SPCSRL   = IHD01L
C           ENDIF
C*
＊数量
C           IF      IHD01  =  'IJ2SRS'
C           EVAL    SPCSRF   = 'IJ2SHC'
C           EVAL    SPCSRL   = IHD01L + 1
C           ENDIF
```

数量に遷移

次の行の商品コードに遷移

図1

優秀賞

IBM i のスプールファイル参照機能をWebで構築

福島 利昭 様

株式会社ランドコンピュータ
代表取締役

株式会社ランドコンピュータ
https://www.rand.co.jp/

高校、大学等のコンピュータ教室で使われる「授業支援システム」の設計、開発、販売を行っており、学校等の教育関連施設に対して５千件以上の納入実績がある。業務ソリューションに必要なソフトウェア開発と、画像・音声処理などの機器製造をトータルで行っている。

業務課題

IBM i のスプールファイルには、印刷して内容を保管する場合と、画面上で確認するだけでよいものがある。

そこで使いやすいスプールファイル参照システムを構築し、以下の課題を解決したい。

・スプールファイルの一覧や内容が簡単に参照できる
・マウスを使って操作しやすい
・テキストのコピー＆ペーストが簡単にできる
・印刷が簡単にできる

技術課題

1　Delphi/400 で開発したいが、実行環境を構築したり、アプリケーションの配布が不要な Web ベースの仕組みとしたい

2　Delphi/400 機能を使って、スプールファイルの内容を確認する手法の確定が必要である

技術課題の解決策

1　IntraWeb を使ってシステムを設計する
2　今回は、CMD400 でスプールファイルを一時ファイルにコピーし、FILE400 で中身を確認する方法を採用した

業務課題解決と効果

・画面上でスプールファイルを手軽にプレビューできるようになり、印刷する機会を削減できた（紙、インクの節約）。
・マウス操作やコピー＆ペーストなどの操作が簡単になった。
・印刷機能も実装し、5250 画面を使用せずに確認作業が実行できるようになった。

M

ソース1　スプールリスト取得

■スプールのリストを取得する

```
// アウトキューを指定
ListSpool4001.OutQName := cmbPrinter.Items[cmbPrinter.ItemIndex];
ListSpool4001.AS400.Userid := 'qsecofr';
ListSpool4001.AS400.PWD := 'xxxxxx';
try
  ListSpool4001.Active := True;

  // 先頭レコード
  ListSpool4001.First;

  // スプールリストの格納
  while not ListSpool4001.Eof do
  begin
    with tmpSPLInfo do
    begin
      Name := Trim(ListSpool4001.FieldByName('Name').AsString);
      UserName := Trim(ListSpool4001.FieldByName('UserName').AsString);
      DataName := Trim(ListSpool4001.Fields[2].AsString);
      PageCount := ListSpool4001.Fields[3].AsInteger;
      CreateD := StrToStrDate(ListSpool4001.Fields[8].Value);
      CreateT := StrToStrTime(ListSpool4001.Fields[9].Value);
      SPLFileNumber := ListSpool4001.FieldByName('SpoolFileNumber').AsInteger;
      JobNumber := Trim(ListSpool4001.FieldByName('JobNumber').AsString);
      JobName := Trim(ListSpool4001.FieldByName('JobName').AsString);
    end;
    SPLList.Add(tmpSPLInfo);

    ListSpool4001.Next;
  end;
```

ソース2　FILE400で開く

■　スプールファイルを一時ファイルにコピーしてFILE400で開く

```
  Cmd4001.CommandLine.Clear;

  // ファイル名、ジョブ番号、ユーザー名、ジョブ名、スプール番号
  SS := Format('CPYSPLF FILE(%s) TOFILE(OAWORK/FL200) JOB(%s/%s/%s)
SPLNBR(%s)',
    [
      IWGrid2.Cell[RowIndex, 0].Text,
      IWGrid2.Cell[RowIndex, 7].Text,
      IWGrid2.Cell[RowIndex, 1].Text,
      IWGrid2.Cell[RowIndex, 8].Text,
      IWGrid2.Cell[RowIndex, 6].Text
    ]);

  // OAWORKのFL200というファイルにスプールをコピー
  Cmd4001.CommandLine.Add(SS);
  Cmd4001.Execute;

  // コピーしたファイルを開き、メモに表示
  File4002.LibraryName := 'OAWORK';
  File4002.FileMember := 'FL200';
  File4002.FileName := 'FL200';
  File4002.FileDescription := 'FL200';
  try
    File4002.Active := True;
    File4002.GetFirst;
    for ii := 0 to File4002.RecordCount - 1 do
    begin
      IWMemo1.Lines.Add(File4002.GetValue(1));
      File4002.GetNext;
    end;
  finally
    File4002.Active := False;
  end;
```

ソース3　テキストファイルダウンロード

■ プレビューで表示したメモをテキストファイルとしてダウンロード

```
  // CSVの作成
  lvslCSV := TStringList.Create;
  try
    // 内容を書き込む
    lvslCSV.Text := IWMemo1.Text;

    // メモリーストリームの作成
    lvmemCSV := TMemoryStream.Create;
    lvmemCSV.Clear;
    // メモリーストリームへの保存
    lvslCSV.SaveToStream(lvmemCSV);

    // ダウンロード
    WebApplication.SendStream(lvmemCSV, True, '', 'SPL' +
FormatDateTime('YYYYMMDDHHNNSS', Now) + '.txt');
  finally
    lvslCSV.Free;
  end;
```

図1　画面設計

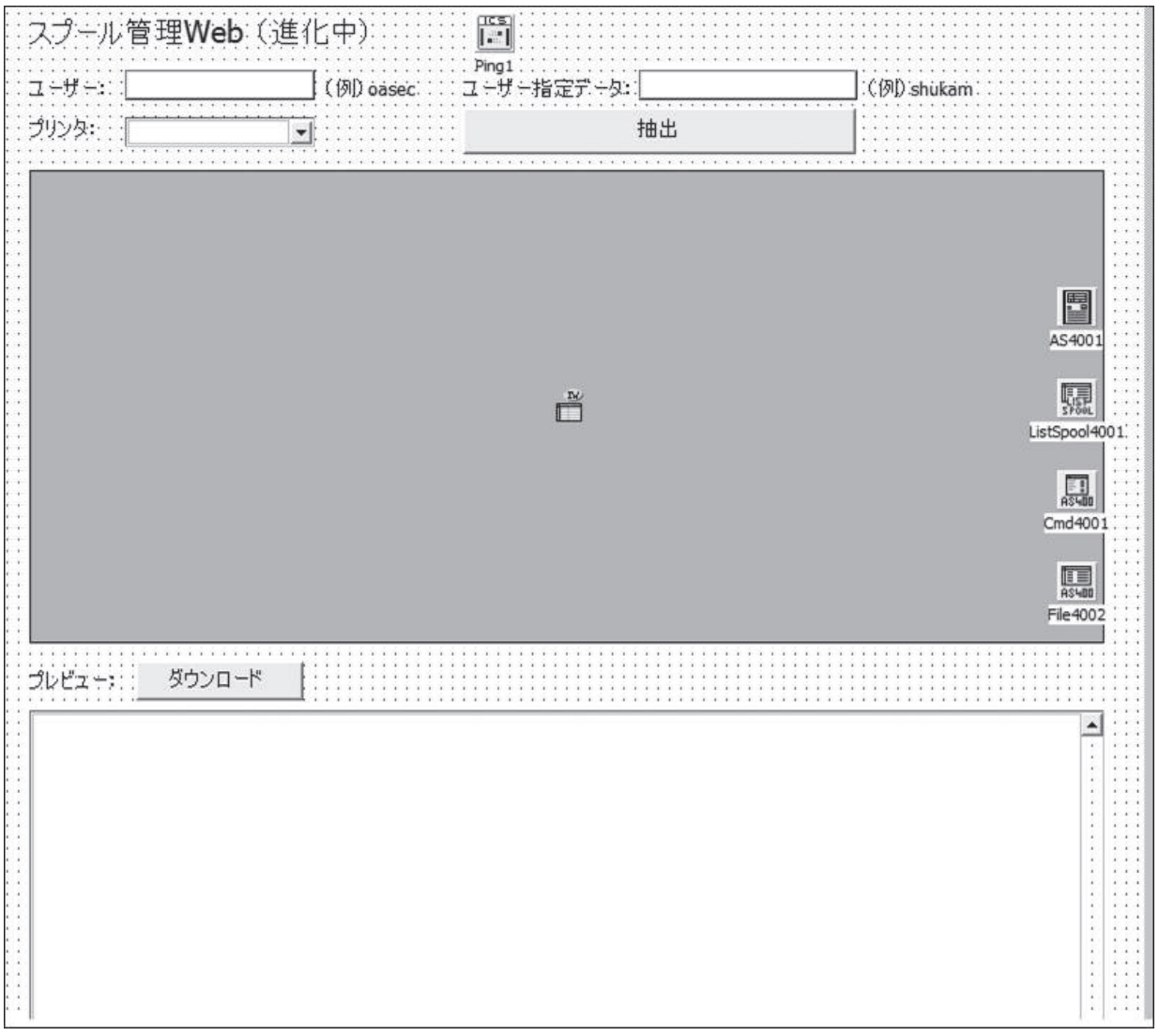


図2　実行画面


スプール管理Web（進化中）
ユーザー:
(例)oasec
ユーザー指定データ:
(例)shukam
プリンタ:
P1
抽出
スプール一覧
ファイル名 ユーザー名 ユーザー指定データ ページ数 作成日 作成時刻 スプール番号 ジョブ番号 ジョブ名 プレビュー
LIST KANRI30 KES024 1 2017/08/01 10:05:49 11 433345 H2 プレビュー
LST400 KANRI30 YH2015 2 2017/08/01 10:02:55 9 433345 H2 プレビュー
WK KANRI30 CF031A 1 2017/08/01 09:59:50 7 433345 H2 プレビュー
LIST KANRI30 KEI06U 1 2017/08/01 09:55:11 6 433345 H2 プレビュー
LST400 KANRI30 ES0514 3 2017/08/01 09:53:16 5 433345 H2 プレビュー
LST400 KANRI30 ES0514 3 2017/08/01 09:51:47 4 433345 H2 プレビュー
LST400 KANRI30 ES0514 3 2017/08/01 09:50:56 3 433345 H2 プレビュー
LIST KANRI30 KISIRE.R 2 2017/08/01 09:48:09 1 433345 H2 プレビュー
LIST OASEC SM0144 1 2017/08/01 09:42:29 2 433340 SS014ZCL プレビュー
LST400 OASEC KW0010 1 2017/08/01 09:25:33 1 433314 KW0003CL プレビュー
LIST OASEC NS0044 6 2017/08/01 09:25:30 1 433317 NS0044 プレビュー
LIST OASEC GYSHIKMEI 2 2017/08/01 00:00:41 57 433098 H0 プレビュー
LIST KANRI30 KES024 1 2017/07/03 10:34:26 11 428260 H2 プレビュー
プレビュー:
ダウンロード
[17/07/01 - 17/07/31] 入金実績リスト (CF031A) 17/08/01 P. 1
支払条件 顧客№ 顧客名 入金額 入金日
0551231530 173,340 7/21
0427071881 126,360 7/31
0326471110 206,712 7/20
0534552311 583,848 7/31
0529713011 97,200 7/25
0648076371 5,294,700 7/31

Migaro.
Technical Report
2017

ミガロ.SE 論文／ミガロ. テクニカルレポート

尾崎 浩司

株式会社ミガロ.

RAD事業部 営業・営業推進課

[Delphi/400] デスクトップアプリケーション開発でも役立つFireMonkey活用入門

略歴
1973年8月16日生まれ
1996年 三重大学工学部卒業
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
2013年4月 RAD事業部配属

現在の仕事内容
ミガロ.製品の営業を担当している。これまでのシステム開発経験を活かして、IBM iをご利用のお客様に対して、GUI化、Web化、モバイル化などをご提案している。

1.はじめに

Delphi/400のバージョンXE3以降では、VCL（Visual Component Library）フレームワークとは別に、FireMonkeyという新しいフレームワークが搭載されるようになった。

Delphi/400は、VCLを使用したWindowsクライアント上で実行できるアプリケーション開発に長年利用されているが、この新しいFireMonkeyフレームワークを使用すれば、iOSやAndroid用のモバイル開発も可能である。

そのためFireMonkeyはモバイル開発専用として使われることが多いが、実はWindowsやMacといったデスクトップOS向けの開発にも対応しており、iOS/Androidと合わせて合計4つのプラットフォームに対応するマルチデバイス開発機能を持っている。

本稿では、これまで主にVCLを使用してアプリケーションを開発している方を対象に、Windowsアプリケーション開発で役立つFireMonkeyのポイントを解説する。

2.FireMonkeyとは

2-1 FireMonkeyの概要

FireMonkeyを説明する前に、まず従来のVCLの特徴を確認する。VCLは、Windowsアプリケーション開発専用のフレームワークであり、Windows APIをラッピングしたものである。【図1】

開発者は、VCLコンポーネントとRTL（ランタイムライブラリ）と呼ばれるモジュールを使用して開発する。VCLコンポーネントは、Windows APIをラッピングしているので、Windowsに用意されたすべての機能を活用できる。

これに対してFireMonkeyフレームワークは、マルチデバイス向けの開発を目的としたフレームワークである。VCLのようにWindowsに機能特化していないものの、複数のOS向けに汎用的なアプリケーションを開発できる。

またさまざまなOS上で表示可能にするため、グラフィック処理装置（GPU）を使用している。Windowsの場合はDirectX、Macの場合はOpenGL、iOSやAndoridの場合はOpenGL ESといったGPU用APIをラッピングしており、各OSで共通的な機能を活用できる。【図2】

各OSに用意されたすべての機能を標準で活用できるわけではないが、特定のOSに特化しない仕組みが、マルチデバイス対応を実現する根幹となっている。

FireMonkeyの場合、GPUを活用するフレームワークなので、とくに3Dグラフィック処理が優れている。また2Dでも、VCLでは表現できないようなビジュアルやアニメーションなどの表示も可能である。なお、VCLと同様にRTLを使えるので、たとえばIntToStr関数等Delphi/400で利用している関数や手続きは、VCLと同じように使用できる。

2-2 FireMonkeyに最適なアプリケーション

PCアプリケーション開発での、従来

図1

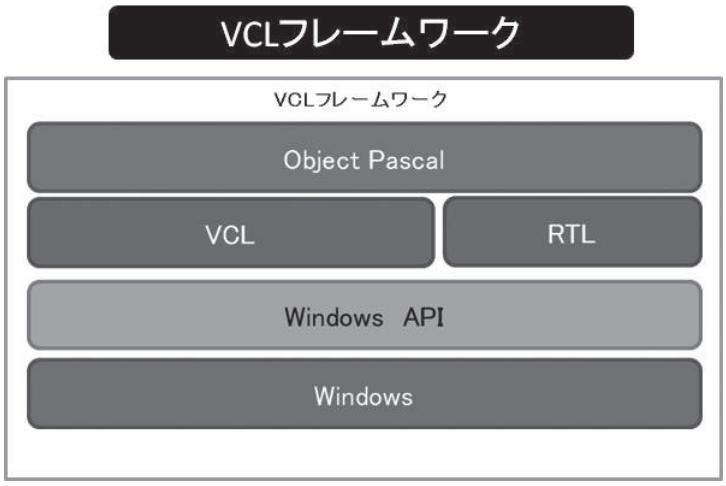

VCLフレームワーク
VCLフレームワーク
Object Pascal
VCL
RTL
Windows API
Windows

図2

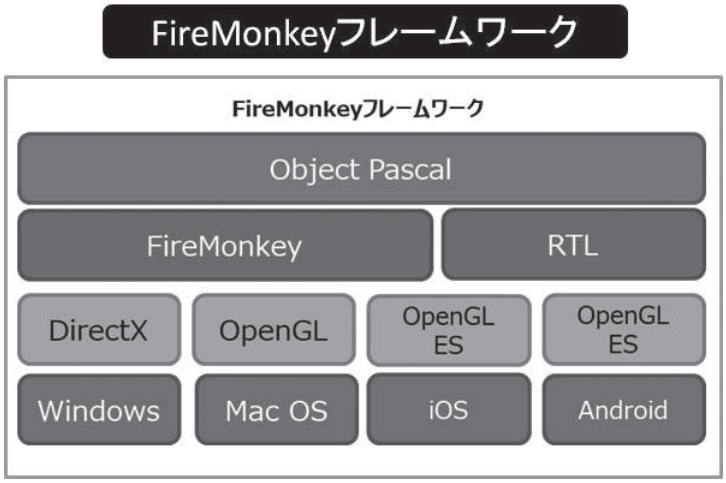
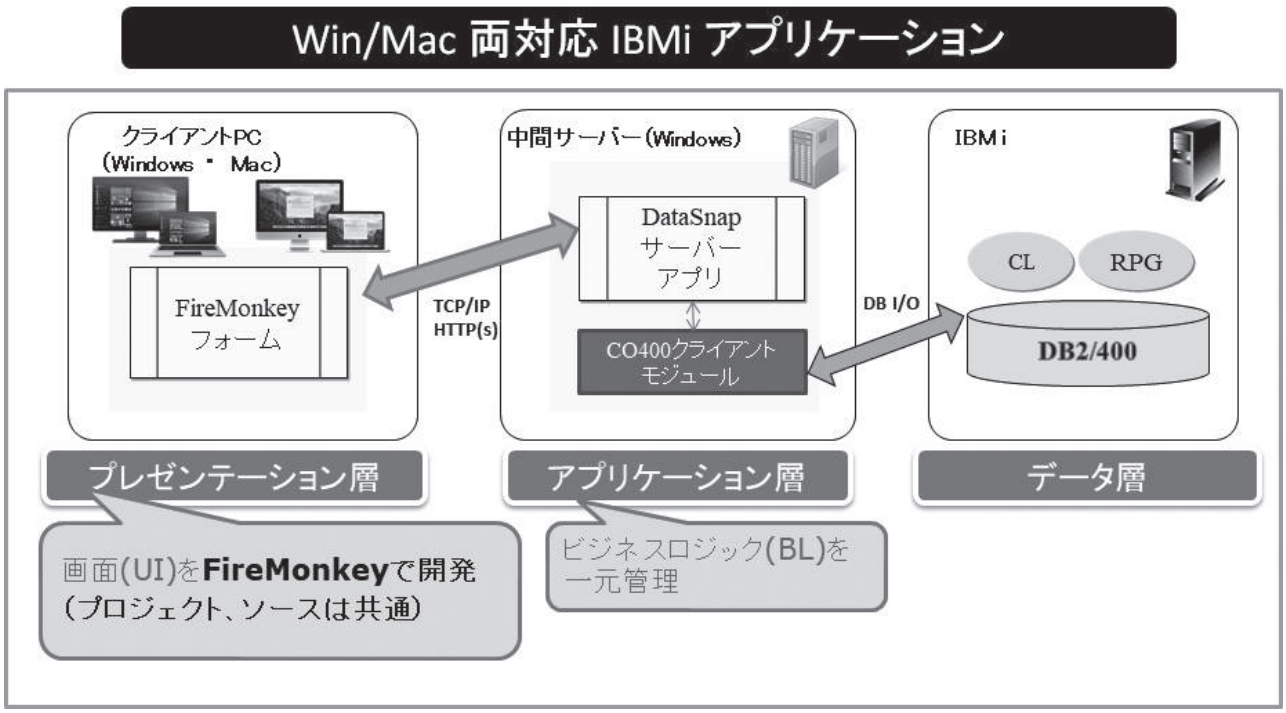

FireMonkeyフレームワーク
FireMonkeyフレームワーク
Object Pascal
FireMonkey
RTL
DirectX
OpenGL
OpenGL ES
OpenGL ES
Windows
Mac OS
iOS
Android

図3


Win/Mac 両対応 IBMi アプリケーション
クライアントPC
(Windows ・ Mac)
FireMonkey
フォーム
TCP/IP
HTTP(s)
中間サーバー(Windows)
DataSnap
サーバー
アプリ
CO400クライアント
モジュール
DB I/O
IBMi
CL
RPG
DB2/400
プレゼンテーション層
アプリケーション層
データ層
画面(UI)をFireMonkeyで開発
(プロジェクト、ソースは共通)
ビジネスロジック(BL)を
一元管理

のVCLとFireMonkeyとの使い分けを考えてみる。もしクライアントPCの対象がWindowsだけでなく、Macが含まれるのであれば、FireMonkeyが必要となる。ただしdbExpressやFireDACといったデータベースエンジンのIBM i用ドライバーは、Windows専用となるので、中間サーバー（DataSnapサーバー）を使用した3階層形式での実装となる。【図3】

もちろんクライアントがWindowsだけであれば、FireMonkeyでもVCLと同様のC/S形式で開発できる。

FireMonkeyが最適なアプリケーションとしては、キーボードを持たないタブレットPC用アプリケーションや、画像等のビューア系アプリケーション、デジタル・サイネージに代表される情報系アプリケーションなどが挙げられる。【図4】

これらのアプリケーションは、キーボードでの情報入力ではなく、情報の参照がメインとなる。こうしたアプリケーションの開発は、視覚的な効果が活かせるグラフィック処理に長けたFireMonkeyが最適といえる。

3. VCLとFireMonkeyの違い

3-1　コンポーネントの違い

FireMonkeyのプログラム開発手順はVCL同様、次の4つの手順で実施する。

①プロジェクトの新規作成
②フォームにコンポーネントを配置
③配置したコンポーネントにプロパティを定義
④イベントハンドラにロジック作成

言語としてはどちらもObjectPascalを使用しているので、Delphi/400開発者はVCL同様のプログラミングスキルで開発できる。

VCLとFireMonkeyとの違いの1つは、使用するコンポーネントの種類が異なる点である。FireMonkeyには、VCLと同じ名前のコンポーネントも多数あるが、完全に同じではなく、設定や使用方法が若干異なる。以下に代表的な違いを挙げる。

まずコンポーネント全般の違いとして、VCLでは表示文字列をCaption、入力文字列をTextプロパティで扱うが、FireMonkeyではどちらもTextプロパティで扱う。またコンポーネントの位置を指定するLeft、Topプロパティは、FireMonkeyではPosition.X,Yプロパティに相当する。

このように同じ機能でも、プロパティの異なる場合があるので注意が必要である。【図5】

次に、画面作成に多用するTEditについて詳しく比較してみる。これもいくつかの機能や使用方法が異なっている。

【図6】にいくつかの違いを挙げているが、たとえばVCLにあるCharCaseプロパティは、FireMonkeyで相当するプロパティが存在しないので、個々の文字が入力された時に発生するイベント（OnChangeTraking）でロジックを記述する必要がある。

またVCLにおけるColorプロパティは、R（赤）、G（緑）、B（青）の3原色で表されるが、FireMonkeyのColorプロパティは、RGBに加え、A（透過度）という要素を持つので、半透明な色も表現できる。

最後に、Fontプロパティの違いを説明する。VCLの場合はフォントのサイズがポイント単位（1ポイントが1/72インチ）であるが、FireMonkeyの場合はデバイス非依存ピクセル（DIP）単位（1DIPが1/96インチ）である。したがって、同じFont.Size値を持つ場合、FireMonkeyのほうが文字が小さくなる。

3-2　FireMonkey独自のコンポーネント機能

次に、FireMonkey独自の機能について説明する。VCLにもコンポーネントの親子関係があるが、フォーム以外で親となりえるコンポーネントは、TPanelやTGroupBoxなどのコンテナコンポーネントに限定されている。

しかしFireMonkeyの場合は、あらゆるコンポーネントを親子関係にできる。たとえば、TButtonの子としてTImageを使用することで、TBitBtnのような機能を実現したり、TEditの子にTLabelを使用することで、TLabeledEditのような機能を実現できる。【図7】

このようにコンポーネントの親子関係を使用して独自の組み合わせが可能なので、よく使用する組み合わせはコンポーネントテンプレートに登録しておけば、ツールパレットからいつでも利用できる。

またFireMonkeyのコンポーネントはすべてグラフィック描画により実現しているので、表示のカスタマイズ機能に長けている。FireMonkey独自のRotationAngleプロパティは、コンポーネントを表示する角度を自由に指定できる。【図8】

データベースアプリ開発では、VCLの場合、TDBEditやTDBGridなどデータベース連結コンポーネントを使用することが多い。しかし、FireMonkeyにはデータベース連結コンポーネントが存在しない。そのため、TEditやTStringGridなどを使用することになる。

ただしDelphi/400には、LiveBindingという仕組みが用意されているので、この機能を使用するとデータベース連結コンポーネントと同様の実装が可能である。【図9】

4. FireMonkey画面作成のポイント

4-1　フォームレイアウトとコンポーネントの配置

VCLでフォームレイアウトを作成する場合、TForm上に直接コンポーネントを配置して作成する。この時に配置されたコンポーネントは、Left,Topプロパティにより固定位置に配置される。もちろんFireMonkeyでも、Position.X,Yプロパティを同様に定義することで、同じような固定配置による作成が可能である。

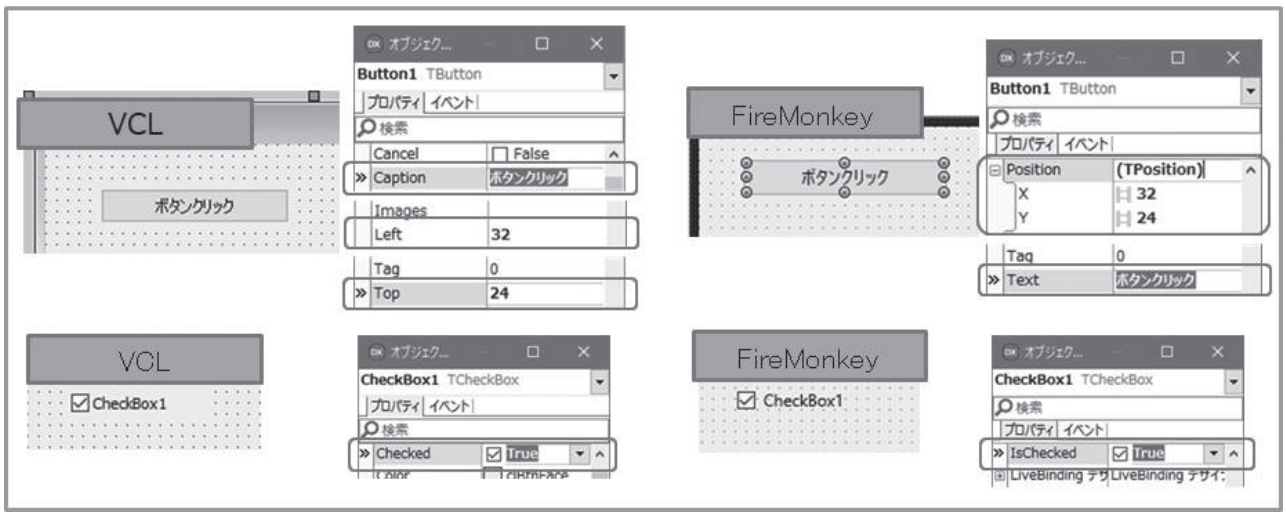
しかし従来の解像度を基準に作成した固定のフォームレイアウトは、近年の高解像度ディスプレイやMacなどで使用されるRetinaディスプレイ上では見えづらいことも多い。【図10】

ビジュアル機能に優れたFireMonkeyでは、LayoutsカテゴリにあるTLayoutというレイアウトコンポーネントを使用することで、解像度によって画面を自動調整するようなアプリケーションも開発できる。【図11】

図4


FireMonkeyに最適なアプリケーション例
タブレットPC用アプリ
画像ビューアソフト
空港のフライト情報

図5


VCLとFireMonkeyのプロパティの違い
VCL
ボタンクリック
Button1 TButton
プロパティ イベント
検索
Cancel False
Caption ボタンクリック
Images
Left 32
Tag 0
Top 24
FireMonkey
ボタンクリック
Button1 TButton
Position (TPosition)
X 32
Y 24
Tag 0
Text ボタンクリック
VCL
CheckBox1
CheckBox1 TCheckBox
Checked True
FireMonkey
CheckBox1
CheckBox1 TCheckBox
IsChecked True

TLayoutは、TPanelのようなコンテナコンポーネントであるが、TLayout自体は表示されない。またTLayout自体のVisibleを変更すると、配下のコンポーネントの表示／非表示が一括設定できる点も便利である。

レイアウトコンポーネントで一番わかりやすいのが、TScaledLayoutである。TScaledLayoutは縮小／拡大が可能なフォームレイアウトで、この配下に配置したコンポーネントは、フォームのサイズ変更に合わせて縮尺が自動的に調整される。

TScaledLayoutを使用すると、固定配置したコンポーネントの場合でも、高解像度ディスプレイの対応が可能となる。【図12】

次に、TFlowLayoutを説明する。これは文章の段落内の単語と同じように、子コンポーネントを整列させるフォームレイアウトである。

左端から順に貼り付けたコンポーネントは、フォームの幅を超えると自動的に改行される。またフォームの幅が変更された場合、その変更に応じて改行位置が自動的に調整されるのが特徴である。【図13】

なお文章の場合、段落ごとに改行を入れるが、それと同じように、並べられたコンポーネントの途中で改行するのが、TFlowLayoutBreakである。

配置したコンポーネントのうしろにTFlowLayoutBreakを挿入すると、任意の位置でコンポーネントを改行できる。

このTFlowLayoutは、たとえばタブレット用アプリケーションで、縦画面と横画面を切り替えた時に項目の配置を自動調整する場合に便利である。

3つ目に説明するのは、TGridLayoutである。これは等間隔のセルにコンポーネントを配置するようなフォームレイアウトである。

1つのセルの高さと幅をItemHeight, ItemWidthプロパティで設定し、その中で順番にコンポーネントを配置して利用する。碁盤目のようなフォームレイアウトを作る時に重宝する。【図14】

最後は、TGridPanalLayoutである。これは、各コンポーネントがグリッドパネル上のセル内に配置されるフォームレイアウトである。

グリッドの列情報を保持するColumnCollection、行情報を保持するRowCollection、グリッドに配置されるコンポーネント情報を保持するControlCollectionを使用してフォームレイアウトを作成する。エクセルのように列ごとの幅や行ごとの高さを指定して、セル結合などを行いながら表を調整できる。【図15】

フォームレイアウトが決まったら、次はその中に配置する各コンポーネントを検討する。コンポーネントを固定配置する場合、位置をPosition.X,Yプロパティで指定し、大きさをWidth,Heightプロパティで指定する。

この場合、フォームの大きさが変わっても、コンポーネントの位置やサイズは変わらない。フォームの大きさが変わった時に、コンポーネントの配置を調整するには、配置を定義するプロパティを使用すればよい。具体的には、Alignプロパティ、Marginsプロパティ、Paddingプロパティの3つである。

Alignプロパティは、親コンポーネントに対し整列するためのもので、VCLにも同名のプロパティが存在する。FireMonkeyのAlignプロパティも同様の機能だが、VCLよりも多くのオプション値が用意されている。【図16】

またMarginsプロパティはコンポーネント間の余白をピクセル単位で指定し、Paddingプロパティは親コンポーネントから子コンポーネントまでの距離をピクセル単位で指定する。【図17】

これらを設定すると、画面比率の異なる環境で実行した時にも、コンポーネントの表示が最適に調整できる。

4-2　Effectとアニメーション効果

FireMonkeyはグラフィック処理に長けたフレームワークであり、画面の表示効果にさまざまな機能が用意されている。その代表が、Effect効果とアニメーション効果である。以下に、それぞれのポイントを説明する。

Effect効果とは、コンポーネントに対する画像効果のことである。たとえば、TEditの子にTGlowEffectを配置すると、入力欄にグロー（発光）効果を適用できる。【図18】

このエフェクト効果だが、効果を有効にする条件（トリガー）も指定できる。Effectに用意されたTriggerプロパティを設定すればよい。【図19】

TGlowEffect以外にも、影の効果を生むTShadowEffectや、反射効果を生むTReflectionEffect、波模様効果を生むTWaveEffectなど多彩な効果がある。

次にアニメーション効果だが、これはプロパティの値を連続的に変化させる仕組みのことである。時間の経過に合わせてプロパティの値を変化させることで、動きをつけられる。

アニメーション効果は、ソースコードを使用して任意のタイミングで開始／終了したり、Effectと同様、トリガーにより実行できる。

アニメーション効果の代表的なコンポーネントは、TFloatAnimationである。これは数値型プロパティに関連付けることで、数値を連続的に変化させられるコンポーネントである。【図20】

シンプルなアニメーション効果の例として、フォーム上に貼り付けたコンポーネントの移動を説明する。

フォーム上にボタンを配置し、ボタンのPositon.Xプロパティに対し、「TFloatAnimationの新規作成」を選択する。するとFloatAnimationに対するプロパティ設定ができるので、StartFromCurrentプロパティをTrueに、StopValueプロパティを300に、Durationプロパティを2に、最後にTriggerプロパティを”IsPress = True”に設定する。

プログラムを実行してボタンをクリックすると、ボタン自体が右側にアニメーションで移動できる。【図21】

このようにいろいろなプロパティ値の操作で、プログラムからアニメーション効果を実現できる。

Effect効果とアニメーション効果をそれぞれ説明したが、もちろんこれらの組み合わせも可能である。最後に、組み合わせた例を説明する。

発光効果であるTGrowEffectの発光度合いは、Softnessプロパティで設定できるが、このSoftnessプロパティに対し、TFloatAnimationを設定できる。【図22】

ボタンをクリックした時、エラーチェックとして項目がブランクだったら発光するアニメーションを実行するよう

図6

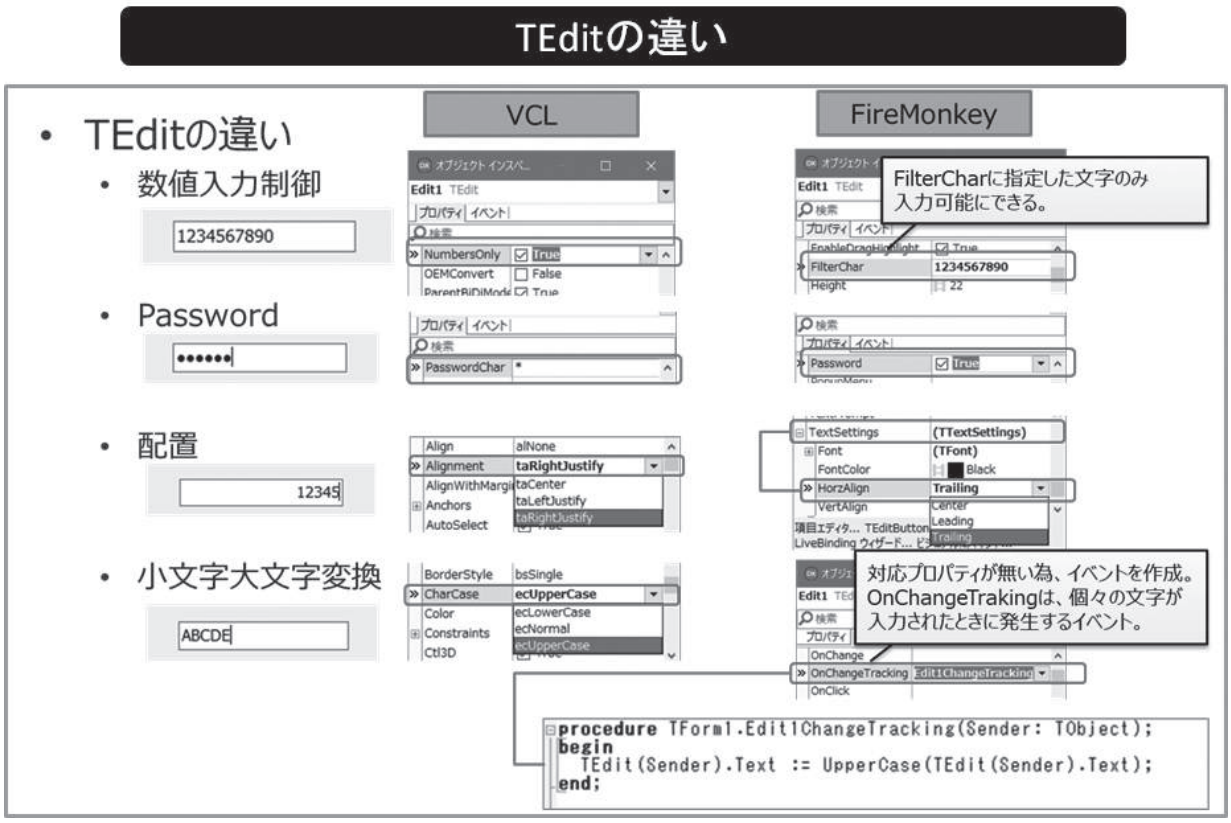

TEditの違い
TEditの違い
数値入力制御
1234567890
Password
配置
12345
小文字大文字変換
ABCDE
VCL
FireMonkey
FilterCharに指定した文字のみ
入力可能にできる。
対応プロパティが無い為、イベントを作成。
OnChangeTrakingは、個々の文字が
入力されたときに発生するイベント。
procedure TForm1.Edit1ChangeTracking(Sender: TObject);
begin
TEdit(Sender).Text := UpperCase(TEdit(Sender).Text);
end;

図7

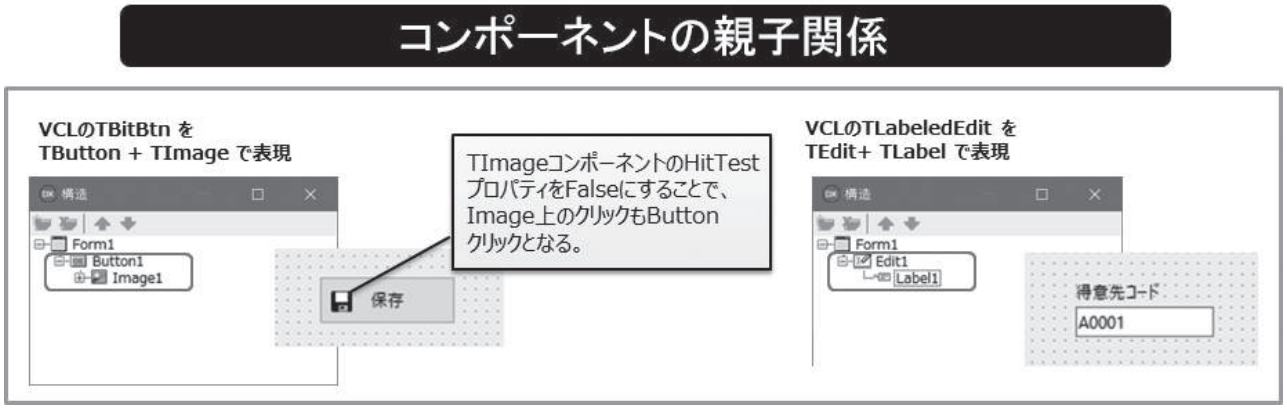
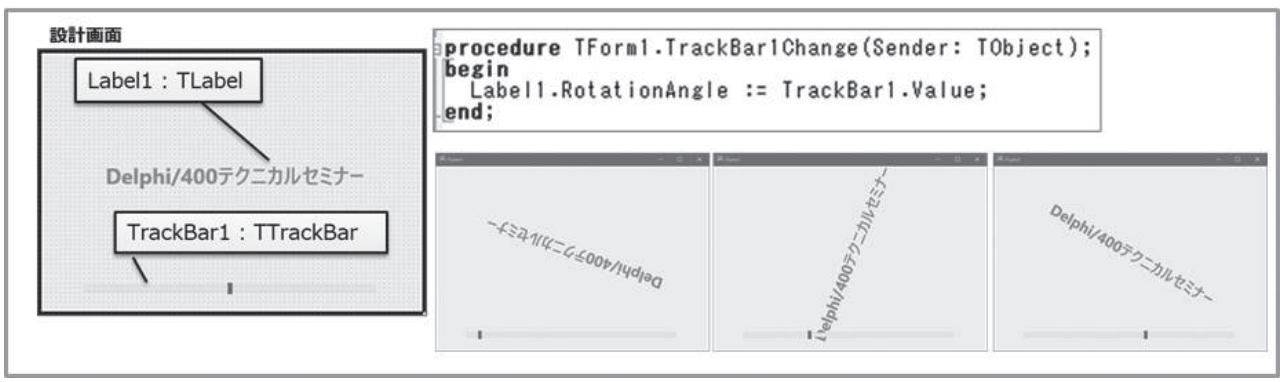

コンポーネントの親子関係
VCLのTBitBtn を
TButton + TImage で表現
TImageコンポーネントのHitTest
プロパティをFalseにすることで、
Image上のクリックもButton
クリックとなる。
保存
VCLのTLabeledEdit を
TEdit+ TLabel で表現
得意先コード
A0001

図8


コンポーネントの回転
設計画面
Label1 : TLabel
Delphi/400テクニカルセミナー
TrackBar1 : TTrackBar
procedure TForm1.TrackBar1Change(Sender: TObject);
begin
Label1.RotationAngle := TrackBar1.Value;
end;

な場合は、【ソース 1】で実現できる。たったこれだけの処理で、エラー発生時にエラー項目に対してのブリンク（点滅）処理を実現できる。

5.おわりに

本稿では、PC アプリケーション開発に役立つ FireMonkey フレームワークの基本を説明してきた。Windows 機能をフル活用したい場合には、VCL 開発が向いているが、視覚的効果が必要なビジュアルアプリケーションを開発する場合は、FireMonkey が適切である。

目的や用途によって VCL と FireMonkey を使い分けることで、これまで以上に Delphi/400 で開発できるアプリケーションの幅を広げられるはずである。

M

図9

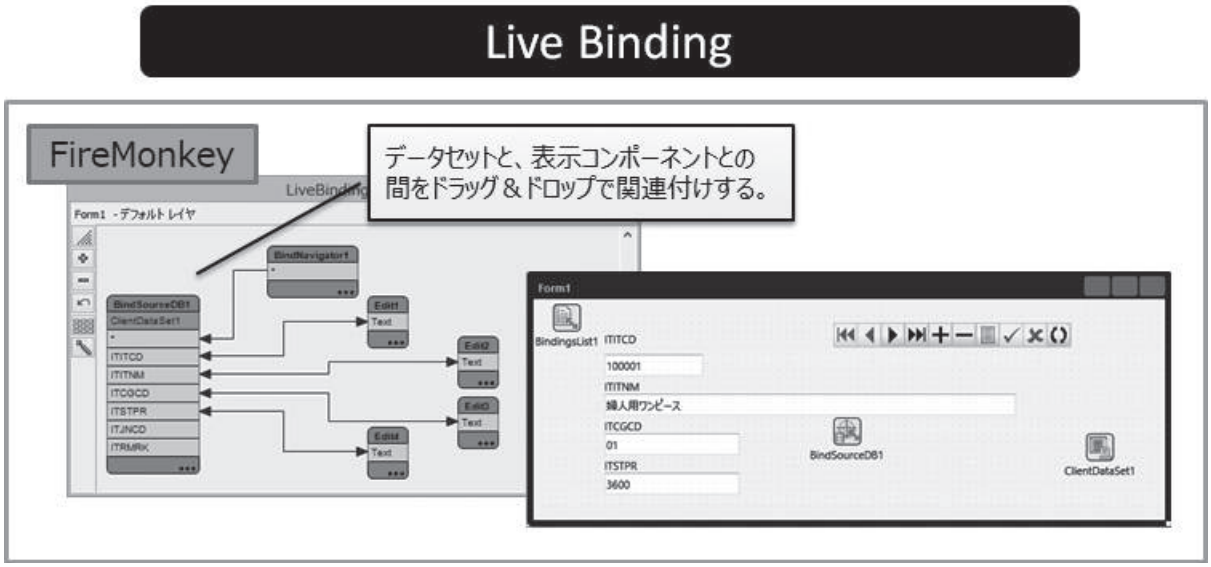

Live Binding
FireMonkey
データセットと、表示コンポーネントとの
間をドラッグ＆ドロップで関連付けする。
Form1
BindingsList1
ITITCD
100001
ITITNM
婦人用ワンピース
ITCGCD
01
ITSTPR
3600
BindSourceDB1
ClientDataSet1

図10

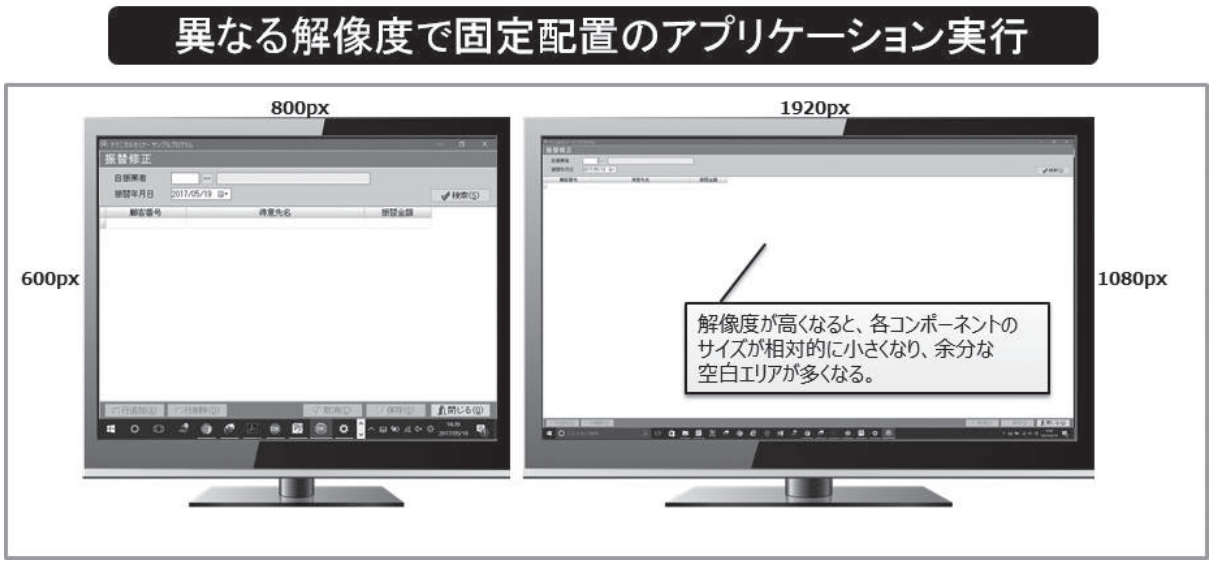

異なる解像度で固定配置のアプリケーション実行
800px
1920px
600px
1080px
解像度が高くなると、各コンポーネントの
サイズが相対的に小さくなり、余分な
空白エリアが多くなる。

図11


Layoutsカテゴリ
ツール パレット
検索
Layouts
TLayout
TScaledLayout
TGridLayout
TGridPanelLayout
TFlowLayoutBreak
TFlowLayout
TScrollBox
TVertScrollBox

図12

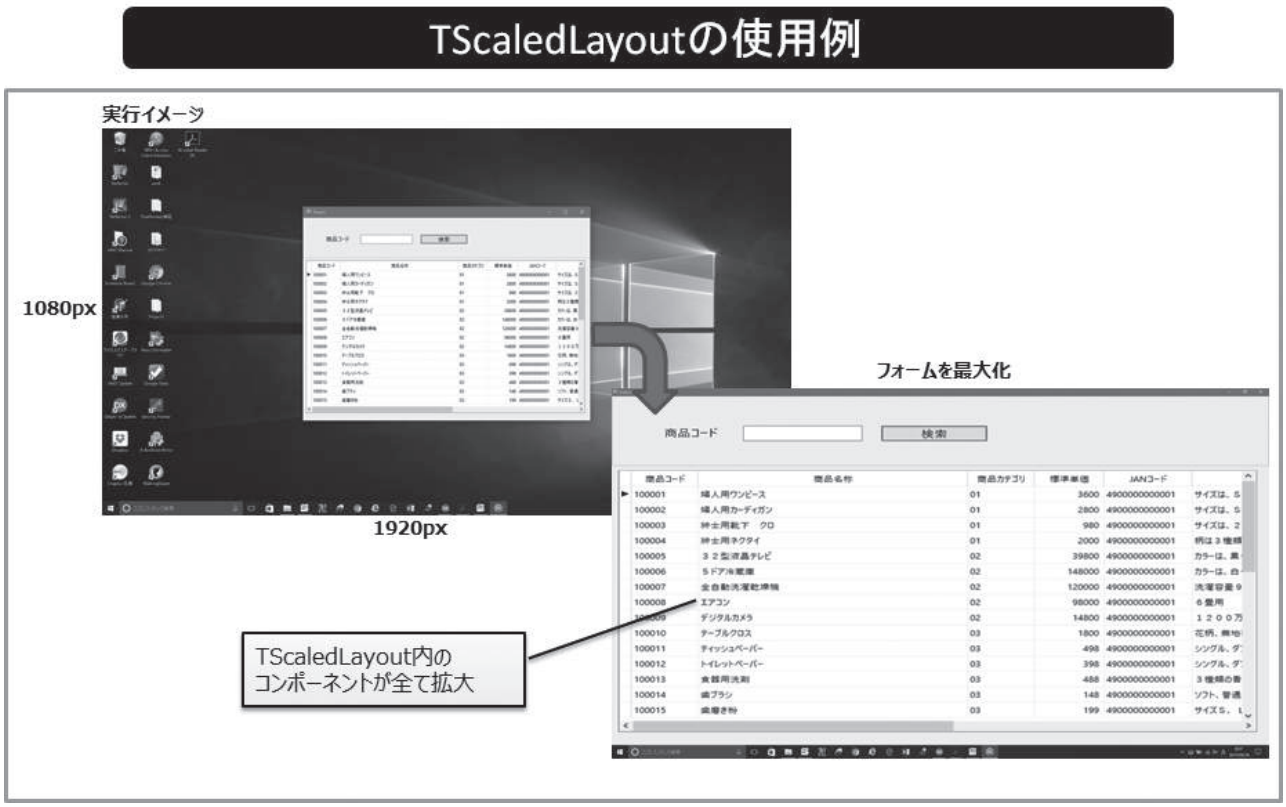

TScaledLayoutの使用例
実行イメージ
1080px
1920px
フォームを最大化
TScaledLayout内の
コンポーネントが全て拡大

図13

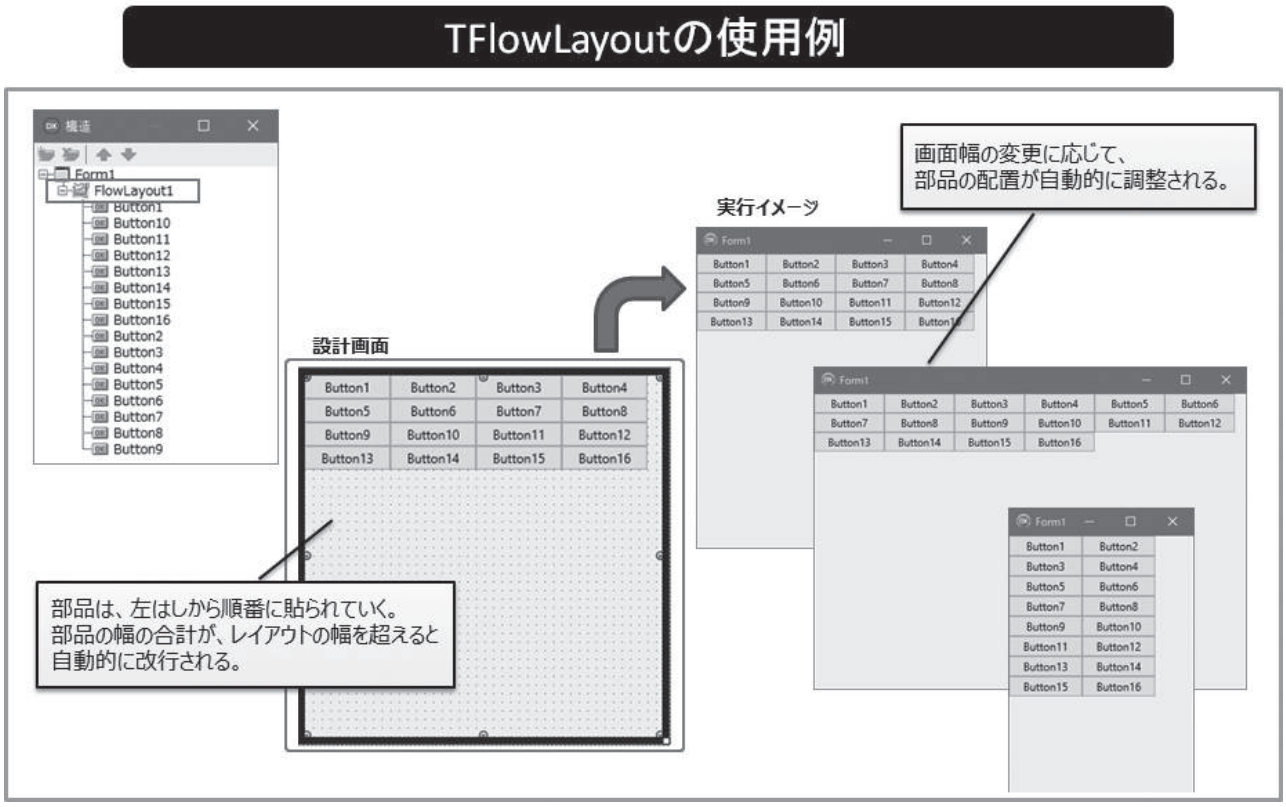

TFlowLayoutの使用例
Form1
FlowLayout1
Button1
Button10
Button11
Button12
Button13
Button14
Button15
Button16
Button2
Button3
Button4
Button5
Button6
Button7
Button8
Button9
設計画面
実行イメージ
画面幅の変更に応じて、
部品の配置が自動的に調整される。
部品は、左はしから順番に貼られていく。
部品の幅の合計が、レイアウトの幅を超えると
自動的に改行される。

図14

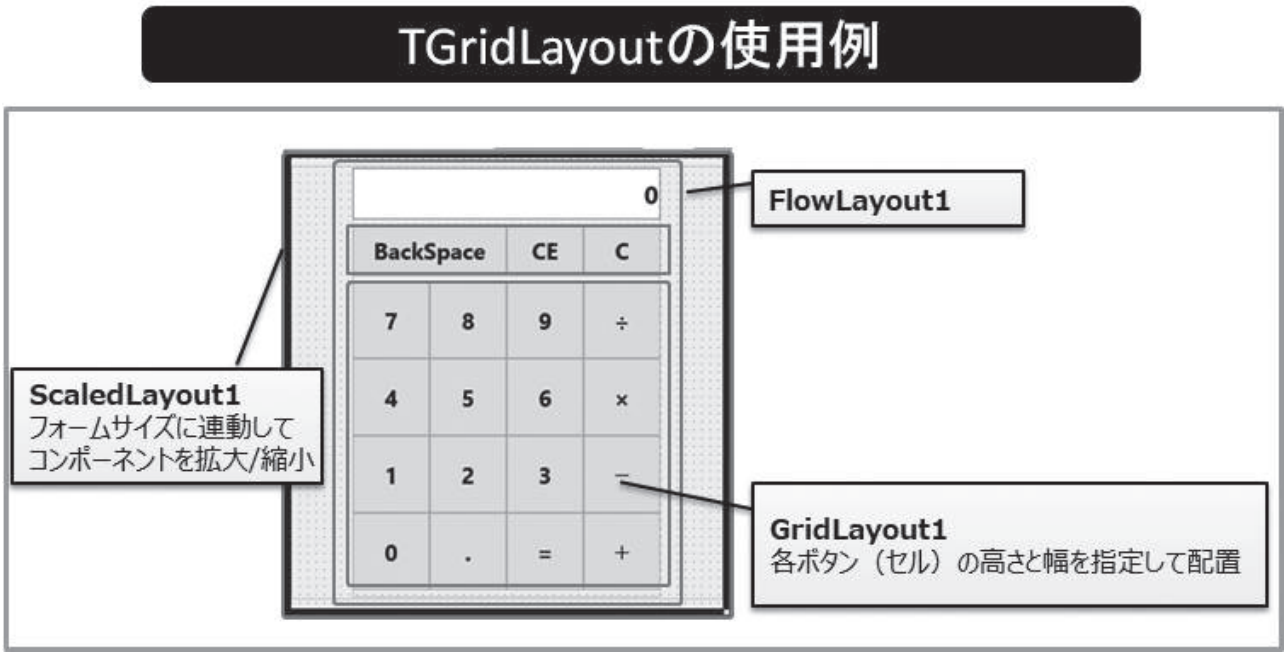

TGridLayoutの使用例
0
BackSpace
CE
C
7
8
9
÷
4
5
6
×
1
2
3
−
0
.
=
+
FlowLayout1
ScaledLayout1
フォームサイズに連動して
コンポーネントを拡大/縮小
GridLayout1
各ボタン（セル）の高さと幅を指定して配置

図15

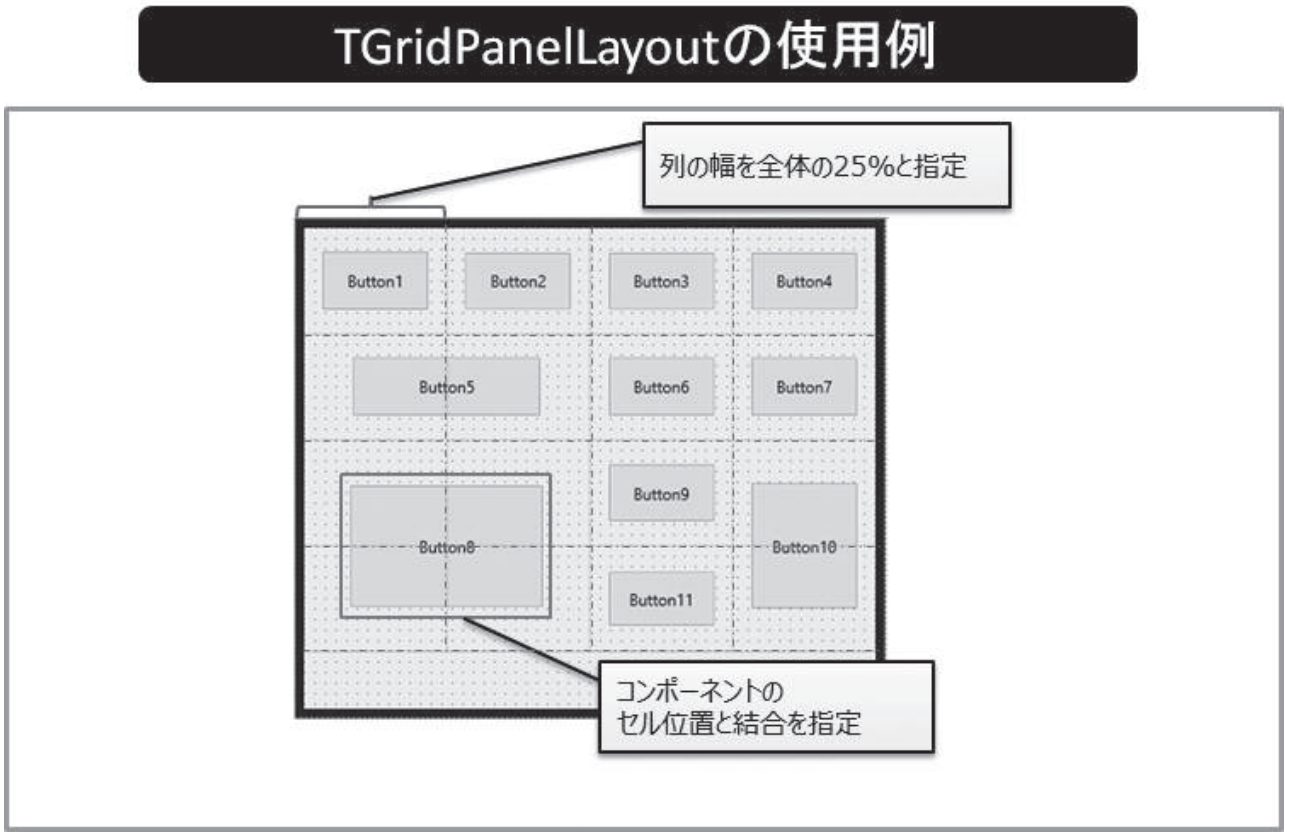

TGridPanelLayoutの使用例
列の幅を全体の25%と指定
Button1
Button2
Button3
Button4
Button5
Button6
Button7
Button8
Button9
Button10
Button11
コンポーネントの
セル位置と結合を指定

図16

Alignプロパティ

Alignプロパティ　設定値一覧

設定値	機能
Top, Left, Right, Buttom	親コンポーネントの1辺に寄せて、空いている領域いっぱいに広げて表示（VCLと同じ）
Client	空いている領域を埋め尽くして表示（VCLと同じ）
Fit, FitLeft, FitRight	親項目の中で最大化（コンポーネントの縦横比は維持）
Vertical, VertCenter, Horizontal, HorzCenter	幅あるいは高さの一方向をリサイズ
Center	親コンポーネントの中央に表示
Contents	親コンポーネント全体に表示
Scale	親コンポーネントのサイズにあわせてサイズが変更

図17

MarginsとPadding

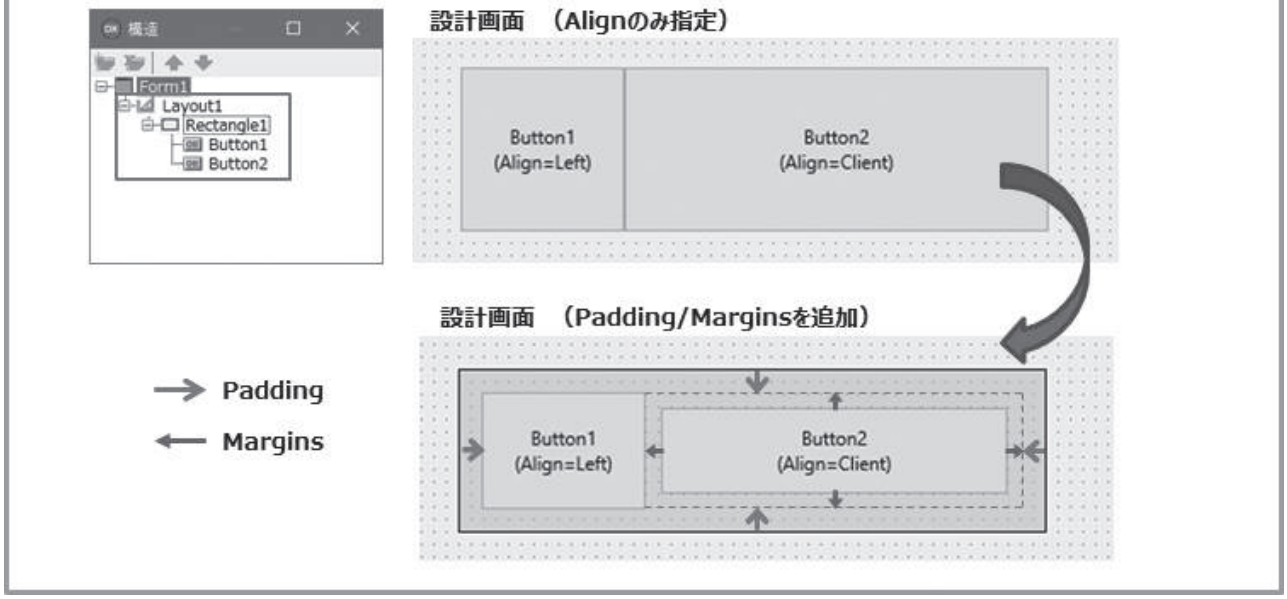


図18

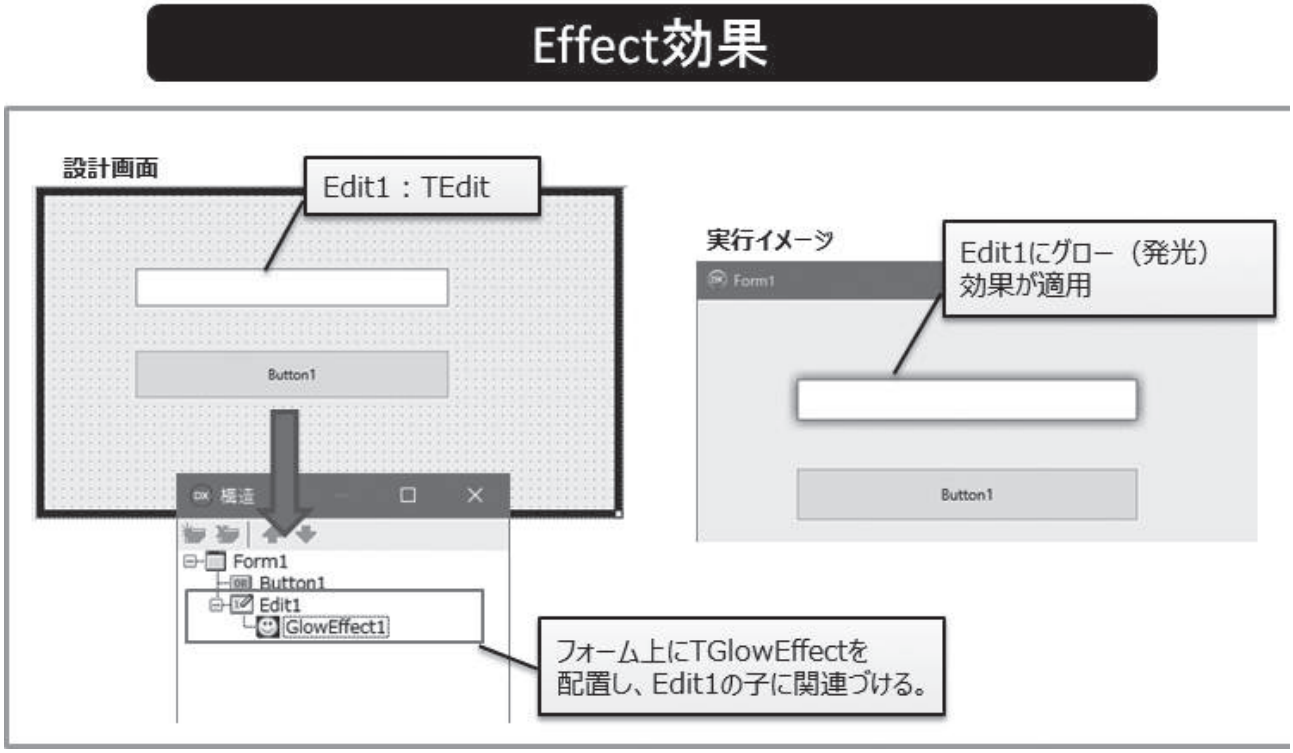

Effect効果
設計画面
Edit1 : TEdit
Button1
構造
Form1
Button1
Edit1
GlowEffect1
フォーム上にTGlowEffectを
配置し、Edit1の子に関連づける。
実行イメージ
Form1
Edit1にグロー（発光）
効果が適用
Button1

図19

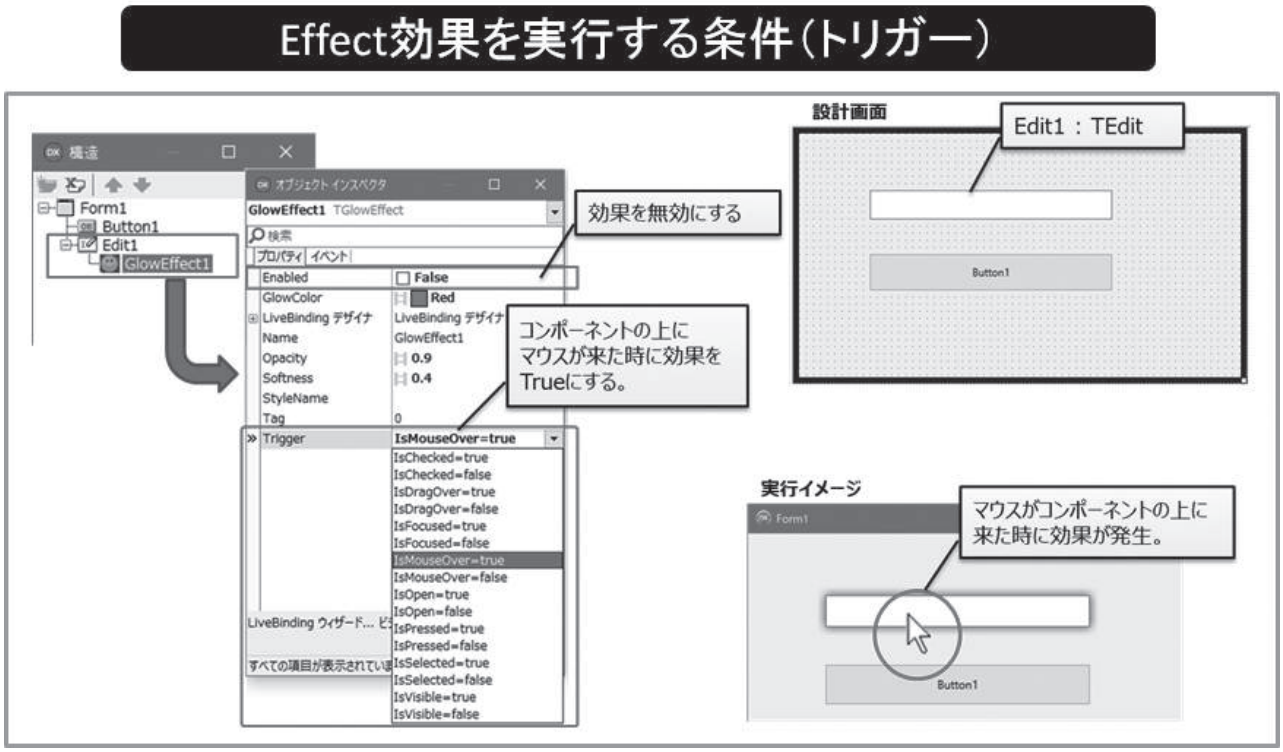

Effect効果を実行する条件（トリガー）
構造
Form1
Button1
Edit1
GlowEffect1
オブジェクト インスペクタ
GlowEffect1 TGlowEffect
検索
プロパティ
イベント
Enabled
False
GlowColor
Red
LiveBinding デザイナ
Name
GlowEffect1
Opacity
0.9
Softness
0.4
StyleName
Tag
0
Trigger
IsMouseOver=true
IsChecked=true
IsChecked=false
IsDragOver=true
IsDragOver=false
IsFocused=true
IsFocused=false
IsMouseOver=true
IsMouseOver=false
IsOpen=true
IsOpen=false
IsPressed=true
IsPressed=false
IsSelected=true
IsSelected=false
IsVisible=true
IsVisible=false
効果を無効にする
コンポーネントの上に
マウスが来た時に効果を
Trueにする。
設計画面
Edit1 : TEdit
Button1
実行イメージ
Form1
マウスがコンポーネントの上に
来た時に効果が発生。
Button1

図20

TFloatAnimation

TFloatAnimation

主なプロパティ

プロパティ	機能
StartFromCurrent	True：現在のプロパティ値を初期値とする
StartValue	開始値（StartFromCurrent=Falseの場合）
StopValue	終了値
Duration	アニメーション時間（秒）
Loop	True：アニメーションを繰り返す

図21

アニメーション効果　実行例

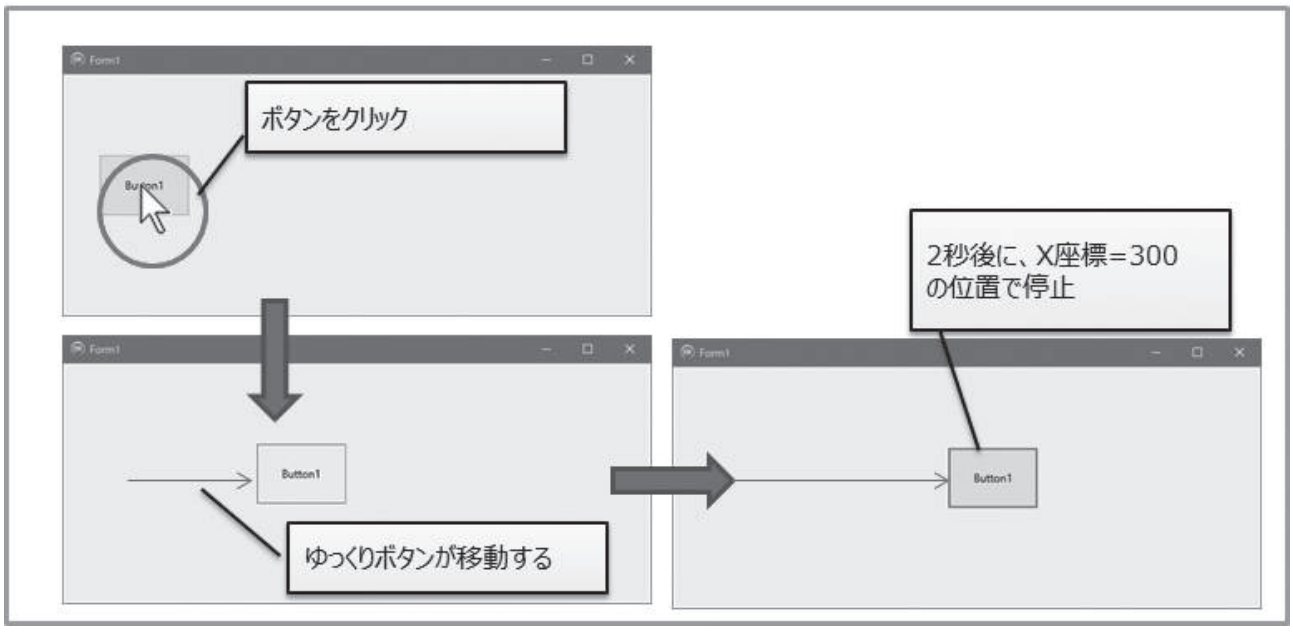
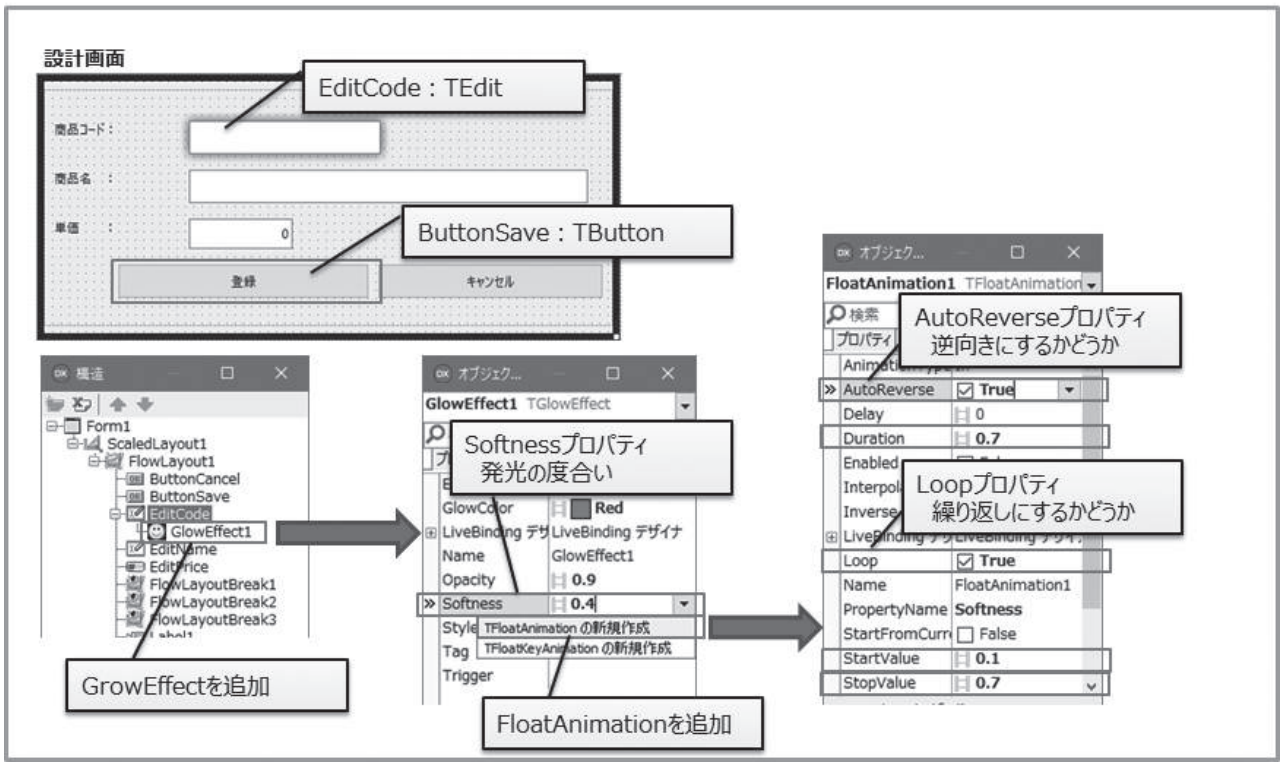


図22

Effect効果とアニメーション効果の組み合わせ



ソース1

登録ボタンのOnClickイベント

```
procedure TForm1.ButtonSaveClick(Sender: TObject);
begin
  GlowEffect1.Enabled := False;  //エフェクトOFF
  FloatAnimation1.Stop;          //アニメーションストップ

  //エラーチェック
  if EditCode.Text = '' then
  begin

    GlowEffect1.Enabled := True; //エフェクトON
    FloatAnimation1.Start;       //アニメーションスタート
    EditCode.SetFocus;

    MessageDlg('商品コードが未入力です。', TMsgDlgType.mtError,
      [TMsgDlgBtn.mbOK], 0);

    Abort;
  end;

  //更新処理
        ...
end;
```

宮坂 優大

株式会社ミガロ.

システム事業部 システム1課

[Delphi/400]

Delphi/400バージョンアップに伴う文字コードの違いと制御

略歴
1982年11月19日生まれ
2006年 近畿大学 理工学部卒業
2006年4月 株式会社ミガロ. 入社
2006年4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サポートを担当している。DelphiおよびDelphi/400のスペシャリストを目指して精進する毎日。

1.はじめに

2015年にWindowsの大きな変革となるWindows 10がリリースされたことにより、ここ数年でDelphi/400のアプリケーションも、Windows 10への正式対応を目的にバージョンアップを実施する企業が増えている。

Delphiはバージョンの互換性が非常に高い言語だが、バージョンアップでの大きなポイントの1つが文字コードである。Windows 9Xなど旧日本語環境のWindowsは、Ansi（Shift_JIS）を標準文字コードとして扱ってきたが、最近のWindowsではUnicodeが標準になっている。Delphi言語も同様に、以前はAnsiが標準文字コードであったが、最近のバージョンではUnicodeに変わっている。

プログラムには文字を扱うコードが非常に多く、その基盤となる文字コードが変わると、アプリケーションの動作も影響を受ける可能性が高い。そのため、バージョン互換が高いDelphi言語であっても、標準文字コードが異なるバージョン間でのバージョンアップでは、影響を考慮した対応が必要となる。

本稿では、こうしたDelphiのバージョンアップで重要なポイントとなる文字コードの扱いについて、違いや制御方法を考察する。

2.Delphi言語で扱う文字コード

冒頭で述べたように、Delphiのバージョンアップで注意すべき重要なポイントとして、文字コードが挙げられる。Delphiが扱う文字コードについて、バージョンで整理すると、Delphi 2007以前ではAnsi型（AnsiString）、Delphi 2009以降ではUnicode型（UnicodeString）が標準となっている。

具体的にはUnicode対応により、プログラム上のString型、Char型、PChar型の標準仕様が変更されている。そのため、Delphi 2007以前のバージョンからDelphi 2009以降のバージョンへプログラムをバージョンアップする場合には、文字コードへの配慮が必要となる。

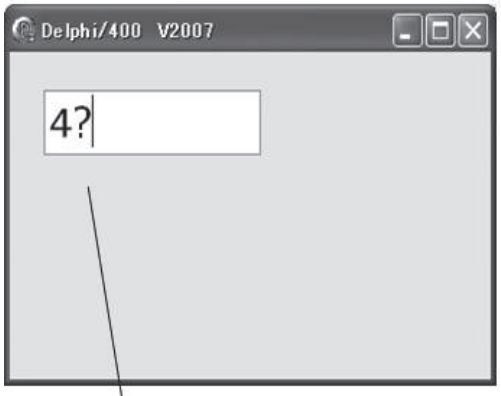
Unicodeとは、符号化文字集合や文字符号化方式などを定めた文字コードの規格である。UnicodeはAnsiと比べると、使える文字種類が非常に多く、世界中で表現されているほぼすべての文字に対応できる。

たとえば、「m^3」という記号文字はAnsiString型には存在しないが、UnicodeString型には存在している。これをDelphiアプリケーション上で入力すると、Unicodeに対応していないDelphi 2007以前のバージョンでは「?」で表示されるが、Unicodeに対応したバージョンでは正しく表示できる。【図1】

Unicodeについて、もう少し補足する。Unicodeの制御文字、文字、記号にはそれぞれに数値が割り当てられており、それをコードポイントと呼んでいる。たとえば、“A”のコードポイントは65（バイナリでは41）、“B”のコードポイントは66（バイナリでは42）と、それぞれの文字に一意のコードポイントが割り当

図1

AnsiStringとUnicodeString



V2007（AnsiString）
では表示できない

10 Seattle（UnicodeString）
では表示可能

図2

AnsiStringとUnicodeStringの桁数とバイト数

AnsiString型

文字	A	B	あ		い	
バイト	1	2	3	4	5	6
バイナリ	41	42	82	A0	82	A2

UnicodeString型

文字	A		B		あ		い	
バイト	1	2	3	4	5	6	7	8
バイナリ	00	41	00	42	82	A0	82	A2

てられている。

ここで重要なのは、UnicodeString 型ではそれぞれの文字が 2 バイトで表現される点である。AnsiString 型では半角文字は 1 バイトで表現されるが、UnicodeString 型では半角・全角という概念がない。この変更のため、文字列のバイト数で計算していたロジックは、【図 2】のように Ansi と Unicode で計算や動作が一致しなくなる可能性がある。これが大きな影響点といえる。

ただし標準の文字コードが変わっているだけなので、これまで使用していた Ansi 文字列は「AnsiString」と明示的にコーディングすれば、新しい Delphi 上でも使用できる。また同様に、古い Char 型、PChar 型 は AnsiChar 型、PAnsiChar 型として使用できる。

実際に文字コードの違いで、どのような挙動の差異があるかを調べるため、【図 3】【ソース 1】のプログラムを Delphi 2007 と Delphi 10 Seattle で作成して比較した。

画面に TEdit と TButton を配置し、ボタンを押すと入力した文字列の 1 ～ 4 バイト目を Copy 関数で取得し、ShowMessage 関数で表示する。ここでは Delphi 2007 のアプリケーションを①、Delphi10Seattle のアプリケーションを②として比較する。

画面項目で「あいうえお」と入力してボタンを押すと、①のアプリケーションでは「あい」と表示される。【図 4】

しかし、②のアプリケーションで同じことを実行しても結果が異なる。同じように「あいうえお」と入力し、ボタンを押すと「あいうえ」と表示される。Delphi 2007 とは異なり、1 ～ 4 バイト目ではなく、1 ～ 4 文字目を取得していることがわかる。【図 5】

このように同じプログラムロジックであっても、文字コードの違いによって動作が異なることがある。では②のアプリケーションで、①のアプリケーションと同じように動作させる方法を説明する。

単純な方法としては、UnicodeString 型を AnsiString 型で扱うことで解消できる。Copy 関数に渡す文字列を AnsiString 型で明示的にキャストすることで、従来の AnsiString 型として文字列を扱える。【ソース 2】

修正したソースで再度実行すると、①のアプリケーションと同じ文字列で表示されることがわかる。【図 6】

ただし、AnsiString 型でキャストすると、UnicodeString 型では扱える特殊な文字が扱えなくなるので、デメリットも理解しておく必要がある。

IBMi 側の環境が日本語 Unicode を取り扱う CCSID1399 であれば、UnicodeString 型のまま扱ったほうがよいが、従来の 5026 ／ 5035 の CCSID であれば、もともと UnicodeString の文字種類を扱えないので、AnsiString で扱っても問題ない。【図 7】

Delphi 2007 以前から Delphi 2009 以降にバージョンアップする際には、こうした文字コードの制御を考慮してプログラムを変更すれば、スムーズに実施できる。

3.文字コードの違いによる制御ポイント

前章では文字コード自体の違いについて考察したが、文字コードが変わることで違いが発生する点はほかにもある。それはプロパティや関数の動作である。

たとえば TEdit では、入力できる最大桁数を制御する MaxLength プロパティがある。これもバイト数単位から文字数単位でカウントされるため、全角文字を入力した時にこれまでと挙動が異なる。ここでは TEdit で MaxLength プロパティを 5 で設定した例を説明する。【図 8】

この TEdit に対して、「123 あい」と入力する。Delphi 2007 では「123 あ」と入力した段階で規制がかかり、入力できなくなる。【図 9】

しかし、Delphi 10 Seattle の同じアプリケーションでは、「123 あい」とすべて文字を入力できるため、挙動が違っていることがわかる。【図 10】

これは Unicode なので、文字数としてカウントされている。これと同じ動作になるのが、文字長を判断する Length 関数である。「123 あいう」という文字列に対して、Length 関数で取得できる結果値は Delphi のバージョンによって異なる。

Delphi 2007 ではバイト数で算出されているため、9 が取得できるが、Delphi 10 Seattle では文字数で算出されるため、6 が取得される。同じ挙動に合わせる対処方法としては、前章で説明したように、対象の文字列を AnsiString 型でキャストすればよい。

関数については、Delphi 2009 以降でも互換性を考慮し、Ansi 専用の関数が別途用意されている場合もある。それらの関数は、「System.AnsiStrings」ユニットに定義されている。

たとえば、StringReplace や Upper/LowerCase、StuffString などよく使われる関数が多数含まれている。プログラムの Uses 節に「System.AnsiStrings」を追加すれば、これらの Ansi 型関数を使用できる。

文字列を扱う関数で動作が違う場合には、互換の Ansi 関数が用意されていることが多いので、プログラムを作り変える前に、このユニットや WindowsAPI を確認するようお勧めする（WindowsAPI 関数についても、Microsoft 社が Ansi 互換の関数を多数用意している）。

4.おわりに

本稿では Delphi のバージョンによって異なる文字コードや制御方法について考察し、対処ポイントをまとめた。

文字コード自体は、普段のプログラミングではそれほど意識しないが、プログラムコードのほとんどは文字コードが密接に関係している。そのため、標準文字コードが変わると、アプリケーションの動作にも影響する可能性がある。

しかし前述の対処例により、文字コードの扱いや互換関数を把握していれば、バージョンアップ対応は定型的なソース変更で実施できる。

Delphi 2007 以前のプログラムから Delphi 2009 以降へバージョンアップをする場合には、本稿の対処ポイントを役立てていただければ幸いである。

M

図3

検証用設計画面1

ソース1

ボタン押下時の処理

```
procedure TSampleForm.Button1Click(Sender: TObject);
var
  sStr: String;
begin
  //Editに入力された1～4文字(ﾊﾞｲﾄ)目を取得
  sStr := Copy(Edit1.Text, 1, 4);

  //取得した文字列を表示
  ShowMessage(sStr);

end;
```

図4

実行結果（Delphi2007）

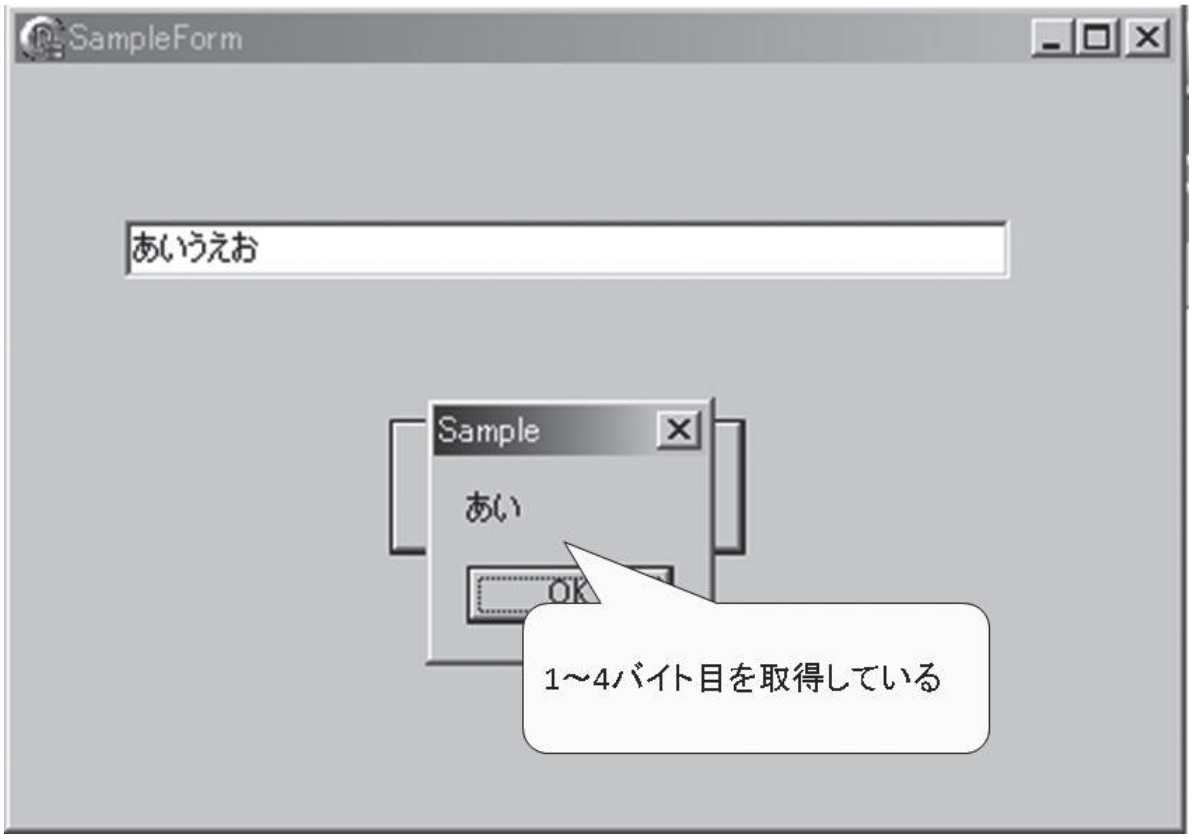


図5

実行結果（ Delphi10Seattle ）

ソース2

ボタン押下時の処理(AnsiString修正後)

```
procedure TSampleForm.Button1Click(Sender: TObject);
var
  sStr: String;
begin
  //Editに入力された1～4文字(ﾊﾞｲﾄ)目を取得
  sStr := Copy(AnsiString(Edit1.Text), 1, 4);

  //取得した文字列を表示
  ShowMessage(sStr);

end;
```

Editの文字列をAnsiString型でキャストする

図6

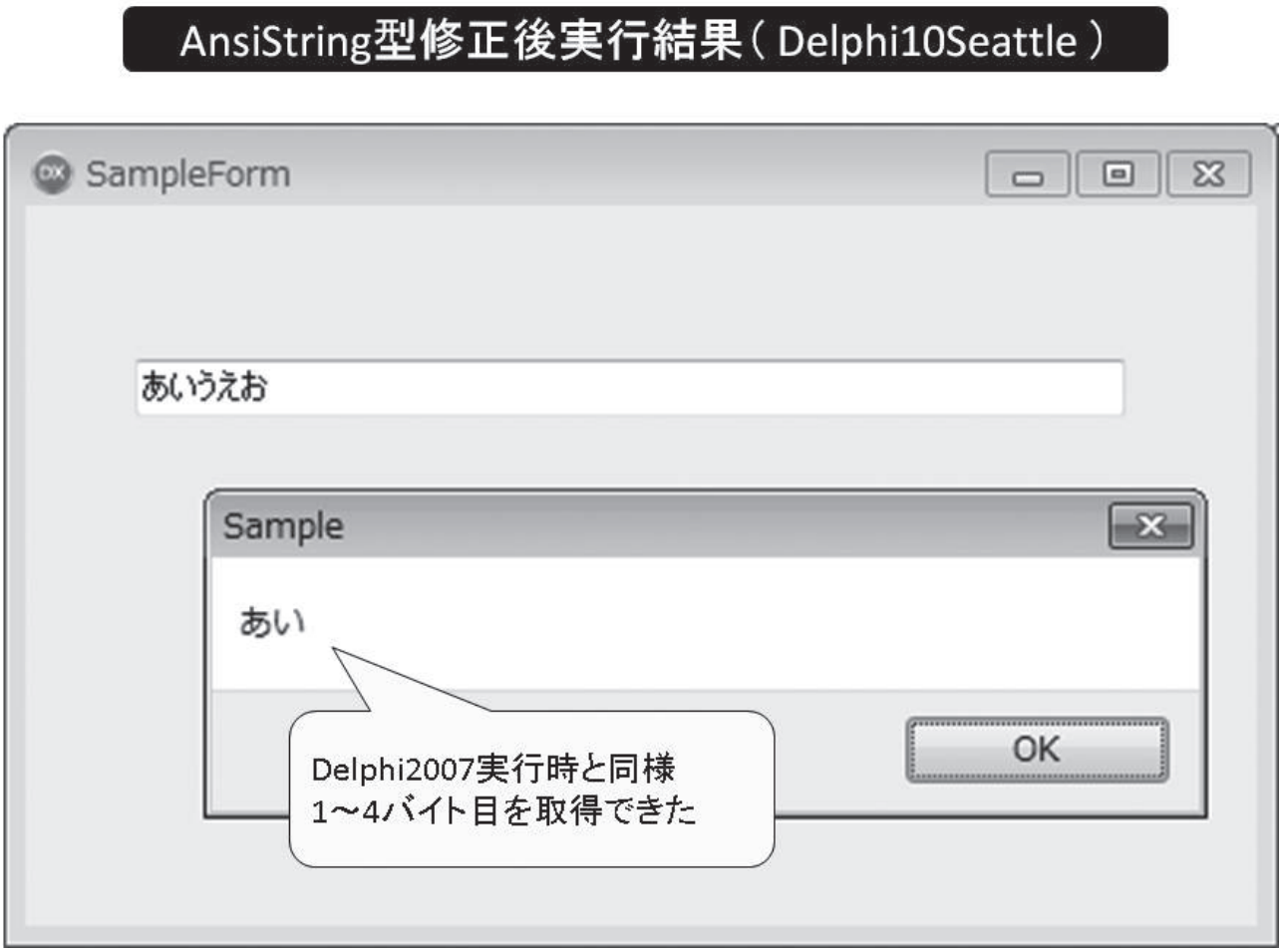


図7

Unicode文字をCCSID（5026/5035）のファイルに保存

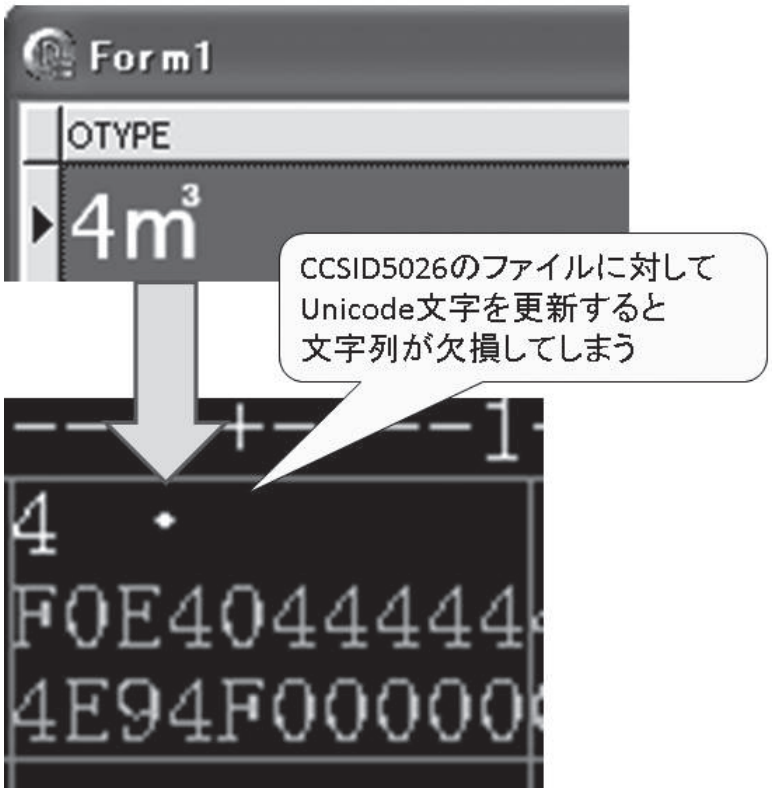


図8

検証用設計画面2

図9

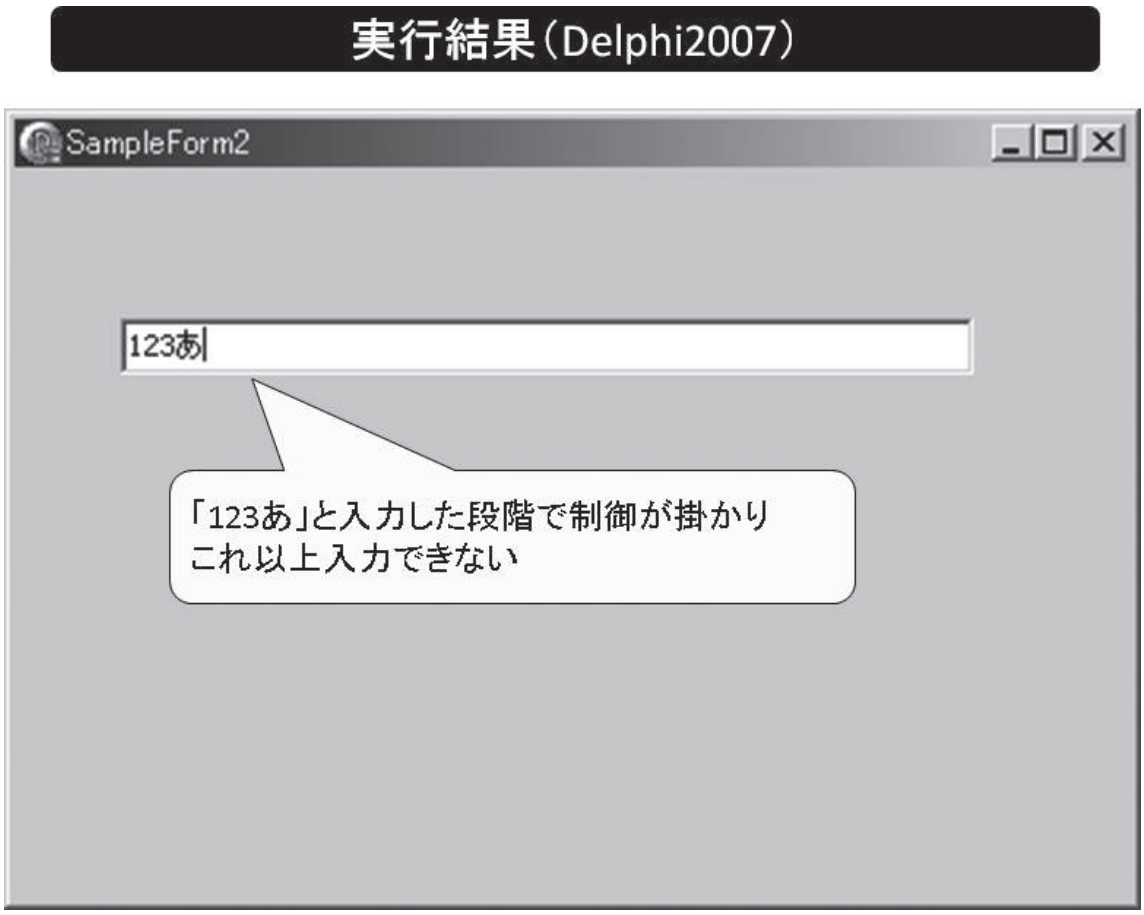

実行結果（Delphi2007）
SampleForm2
123あ
「123あ」と入力した段階で制御が掛かり
これ以上入力できない

図10


実行結果（Delphi10Seattle）
SampleForm2
123あい
「123あい」と5文字まで
入力できる

畑中 侑

株式会社ミガロ.

システム事業部 システム2課

[Delphi/400]

FastReportへの効率的な帳票レイアウトコンバート

略歴
1983年7月6日生まれ
2006年 京都産業大学 法学部卒業
2006年4月 株式会社ミガロ. 入社
2006年4月 システム事業部配属

現在の仕事内容
システムの受託開発を担当しており、要件確認から納品・フォロー、保守作業に至るまで、システム開発全般に携わっている。

1.はじめに

アプリケーションのマイグレーションにはさまざまな目的があるが、最近の傾向としてはシステム基盤の最新OSへの対応や、新しいIT技術、たとえばスマートデバイス活用の検討などが挙げられる。

2016年7月にリリースされたDelphi/400 10 Seattleでは、Windows 10への正式対応や、マルチデバイス開発環境が強化されており、Delphi/400のマイグレーションを検討・実施された企業は非常に多い。こうしたマイグレーションの際、事前に検討すべき課題の1つに帳票ツールがある。

Delphi 7以前のバージョンでは、帳票ツール「QuickReport」により帳票機能を作成していることが多い。しかしバンドルされる帳票ツールが変わり、QuickReportが標準サポートされなくなってから、帳票機能を別の帳票ツールに作り直す必要が生じてきた。

このような作り直しの手間が、マイグレーションに踏み切れない理由になる場合もある。この課題に対する解決策の1つとして、本稿ではQuickReportからDelphi最新版に付属している帳票ツール「FastReport」へのコンバートを題材に効率的な手法を説明する。

2.効率的な帳票レイアウトコンバート手法

通常、帳票ツールを変更する場合には、帳票レイアウトを新しい帳票ツールで作り直すことになる。そうした作業はプログラムの移行に時間を要するが、QuickReportからFastReportへコンバートする場合には、FastReportで用意されているコンバート機能を利用することで、効率的に移行できる。大きな流れとしては、コンバート作業の環境を準備し、移行用のプロジェクトを作成してコンバート処理を実行する。

本稿では、QuickReportが付属する最終バージョンのDelphi 7から、FastReportが最初に付属したDelphi XE3へのマイグレーション環境で検証している。

2-1. コンバート環境の準備

コンバートを実施する作業環境として、QuickReportを使用していたDelphi開発環境（本稿ではDelphi 7）にFastReportのコンポーネントをインストールする。

手順は、次のとおりである。

①FastReportコンポーネントの「.bpl」ファイル作成

FastReportのインストールフォルダ内に、「recompile.exe」という実行ファイルが存在する。もしくは、Windowsボタンから「すべてのプログラム｜FastReports｜FastReport」内を見ると、「Recompile Wizard」が見つかる。

これを起動すると、【図1】のウィザード画面が表示される。

このウィザード画面で上から順に設定値を指定し、最後に"Compile"ボタン

図1

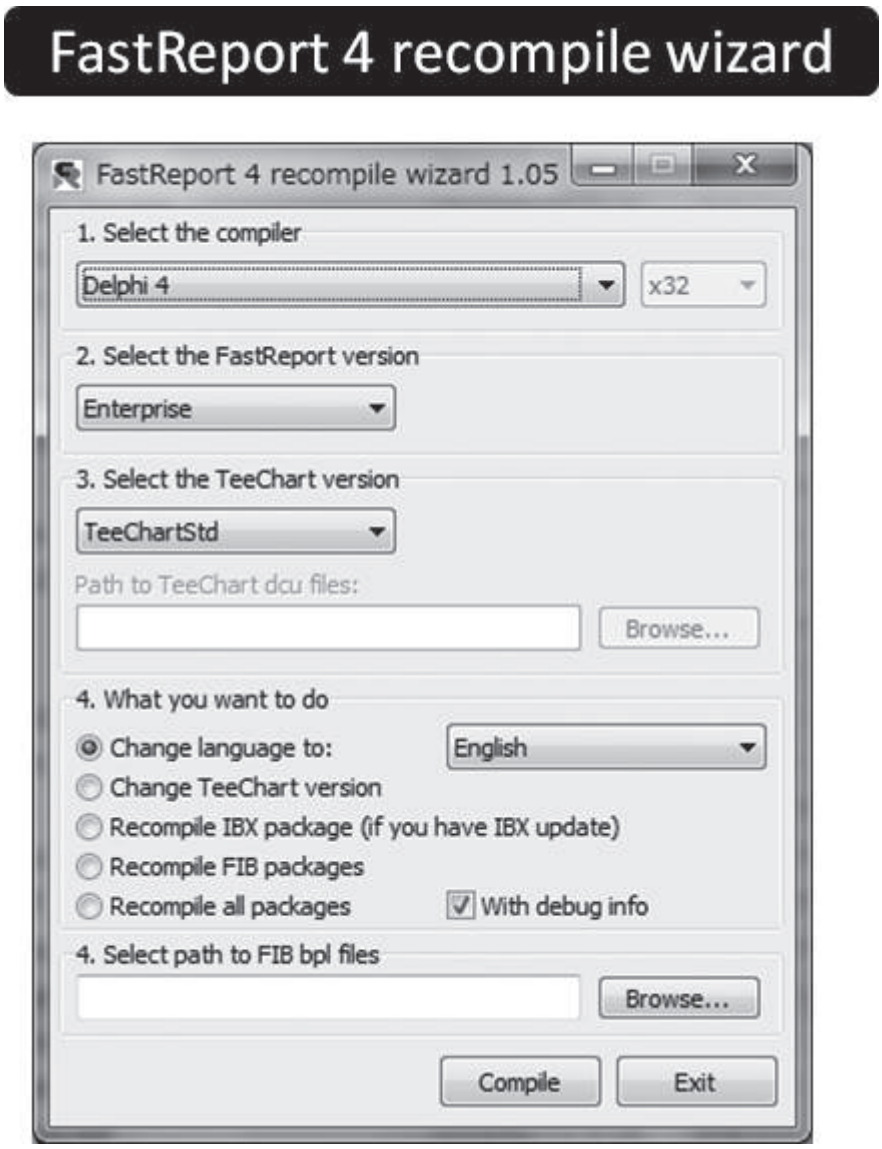
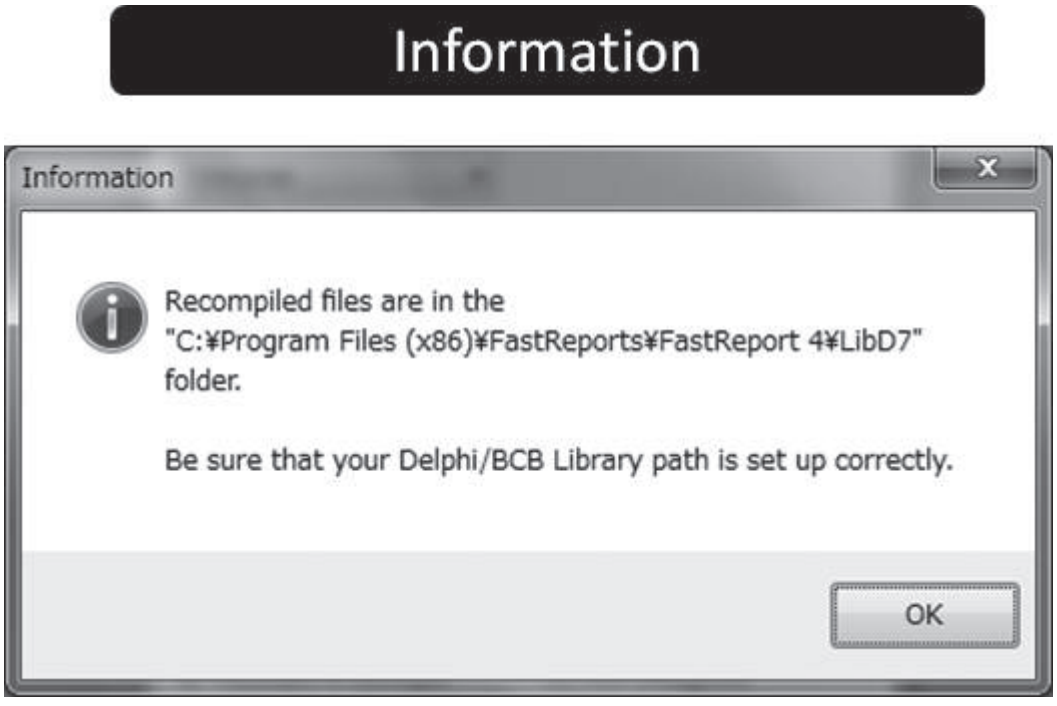

FastReport 4 recompile wizard
FastReport 4 recompile wizard 1.05
1. Select the compiler
Delphi 4
x32
2. Select the FastReport version
Enterprise
3. Select the TeeChart version
TeeChartStd
Path to TeeChart dcu files:
Browse...
4. What you want to do
Change language to:
English
Change TeeChart version
Recompile IBX package (if you have IBX update)
Recompile FIB packages
Recompile all packages
With debug info
4. Select path to FIB bpl files
Browse...
Compile
Exit

図2


Information
Information
Recompiled files are in the
"C:¥Program Files (x86)¥FastReports¥FastReport 4¥LibD7"
folder.
Be sure that your Delphi/BCB Library path is set up correctly.
OK

を押すことで、「FastReport 4」フォルダ内に「LibD*」というフォルダが作成され、各種「.bpl」ファイルを一括で作成できる（「*」は指定バージョンに応じてバージョン値が入る）。

以下に、ウィザード画面の設定内容を確認する。

(1) Select the compiler

開発環境のバージョンを選択する。本稿では「Delphi7」を指定。

(2) Select the FastReport version

FastReport のバージョンを選択する。本稿では「Enterprise」を指定。

(3) Select the TeeChart version

FastReport で使用するグラフ「TeeChart」のバージョンを選択する。本稿では「TeeChartStd」を指定。

(4) What you want do

リコンパイル目的を選択する。本稿では「Recompile all packages」を指定。

(4) Select path to FIB bpl files

レポート内のデータソースで FIB 接続を用いる場合に選択する。本稿では不要なので設定はしない（連番（4）が続いているが、画面表記に合わせている）。

各項目を設定し、"Compile" ボタンを押すことで、コンパイル処理が自動で始まり、【図 2】の確認メッセージが表示されれば完了である。

この操作で、「FastReport 4」フォルダ内に「LibD7」というフォルダが作成され、その中に各種「.bpl」が存在することが確認できる。【図 3】

②FastReport 関連コンポーネントの登録

Delphi を起動し、メニューバーより「コンポーネント｜パッケージのインストール…」を順番に選択し、「デフォルトプロジェクトオプション」のウインドウを起動する。

中段の「追加（A）…」ボタンより、先ほど「LibD7」フォルダ内に生成された各種「.bpl」を順次追加する。追加が終われば、"OK" ボタンを押下して、「デフォルトプロジェクトオプション」を終了する。【図 4】

③ライブラリパスの追加

Delphi 開発環境のメニューバーより、「ツール｜環境オプション…」を順番に選択し、「環境オプション」のウインドウを起動する。

次に「ライブラリ」タブに切り替えて、「ライブラリパス（B)」の右側にある参照ボタンを押下し、「ディレクトリ」のウインドウを起動する。「ライブラリパスの一覧」に先ほどの「LibD7」フォルダのパスを加える。【図 5】

具体的には、中断の参照ボタンを押下し、「LibD7」フォルダのパスを指定すると、" 追加（A）" ボタンがアクティブになるので押下する。その後、"OK" ボタンで「ディレクトリ」「環境オプション」の各ウインドウを終了する。

2-2. コンバート用プロジェクトの新規作成

QuickReport から FastReport へ帳票レイアウトをコンバートするには、FastReport に用意されている「ConverterQR2FR」ユニットを利用する。そのため、この作業ではコンバート用にプロジェクトの作成が必要となる。

①プロジェクトの新規作成

Delphi 開発環境を起動し、メニューバーより「ファイル｜新規作成｜アプリケーション」を選択し、新規プロジェクトを作成する。次に初期生成された Unit1 のデザインに対して、任意のフォームサイズに変更し下記のコンポーネントを配置する。

Button　ツールパレット = Standard
frxReport
　　ツールパレット = FastReport 4.0

最後に、コンバート対象の TQuickRep をプロジェクトに追加する。本稿では「QuickFrm」とする。【図 6】

フォームに配置したボタンを押下することで、コンバート対象の TQuickRep の帳票レイアウトを FastReport のデザインファイルにコンバートする。

プロジェクトの保存内容と各プロパティ設定は、【図 7】のとおりである。

●プロジェクト
　保存ファイル名：ConvertSample.dpr
●初期生成フォーム（TForm1）
　Name プロパティ：frmMain
　保存ファイル名：MainFrm.pas
●配置した Button
　Name プロパティ：
　　btnConvertQuickReport
　Caption プロパティ：
　　ConvertQuickReport
●配置した frxReport
● Name プロパティ：frxReport

②コンバート手続きの実装

MainFrm のソース部を開き、uses 節に「ConverterQR2FR」とコンバート対象の QuickFrm を追加する。続いて、btnConvertQuickReport の OnClick イベントにコンバート手続きを実装する。【ソース 1】

TConverterQR2FR の変数を定義し、Source プロパティにコンバート対象の TQuickRep、Target プロパティに TfrxReport を指定したあと、Convert 手続きによりコンバートが実行される。その後、SaveToFile 手続きにより FastReport のデザインファイルとして保存する。

③コンパイルして EXE を生成する

プロジェクトをコンパイルし、エラーがないことを確認し、EXE ファイルを生成する。QuickReport のバージョンによっては未定義の変数が TConverterQR2FR で宣言されているため、エラーとなる場合はコメントアウトする。【ソース 2】

2-3. コンバート処理の実行

作成した EXE ファイルを実行し、QuickReport の帳票レイアウトを FastReport へコンバートする。

EXE ファイル実行後、" Convert QuickReport " ボタンを押下すると、EXE ファイルと同階層に FastReport のデザインファイルである拡張子「fr3」のファイルが生成されたことを確認できる。続いて FastReport を起動し、生成された FastReport のデザインファイルを開いて確認する。【図 8】

QuickReport の TQuickRep が、FastReport の TfrxReportPage に変換され、TQRLabel が、TfrxMemoView に置き換えられており、レイアウトが変換されていることが確認できる。

図3

フォルダ「LibD7」

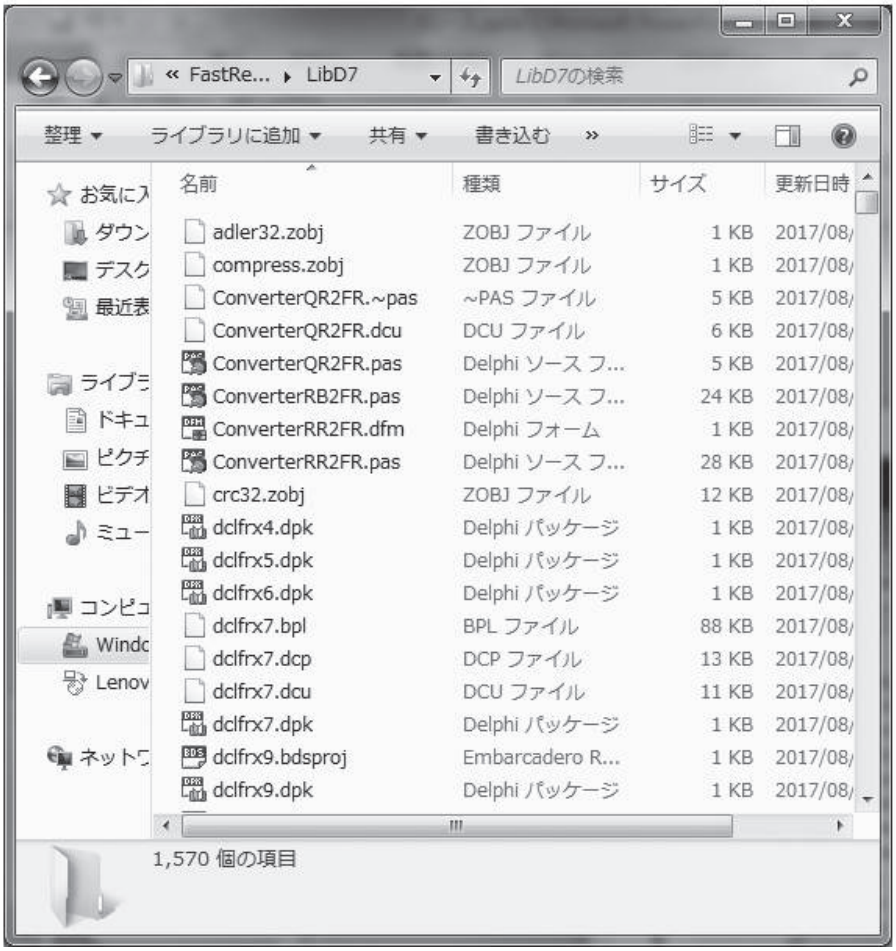
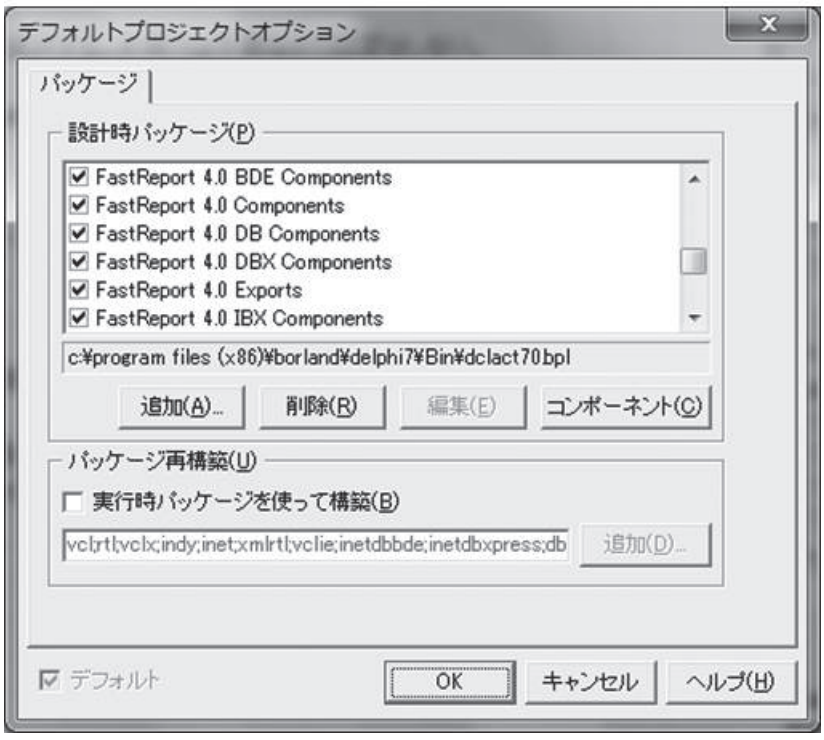


図4

デフォルトプロジェクトオプション



しかし【図 8】のように、TfrxMemo View の Text 部で一部の文字列が文字化けする場合もある。こうした文字化けが発生した場合は、コンバート後のファイルを FastReport で確認し、手動で設定などを調整する必要がある。

またコンバート後の環境に合わせて、適時データセットの割り当てを補填すれば、新しい Delphi 開発環境（本稿では XE3）ですぐに FastReport のプログラムを利用できる。これで QuickReport から FastReport への帳票レイアウトのコンバートが完了である。

3.おわりに

本稿では QuickReport から FastReport の帳票レイアウトのコンバートを題材に効率的な手法を調査・検証しているが、QuickReport だけでなく、Rave Raports についても類似したコンバート用ユニットは用意されている。

Delphi ではバージョンによって付属される帳票ツールが変更されることがあるが、本稿の手法を使えば、最新の帳票ツールである FastReport へ少ない作業で移行できる。帳票ツールの変更が作業課題となっている場合は、この手法で効率的なマイグレーションの実施に役立てていただきたい。

M

図5

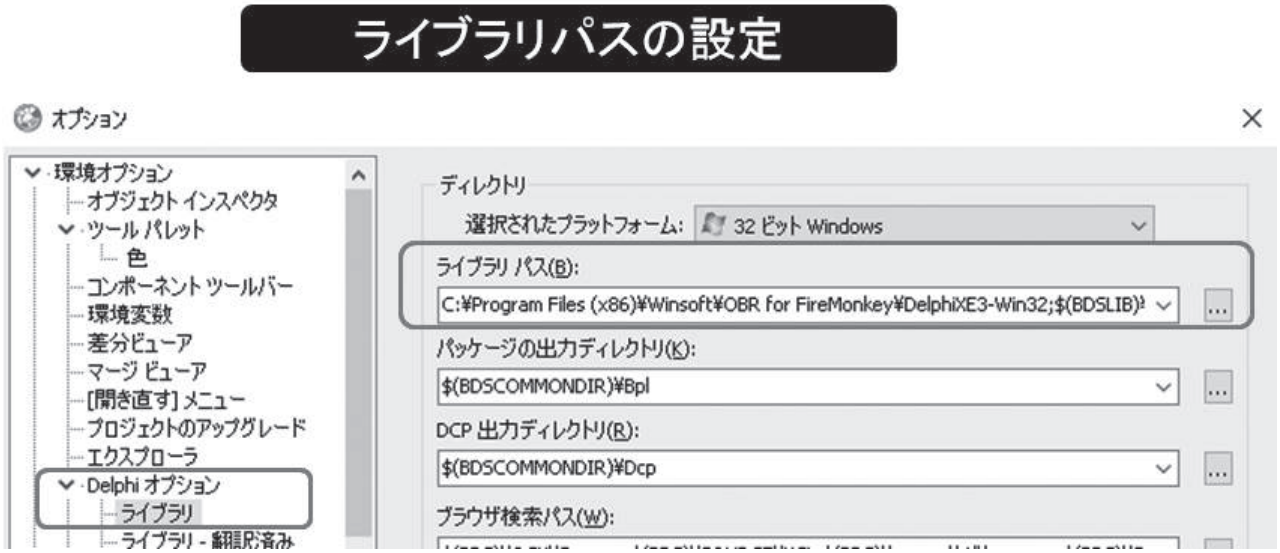

ライブラリパスの設定
オプション
環境オプション
オブジェクト インスペクタ
ツール パレット
色
コンポーネント ツールバー
環境変数
差分ビューア
マージ ビューア
[開き直す] メニュー
プロジェクトのアップグレード
エクスプローラ
Delphi オプション
ライブラリ
ディレクトリ
選択されたプラットフォーム:
32 ビット Windows
ライブラリ パス(B):
C:¥Program Files (x86)¥Winsoft¥OBR for FireMonkey¥DelphiXE3-Win32;$(BDSLIB)¥
パッケージの出力ディレクトリ(K):
$(BDSCOMMONDIR)¥Bpl
DCP 出力ディレクトリ(R):
$(BDSCOMMONDIR)¥Dcp
ブラウザ検索パス(W):

図6

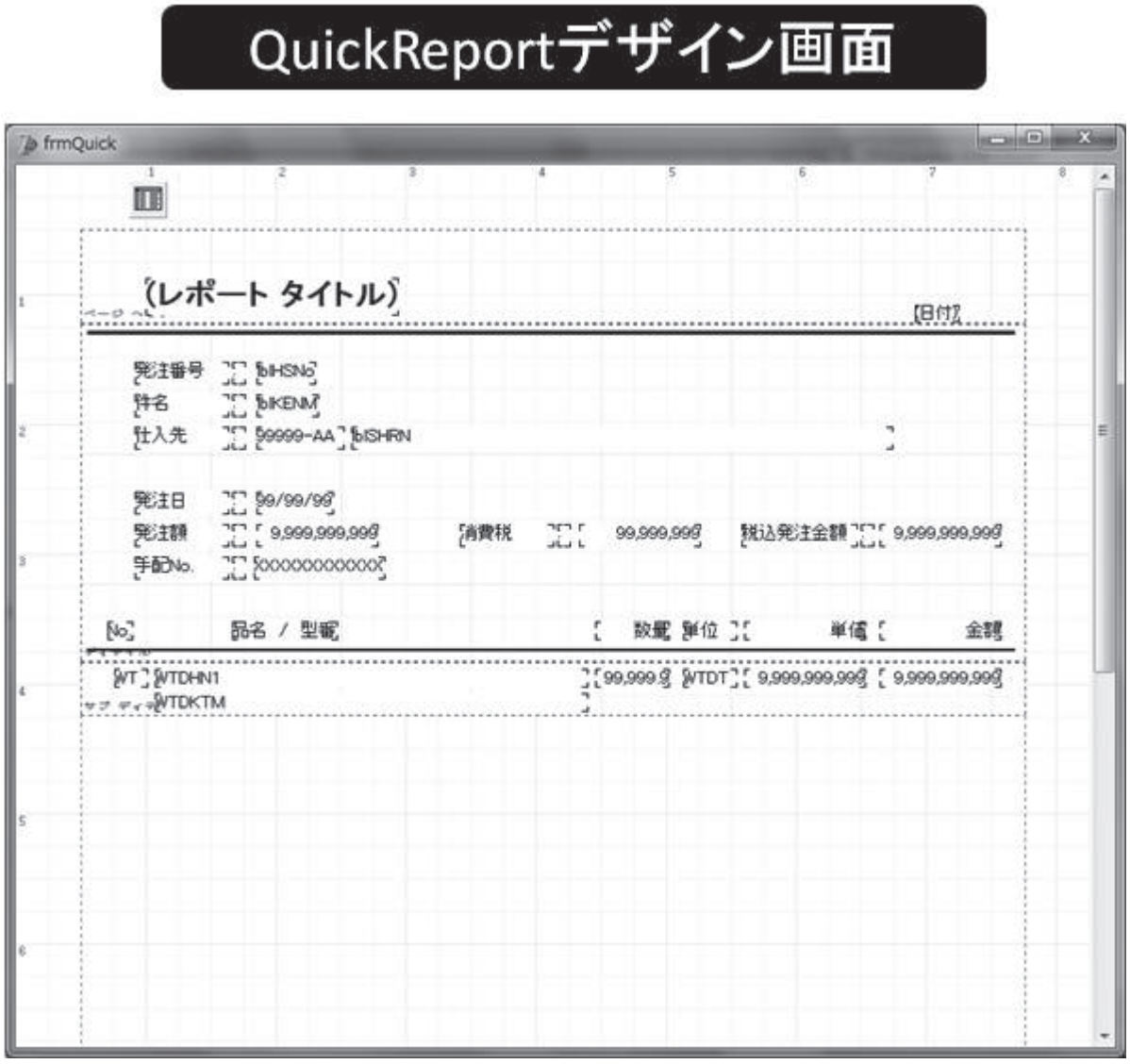
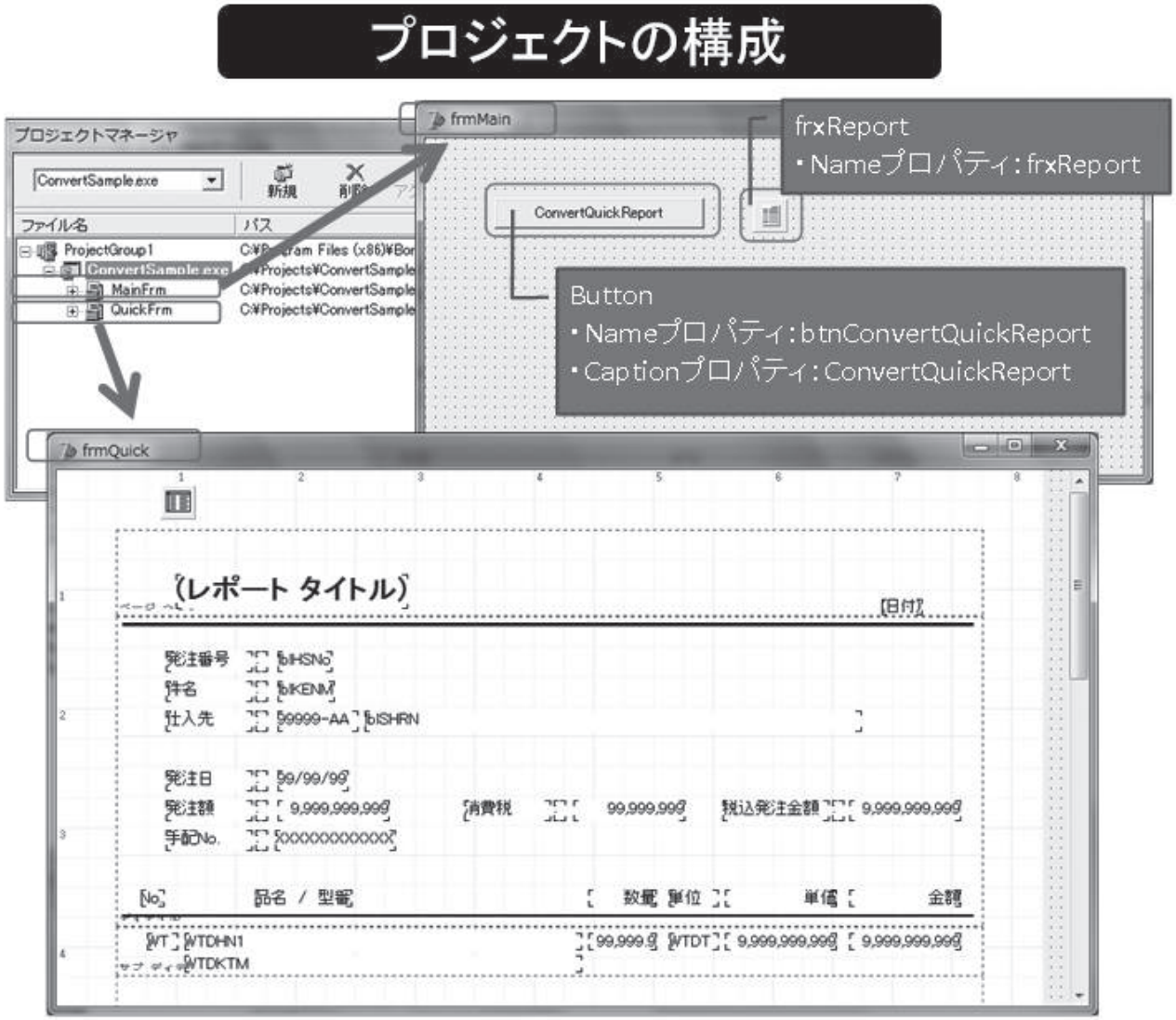

QuickReportデザイン画面
frmQuick
〔レポート タイトル〕
〔日付〕
発注番号 〔bIHSNo〕
件名 〔bIKENM〕
仕入先 〔99999-AA〕〔bISHRN〕
発注日 〔99/99/99〕
発注額 〔9,999,999,999〕
消費税 〔99,999,999〕
税込発注金額 〔9,999,999,999〕
手配No. 〔XXXXXXXXXXX〕
No. 品名 / 型番 数量 単位 単価 金額
WT WTDHN1 WTDKTM 99,999 WTDT 9,999,999,999 9,999,999,999

図7


プロジェクトの構成
プロジェクトマネージャ
ConvertSample.exe
新規
ファイル名
パス
ProjectGroup1
ConvertSample.exe
MainFrm
QuickFrm
C:¥Projects¥ConvertSample
C:¥Projects¥ConvertSample
frmMain
ConvertQuickReport
frxReport
・Nameプロパティ：frxReport
Button
・Nameプロパティ：btnConvertQuickReport
・Captionプロパティ：ConvertQuickReport
frmQuick
〔レポート タイトル〕
〔日付〕
発注番号
件名
仕入先
発注日
発注額
手配No.
消費税
税込発注金額
No.
品名 / 型番
数量
単位
単価
金額

ソース1

btnConvertQuickReportのOnClickイベント

```
procedure TfrmMain.btnConvertQuickReportClick(Sender: TObject);
var
  conv: TConverterQR2FR;
begin

  //コンバート対象を生成
  frmQuick := TfrmQuick.Create(Self);
  try
    //TConverterQR2FRクラスを生成し、各種プロパティをセット
    conv := TConverterQR2FR.Create;
    conv.Source := frmQuick.RepHattyuList;  //コンバート対象のTQuickRepを指定
    conv.Target := frxReport;               //自身のTfrxReportを指定
    conv.Convert;                           //変換処理の実行

    //EXEファイルと同階層にレポートタイトル名として保存
    frxReport.SaveToFile(frmQuick.RepHattyuList.ReportTitle + '.fr3');
  finally
    frmQuick.Release;
  end;
end;
```

ソース2

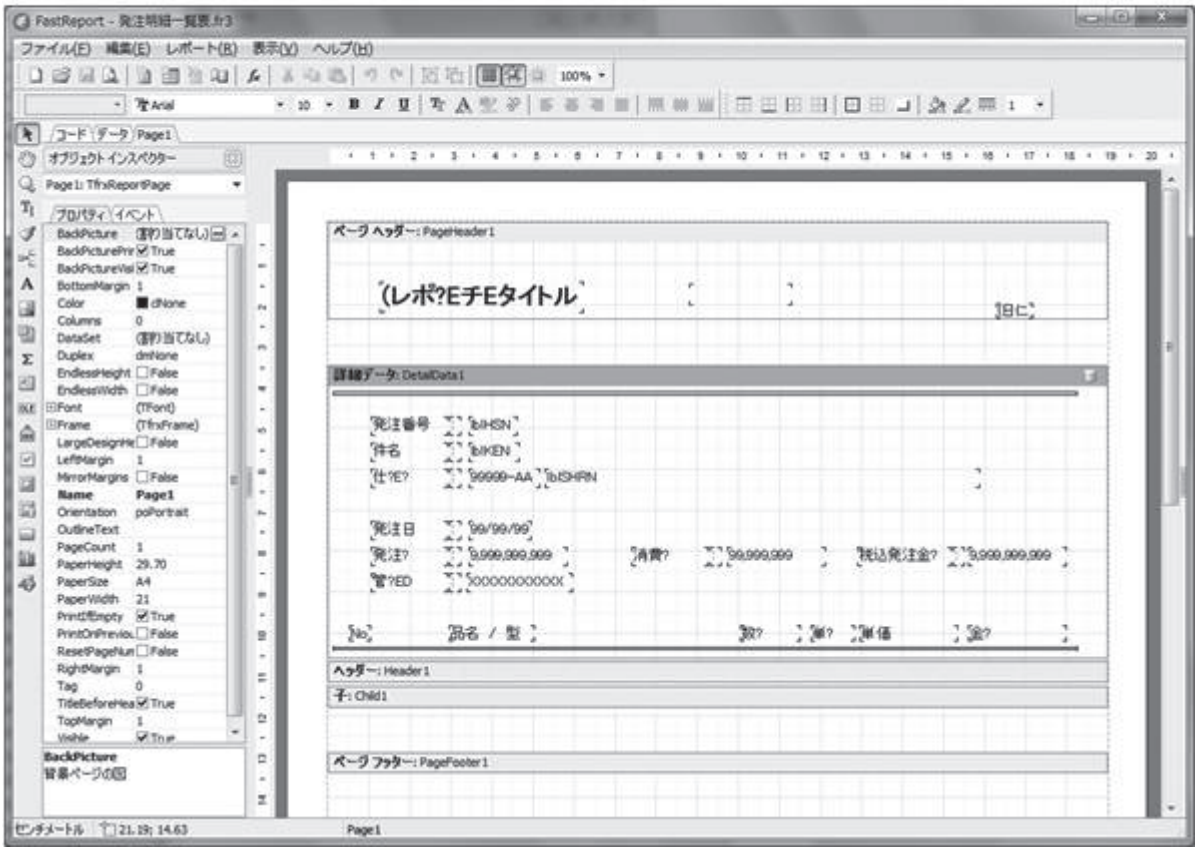
コメントアウト箇所（ConverterQR2FR.pas）

```
procedure TConverterQr2Fr.AddObjects(const ABand: TQRCustomband;
  const Band: TfrxBand);
var
  i: integer;
〜〜省略〜〜
 if ABand.Controls[i] is TQRShape then
    begin
      Sh := ABand.Controls[i] as TQRShape;
      Shape := TfrxShapeView.Create(Band);
      Shape.CreateUniqueName;
      case Sh.Shape of
        qrsRectangle: Shape.Shape := skRectangle;
        qrsCircle: Shape.Shape := skEllipse;
//QuickReportのバージョンにより未定義のプロパティのためコメントアウト
//        qrsRoundRect: Shape.Shape := skRoundRectangle;
      end;
      Shape.SetBounds(Sh.Left, Sh.Top, Sh.Width, Sh.Height);
    end
〜〜省略〜〜

end;
```

図8

FastReportで確認



八木沼 幸一

株式会社ミガロ.

システム事業部 プロジェクト推進室

[Delphi/400] IBM i トリガー機能を活かしたセキュリティログ対応

略歴
1969 年 2 月 22 日生まれ
1991 年 獨協大学 経済学部卒業
2005 年 7 月 株式会社ミガロ. 入社
2005 年 7 月 システム事業部配属

現在の仕事内容
Delphi/400、SmartPad4i（JC/400）を利用したシステムの受託開発の内、RPG 開発をメインに担当し、HA ツールである MAXAVA の設定やサポート業務も担当している。

1.はじめに

トリガー機能というと、Microsoft SQL Server や Oracle Database などのオープン系データベースを思い浮かべる方も多いと思うが、IBM i でもトリガー機能を使用してさまざまな処理を実行できる。

更新ログを登録したり、明細登録した情報を自動集計したり、ログ出力時のタイムスタンプを保持したりと、使い方次第では非常に便利である。

さらに IBM i では更新トリガーだけではなく、レコードごとの読み取り（READ）トリガーも可能なので、セキュリティの観点からも有用である。

本稿では、トリガー機能の中でも多く用いられる、データの変更を監視するための更新ログファイルを出力する機能と Delphi/400 からの利用例について記述する。

2.トリガー機能の概要

2-1. トリガー機能とは

トリガー機能とは、IBM i のリレーショナルデータベースが備える一般的な機能の 1 つで、データベースファイルに、INSERT/UPDATE/DELETE/READ などのアクセスがあった際に、事前定義しておいたプログラムを自動的に実行できる。

2-2. トリガー機能のメリット

トリガー機能には、次のようなメリットがある。

・アプリケーション開発の効率化

トリガーはデータベースに保管されるため、アプリケーションごとにトリガーに相当するプログラミングを実装する必要がない。

・保守の簡素化

仕様や方針が変更になった場合でも、各アプリケーションプログラムではなく、該当するトリガープログラムを変更するだけで、共通で動作変更に対応できる。

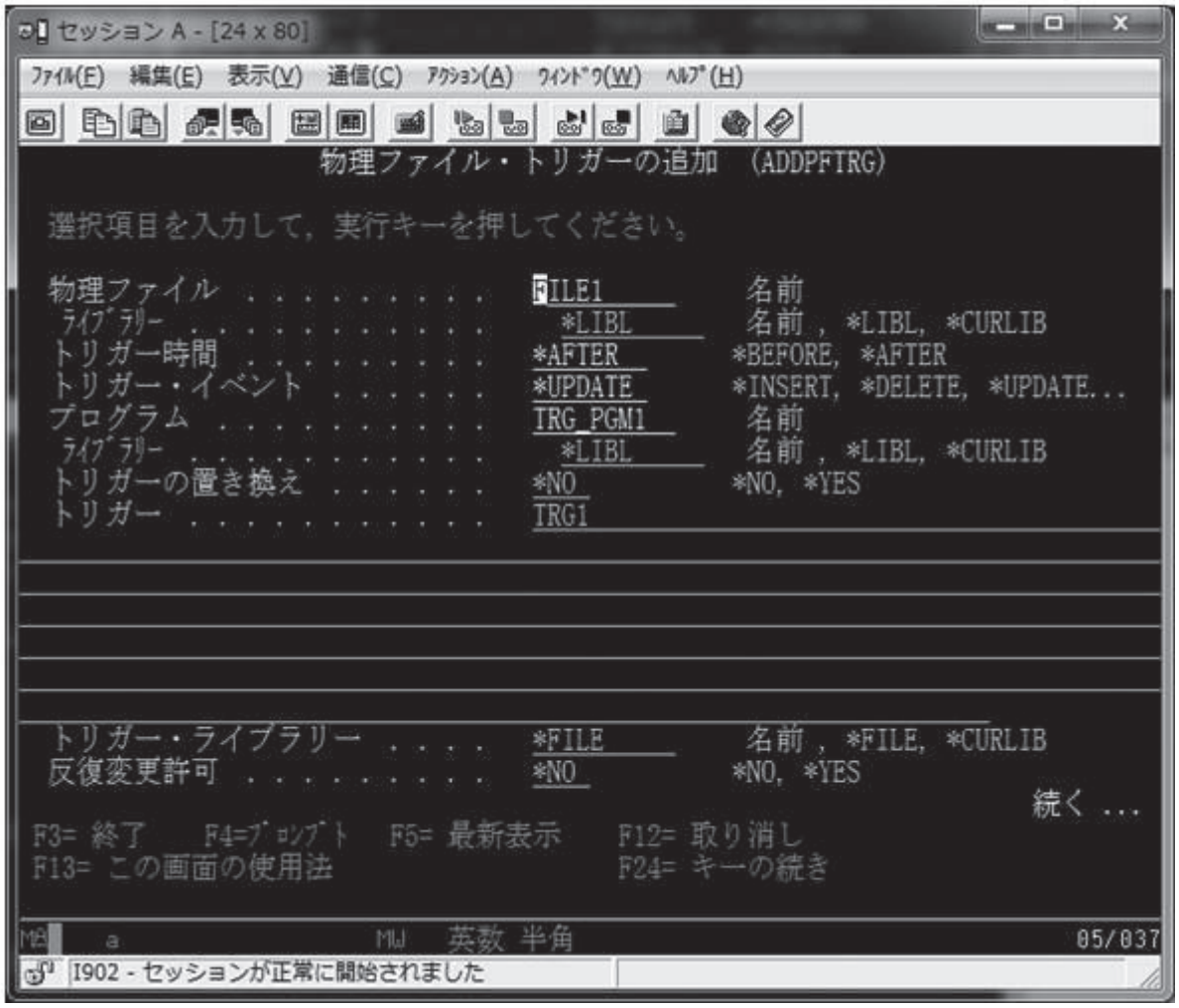
2-3. トリガー機能使用時の考慮点

トリガー機能は便利な機能ではあるが、使用時に注意すべき点がいくつかある。

トリガーを一度設定すると、原理的に毎回実行されてしまうので、パフォーマンスへの影響を考慮する必要がある。たとえばファイル更新時にトリガーを設定している場合（*READ 以外）、入力画面プログラムや照会画面プログラムなどの少量のデータ更新では影響はほとんどないが、バッチ処理など大量にデータを扱う場合は、トリガー機能を一時的に無効にするなどの措置を実施すべきである。

トリガーを変更する際は、コマンド（CHGPFTRG）で一時的にトリガーを無効化できる。通常バッチ処理で、複数のトリガーを設定済みのファイルを扱う場合、トリガーを一括で無効にする『無

図1



コマンド1

```
ADDPFTRG FILE(TRGLIB/SHAINP)
TRGTIME(*AFTER) TRGEVENT(*UPDATE)
PGM(TRGLIB/TRG_PGM1)
```

Delphi/400からの実行例

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  AS4001.RemoteCmd('ADDPFTRG FILE(TRGLIB/SHAINP) TRGTIME(*AFTER) TRGEVENT(*UPDATE) PGM(TRGLIB/TRG_PGM1)');
end;
```

効化 CL』と、トリガーを一括で有効にする『有効化 CL』を作成しておき、バッチ処理の先頭で『無効化 CL』を起動して無効化し、バッチ処理本体が終了したあとに『有効化 CL』を起動して有効化するなどの工夫が必要になる場合もある。これは、Delphi/400 からデータを操作する場合も同様である。

2-4. トリガーのタイミング

トリガーの発行されるタイミングについては、対象ファイルに対してのトリガー設定方法で指定できる（詳しい設定方法は後述）。更新の前（Before）／後（After）が指定でき、更新前であれば何らかのチェック、更新後であればログファイルの登録などの用途に使われることが多い。

ちなみにトリガー機能は、Microsoft SQL Server では Before トリガーはなく After トリガーのみをサポートするなど、データベースの種類によって多少異なる。

2-5. その他の機能

セキュリティログを登録する目的でトリガー機能を使用する場合、ログファイルに対するトリガープログラムを一度作成し、対象ファイルに対してトリガーを設定してしまえば、そのあとは意識することなく、ログ出力を自動的に実行できる。

またファイルに対する更新処理だけではなく、レコードごとの READ が行われたタイミングでトリガーを起動することもできる。いわゆる、READ トリガーである。

これは、レコード単位にどのような情報を読み取ったかを把握できるので、監査目的にも使用できる。また、「ある特別のレコードだけをある特定のユーザーのみにしか見せない」といったセキュリティ目的にも使用できる。

Delphi/400 と連携してセキュリティログなどを出力する場合のメリットとして、通信回数の削減効果が期待できる。

Delphi 側のプログラムでセキュリティログの登録処理を行う場合、トランザクションファイルやマスタファイルのような本番データ 1 件の更新処理とログデータ 2 件（変更前後のイメージを持つ場合）の更新となるので、計 3 回の通信に分離される。しかしトリガー機能を使用すれば 1 回の通信のみで実行できるので、パフォーマンスの向上にも役立つ。

3.対象ファイルに対するトリガー設定

トリガー機能を使用する際には、対象となるファイルに対してトリガーを設定する必要がある。

トリガーの「追加」「除去」「有効／無効化」「定義の確認」は、IBM i のコマンドラインから各コマンド発行で実行できる。

トリガーを設定することで、RPG などのプログラムでファイルにアクセスした場合はもちろん、Delphi/400 から直接そのファイルにアクセスした場合もトリガーの効力が及ぶので非常に便利である。

物理ファイルにトリガーが追加されると、指定されたそのファイルの全メンバーおよび従属する論理ファイルが、トリガーの影響を受けることになる。

指定されたそのファイルにより、メンバーに対して変更操作が行なわれると、トリガープログラムが呼び出される。また物理ファイルに従属している論理ファイルに対して変更操作が行なわれた時にも、トリガープログラムが呼び出されるので注意が必要である。

3-1. トリガーの追加（ADDPFTRG）

対象ファイルに対するトリガーの追加は、「物理ファイルトリガーの追加（ADDPFTRG）」コマンドで設定できる。

1 つのデータベースファイルには、最大 300 までのトリガーを関連付けられ、それぞれの挿入、削除、または更新操作時の前後に複数のトリガーを呼び出せる。

またそれぞれの読み取り操作の場合は、その操作が行われたあとに複数のトリガーを呼び出せる。

それでは、具体的にトリガーの追加手順を解説する。【図 1】

(1) コマンド ADDPFTRG を入力

コマンド ADDPFTRG を入力し、F4 キーを実行する。

(2) 物理ファイル／ライブラリ パラメータの指定

トリガーを設定するファイル／ライブラリを指定する。

(3) トリガー時間（TRGTIME）パラメータの指定

*BEFORE および *AFTER のどちらか 1 つを指定する。

*BEFORE を指定した場合、トリガープログラムはファイルに対する変更操作の前に実行される。*AFTER を指定した場合、トリガープログラムはファイルに対する変更操作のあとに実行される。

(4) トリガーイベント（TRGEVENT）パラメータの指定

*INSERT、*DELETE、*UPDATE、*READ のうちの 1 つを指定する。

*INSERT を指定した場合、ファイルに対してのレコード挿入操作でトリガープログラムが実行される。*DELETE を指定した場合、ファイルに対してのレコード削除操作でトリガープログラムが実行される。*UPDATE を指定した場合、ファイルに対してのレコード更新操作でトリガープログラムが実行される。*READ を指定した場合、ファイルに対しての読み取り操作でトリガープログラムが実行される。

(5) プログラム（PGM）パラメータの指定

対象のファイルに対して操作が行われた際に、起動するプログラム名を設定する。

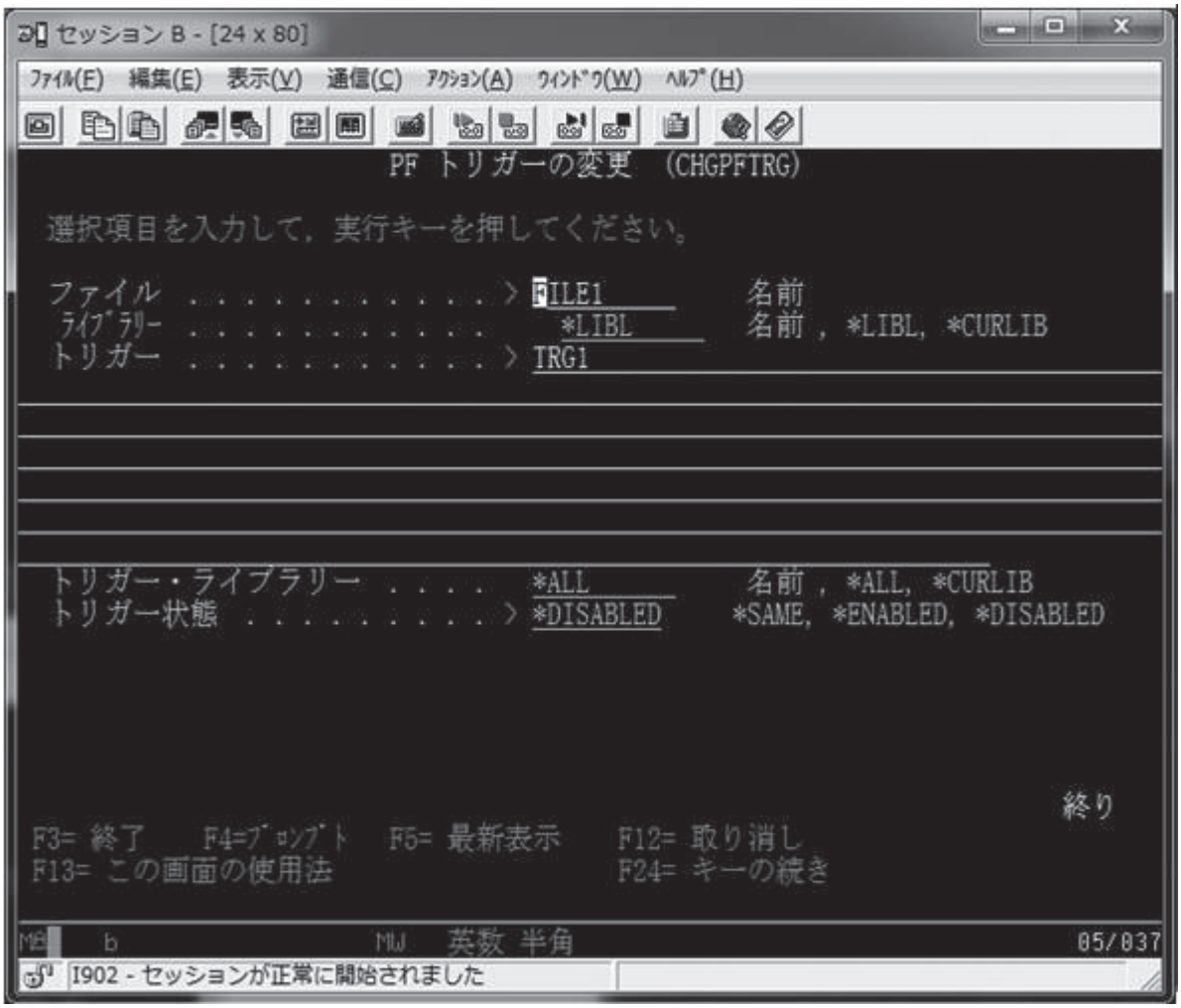
(6) トリガーの置き換え（RPLTRG）パラメータの指定

*NO および *YES のどちらか 1 つを指定する（デフォルト値は *NO である）。これは、同一のトリガー時間（TRGTIME）パラメータおよびトリガーイベント（TRGEVENT）パラメータを持つトリガーがすでに設定されていた場合に、今回のコマンドの内容で置き換えるかどうかを設定できる。

*NO を指定した場合は置き換えられない（既存のトリガー内容を優先する）。*YES を指定した場合は、既存のトリガーが置き換えられる。

(7) トリガー（TRG）パラメータの指

図2



コマンド2

```
CHGPFTRG FILE(TRGLIB/SHAINP) TRG(*ALL)
TRGLIB(*ALL) STATE(*DISABLED)
```

Delphi/400からの実行例

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  AS4001.RemoteCmd('CHGPFTRG FILE(TRGLIB/SHAINP) TRG(*ALL) TRGLIB(*ALL) STATE(*DISABLED)');
end;
```

定

*GEN および任意のトリガー名を指定する（デフォルト値は *GEN である）。

*GEN を指定した場合、システムがトリガー名を自動生成してくれるが、わかりやすい名前を付けておいたほうが、このあと便利である。

トリガーの有効／無効化（CHGPFTRG）やトリガーの除去（RMVPFTRG）を行う際に、すべて（*ALL）という選択肢も当然あるが、特定のトリガーだけに効力を効かせたいといった要望が出てくる可能性も充分考えられる。

その場合、トリガー名をコマンド内に明記する必要があるので、任意のトリガー名を付けておくようお勧めする。システムで自動生成されたトリガー名はDSPFD で調べる必要があり、名前も長くわかりにくい場合が多い。そこで任意の名前を付けておけば、調べる手間が省ける。

Delphi/400 からこれらを操作する場合は、【コマンド 1】の内容を TAS400 コンポーネントの RemoteCmd メソッドで実行するか、もしくは CL を作成して TCall400 から実行することでトリガーが追加できる。

3-2. トリガーの有効／無効化（CHGPFTRG）

対象となるファイルに設定済みのトリガー（1 つまたはすべて）の状態を変更する際には、CHGPFTRG コマンドを使用する。このコマンドは、一時的にトリガー機能を停止したい場合などによく使用される。

バッチ処理などの大量データを扱い、しかもその更新ログが必要ない場合は、一時的にトリガーを無効化し、バッチ処理後に再度有効化するといったように使用する。

Delphi/400 からトリガーの有効／無効を切り替える場合、有効化 CL および無効化 CL を作成しておき、それぞれの CL を呼び分けることで実現できる。

また CL を作成せず、Delphi/400 から直接上記の IBM i コマンドを発行するのも有効である。

それでは、具体的にトリガーの変更手順を解説する。【図 2】

(1) コマンド CHGPFTRG を入力

コマンド CHGPFTRG を入力し、F4 キーを実行する。

(2) 物理ファイル／ライブラリ パラメータの指定

トリガーを設定済みのファイル／ライブラリを指定する。

(3) トリガー（TRG）パラメータの指定

*ALL もしくは任意のトリガー名を指定する。

*ALL を指定した場合、対象ファイルに設定されているすべてのトリガーが変更される。任意のトリガー名を指定した場合は､そのトリガーのみが変更される。

(4) トリガー状態（STATE）パラメータの指定

*SAME、*ENABLED、*DISABLED のうちの 1 つを指定する。

*SAME を指定した場合、値は変更されない。*ENABLED を指定した場合、指定されたトリガーが有効になる。*DISABLED を指定した場合、指定されたトリガーが無効になる。

Delphi/400 から操作する場合は、前述した「追加」と同様に、【コマンド 2】の内容を実行するか、CL を作成して実行することでトリガーを変更できる。

有効化の場合は、STATE パラメータを *ENABLED にする。

3-3. トリガーの除去（RMVPFTRG）

トリガー設定済みのファイルからトリガーを除去する際には､「物理ファイルトリガーの除去（RMVPFTRG）」コマンドを使用できる。

除去するトリガーは、トリガー時間（TRGTIME）、トリガーイベント（TRGEVENT）、トリガー名（TRG）の指定で個別に除去できるが、それぞれの項目に *ALL を指定することで、一括除去も可能である。

それでは、具体的にトリガーの除去手順を解説する。【図 3】

(1) コマンド RMVPFTRG を入力

コマンド RMVPFTRG を入力し、F4 キーを実行する。

(2) 物理ファイル／ライブラリ パラメータの指定

トリガーを設定済みのファイル／ライブラリを指定する。

(3) トリガー時間（TRGTIME）、トリガーイベント（TRGEVENT）トリガー（TRG）の指定

上記パラメータはトリガーの追加（ADDPFTRG）の指定方法と同様だが、*ALL の指定も可能である。

Delphi/400 から操作する場合は、追加の場合と同様に、【コマンド 3】の内容を実行するか、CL を作成して実行することでトリガーを除去できる。

3-4. トリガー定義の確認（DSPFD）

対象ファイルに対するトリガーの状態や設定数などの現状内容を確認する場合は、ファイル記述表示（DSPFD）コマンドを実行する。

【図 4】は DSPFD コマンドで表示されるトリガー情報部分であるが、実際には DSPFD の内容の 4 ページ目以降に表示される｡これによりトリガーの名前、状態、イベント、時刻、プログラム名などの情報がすべて確認できる。

4.トリガーを活かした更新ログ作成

ここでは、社員マスタ（SHAINP）への更新内容を社員マスタログファイル（SHAINPR）に出力する方法を例に説明する。

まず、更新対象ファイルの社員マスタ（SHAINP）は､【DDS1】のとおりにする。

次に、ログを出力する社員マスタログファイル（SHAINPR）を作成する。このファイルには社員マスタ（SHAINP）と同一の項目を持ち（項目 ID も同じ）、さらに【DDS2】のようにログ情報を保持する項目も追加している。

この項目の追加により、社員マスタの変更情報プラス、ログの出力日時や更新ユーザー、トリガー事象などの情報も把握できる。

続いて、社員マスタ（SHAINP）に対してトリガー設定を行う｡今回は､データ更新（*UPDATE）後に、変更後の情報をログファイルに出力するよう設定す

図3

コマンド3

```
RMVPFTRG FILE(TRGLIB/SHAINP) TRGTIME(*ALL)
TRGEVENT(*ALL) TRG(*ALL)
```

Delphi/400からの実行例

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  AS4001.RemoteCmd('RMVPFTRG FILE(TRGLIB/SHAINP) TRGTIME(*ALL) TRGEVENT(*ALL) TRG(*ALL)');
end;
```

る。トリガープログラムは、このあとで作成するトリガープログラム（TRG_PGM1）を【図5】のように指定する。

最後に、TRG_PGM1をRPGで作成する例を示す（【ソース1】～【ソース5】参照）。

このRPGプログラムはIBM i側で設定しておくだけだが、対象ファイルをDelphi/400などのアプリケーションで操作した場合には、このトリガー機能が自動的に実行される。

【ソース1】～【ソース5】のRPGプログラムの概要は、次のとおりである。

F仕様書には、社員マスタログファイル（SHAINPR）のみを出力モードで定義する。社員マスタ（SHAINP）の変更前後の情報は、パラメータで受け取る。トリガープログラムのパラメータとしては、トリガーバッファとそのバッファの長さの2つを受け取る（受け取りパラメータの詳細は、【ソース1】～【ソース5】を参照）。

社員マスタ（SHAINP）の変更前後の情報をパラメータから取得するには、ポインターを使用してプログラム内の構造体に取り込む。

まず、変更前の情報を取り込む方法を解説する。

【ソース1】で構造体OLDRECを定義する際、EXTNAMEに社員マスタ（SHAINP）指定することで、構造体OLDRECのレイアウトと社員マスタのレイアウトを同期させる。

さらにBASEDで、基底ポインター名としてOLDRPを指定しておく。その上で、【ソース4】の81.00ステップで、%ADDR命令を使用してトリガーバッファ（PARM1）内の元レコード位置のポインターを基底ポインターのOLDRPにセットすることで、構造体OLDRECに変更前の社員マスタのレコード情報がセットされる。

同じ要領で、変更後の情報も取得できる。

そして【ソース4】【ソース5】で、ログ情報（ログ出力日時、端末ID、ユーザー名、トリガー事象）を付加し、社員マスタログファイル（SHAINPR）を出力（WRITE）する。

【ソース1】で、構造体OLDREC定義時にPREFIX（O）を指定しているので、構造体OLDRECの項目名の先頭には、”O”が付加されている。そのため、社員マスタログファイル（SHAINPR）の項目名とは一致していない。

逆に構造体NEWREC定義時にはPREFIXを指定していないので、社員マスタログファイル（SHAINPR）の項目名と一致している。その結果、社員マスタログファイル（SHAINPR）へのWRITE命令によって変更後の情報が出力される。

変更前の情報をログファイルに出力したい場合は、PREFIXの定義を逆転させることで実現できる。

5.まとめ

IBM iでシステムを構築する際にトリガー機能を取り入れていない環境も多いが、使い方次第ではDelphi/400側のプログラムも省略・共通化でき、開発面でも非常に有用な機能と言える。

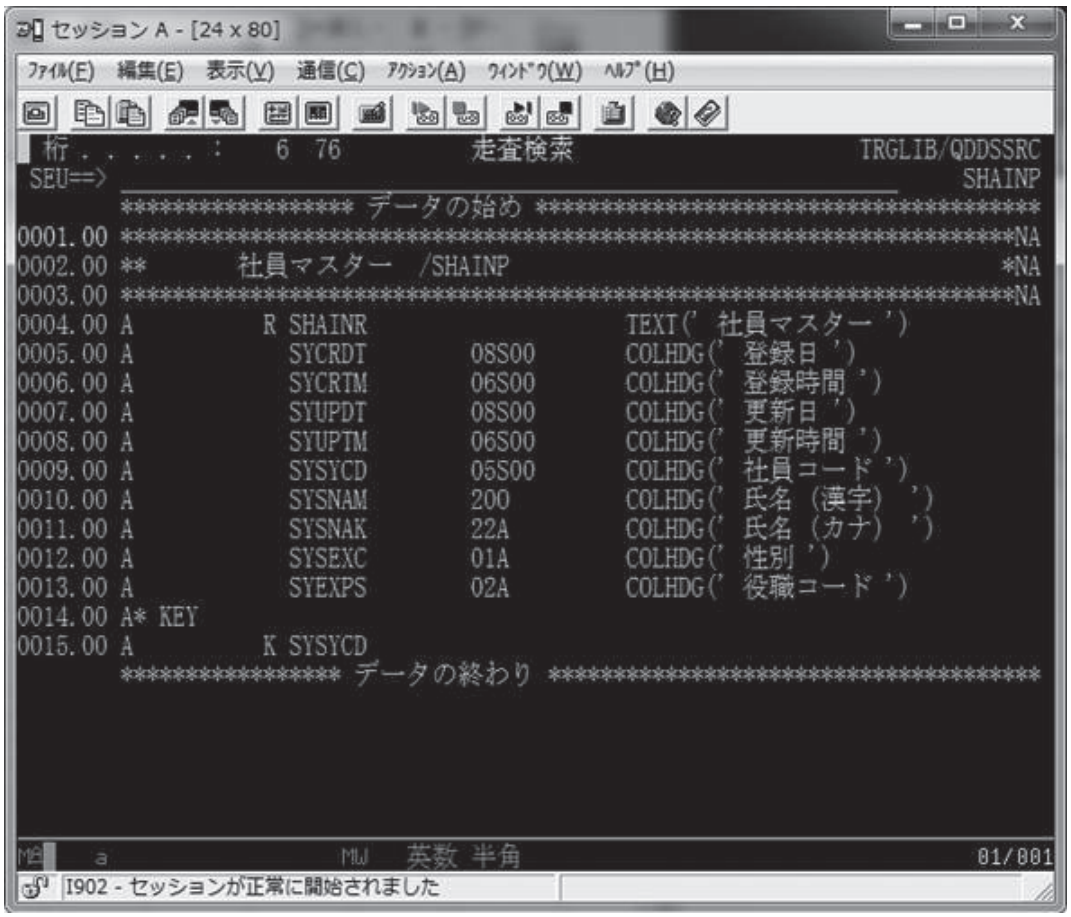
本稿ではセキュリティログを出力する機能を例にしたが、データ更新前のチェック機能や登録情報の自動集計機能などにも有効なので、IBM iの運用に取り入れてぜひ活用いただきたい。

M

図4


トリガー記述
トリガー名 . . . . . : TRG QSYS_TRIG_TRGLI
FILE1 000001
トリガー・ライブラリー . . . . . : TRGLIB
トリガーの状態 . . . . . : STATE *ENABLED
トリガー状況 . . . . . : *OPERATIVE
トリガー・イベント . . . . . : TRGEVENT *UPDATE
トリガー時刻 . . . . . : TRGTIME *AFTER
反復変更の許可 . . . . . : ALWREPCHG *NO
トリガー更新条件 . . . . . : TRGUPDCND *ALWAYS
プログラム名 . . . . . : PGM TRG_PGM1
ライブラリー . . . . . : TRGLIB
プログラムはスレッド・セーフ . . . . . : THDSAFE *UNKNOWN
マルチスレッド・ジョブの処置 . . . . . : MLTTHDACN *SYSVAL
トリガー・タイプ . . . . . : *SYS
トリガーの方向 . . . . . : *ROW
トリガー作成日／時刻 . . . . . : 17/08/18 15:48
トリガー更新桁の数 . . . . . : 0
メンバー記述
メンバー . . . . . : MBR FILE1
メンバー・レベル ID. . . . . : 1170818154323
メンバー作成日 . . . . . : 17/08/18
テキスト ' 記述 ' . . . . . : TEXT
メンバーの満了日 . . . . . : EXPDATE *NONE
メンバー・サイズ . . . . . : SIZE
初期レコード数 . . . . . : 10000
増分レコード数 . . . . . : 1000
続く ...
F3= 終了 F12= 取消し F19= 左 F20= 右 F24= キーの続き
MW 英数 半角
03/022
I902 - セッションが正常に開始されました

DDS1


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)
桁 . . . . . : 6 76 走査検索 TRGLIB/QDDSSRC
SEU==> SHAINP
********** データの始め **********
0001.00 **********NA
0002.00 ** 社員マスター /SHAINP *NA
0003.00 **********NA
0004.00 A R SHAINR TEXT(' 社員マスター ')
0005.00 A SYCRDT 08S00 COLHDG(' 登録日 ')
0006.00 A SYCRTM 06S00 COLHDG(' 登録時間 ')
0007.00 A SYUPDT 08S00 COLHDG(' 更新日 ')
0008.00 A SYUPTM 06S00 COLHDG(' 更新時間 ')
0009.00 A SYSYCD 05S00 COLHDG(' 社員コード ')
0010.00 A SYSNAM 200 COLHDG(' 氏名 (漢字) ')
0011.00 A SYSNAK 22A COLHDG(' 氏名 (カナ) ')
0012.00 A SYSEXC 01A COLHDG(' 性別 ')
0013.00 A SYEXPS 02A COLHDG(' 役職コード ')
0014.00 A* KEY
0015.00 A K SYSYCD
********** データの終わり **********
英数 半角
01/001
I902 - セッションが正常に開始されました

DDS2

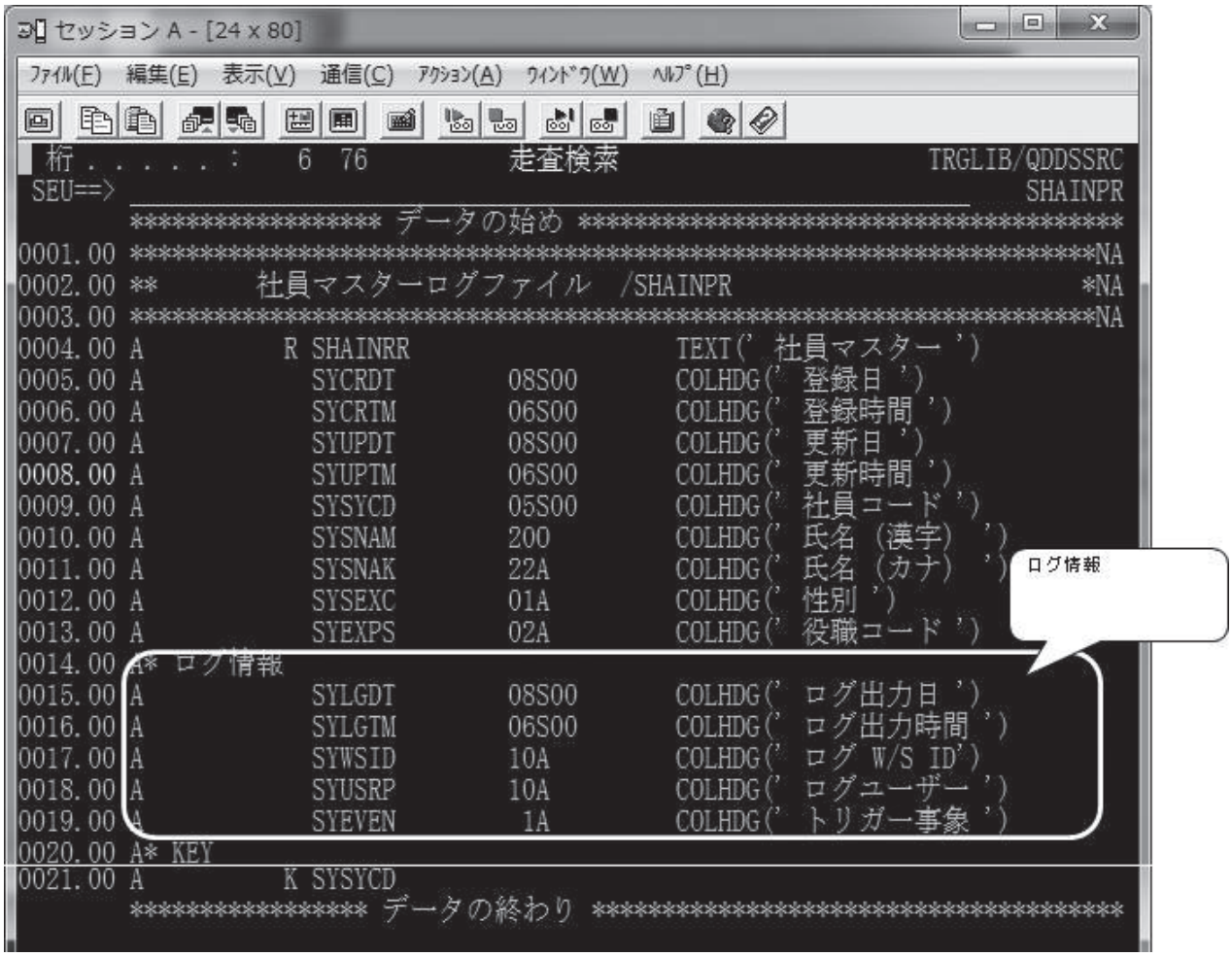
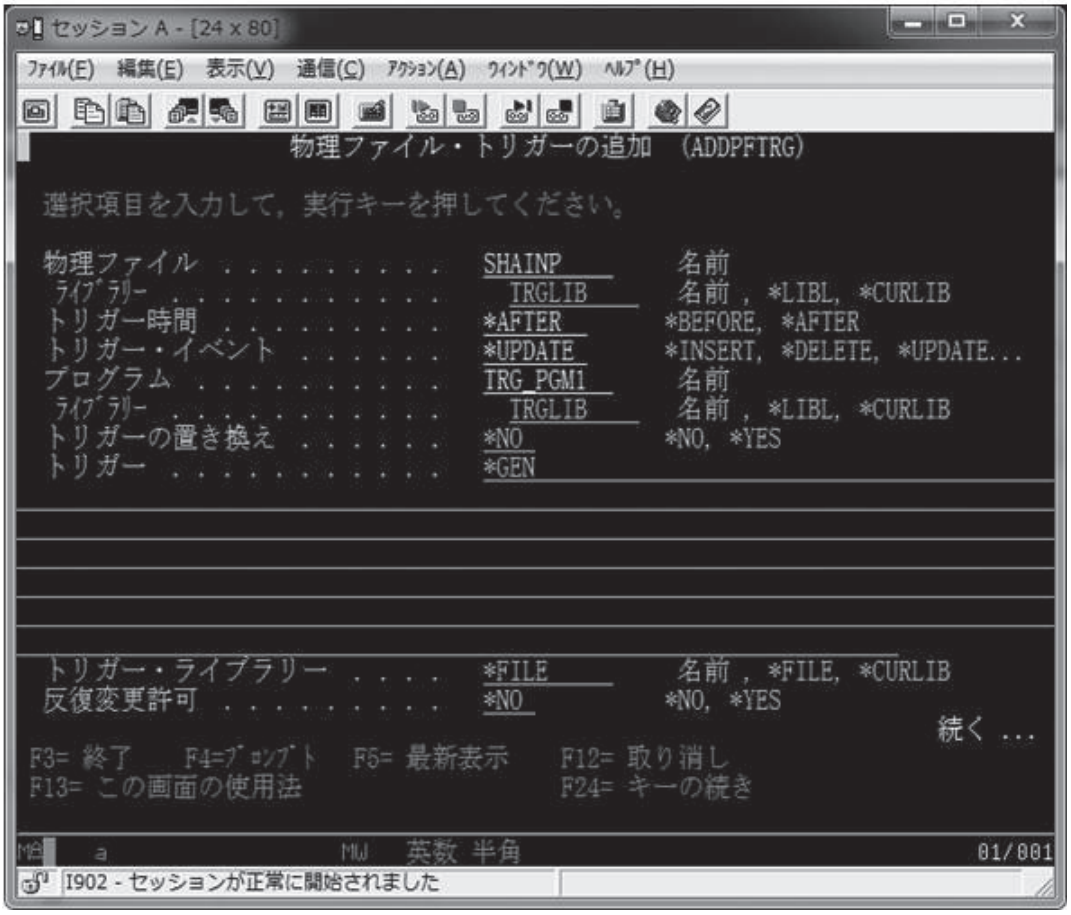
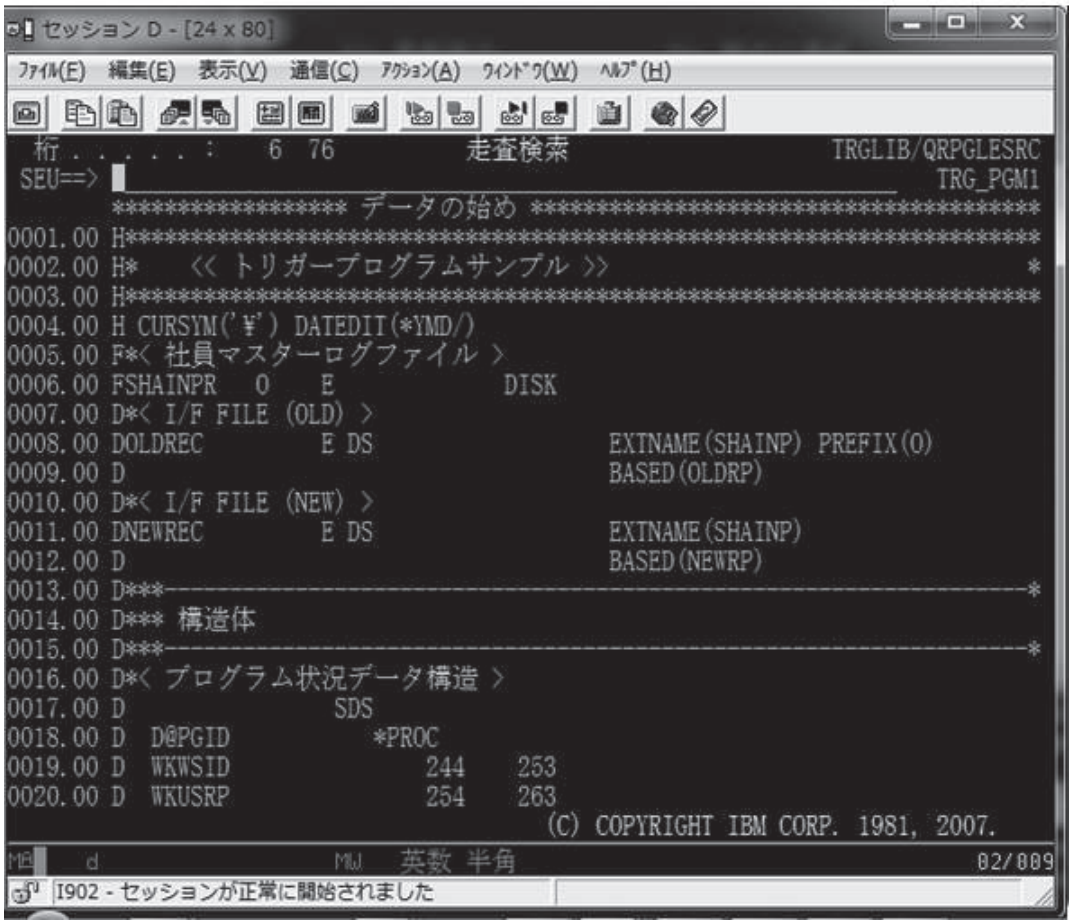

セッション A - [24 x 80]
ファイル(F) 編集(E) 表示(V) 通信(C) アクション(A) ウィンドウ(W) ヘルプ(H)
桁 . . . . . : 6 76 走査検索 TRGLIB/QDDSSRC
SEU==> SHAINPR
********** データの始め **********
0001.00 **********NA
0002.00 ** 社員マスターログファイル /SHAINPR *NA
0003.00 **********NA
0004.00 A R SHAINRR TEXT(' 社員マスター ')
0005.00 A SYCRDT 08S00 COLHDG(' 登録日 ')
0006.00 A SYCRTM 06S00 COLHDG(' 登録時間 ')
0007.00 A SYUPDT 08S00 COLHDG(' 更新日 ')
0008.00 A SYUPTM 06S00 COLHDG(' 更新時間 ')
0009.00 A SYSYCD 05S00 COLHDG(' 社員コード ')
0010.00 A SYSNAM 200 COLHDG(' 氏名 (漢字) ')
0011.00 A SYSNAK 22A COLHDG(' 氏名 (カナ) ')
0012.00 A SYSEXC 01A COLHDG(' 性別 ')
0013.00 A SYEXPS 02A COLHDG(' 役職コード ')
0014.00 A* ログ情報
0015.00 A SYLGDT 08S00 COLHDG(' ログ出力日 ')
0016.00 A SYLGTM 06S00 COLHDG(' ログ出力時間 ')
0017.00 A SYWSID 10A COLHDG(' ログ W/S ID')
0018.00 A SYUSRP 10A COLHDG(' ログユーザー ')
0019.00 A SYEVEN 1A COLHDG(' トリガー事象 ')
0020.00 A* KEY
0021.00 A K SYSYCD
********** データの終わり **********
ログ情報

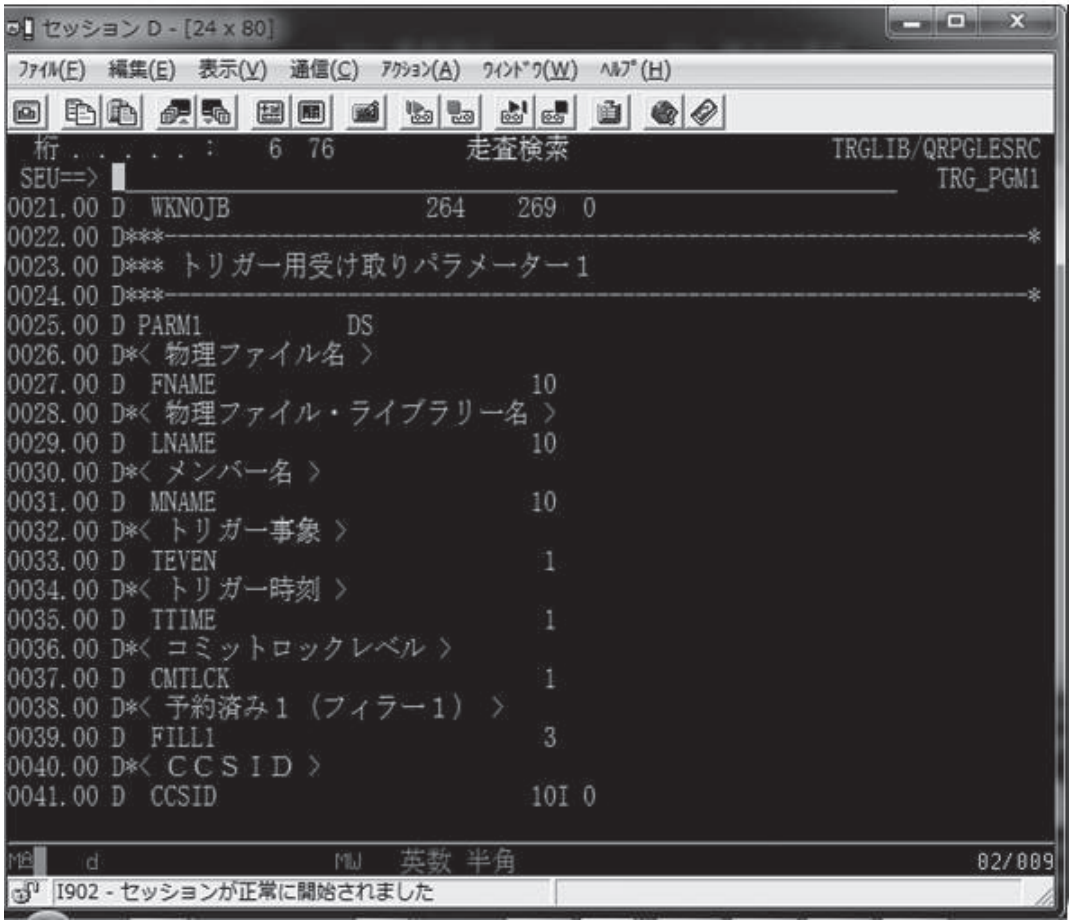
図5


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)
物理ファイル・トリガーの追加 (ADDPFTRG)
選択項目を入力して，実行キーを押してください。
物理ファイル . . . . . SHAINP 名前
ライブラリー . . . . . TRGLIB 名前 , *LIBL, *CURLIB
トリガー時間 . . . . . *AFTER *BEFORE, *AFTER
トリガー・イベント . . . . . *UPDATE *INSERT, *DELETE, *UPDATE...
プログラム . . . . . TRG_PGM1 名前
ライブラリー . . . . . TRGLIB 名前 , *LIBL, *CURLIB
トリガーの置き換え . . . . . *NO *NO, *YES
トリガー . . . . . *GEN
トリガー・ライブラリー . . . . *FILE 名前 , *FILE, *CURLIB
反復変更許可 . . . . . *NO *NO, *YES
続く ...
F3= 終了 F4=プロンプト F5= 最新表示 F12= 取り消し
F13= この画面の使用法 F24= キーの続き
MA a MW 英数 半角 01/001
I902 - セッションが正常に開始されました

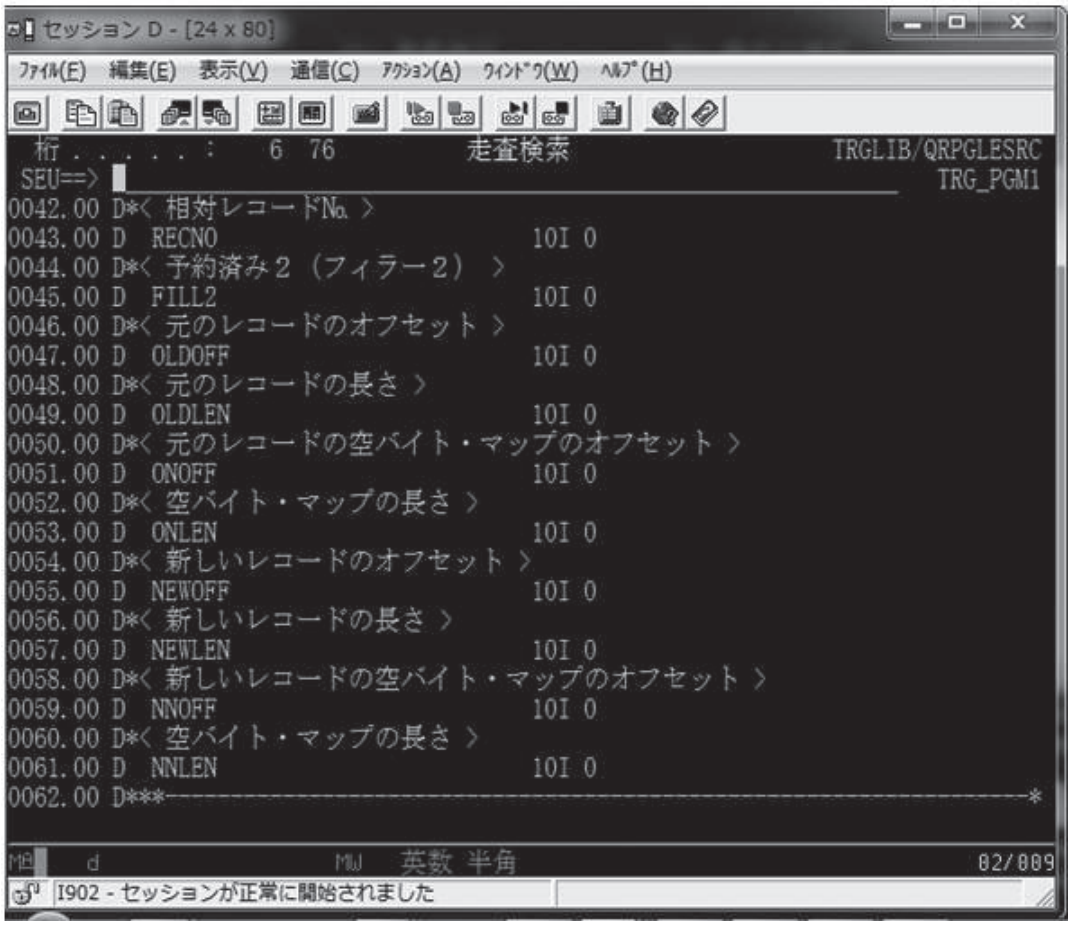
ソース1


セッション D - [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)
桁 . . . . . : 6 76 走査検索 TRGLIB/QRPGLESRC
SEU==> TRG_PGM1
0001.00 H********** データの始め **********
0002.00 H* << トリガープログラムサンプル >>
0003.00 H**********
0004.00 H CURSYM('¥') DATEDIT(*YMD/)
0005.00 F*< 社員マスターログファイル >
0006.00 FSHAINPR O E DISK
0007.00 D*< I/F FILE (OLD) >
0008.00 DOLDREC E DS EXTNAME(SHAINP) PREFIX(O)
0009.00 D BASED(OLDRP)
0010.00 D*< I/F FILE (NEW) >
0011.00 DNEWREC E DS EXTNAME(SHAINP)
0012.00 D BASED(NEWRP)
0013.00 D***----------*
0014.00 D*** 構造体
0015.00 D***----------*
0016.00 D*< プログラム状況データ構造 >
0017.00 D SDS
0018.00 D D@PGID *PROC
0019.00 D WKWSID 244 253
0020.00 D WKUSRP 254 263
(C) COPYRIGHT IBM CORP. 1981, 2007.
MA d MW 英数 半角 02/009
I902 - セッションが正常に開始されました

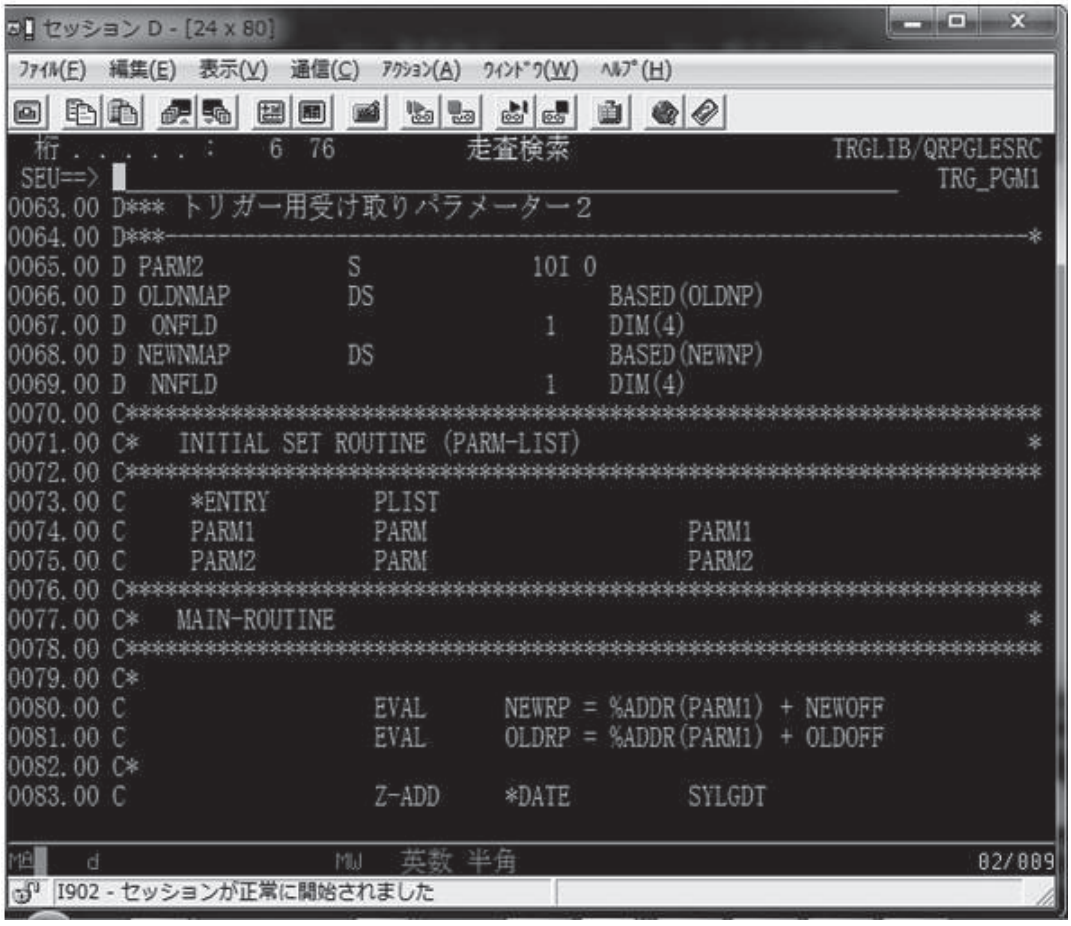
ソース2


セッション D - [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)
桁 . . . . . : 6 76
走査検索
TRGLIB/QRPGLESRC
SEU==>
TRG_PGM1
0021.00 D WKNOJB 264 269 0
0022.00 D***
0023.00 D*** トリガー用受け取りパラメーター1
0024.00 D***
0025.00 D PARM1 DS
0026.00 D*< 物理ファイル名 >
0027.00 D FNAME 10
0028.00 D*< 物理ファイル・ライブラリー名 >
0029.00 D LNAME 10
0030.00 D*< メンバー名 >
0031.00 D MNAME 10
0032.00 D*< トリガー事象 >
0033.00 D TEVEN 1
0034.00 D*< トリガー時刻 >
0035.00 D TTIME 1
0036.00 D*< コミットロックレベル >
0037.00 D CMTLCK 1
0038.00 D*< 予約済み1 (フィラー1) >
0039.00 D FILL1 3
0040.00 D*< CCSID >
0041.00 D CCSID 10I 0
MW 英数 半角
02/009
I902 - セッションが正常に開始されました

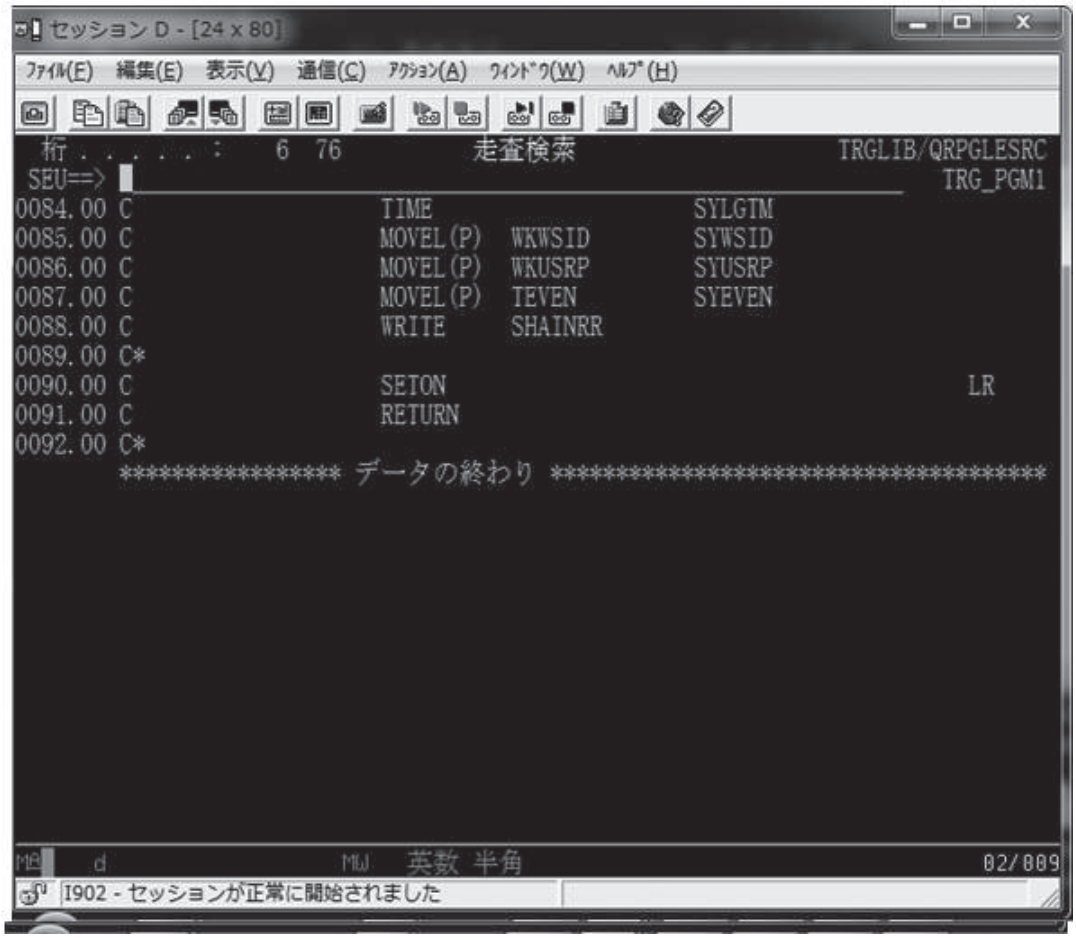
ソース3


セッション D - [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)
桁 . . . . . : 6 76
走査検索
TRGLIB/QRPGLESRC
SEU==>
TRG_PGM1
0042.00 D*< 相対レコードNo. >
0043.00 D RECNO 10I 0
0044.00 D*< 予約済み2 (フィラー2) >
0045.00 D FILL2 10I 0
0046.00 D*< 元のレコードのオフセット >
0047.00 D OLDOFF 10I 0
0048.00 D*< 元のレコードの長さ >
0049.00 D OLDLEN 10I 0
0050.00 D*< 元のレコードの空バイト・マップのオフセット >
0051.00 D ONOFF 10I 0
0052.00 D*< 空バイト・マップの長さ >
0053.00 D ONLEN 10I 0
0054.00 D*< 新しいレコードのオフセット >
0055.00 D NEWOFF 10I 0
0056.00 D*< 新しいレコードの長さ >
0057.00 D NEWLEN 10I 0
0058.00 D*< 新しいレコードの空バイト・マップのオフセット >
0059.00 D NNOFF 10I 0
0060.00 D*< 空バイト・マップの長さ >
0061.00 D NNLEN 10I 0
0062.00 D***
MW 英数 半角
02/009
I902 - セッションが正常に開始されました

ソース4


セッション D - [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)
桁 . . . . . : 6 76 走査検索 TRGLIB/QRPGLESRC
SEU==> TRG_PGM1
0063.00 D*** トリガー用受け取りパラメーター2
0064.00 D***
0065.00 D PARM2 S 10I 0
0066.00 D OLDNMAP DS BASED(OLDNP)
0067.00 D ONFLD 1 DIM(4)
0068.00 D NEWNMAP DS BASED(NEWNP)
0069.00 D NNFLD 1 DIM(4)
0070.00 C*
0071.00 C* INITIAL SET ROUTINE (PARM-LIST)
0072.00 C*
0073.00 C *ENTRY PLIST
0074.00 C PARM1 PARM PARM1
0075.00 C PARM2 PARM PARM2
0076.00 C*
0077.00 C* MAIN-ROUTINE
0078.00 C*
0079.00 C*
0080.00 C EVAL NEWRP = %ADDR(PARM1) + NEWOFF
0081.00 C EVAL OLDRP = %ADDR(PARM1) + OLDOFF
0082.00 C*
0083.00 C Z-ADD *DATE SYLGDT
MA d MW 英数 半角 02/009
I902 - セッションが正常に開始されました

ソース5


セッション D - [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)
桁 . . . . . : 6 76 走査検索 TRGLIB/QRPGLESRC
SEU==> TRG_PGM1
0084.00 C TIME SYLGTM
0085.00 C MOVEL(P) WKWSID SYWSID
0086.00 C MOVEL(P) WKUSRP SYUSRP
0087.00 C MOVEL(P) TEVEN SYEVEN
0088.00 C WRITE SHAINRR
0089.00 C*
0090.00 C SETON LR
0091.00 C RETURN
0092.00 C*
********** データの終わり **********
MA d MW 英数 半角 02/009
I902 - セッションが正常に開始されました

吉原 泰介

株式会社ミガロ.

RAD事業部 技術支援課

[Delphi/400] アプリケーションテザリングを利用したPC & モバイルアプリケーション連携

略歴
1978 年 3 月 26 日生まれ
2001 年 龍谷大学 法学部卒業
2005 年 7 月 株式会社ミガロ. 入社
2005 年 7 月 システム事業部配属
2007 年 4 月 RAD 事業部配属

現在の仕事内容
Delphi/400 を中心に製品試験および月 100 件に及ぶ問い合わせサポートやセミナー講師などを担当している。

1.はじめに

スマートフォンやタブレットなどのモバイル端末の普及が始まって数年が経ち、現在では個人はもちろん、企業内での活用も一般的になってきた。こうしたモバイル端末は、名前のとおり、持ち歩いて使用できる特性を活かした場面で利用されることが多い。

企業でモバイル端末が使われる主な用途としては、ブラウザ、電話、カメラ（写真撮影）などが挙げられる。

これらの機能は、モバイル端末での特殊な機能というよりも、基本的には既存のデバイス（たとえばPC やデジタルカメラ）で使用していた機能をモバイル端末で代用していることがわかる。業務によってはタブレットがPC 自体の代用となり、置き換わっていることも多い。

つまりモバイル端末のアプリケーションでは、特殊な機能だけが求められるわけではなく、従来の業務で使っている機能をモバイル端末で利用できるだけでも、大きな価値や利便性が得られる。

本稿では、モバイルアプリケーション開発を難しくとらえず、従来の機能を代替えするという観点で「アプリケーションテザリング」という技術の活用方法について考察する。

2.アプリケーションテザリング技術

2-1. アプリケーションテザリングとは

最近スマートフォンによく搭載されている「テザリング」という機能がある。これはスマートフォンがWi-Fi ルータの役割を果たし、スマートフォン経由でPC などのインターネット通信を共有する機能である。

Delphi/400 では XE7 より、「アプリケーションテザリング」という技術が実装されている。これは前述したインターネット共有とは若干異なるが、同じネットワーク（あるいは Bluetooth 接続された）アプリケーション同士でデータや処理を共有する機能である。アップテザリングと呼ばれることもある。

このアプリケーションテザリングは、モバイルアプリケーション開発のFireMonkey でも、PC アプリケーション開発の VCL でも、同じコンポーネントを使って実装可能である。

つまり従来のPC アプリケーションと新規開発したモバイルアプリケーションを容易に連携できるわけだ。たとえばモバイルアプリケーションからPC アプリケーションに処理結果を連携して、PC アプリケーション側でIBM i に更新するといった仕組みも構築できる。

本稿ではこうしたアプリケーションテザリングの技術に着目し､PCアプリケーションとモバイルアプリケーションを連携する実装方法をまとめている。なお、開発環境の対象バージョンはDelphi/400 XE7 以降を想定している。

2-2. アプリケーションテザリングでモバイル端末を活用

アプリケーションテザリングでモバイルアプリケーションを活用する実装例として、従来のシステムでPC と連携して

図1

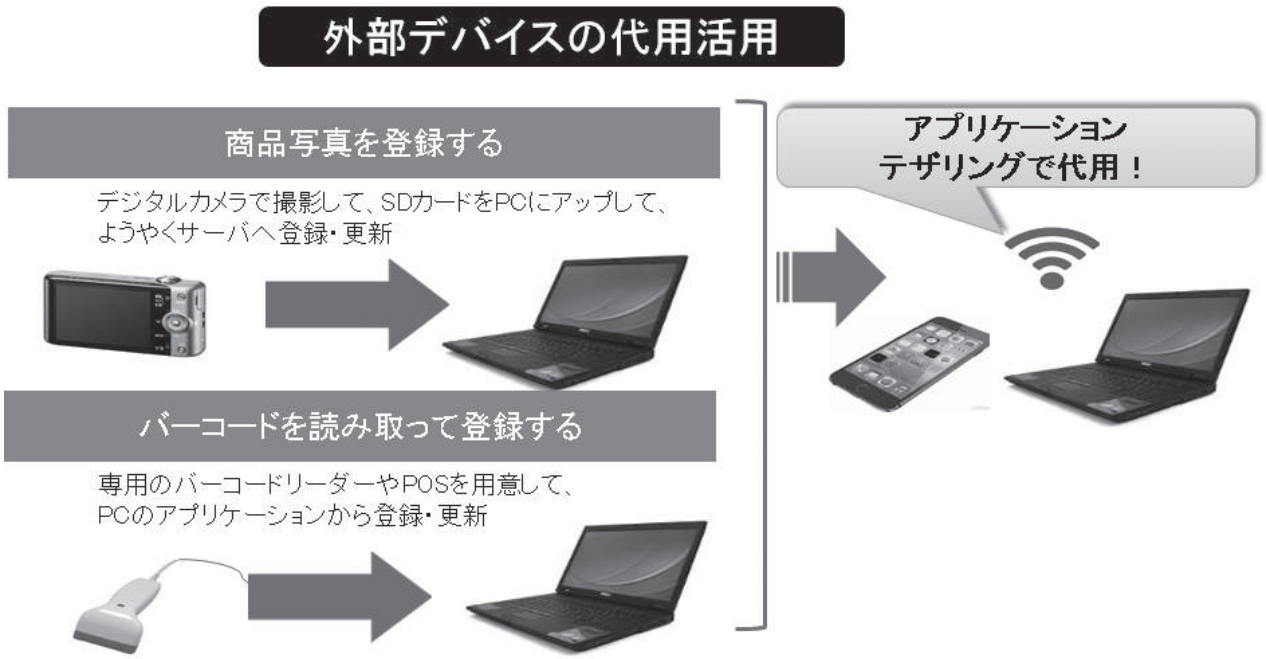

外部デバイスの代用活用
商品写真を登録する
デジタルカメラで撮影して、SDカードをPCにアップして、
ようやくサーバへ登録・更新
バーコードを読み取って登録する
専用のバーコードリーダーやPOSを用意して、
PCのアプリケーションから登録・更新
アプリケーション
テザリングで代用！

図2


専用コンポーネント
TTetheringManagerコンポーネント
TetheringManager1
ネットワーク上でテザリング
するための接続等の管理
TTetheringAppProfileコンポーネント
TetheringAppProfile1
テザリングで接続した
アプリケーション間で共有する
リソースの制御

いる外部デバイスを検討した。モバイル端末を活かせるものとして、次の2機能を題材とする。【図1】

①デジタルカメラで商品写真を撮り、画像データとしてシステムに取り込む機能
②バーコードリーダーやPOSでバーコードやQRコードをシステムに取り込む機能

いずれの機能も業務で使われることが多く、別デバイスで運用されているケースもよく見られる。また機能的にもモバイルアプリケーションに適しており、実装が容易である。

3.アプリケーションテザリングを活用した連携開発

3-1. アプリケーションテザリング用コンポーネント

アプリケーションテザリング機能を使用するためのコンポーネントを確認する。専用のコンポーネントとして、TTetheringManagerとTTetheringAppProfileの2つのコンポーネントが用意されている。それぞれの役割は、次のとおりである。【図2】

TTetheringManagerコンポーネント

同一ネットワークあるいはBluetoothの通信上で、アプリケーション同士の接続を管理する

TTetheringAppProfileコンポーネント

テザリングで接続したアプリケーション間で共有するリソースを制御する

この2つのコンポーネントは基本セットで使用し、TTetheringManagerで接続してTTetheringAppProfileで共有リソースを送受信する。たとえばコードや文字列、画像などのファイルをStream形式で扱うこともできる。【図3】

本稿では、モバイルアプリケーションで写真撮影やバーコード読み取りの機能を実装し、それをPCアプリケーションで受信して利用するアプリケーションを作成する。【図4】

3-2. モバイルアプリケーションの開発

まず、モバイル端末側のアプリケーションを作成する。

使用コンポーネントは写真撮影用にTActionList、TImageコンポーネント、バーコード読み取り用にTTMSFMXZBarReaderコンポーネント、それぞれの実行用にTButtonを2つ配置する。またアプリケーションテザリング用には、前述したTTetheringManagerコンポーネントとTTetheringAppProfileを配置しておく。【図5】

ここで使用するTTMSFMXZBarReaderコンポーネントは標準コンポーネントではないが、TMSSoftware社のHPから無償で使用できるiOS用バーコード読み取りコンポーネントである（http://www.tmssoftware.com/）。Androidを使う場合は、拡張したフリーソースのTTKRBarCodeScannerが使いやすい。

次にコンポーネントを設定する。テザリングの設定は、TTetheringAppProfileのGroupプロパティに通信するアプリケーションで共通の値を設定しておく。グループ名のようなものだと考えるとわかりやすい。【図6】

そしてアプリケーション間で共有するデータについては、Resourcesプロパティにアイテムを、次のように2つ追加して用意する。

1つ目は写真転送用で、アイテムのNameプロパティには"Camera"、ResTypeプロパティには写真（画像ファイル）を扱うために"Stream"を設定する。

2つ目はバーコード転送用で、アイテムのNameプロパティには"Barcode"、ResTypeプロパティにはバーコード値を扱うために"Data"を設定する。いずれもKindプロパティはデフォルト値"Shared"のままにしておく。【図6】

設定が完了したら、配置したコンポーネントにプログラム動作をコーディングしていく。撮影用のボタンには、カメラ操作の標準アクションであるTTakePhotoFromCameraActionを割り当て、OnDidFinishTakingイベントを作成する。【図7】

写真撮影終了時に発生するOnDidFinishTakingイベントには、【ソース1】のようなコードを記述する。

撮影した写真（画像ファイル）をTTetheringAppProfileのアイテムに代入することで、テザリングされた別のアプリケーションでアイテム情報を共有できる。

バーコード読み取り用のボタンには、OnClickイベントでTTMSFMXZBarReaderのShowメソッドを実行することで、バーコードスキャン画面を起動。バーコードスキャンされた際にOnGetResultイベントで取得したバーコード値を、TTetheringAppProfileのアイテムに代入するだけで実装できる。【ソース2】

最後に、フォームのOnCreateイベントでTTetheringManagerを接続するようにコーディングしておくことで、起動直後にPCアプリケーションとテザリング通信状態になる。

これだけの開発でモバイル端末の写真撮影とバーコード読み取りを実行し、PCへデータを転送（共有）するアプリケーションが完成である。

3-3. PCアプリケーションの開発（拡張実装）

次に、PC側のアプリケーションを実装する。PCアプリケーションは、写真やコードを扱う既存アプリケーションがすでに導入されていることを想定する。本稿では、飲料水の商品一覧・明細を扱う単純なアプリケーション例を題材に拡張実装している。

まずアプリケーションテザリング用として、モバイルアプリケーションと同様にTTetheringManagerコンポーネントとTTetheringAppProfileを追加で配置する。【図8】

プロパティの設定もモバイルアプリケーションと同様ではあるが、TTetheringAppProfileのアイテムでKindプロパティには"Mirror"を設定する。これは送られてきた共有アイテムを受信する側の設定である。

そして、それぞれのアイテムにはOnResourceReceivedのイベントがあるので、ここに転送されてきた情報を受信した際の処理をコーディングする。写真の受信については、【ソース3】のように、送られてきたStreamをLoadFromStreamに読み込むことで画面のTImageへ写真画像を反映できる。

図3

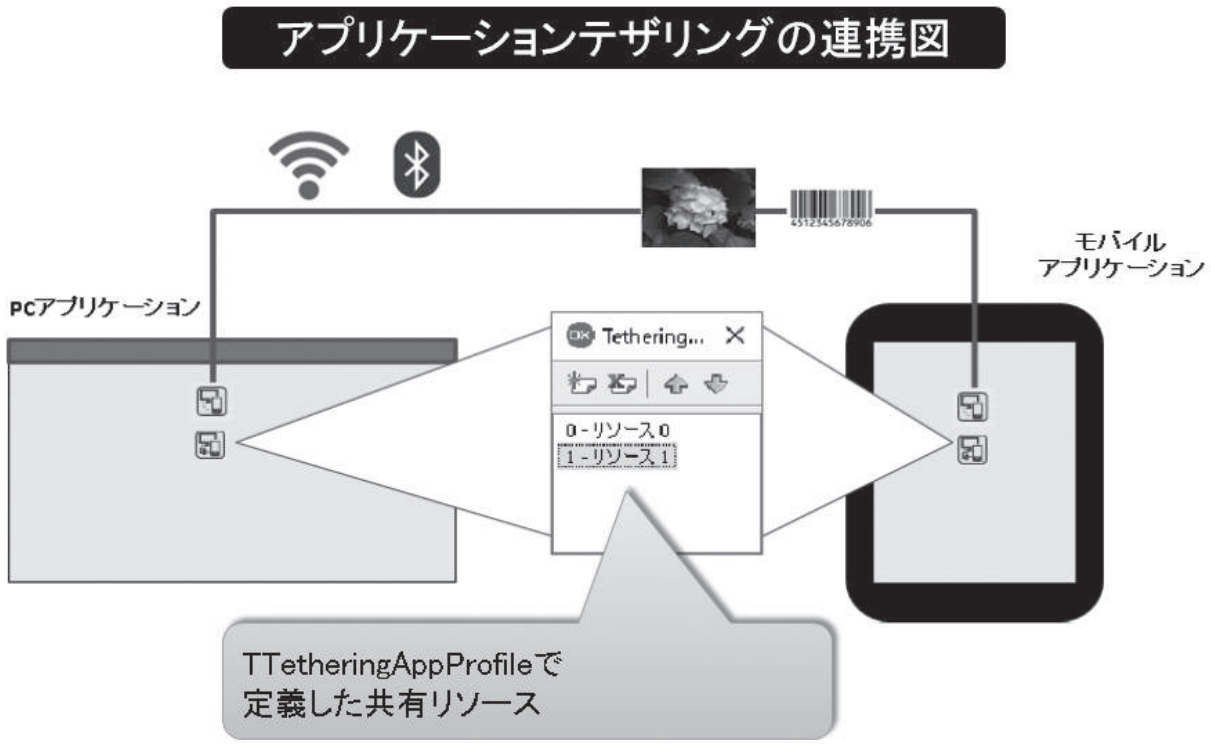
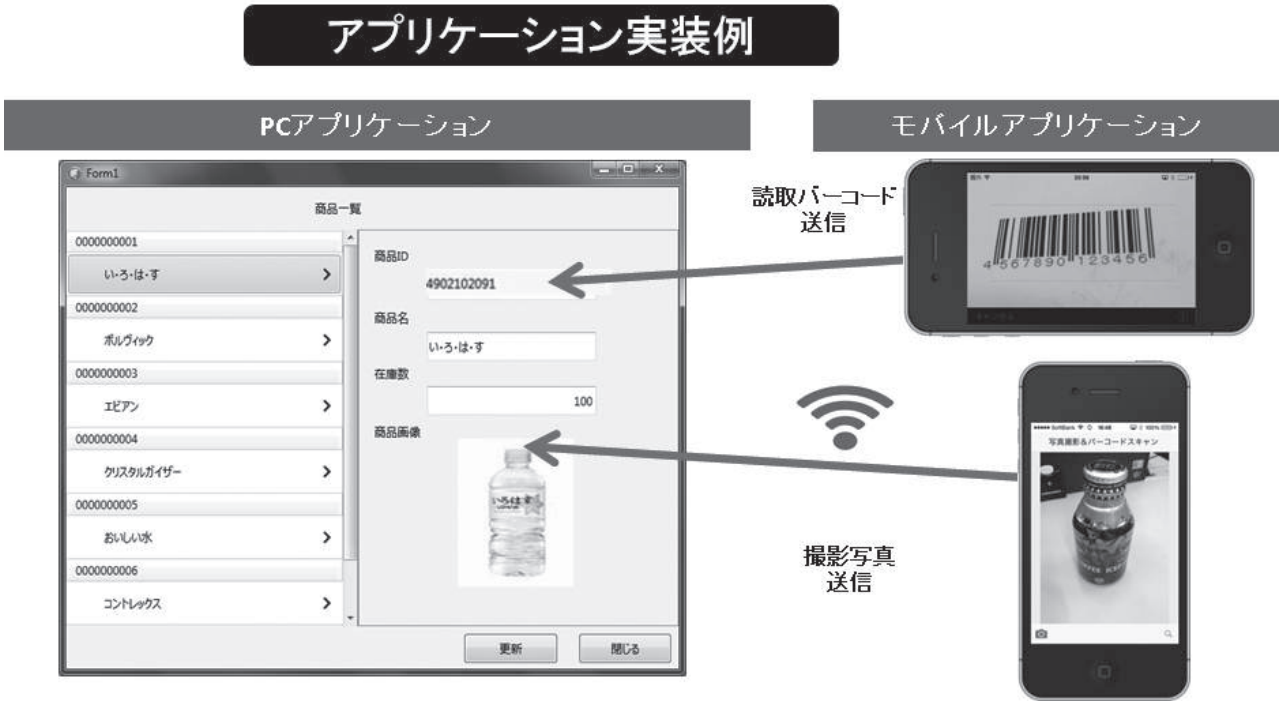

アプリケーションテザリングの連携図
PCアプリケーション
モバイル
アプリケーション
Tethering...
0 - リソース 0
1 - リソース 1
TTetheringAppProfileで
定義した共有リソース

図4


アプリケーション実装例
PCアプリケーション
モバイルアプリケーション
Form1
商品一覧
0000000001
いろはす
0000000002
ボルヴィック
0000000003
エビアン
0000000004
クリスタルガイザー
0000000005
おいしい水
0000000006
コントレックス
商品ID
4902102091
商品名
いろはす
在庫数
100
商品画像
更新
閉じる
読取バーコード
送信
撮影写真
送信

バーコードの受信については、【ソース 4】のように、画面項目（ここではTEdit）に送られてきたコードを設定するだけである。これで、PC アプリケーション側のテザリング実装が完成した。

最後にコンパイルした PC アプリケーションとモバイルアプリケーションを同じネットワーク上、あるいは Bluetooth で通信が可能な環境で起動して動作確認する。

モバイルアプリケーションで写真を撮れば、PC 側に写真画像が表示され、バーコードをスキャンすれば、PC 側に読み取ったコードが自動入力される（【図 4】の実装例が完成する）。

これにより、デジタルカメラから画像を手動でアップしたり、専用端末でスキャンしていたバーコードなどの外部デバイス作業をモバイル端末と簡易なアプリだけで代用できる。

3-4. アプリケーションテザリングの応用活用

補足として、いくつかのアプリケーションテザリングの応用テクニックについても検証した。

アプリケーションテザリングでは、TTetheringAppProfile の Resources プロパティのアイテムを共有するだけでなく、アクション（処理）も共有できる。これは Resources プロパティと同様に、TTetheringAppProfile の Actions プロパティのアイテムとして設定する。【図 9】

Actions にアイテムを追加すると、アイテムの Action プロパティのリストに割り当てられる。この Action を共有しておけば、PC 側で設定している Action をモバイルアプリケーション側から実行可能になる。

たとえば、PC 上で PowerPoint を OLE で操作する Action 処理を作成しておき、モバイル端末でプレゼンテーション資料をリモコン操作するアプリケーションも簡単に実現できる。実際にそうしたリモコンとして使うモバイルアプリケーションは、市販製品として販売されている。

もちろん、このアプリケーションテザリング技術は PC アプリケーションとモバイルアプリケーションの連携だけではなく、PC アプリケーション同士、あるいはモバイルアプリケーション同士でも活用できる。同一システムでログインしている端末やデバイスのステータス情報を共有したり、アプリケーション間でのチャットなどのグループメッセージ機能としても幅広く応用が考えられる。

ただしこの技術は、冒頭で説明したとおり、同じネットワークあるいは Bluetooth などでの通信が前提である。たとえば、ネットワークに接続していない拠点や事業所間などでは活用できない。そのため、実際にアプリケーション同士で通信していなければ、データやアクションも共有できない。

本稿のプログラムでは、起動時にテザリング接続されることを前提としているが、接続されているかを可視化できるよう工夫しておいたほうが、使い勝手がよい。

たとえば、”接続中”と表示する TLabel を配置しておき、Visible プロパティを False にしておく。そして TTimer コンポーネントを配置し、OnTimer イベントを使って【ソース 5】のようにアプリケーションテザリングの接続カウントを監視して、Visible プロパティを True に設定する仕組みで実装できる。

接続のカウントは、TTethering Manager の RemoteProfiles.Count プロパティでリアルタイムな接続数を取得できる。もちろん TTimer のような常時監視型ではなく、ボタンなどで明示的に実行してチェックする仕組みでもよい。こうした実装を加えると、実行時に PC アプリケーションにモバイルアプリケーションが接続されると、”接続中”の画面表示になるので、操作時にわかりやすい。

4.おわりに

本稿ではアプリケーションテザリングという技術を検証し、既存の外部デバイスを使った機能をモバイルアプリケーションで簡単に代用できることを紹介した。単純に代用が可能になっただけでなく、どちらも Delphi/400 のアプリケーションで作成できる点が大きなメリットである。これにより、自社業務に合わせてアプリケーション側を柔軟に対応していける。

またアプリケーションテザリングを使ったモバイルアプリケーションであれば、DataSnap サーバーのような中間サーバーを構築しなくても、【図 10】のように PC アプリケーションがサーバーの代用となるので、簡単に実装・運用できる。DataSnap のような高度なサーバー機能はないが、モバイルアプリケーション開発の入門としても手軽に試せて便利である。

モバイル端末の企業導入は非常に増えてきたが、アプリケーションまで自社開発している企業はまだまだ少ない。

しかし本稿のプログラム例を一読いただければわかるが、これまでの PC アプリケーション開発と同じように、モバイルアプリケーションも Delphi 言語で開発できる。

日本では業務アプリケーションを自社開発する企業が多く、今後は PC アプリケーション同様にモバイルアプリケーションも自社開発の需要が増えてくると予想される。

そうしたシステム開発の中で、この Delphi/400 のアプリケーションテザリングやモバイルアプリケーション開発技術が役立つように、今後も技術検証や情報発信に努めていきたい。

M

図5

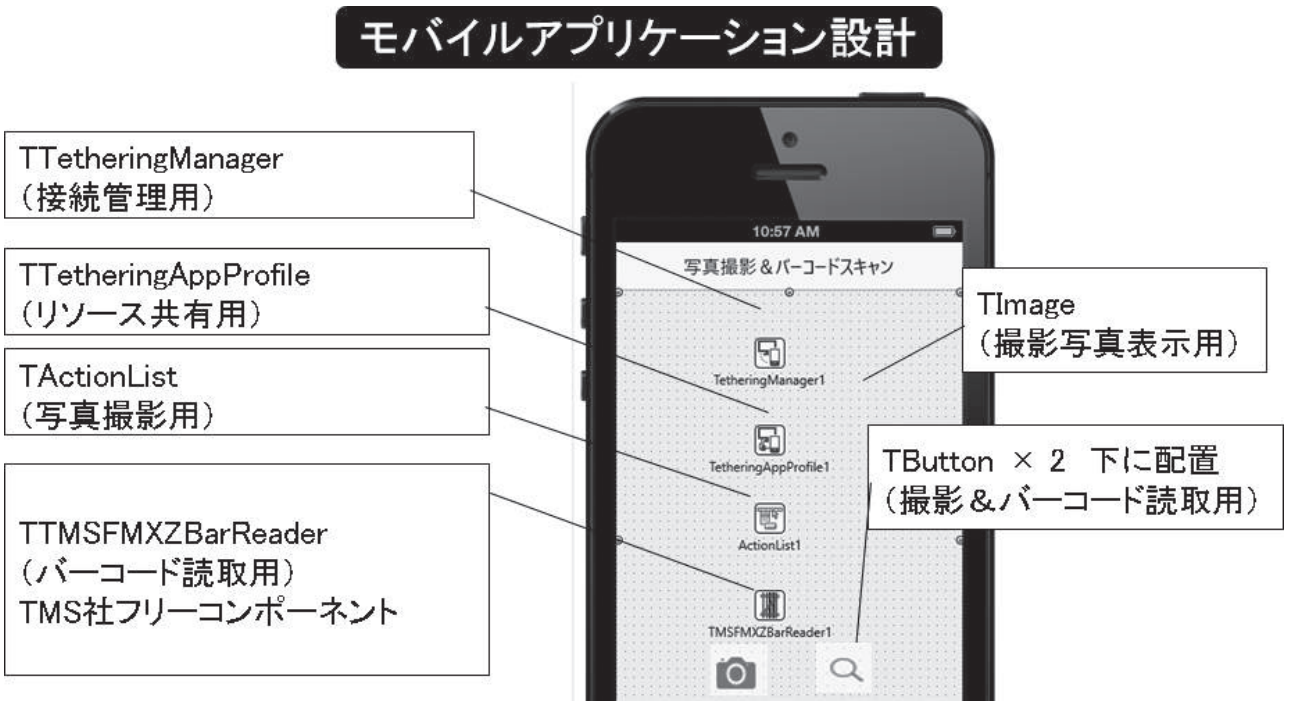

モバイルアプリケーション設計
TTetheringManager
（接続管理用）
TTetheringAppProfile
（リソース共有用）
TActionList
（写真撮影用）
TTMSFMXZBarReader
（バーコード読取用）
TMS社フリーコンポーネント
10:57 AM
写真撮影＆バーコードスキャン
TetheringManager1
TetheringAppProfile1
ActionList1
TMSFMXZBarReader1
TImage
（撮影写真表示用）
TButton × 2 下に配置
（撮影＆バーコード読取用）

図6

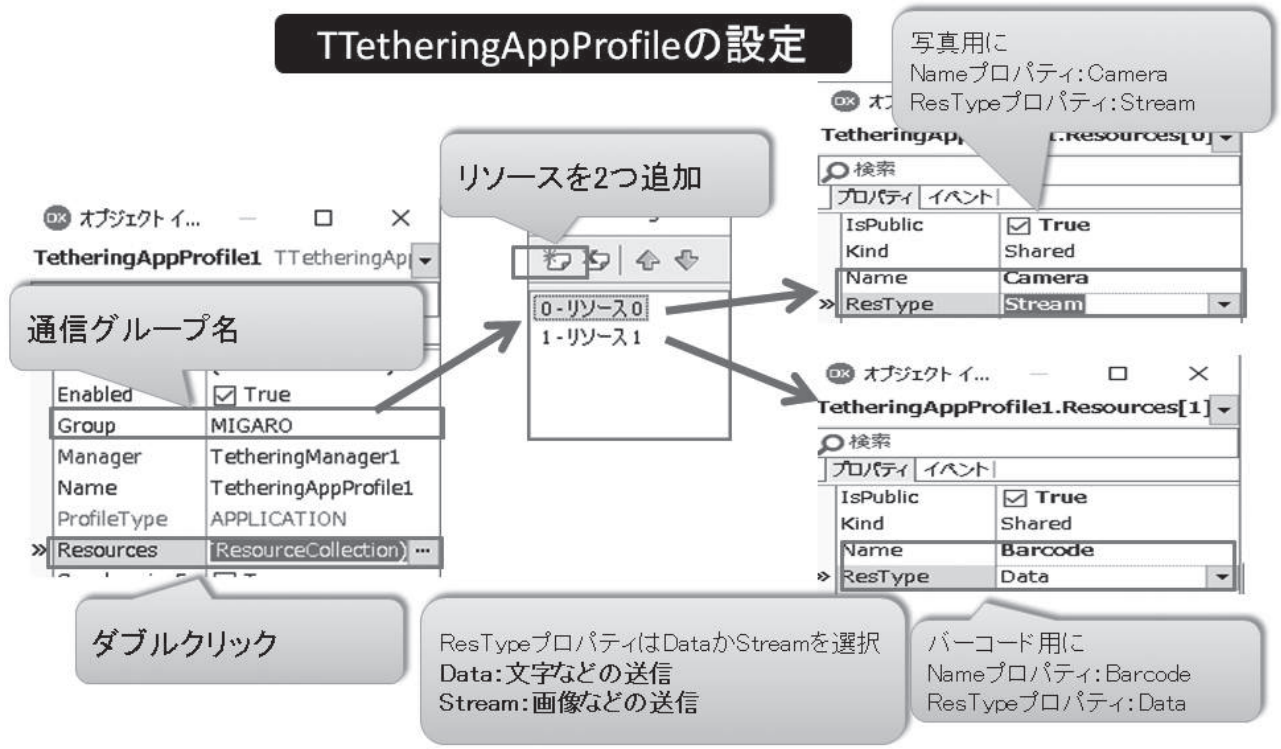
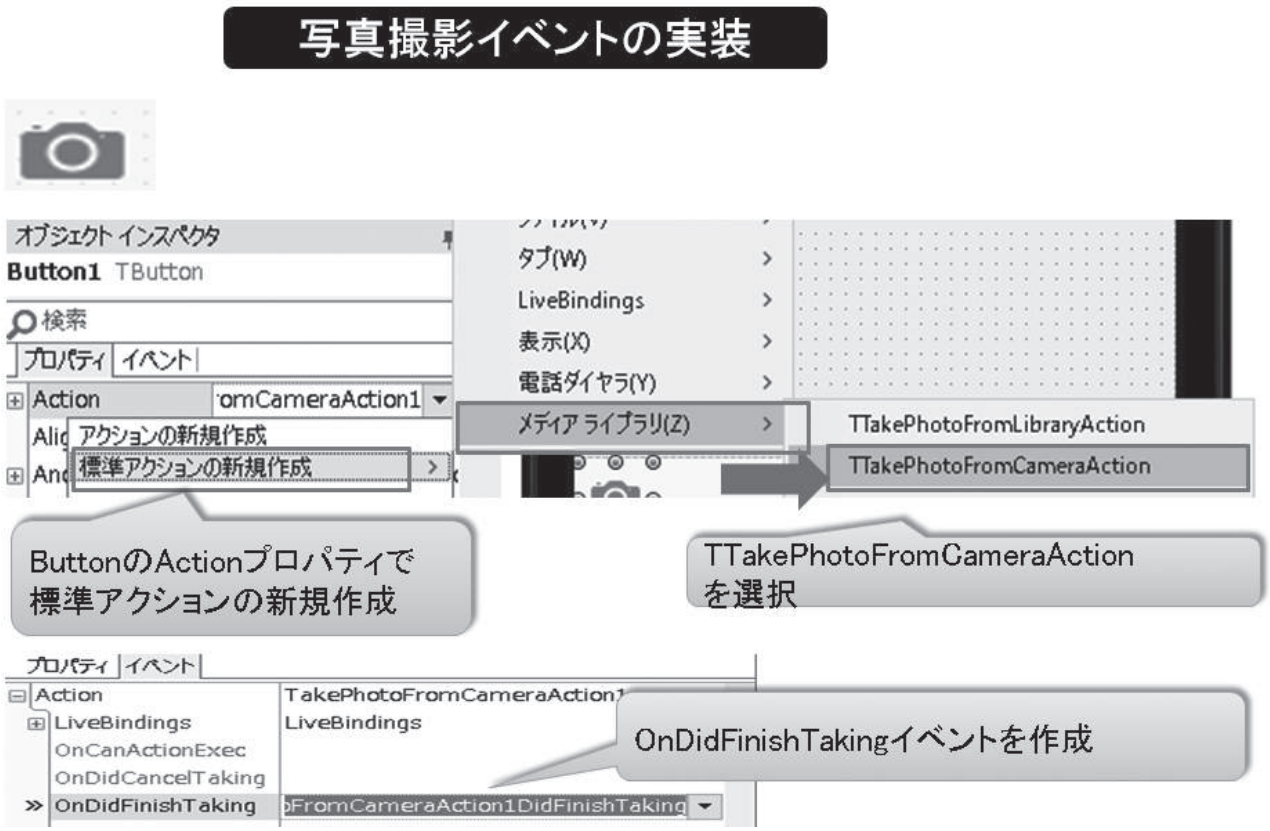

TTetheringAppProfileの設定
写真用に
Nameプロパティ:Camera
ResTypeプロパティ:Stream
リソースを2つ追加
通信グループ名
TetheringAppProfile1
Enabled True
Group MIGARO
Manager TetheringManager1
Name TetheringAppProfile1
ProfileType APPLICATION
Resources (ResourceCollection)
0 - リソース 0
1 - リソース 1
IsPublic True
Kind Shared
Name Camera
ResType Stream
TetheringAppProfile1.Resources[1]
IsPublic True
Kind Shared
Name Barcode
ResType Data
ダブルクリック
ResTypeプロパティはDataかStreamを選択
Data:文字などの送信
Stream:画像などの送信
バーコード用に
Nameプロパティ:Barcode
ResTypeプロパティ:Data

図7



ソース1

OnDidFinishTakingイベント(撮影写真を送信)

```
procedure TForm1.TakePhotoFromCameraAction1DidFinishTaking(Image: TBitmap);
var
  FStream: TMemoryStream;
begin
  FStream := TMemoryStream.Create;  //写真用のStreamを作成
  image.SaveToStream(FStream);      //撮影写真をStreamに格納
  TetheringAppProfile1.Resources.Items[0].Value := FStream; //共有リソースに送信
  Image1.Bitmap.Assign(Image);      //画面に写真を表示
end;
```

ソース2

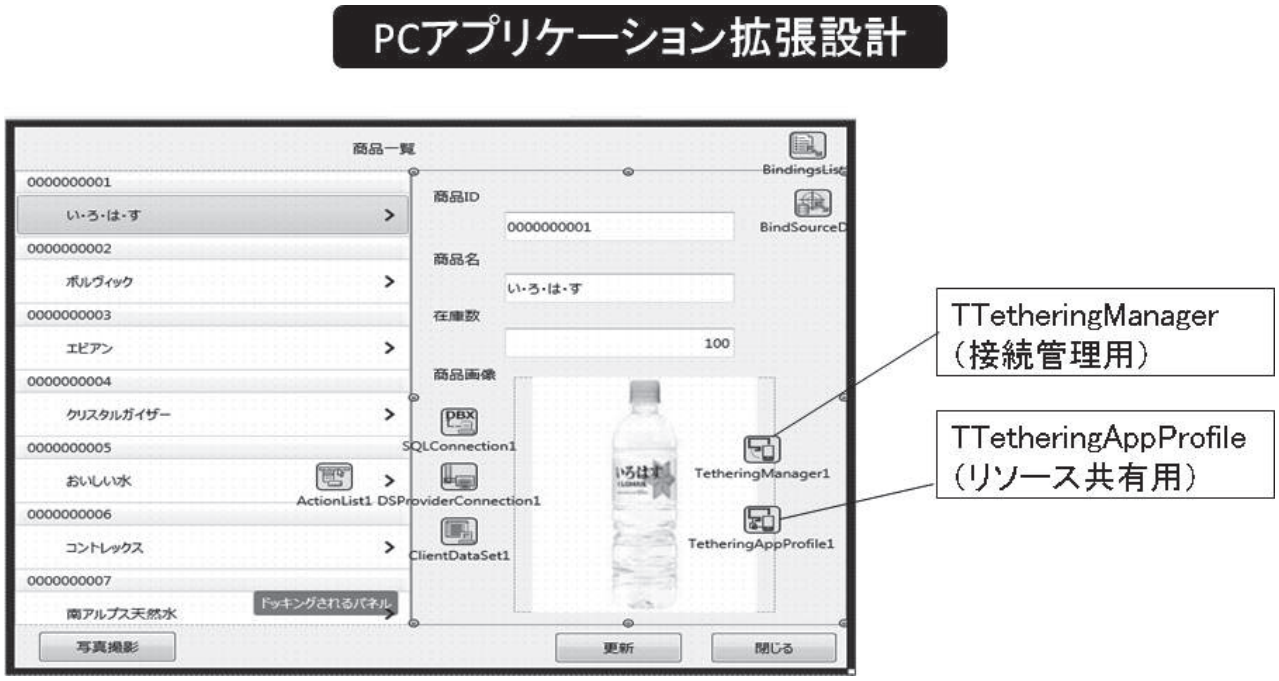
OnClickイベント(バーコード撮影起動)

```
procedure TForm1.Button2Click(Sender: TObject);
begin
  TMSFMXZBarReader1.Show; //バーコード撮影を起動
end;
```

OnGetResultイベント(取得バーコード送信)

```
procedure TForm1.TMSFMXZBarReader1GetResult(Sender: TObject; AResult: string);
Begin
 //読み取ったバーコード値を共有リソースに送信
  TetheringAppProfile1.Resources.Items[1].Value := AResult;
end;
```

図8



ソース3

OnResourceReceivedイベント(撮影写真を受信)

```
//ソース1で送信された画像を受信する
procedure TForm1.TetheringAppProfile1ResourcesOResourceReceived
  (const Sender: TObject; const AResource: TRemoteResource);
begin
  AResource.Value.AsStream.Position := 0;                    //Streamのポジション
  Image1.Bitmap.LoadFromStream(AResource.Value.AsStream);    //画面に受信画像を設定
  Image1.Repaint;                                            //再描画
end;
```

ソース4

OnResourceReceivedイベント(取得バーコードを受信)

```
//ソース2で送信されたバーコード値を受信する
procedure TForm1.TetheringAppProfile1Resources1ResourceReceived
  (const Sender: TObject; const AResource: TRemoteResource);
begin
  Edit1.Text := AResource.Value.AsString;  //画面に受信値を設定
end;
```

図9

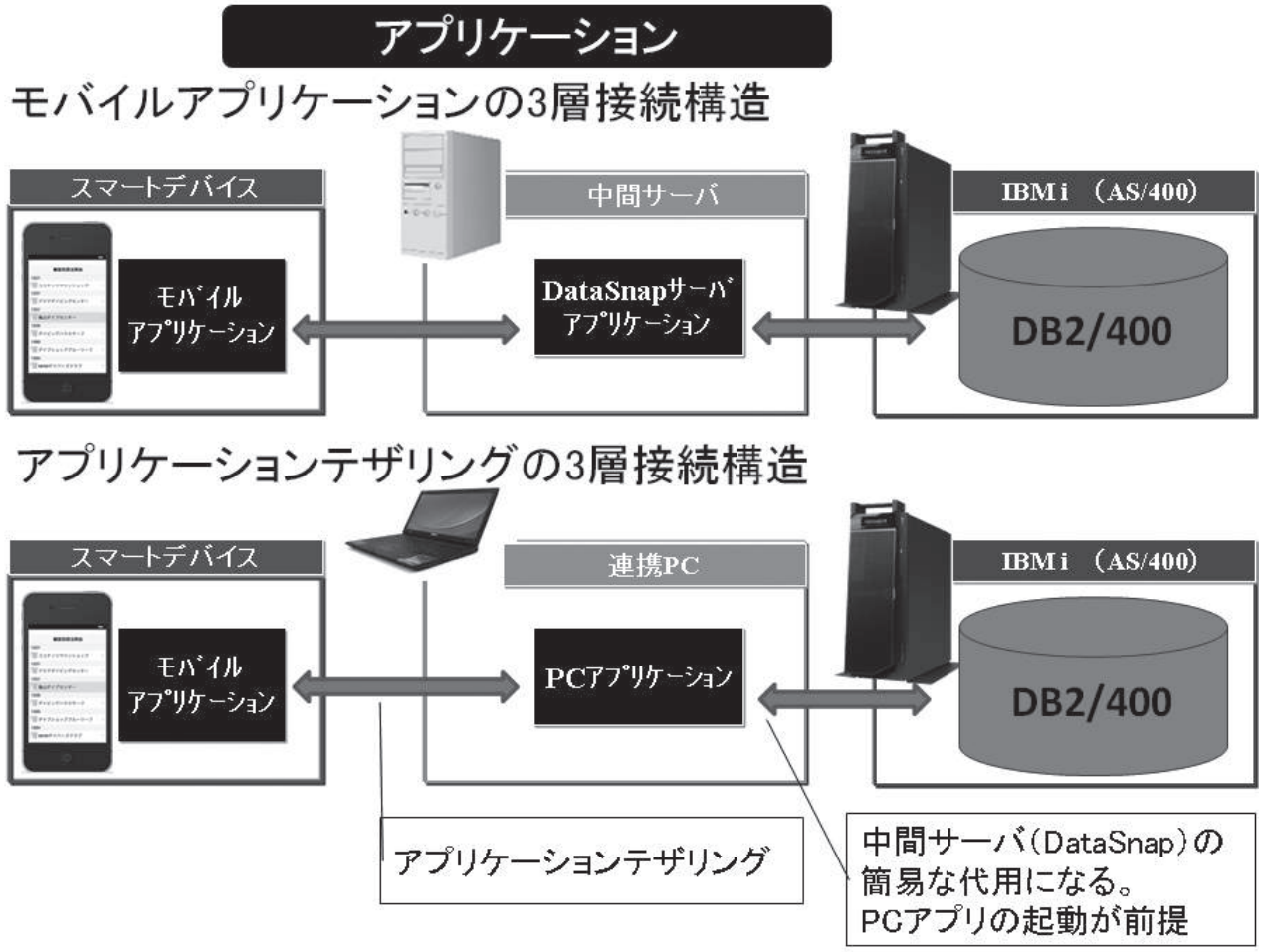
ソース5

OnTimerイベント(接続表示)

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
  Label1.Visible := (TetheringManager1.RemoteProfiles.Count > 0);
end;
```

接続カウントがあれば
Labelを表示

図10



國元 祐二

株式会社ミガロ.

RAD事業部 技術支援課

[SmartPad4i]

SmartPad4iの運用で役立つWEBサーバー機能

略歴
1979年3月27日生まれ
2002年 追手門学院大学 文学部アジア文化学科卒業
2010年10月 株式会社ミガロ. 入社
2010年10月 RAD事業部配属

現在の仕事内容
SmartPad4i（JC/400）、Business4Mobile、Valenceの製品試験やサポート業務、導入支援などを担当している。

1.はじめに

最近ではWEBアプリケーションの技術が向上し、画面や機能もC／Sアプリケーションに近づいている。それに伴い、企業でも基幹システムの一部をWEBアプリケーションで構築するケースが格段に増えてきた。

WEBアプリケーションを採用する場合の大きなメリットは、システムを利用するユーザーごとのPCに環境を構築する必要がない点である。ブラウザからWEBアプリケーションのURLにアクセスすれば、システムを利用できる。これはシステム管理者にとって、運用面で非常に大きな作業軽減となる。

この仕組みを提供しているのが、WEBサーバーである。WEBアプリケーションでは、クライアントPCにセットアップせず、WEBサーバーの端末で代表してセットアップし、クライアントPCの代わりにプログラム処理の実行やDBへのアクセスを実行する。そして、リクエストがあったクライアントのブラウザへ処理結果を送信する。

このように、WEBサーバーはクライアントに代わって処理を実行するためのアプリケーションの一部として、非常に重要な役割を担っている。このWEBサーバーの機能や設定を把握していれば、運用が便利になり、トラブル時にも役立つ。

本稿はこうしたWEBサーバーの機能や設定で、SmartPad4iの利用に役立つ内容を検証し、まとめている。

なお、SmartPad4iはWebSphere Application Server（WAS）環境で動作するので、本稿ではWASのWEBサーバー機能として組み込まれているIBM HTTP Server（IHS）の利用を前提にしている。以降、WEBサーバーはIHSの意味で記述する。

2.WEBサーバーのログ機能

前述したように、WEBサーバーはクライアントからのアクセスへの応答や、WEBアプリケーションの処理といった役割を担う。そのため、WEBサーバーにはWEBアプリケーションが動作しているさまざま情報が集積されており、その情報はログという形で照会できる。

WEBサーバーはさまざまなログを管理するが、その中でもWEBサーバーの運用で非常に重要な情報がアクセスログである。

2-1 アクセスログの活用

アクセスログの内容

アクセスログはWEBサーバーに記録される動作の履歴であるが、その内容を理解することで、有用な解析データとして活用できる。アクセスログは、IHSのインストールディレクトリ直下の[logs]フォルダに出力される。

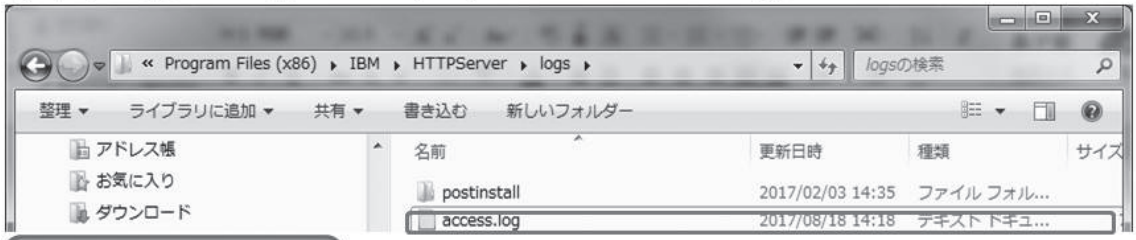
アクセスログにはデフォルトで、次のような情報が記録される。【図1】

・アクセス元のIPアドレス
・アクセスの日付と時刻
・アクセスされたファイル名

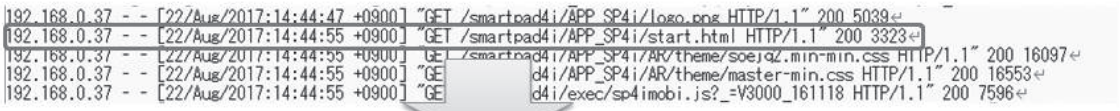
図1

アクセスログ

(出力先　例: C:¥Program Files (x86)¥IBM¥HTTPServer¥logs)



アクセスログ



アクセスログ詳細

IPアドレス　　アクセス日付

192.168.0.37 - - [22/Aug/2017:14:44:55 +0900]

"GET /smartpad4i/APP_SP4i/start.html HTTP/1.1"　200　3323

アクセスされたファイル名　　ステータスコード　　送信バイト数

図2

日単位でアクセスログファイルを分ける

モジュールの有効化

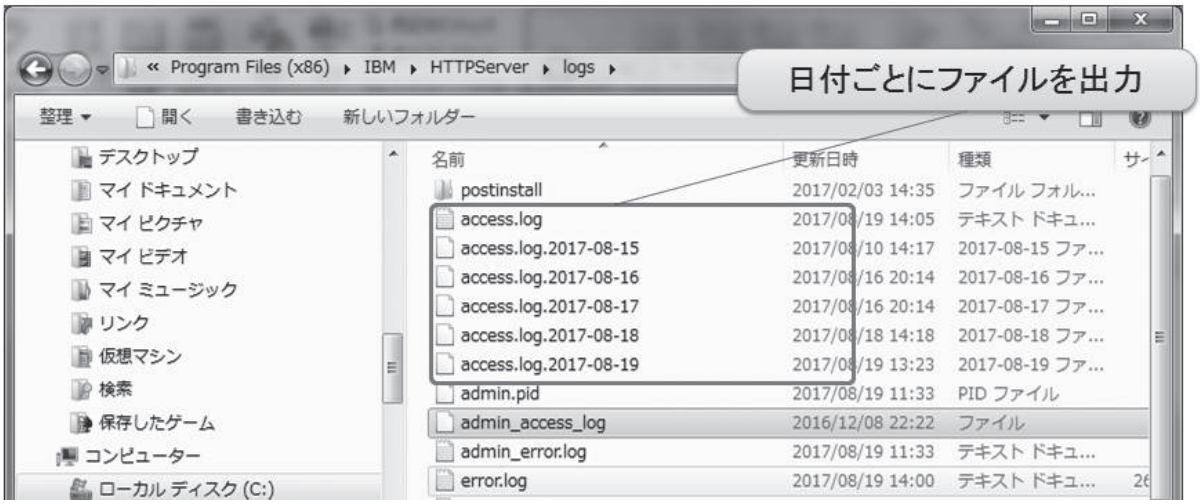
#LoadModule log_config_module modules/mod_log_config.so

→

LoadModule log_config_module modules/mod_log_config.so

CustomLog の設定を変更

CustomLog "|¥"bin/rotatelogs.exe¥" ¥"logs/access.log.%Y-%m-%d¥" 86400" common



日付ごとにファイルを出力

・ステータスコード
・送信バイト数

SmartPad4i を社内ネットワーク内で利用し、クライアント PC の IP アドレスが固定である場合は、アクセスログの記録から対象のクライアント PC を特定できる。

さらにアクセスされたパス、ファイル名や、ステータスコードを確認することで、プログラムが正しいパスのファイルや画像を操作できているかを検証できる。

またアプリケーションでエラーが発生した場合には、ステータスコードが出力される。このコードはエラー内容を判別するエラーコードとしての意味もあるので、アクセスログは運用面でも重要な役割を果たしている。

アクセスログファイルを
日単位で管理する方法

アクセスログは、デフォルトではファイルサイズに制限がなく、単一ファイルに出力され続ける設定となっている。デフォルト設定の場合、1 万リクエストごとに 1 MB 以上のファイルサイズが増える。そのためアクセス数が多いシステムの場合には、アクセスログファイルが膨大なサイズとなる。

このようにアクセスの多い環境では、日単位でアクセスログファイルを出力するのが有効である。ファイルが日単位で分割されることで、サイズも 1 日分のログだけになり、調査時も目的のログを探しやすい。

アクセスログで記録する情報を変更するには、IHS の設定ファイル httpd.conf を編集する。httpd.conf はデフォルトでは、［HTTP Server インストールディレクトリ］\conf\httpd.conf に配置されている。日単位でアクセスログファイルを出力するには、IHS に付属している「rotatelogs プログラム」を利用する。

まず、アクセスログファイルの出力方法をカスタマイズするため、mod_log_config.so モジュールを有効にする。モジュールを有効にする方法は、httpd.conf 内に記述されている行の #（コメント）を削除するだけである。

次に CustomLog に rotatelogs.exe のパスを設定し、アクセスログの形式と分割する秒数を指定する。【図 2】

設定完了後、WEB サーバーを再起動すると設定が反映され、アクセスログファイルが日単位に分かれて出力されるようになる。

なお rotatelogs プログラムの設定・機能については、下記 WEB マニュアルにも詳細が記載されている。

https://publib.boulder.ibm.com/httpserv/manual70/programs/rotatelogs.html

3.WEBサーバーのファイル制御機能

3-1　ファイルのアクセス制限

WEB サーバーはログだけでなく、ファイル管理も運用面での重要な機能となる。WEB サーバーには HTML だけでなく、データとしての画像ファイルや設定ファイルなどが保存されている場合もある。そうしたファイルも WEB サーバーにある限り、パスさえ分かれば、URL を直接入力して自由に参照できる。

そのため WEB サーバーを外部公開するような場合には、アプリケーションで扱うファイルへのアクセスを制限することが多い。WEB サーバーによるファイルのアクセス制限では、特定のファイルをブラウザから直接参照できないように設定できる。

例として、WEB サーバーのディレクトリにある特定ファイルへのリクエストが要求された場合に、403 エラー（閲覧禁止）で制限する方法を説明する。設定はログの変更と同じく、IHS の httpd.conf で制御できる。【図 3】

Directory の記述に、アクセス制限対象のファイルが存在するフォルダパスを指定し、FilesMatch には制限するファイルの拡張子を指定する。こうした簡単な設定だけで、ユーザーからの意図しないアクセスを規制できる。

3-2　ファイルのダウンロード設定

SmartPad4i では、CSV ファイルのダウンロード機能がサポートされている。また、WEB サーバーに配置した PDF ファイルやエクセルファイルなどを、ハイパーリンクを利用してクライアントにダウンロードをさせられる。

しかし一般に WEB アプリケーションでのファイルダウンロードでは、ブラウザの設定によって動作が異なる。たとえばダウンロードとならずに、ブラウザのページ内でファイルが表示される動作になる設定もある。

こうしたクライアント側のブラウザ設定に依存せずにファイルをダウンロードさせるには、WEB サーバー側での制御が必要となる。ここでも httpd.conf を設定することで、ファイルのダウンロードを制御できる。【図 4】

まずは、WEB サーバーが header 情報を返却できるように、ヘッダーモジュールを有効にする。

次に FilesMatch を使用し、特定の拡張子のリクエストが送られた場合にのみ有効になるよう設定する。content-disposition の設定は 2 種類で、inline（デフォルト）に設定すると、ファイルは WEB ページの一部としてブラウザで開く。attachment を設定すると、「名前を付けて保存」のダイアログを表示できる。

つまりファイルをダウンロードさせたい場合は、attachment を設定すればよい。

WEB サーバー側でこのように設定しておくことで、利用するクライアント環境に依存せずにアプリケーションの動作を統一できる。

4. リバースプロキシを使ったアクセス構成

IBM i では V6R1 以降、WAS がデフォルトで付属しており、これを利用する企業も多い。しかし IBM i は基幹データを扱っていることが多く、外部向けに WEB サーバーとして公開する場合には、セキュリティ面の検討が必要となる。

本稿ではセキュリティを高める仕組みの 1 つとして、リバースプロキシについて説明する。

4-1　リバースプロキシとは

リバースプロキシは不特定多数のクライアントからの要求に対して、応答を代行する機能である。WEB サーバーを外部公開する場合、WEB サーバーをインターネット上に公開することになるが、リバースプロキシ経由で WEB サーバー

図3

アクセス制限設定

特定ディレクトリのJPEGファイルへアクセスさせない

```
<Directory "C:/Program Files (x86)/IBM/HTTPServer/htdocs/ja_JP/smartpad4i/html/SP4IV2SMP/PIC">
 <FilesMatch "¥.(jpg|jpeg)$">
  Deny from all
 </FilesMatch>
</Directory>
```

アクセス時には403エラーが返却される

図4

ダウンロード設定

Header モジュールを有効化する

```
#LoadModule headers_module modules/mod_headers.so
```

↓

```
LoadModule headers_module modules/mod_headers.so
```

特定の拡張子のファイルをダウンロードさせる

```
<FilesMatch "¥.(csv|pdf|xls|xlsx)$">
Header set content-disposition attachment
</FilesMatch>
```

にアクセスすることで、IBM i を直接インターネット上に公開する必要がなくなる。

このリバースプロキシは WEB サーバーに付属する一般的な機能なので、SmartPad4i にも利用できる。【図 5】

4-2 リバースプロキシを活用したセキュリティ

ここでは、SmartPad4i をリバースプロキシ経由でアクセスする方法を説明する。なお、リバースプロキシには Windows 版の Apache2.4 を例として使用する。

まず Apache2.4 を下記 URL からダウンロードして、リバースプロキシ機能を持たせるサーバーにインストールする。

https://www.apachelounge.com/

Apache をインストールし、WEB サーバーとして動作することを確認後、Apache の設定ファイル httpd.conf を編集する。httpd.conf は IHS と同様に、インストールディレクトリの conf に配置されている。リバースプロキシを利用するには、まず 3 つのモジュール mod_headers.so、mod_proxy.so、mod_proxy_http.so を有効にする。【図 6】

次に、リクエストをどこに転送するかを設定する。

たとえば、IBM i の WEB サーバーが IP アドレス 192.168.0.2、ポート番号 10000 で構築されている場合には、【図 7】の設定でリクエストを転送できる。

「ProxyRequests OFF」に設定していると、ProxyPass で明示的に指定した先のホストにのみアクセスできるようになる。

ProxyPass には、リバースプロキシのドキュメントルートへのリクエストを http://192.168.0.2:10000/ へ転送する。ProxyPassReverse はレスポンスのヘッダー情報（リクエスト時の属性）を書き換えるための設定項目で、書式は ProxyPass と同じである。

設定後、リバースプロキシの WEB サーバーを再起動することで機能が有効になる。こうした WEB サーバーの 2 重構成を用意することで、セキュリティを高めた IBM i の運用を実現できる。

5.おわりに

本稿ではアクセスログ活用、ファイル制御、リバースプロキシについて説明してきた。「3.WEB サーバーのファイル制御機能」ではブラウザからのファイルダウンロード動作を例に説明したが、WEB アプリケーションでは、クライアント環境設定の違いで運用に苦戦するなどの話を聞くことも多い。しかし WEB サーバーの機能を知っていれば、WEB サーバー側で簡単に対応できる。

WEB サーバーのカスタマイズを知らなくても、WEB アプリケーションは運用できるが、知っておくと運用での負担が大きく軽減できる。

冒頭で述べたとおり、WEB アプリケーションはクライアント環境のセットアップを必要とせず、運用管理をサーバーに集中できる点が大きなメリットである。

この WEB アプリケーションのメリットを最大限に活かすために、本稿を WEB サーバー運用の参考として役立てていただけると幸いである。

M

図5

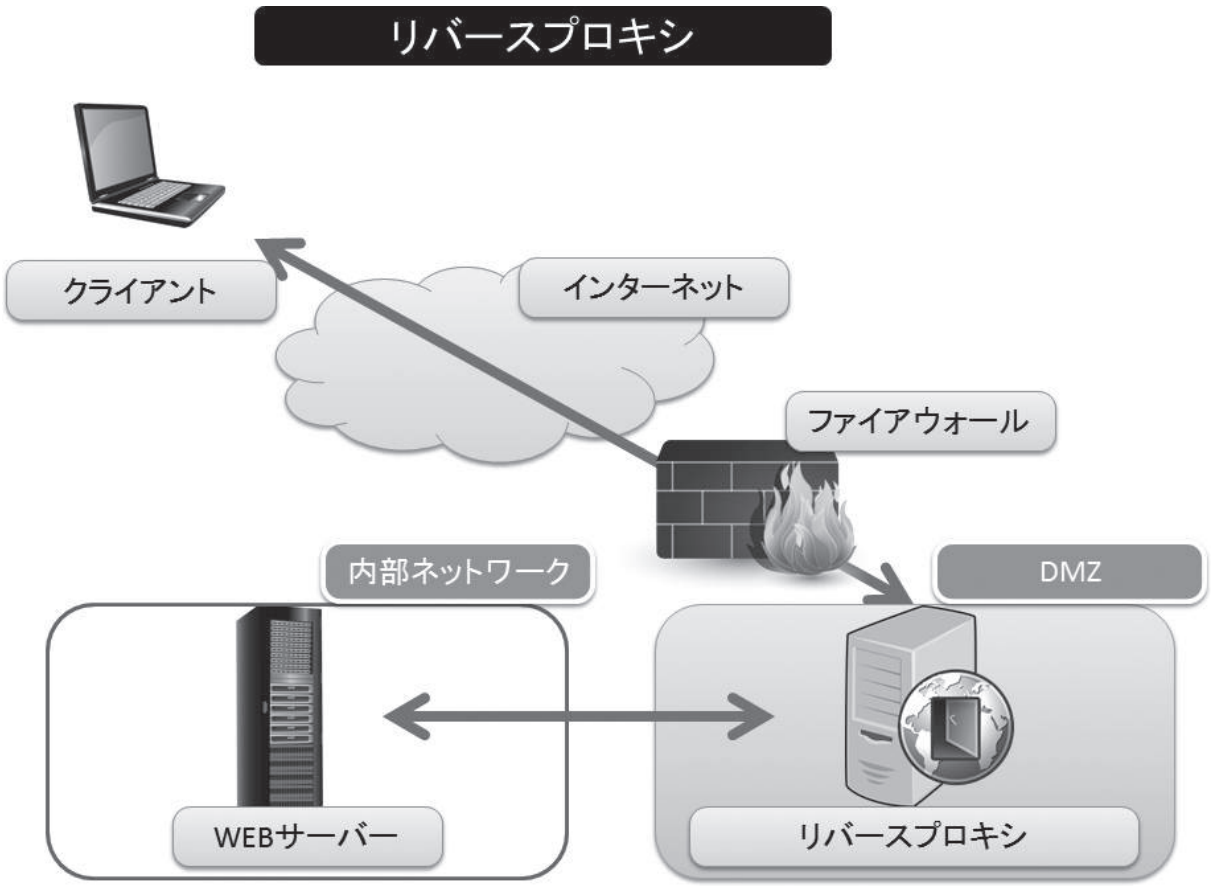


図6

リバースプロキシ設定

モジュールを有効化する

◆mod_headers モジュール

#LoadModule headers_module modules/mod_headers.so

⇩

LoadModule headers_module modules/mod_headers.so

◆mod_proxy モジュール

#LoadModule proxy_module modules/mod_proxy.so

⇩

LoadModule proxy_module modules/mod_proxy.so

◆mod_proxy_http モジュール

#LoadModule proxy_http_module modules/mod_proxy_http.so

⇩

LoadModule proxy_http_module modules/mod_proxy_http.so

図7

リバースプロキシ設定

リバースプロキシ設定

ProxyRequests OFF

ProxyPass / http://192.168.0.2:10000/
ProxyPassReverse / http://192.168.0.2:10000/

Migaro. Technical Report
既刊号バックナンバー

電子版・書籍(紙)媒体で提供中!
http://www.migaro.co.jp/contents/support/technical_report/

No.1　2008 年秋

お客様受賞論文

●最優秀賞

直感的に理解できるシステムを目指して―情報の“見える化”の取り組み
石井 裕昭様／豊鋼材工業株式会社

●ゴールド賞

運用部間にサプライズをもたらした Delphi/400
春木 治様／株式会社ロゴスコーポレーション

●シルバー賞

JACi400 使用による Web アプリケーション開発工数削減
中富 俊典様／日本梱包運輸倉庫株式会社

Delphi/400 を利用した Web 受注システム
飯田 豊様／東洋佐々木ガラス株式会社

●優秀賞

Delphi/400 による販売管理システム（FAINS）について
藤田 建作様／株式会社船井総合研究所

技研化成の新基幹システム再構築
藤田 健治様／技研化成株式会社

SE 論文

はじめての Delphi/400 プログラミング
畑中 侑／システム事業部 システム２課

Delphi/400 と Excel との連携
中嶋 祥子／ RAD 事業部 技術支援課

連携で広がる Delphi/400 活用術
尾崎 浩司／システム事業部 システム２課

フォーム継承による効率向上開発手法
吉原 泰介／ RAD 事業部 技術支援課

API を利用した出力待ち行列情報の取得方法
鶴巣 博行／ RAD 事業部 技術支援課

Delphi テクニカルエッセンス Q&A 集
吉原 泰介／ RAD 事業部 技術支援課

JACi400 を使って RPG で Web 画面を制御する方法
松尾 悦郎／システム事業部 システム２課

あなたはプラインドタッチができますか？
福井 和彦／システム事業部 システム１課

No.2　2009 年秋

お客様受賞論文

●最優秀賞

JACi400 で 既存 Web サービスの内製化を実現
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
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
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
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
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
島根 英行様／シルフ

SE 論文

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

Delphi/400 を利用したはじめての Web 開発
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
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
吉原 泰介／ RAD 事業部 技術支援課 顧客サポート

RPG でパフォーマンスを制御
松尾 悦郎／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
宮坂 優大・田村 洋一郎／システム事業部 システム１課

No.3　2010 年秋

お客様受賞論文

●最優秀賞

建物のクレーム情報管理システム「アフターサービス DB」について

大橋 良之様／東レ建設株式会社

●ゴールド賞

Delphi/400 で「写真管理ソフト」と「スプールファイルの PDF 化ソフト」を自社開発

寒河江 幸喜様／日綜産業株式会社

●シルバー賞

Delphi/400 で鉄鋼受発注業務を統一し 鉄鋼 EDI も実現

柿本 直樹様／合鐵産業株式会社

●優秀賞

Delphi/400 で EIS（Executive Information System）の高速化

小島 栄一様／西川計測株式会社

イントラでの PHP-Delphi-RPG 連携

仲井 学様／西川リビング株式会社

Delphi/400 を使った取引先管理システム

大崎 貴昭様／森定興商株式会社

SE 論文

Delphi/400 ローカルキャッシュ活用術

中嶋 祥子／ RAD 事業部 技術支援課

Delphi/400 帳票開発ノウハウ公開

尾崎 浩司／システム事業部 システム３課

Delphi/400 でドラッグ&ドロップを制御

辻林 涼子／システム事業部 システム２課

Delphi/400 のモジュールバージョン管理手法

前田 和寛／システム事業部 システム２課

Delphi/400　Web からの PDF 出力

福井 和彦・清水 孝将／システム事業部システム３課・システム２課

Delphi/400 で Flash 動画の実装

吉原 泰介／ RAD 事業部 技術支援課 顧客サポート

No.4　2011 年秋［創立 20 周年記念号］

お客様受賞論文

●最優秀賞

全社の経費処理業務を効率化した「e 総務システム」

鈴木 英明様／阪和興業株式会社

●ゴールド賞

「Web 進捗管理システム」でリアルタイム性を実現

堀内 一弘様／エスケーロジ株式会社

●シルバー賞

「営業奨励金申請書」をたった２日間で開発

簑島 宏明様／株式会社ケーユーホールディングス

液体輸送における「配車支援システム」の構築

桂 哲様／ライオン流通サービス株式会社

SE 論文

グラフ活用リファレンス

中嶋 祥子／ RAD 事業部 技術支援課

Web サービスを利用して機能 UP ！

福井 和彦・畑中 侑／システム事業部 システム２課

OpenOffice 実践活用

吉原 泰介／ RAD 事業部 技術支援課 顧客サポート

VCL for the Web 活用 TIPS 紹介

尾崎 浩司／システム事業部 プロジェクト推進室

JC/400 で JavaScript 活用

清水 孝将／システム事業部 システム１課

jQuery 連携で機能拡張

國元 祐二／ RAD 事業部 技術支援課 顧客サポート

No.5　2012 年秋［創刊 5 周年記念］

お客様受賞論文

【部門 1】

●最優秀賞

JC/400 による取引先との Web-EDI システム構築

久保田 佳裕様／極東産機株式会社

●ゴールド賞

Delphi と Excel を使用した帳票コストの削減

大久保 治高様／合鐵産業株式会社

もっと見やすく、もっと使いやすい画面を

新谷 直正様／株式会社アダル

【部門 2】

●優秀賞

Delphi/400 で確認業務の効率化

為国 順子様／ベネトンジャパン株式会社

取引先申請システムでの稟議書作成ワークフロー

大崎 貴昭様／森定興商株式会社

Delphi/400 で IBM i のストアードプロシージャを利用し、SQL 処理を高速化

島根 英行様／シルフ

SE 論文

InstallAware を使った Delphi/400 運用環境の構築

中嶋 祥子／ RAD 事業部 技術支援課 顧客サポート

カスタマイズコンポーネント入門 Delphi/400 開発効率向上

前田 和寛／システム事業部 システム 2 課

Delphi/400 スマートデバイスアプリケーション開発

吉原 泰介／ RAD 事業部 技術支援課 顧客サポート

DataSnap を使用した 3 層アプリケーション構築技法

尾崎 浩司／システム事業部 プロジェクト推進室

JC/400 でポップアップウィンドウの制御&活用ノウハウ

清水 孝将・伊地知 聖貴／システム事業部 システム 1 課

【創刊 5 周年記念】

ミガロ.SE 座談会―お客様と共に歩む、お客様への熱い思い

No.6　2013 年秋

お客様受賞論文

【部門 1】

●最優秀賞

自社用開発フレームワークの構築

駒田 純也様／ユサコ株式会社

●ゴールド賞

Delphi/400 で CTI 開発および関連機能組み込み

仲井 正人様／株式会社スマイル・ジャパン

●シルバー賞

IBM WebFacing から JC/400 への移行・リニューアル手法

八木 秀樹様／極東産機株式会社

Delphi/400 と Delphi を利用した IBM i 資源の有効活用

小山 祐二様／澁谷工業株式会社

発注システムを VB から Delphi へ移植しリニューアル

川島 寛様／株式会社タツミヤ

【部門 2】

●優秀賞

5250 画面を使用せずに AS/400 スプールファイルをコントロールする

白井 昌哉様／太陽セメント工業株式会社

Delphi/400 を利用した 承認フロー導入による IT 内部統制構築

塚本 圭一様／ライオン流通サービス株式会社

SE 論文

FastReport を使用した帳票作成入門

尾崎 浩司／ RAD 事業部 営業推進課

Delphi/400 で開発する 64bit アプリケーション

吉原 泰介／ RAD 事業部 技術支援課 顧客サポート

Web コンポーネントのカスタマイズ入門

佐田 雄一／システム事業部 システム 1 課

Indy を利用したメール送信機能開発

辻野 健・前坂 誠二／システム事業部 システム 2 課

Windows テキストファイル操作ノウハウ

小杉 智昭／システム事業部 プロジェクト推進室

JC/400 Web アプリケーションのユーザー管理・メニュー管理活用術

吉原 泰介・國元 裕二／ RAD 事業部 技術支援課 顧客サポート

No.7　2014 年秋

お客様受賞論文

【部門 1】

●最優秀賞

Delphi/400 による生産スケジューラの再構築

柿村 実様／東洋佐々木ガラス株式会社

●ゴールド賞

Delphi/400 および Delphi を利用したオンライン個人別メニューの構築

小山 祐二様／澁谷工業株式会社

●シルバー賞

IBM i と Delphi/400 のコラボレーション

新谷 直正様／株式会社アダル

●シルバー賞

荷札発行システムリプレースについて

仲井 学様／西川リビング株式会社

【部門 2】

●優秀賞

Delphi/400 バージョンアップのためのクライアント環境構築

普入 弘様／株式会社エイエステクノロジー

●優秀賞

外出先からメールでリアルタイム在庫を問い合せ

島根 英行様／シルフ

SE 論文

iOS/Android ネイティブアプリケーション入門

吉原 泰介／ RAD 事業部 技術支援課

ファイル加工プログラミングテクニック

小杉 智昭／システム事業部 プロジェクト推進室

FastReport を使用した帳票作成テクニック

前坂 誠二／システム事業部

大量データ処理テクニック

佐田 雄一／システム事業部

スマートデバイス WEB アプリケーション入門

尾崎 浩司／ RAD 事業部 技術支援課

國元 祐二／ RAD 事業部 技術支援課

No.8　2015 年秋

お客様受賞論文

【部門 1】

●最優秀賞

iPod Touch の業務利用開発と検証

石井 裕昭様／豊鋼材工業株式会社

●ゴールド賞

ブランク加工図管理システムの構築

小山 祐二様／澁谷工業株式会社

●シルバー賞

Delphi/400 でスプールファイル管理（WRKSPLF コマンドの活用）

三好 誠様／ユサコ株式会社

●シルバー賞

予算管理システムの構築

川島 寛様／株式会社タツミヤ

●シルバー賞

送状データ送信システムの Web 化について

仲井 学様／西川リビング株式会社

【部門 2】

●優秀賞

繰り返し DB 参照時の ClientDataSet の First 機能について

牛嶋 信之様／株式会社佐賀鉄工所

●優秀賞

IBM i のカレンダーを基準に他のシステムを稼働

福島 利昭様／株式会社ランドコンピュータ

SE 論文

フレームを利用した開発手法

前坂 誠二／システム事業部 システム 2 課

Windows タブレット用にカスタムソフトウェアキーボードを実装

福井 和彦／システム事業部 プロジェクト推進室

マルチスレッドを使用したレスポンスタイム向上

尾崎 浩司／ RAD 事業部 営業・営業推進課

Android アプリケーションの NFC 機能活用

吉原 泰介／ RAD 事業部 技術支援課 顧客サポート

スマートデバイス開発で役立つ画面拡張テクニック

國元 祐二／ RAD 事業部 技術支援課 顧客サポート

No.9　2016 年秋

お客様受賞論文

【部門 1】

●最優秀賞

IBM i の見える化で実現するアジャイル開発

吉岡 延泰様／日本調理機株式会社社

●ゴールド賞

Windows Like 5250 への道のり

小山 祐二様／澁谷工業株式会社

●シルバー賞

Delphi プログラム管理ソフトの開発

牛嶋 信之様／株式会社佐賀鉄工所

【部門 2】

●優秀賞

Delphi/400 を利用した定型業務の PDF 化

佐藤 岳様／ライオン流通サービス株式会社

●優秀賞

ちょい足しモバイル

仲井 正人様／株式会社スマイル・ジャパン

●優秀賞

AS/400 の受注データを Web で社員に公開

福島 利昭様／株式会社ランドコンピュータ

SE 論文

iOS モバイルアプリ開発のデザイニングテクニック

前坂 誠二／システム事業部 システム 2 課

新データベースエンジン FireDAC を使ってみよう！

福井 和彦／システム事業部 プロジェクト推進室

Delphi/400 最新プログラム文法の活用法

尾崎 浩司／ RAD 事業部 営業・営業推進課

FastReport を活用した電子帳票作成テクニック

宮坂 優大／システム事業部 システム 1 課

Beacon 技術による IoT 活用の第一歩

吉原 泰介／ RAD 事業部 技術支援課 顧客サポート

Web & ハイブリッドアプリ開発で役立つ IBM i & ブラウザデバッグテクニック

國元 祐二／ RAD 事業部 技術支援課 顧客サポート

MEMO

MEMO

MEMO

MEMO

MEMO

MEMO

MIGARO.
TECHNICAL
REPORT

Migaro.Technical Report
No.10　2017 年秋　［創刊 10 周年記念号］
ミガロ.テクニカルレポート

2017 年 12 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

東京事業所
〒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