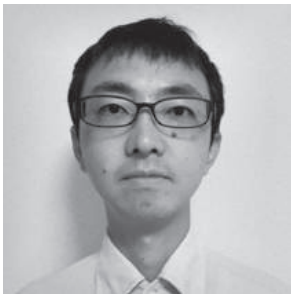
シルバー賞

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
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、日本はもちろん、海外でも高い評価を得ている。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。

はじめに

昭和 13 年に創業した佐賀鉄工所は、昭和 20 年代後半より、高品質・高機能のボルトの提供に特化し、現在では日本はもちろん、海外でも高い評価を得るに至った。

この評価を励みに私たち佐賀鉄工所は、これからもボルトの専門技術者集団としてユーザーのニーズを的確に受け止め、最適な締結技術を提供し続けたいと考えている。

引張強さ 1200MPa 以上の性能を備えるボルトは、自動車のエンジン回りでは 40 ～ 50 本が使用され、ミッションなどでも活躍している。この自動車用の高強度ボルトは、高い信頼性を要求されることから、日本では生産するボルトメーカーが数社に限られている。

「高強度ボルトを生産できるかどうかが、ボルトメーカーの技術力を計る尺度」とまで言われる。当社では現在、高強度ボルトの生産が月産数百トンを超えている。

今後、エンジンの高性能化に伴い、さらに高いスペックが要求されている。当社では新材料により、さらに高強度のボルトの開発を進めている。

システム紹介

当社では日立のメインフレームを使用していたが、2011 年より IBM i を使用し始め、基幹システムはすべて IBM i で処理している。

エンドユーザーが利用する処理画面は、そのほとんどが Visual Basic（以下、VB）6 でインターフェースが構築されている。しかし最新 OS の導入に伴い、VB6 の開発および運用が困難になってきた。

そのため代替手段として導入したのが、IBM i と親和性が高く、さまざまな OS やデータベースに対応できる Delphi/400 であった。現在は、VB6 から Delphi/400 への移行を行っている。

プログラム管理ソフト開発の経緯

IBM i でのプログラムソース管理は、アイエステクノポート製の S/D Manager を使用しており、VB6 に関しては当社で開発したプログラム管理ソフトを使用していた。しかし、Delphi/400 のプログラム管理ソフトは未開発だったので、手動でファイルサーバーへのソース管理を行っていた。

Delphi/400 導入直後は、メインで開発する担当者が 1 人だけだったので、さほど問題はなかった。しかし次第に他の情報システム課員も開発する機会が多くなってきたため、プログラムソース管理（排他制御がないことによるバージョン不整合、および人為的ミスによるプログラムソース消失）に不安を抱えるようになっていた。

そこでプログラム管理ソフトを導入することにより、プログラムソース管理を厳格化し、ファイルサーバーにおける一元管理を実現した。Delphi/400 はオ

図1　チェックアウト概要図

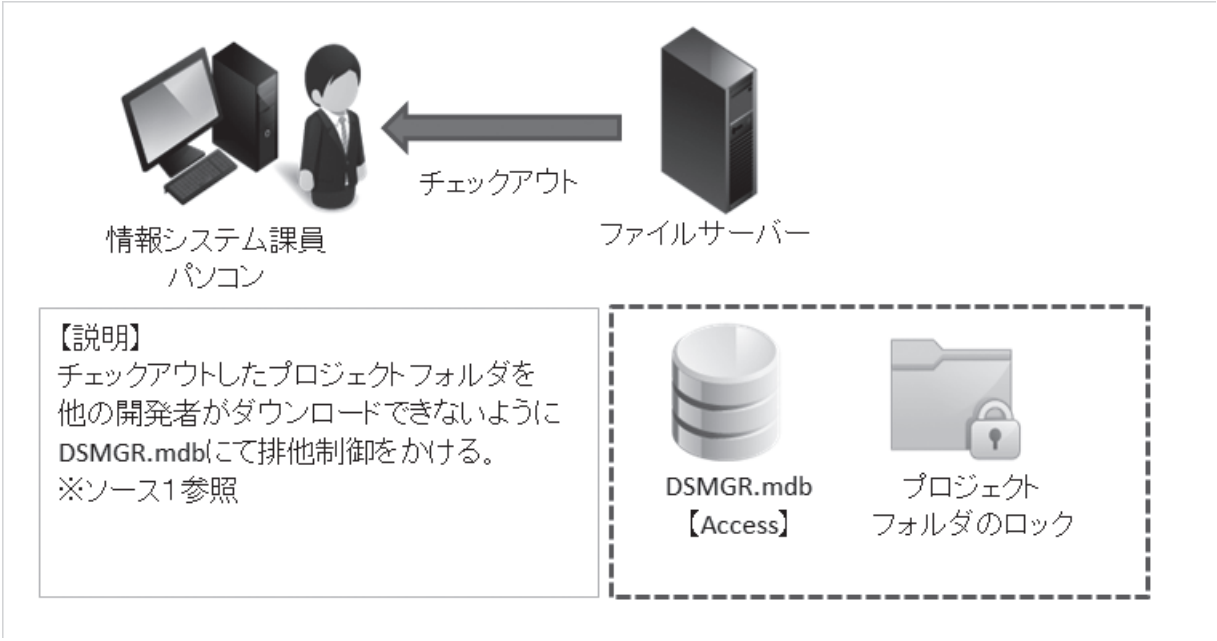
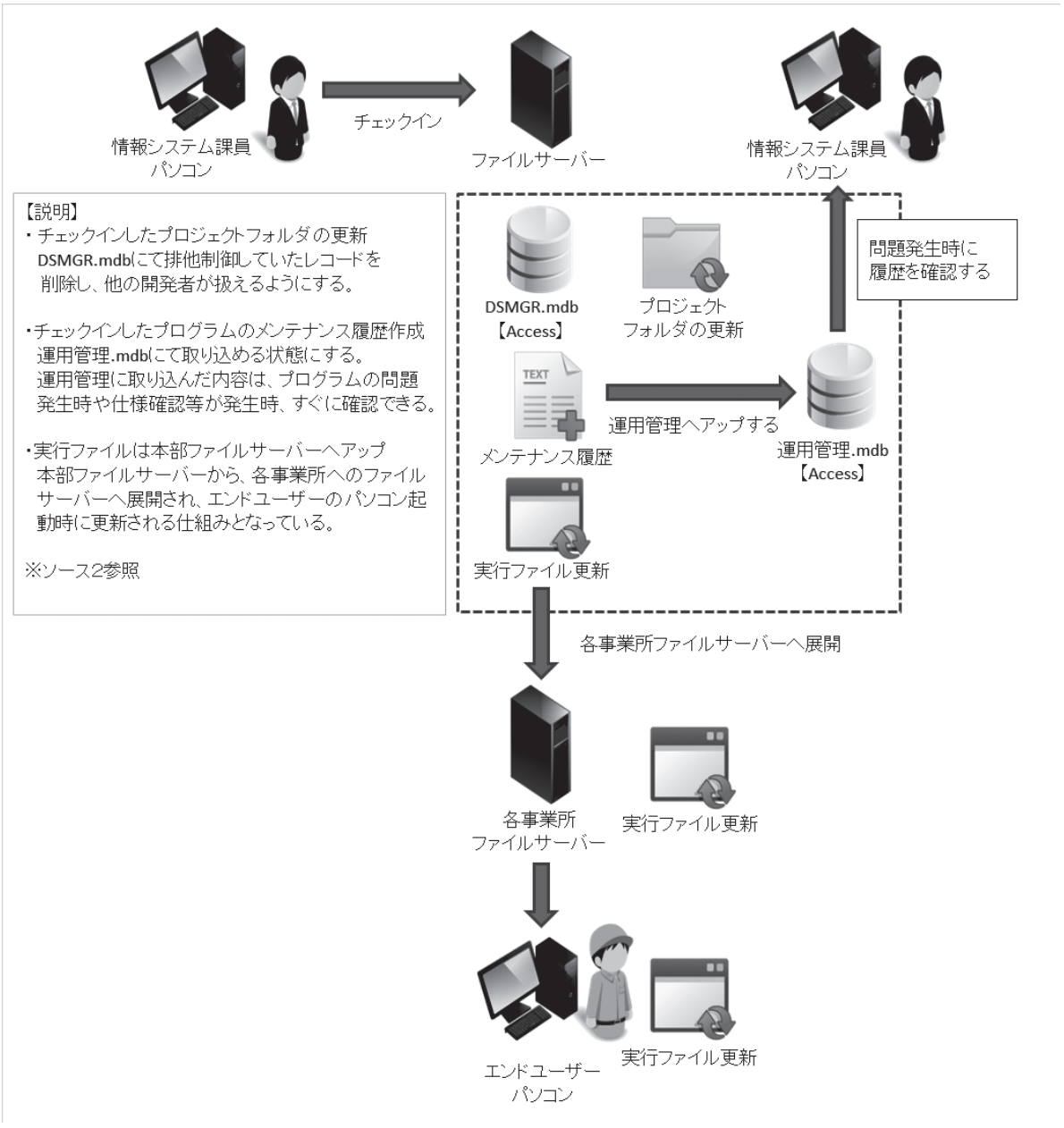


図2　チェックイン概要図



ブジェクト指向言語なので、プログラム開発を標準化すべく、開発した継承元プログラムおよび部品コンポーネントをすぐに開発端末に同期する仕組みを追加した。

Delphiプログラム管理ソフトの開発

要件定義

プログラム管理ソフトの開発に際して、次のシステム要件を決定した（【図1】～【図5】、および【ソース1】【ソース2】を参照）。

・プログラムソースの一元管理（排他制御）

チェックアウト時、プロジェクトフォルダを他の開発者がダウンロードできないように排他制御を実施する。チェックイン時は、チェックアウトしたプロジェクトフォルダについて排他制御を解除し、他の開発者が扱えるようにする

・修正前のプログラムソースのバックアップ

・継承元および部品コンポーネントの同期機能

チェックアウト処理画面で、継承元プログラムを情報システム課のPCに同期する機能を追加する

・チェックイン時のメンテナンス記録用ファイル作成

チェックインしたプログラムのメンテナンス履歴を作成し、運用管理データベースに取り込む

・プログラムソース開発状況の可視化

運用管理データベースに取り込んだ内容は、プログラムの問題発生時や仕様確認等が発生した場合に、すぐに確認できる体制とする

構築システムの仕様

・プログラムソースの開発状況は、ファイルサーバーのAccessで管理する
・プログラムソースの排他制御は、同じくファイルサーバーのAccessで管理する
・プログラムソースおよび実行ファイルは、決められたディレクトリへコピーする
・開発端末における開発用ディレクトリ構造は、すべて同じ構成とする

導入効果と今後の展望

現在、VB6からDelphi/400への移行も本格的に進み始め、Delphi/400のプログラムソースの管理が多くなっている。開発したプログラム管理ソフトにより、ソース管理の人為的ミスを防げるようになった。プログラム管理の改善により、プログラム品質向上に少しでもつなげていきたい。

今後は、チェックインしたプログラムの旧バージョンをすぐに取り出せる機能を追加し、デグレ発生による応急処理の一環として機能追加を検討する予定である。

M

図3　チェックアウト処理内容

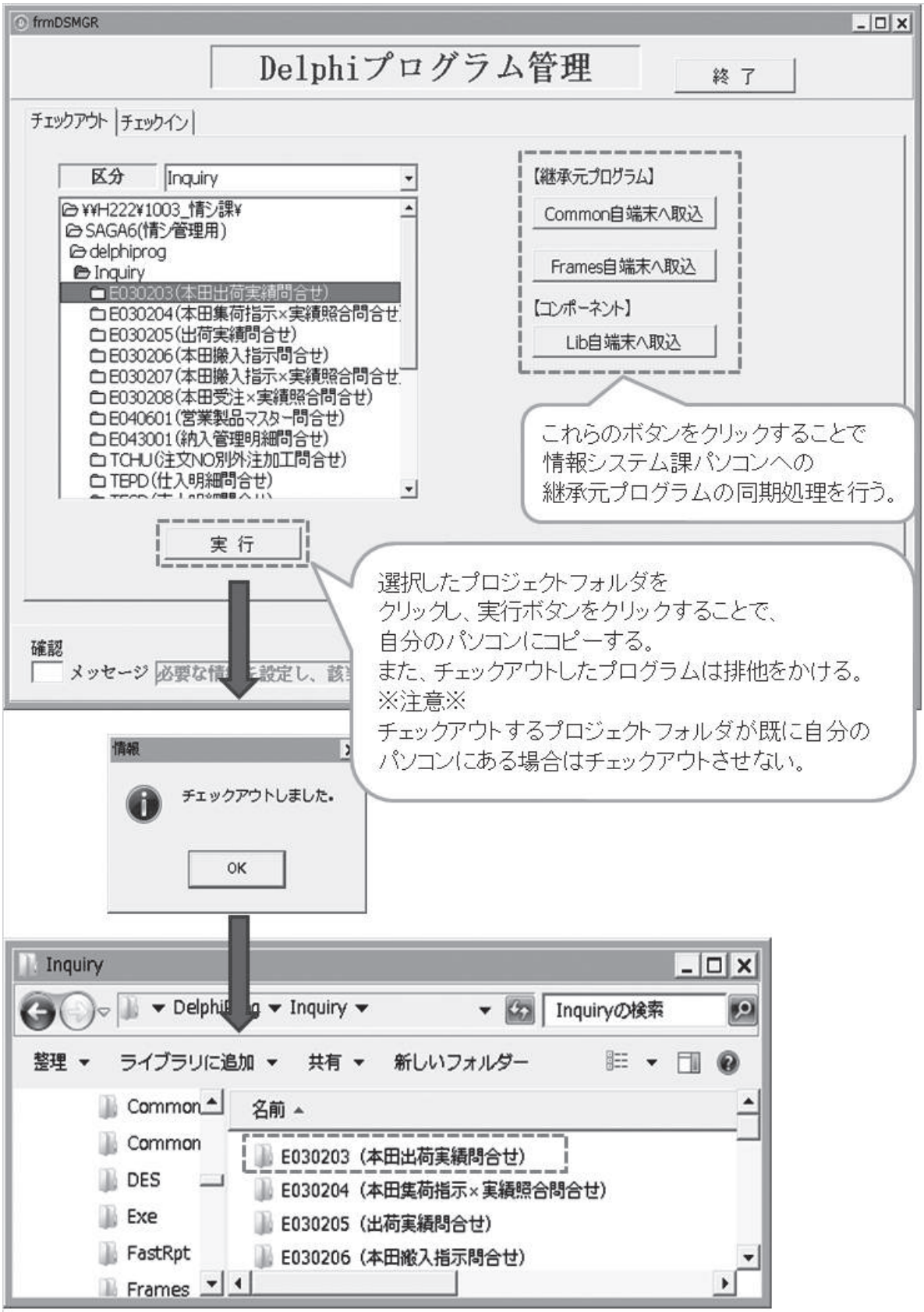


図4　チェックアウト取消処理内容

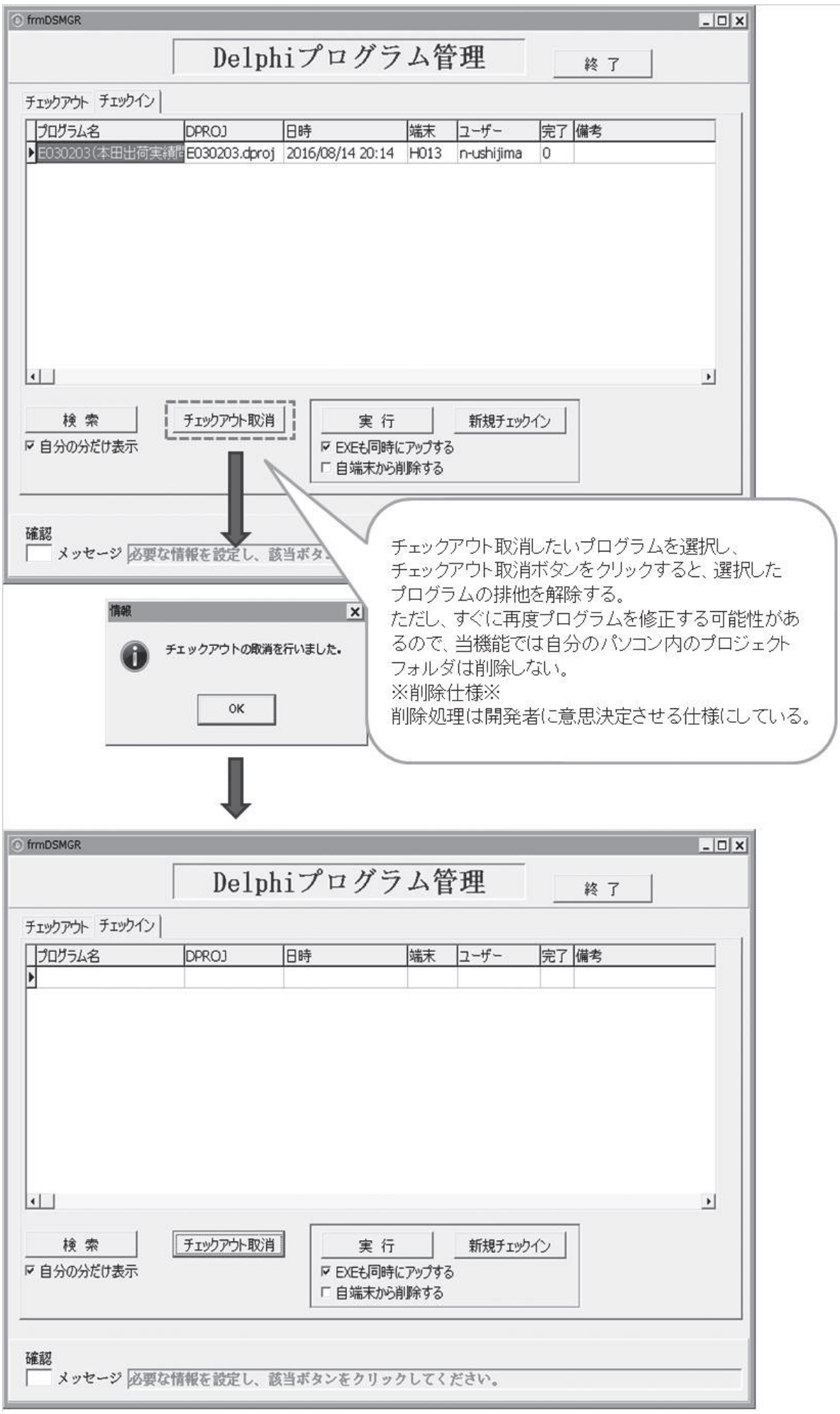
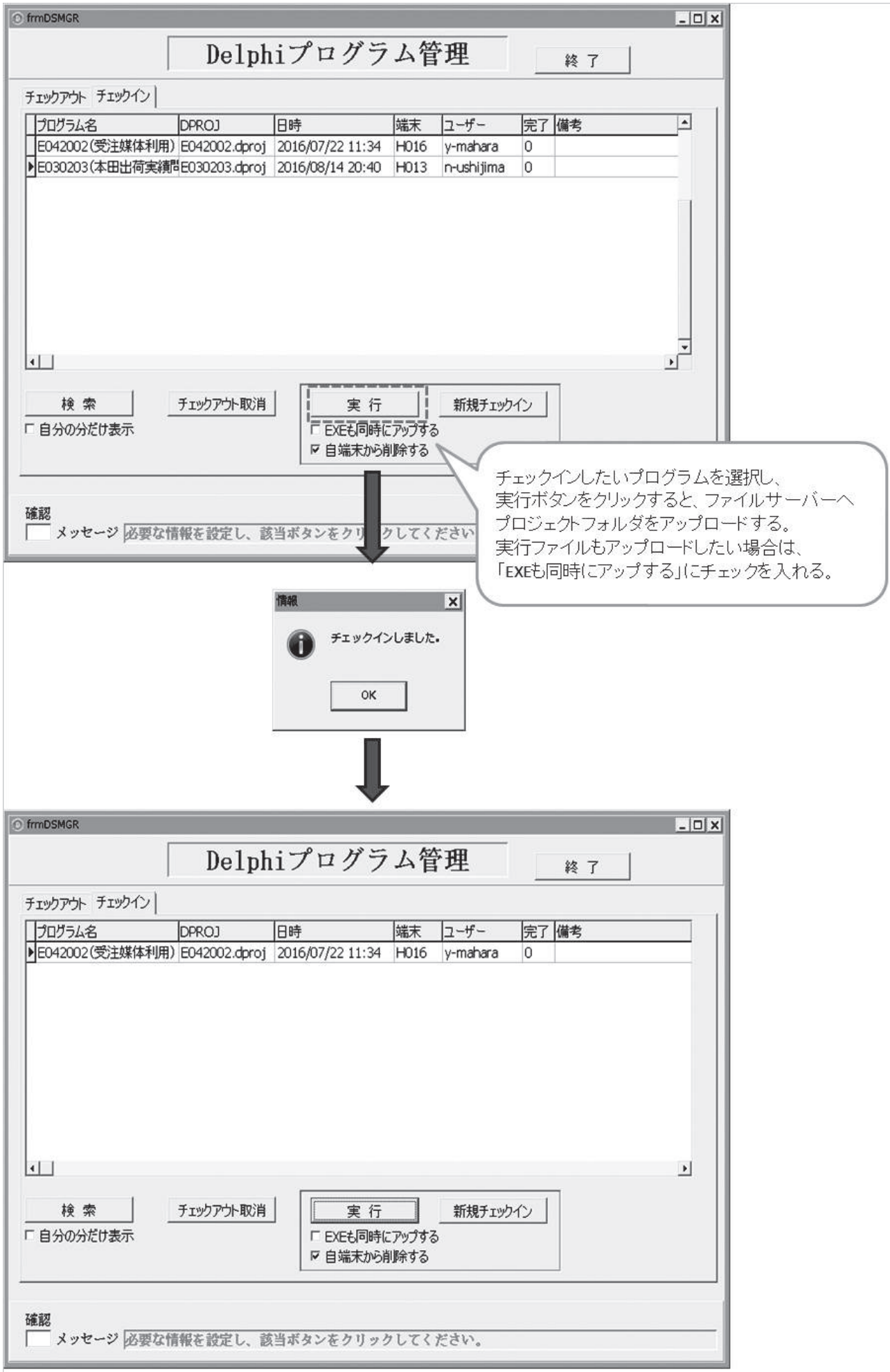


図5　チェックイン処理内容



ソース1　チェックアウト処理

```
{*****************************************************************************
 目的: チェックアウト処理
 引数:
 戻値:
*****************************************************************************}
procedure TfrmDSMGR.Check_Out;
var
  sInfolder,sInDir:string;
  CurDir : String;      //カレントディレクトリを格納する変数
  SelectFolder: String;  //フォルダのパスを格納する変数
  sREC:String;
  SHFILEOPSTRUCT : TSHFileOpStruct;
begin
  //プログラム分類名格納
  sInfolder :=
  Copy(Dir1.GetItemPath(Dir1.ItemIndex),Pos(Combo.Text,Dir1.GetItemPath(Dir1.ItemIndex) )
   + Length(Combo.Text) + 1);

  //チェックアウト先確認用パス設定(自端末)
  sIndir := CT_Path + combo.Text + '¥' + sInfolder + '¥';

  //自端末に同じフォルダがあるかチェックを行う。
  if DirectoryExists(sIndir) then
  begin
    MessageDlg('コピー先に同じフォルダがあります。確認してください。', mtInformation, [mbOK], 0);
    //procedure抜ける
    exit;
  end;

  //自端末コピー
  CurDir := Dir1.GetItemPath(Dir1.ItemIndex);        //from
  SelectFolder := CT_Path + combo.Text; //to
  //構造体の初期設定
  With SHFILEOPSTRUCT do
  begin
    wnd    := Handle;
    wFunc  := FO_COPY;
    pFrom  := PChar(CurDir + #0#0) ;
    pTo    := PChar(SelectFolder + #0#0);
    fFlags := FOF_ALLOWUNDO or FOF_FILESONLY;
    fAnyOperationsAborted := False;
    hNameMappings := nil;
    lpszProgressTitle := nil;
  end;
  //コピー実行
  SHFileOperation(SHFILEOPSTRUCT);

  with dmMain.cdsAccess do
  begin
    //ACCESSに排他フラグを立てる(1レコード追加)。
    Open;
    //追記モードオン
    Append;
    //追記レコードのパラメータ設定
    FieldByName('プログラム名').AsString   := sInfolder;
    FieldByName('DPROJ').AsString  := File1.Items[0];
    FieldByName('日時').AsString   := FormatDateTime('yyyy/mm/dd hh:mm', Now());
    FieldByName('端末').AsString   := GetComputerNameStr();
    FieldByName('ユーザー').AsString   := GetLoginNameStr();
    FieldByName('完了').AsString   := '0';
    FieldByName('PATH').AsString   := StringReplace(Dir1.GetItemPath(Dir1.ItemIndex),sInfolder,'', [rfReplaceAll, rfIgnoreCase]);
    //cds確定
    Post;
    //cdsとAccessの同期
    ApplyUpdates(-1);
    //レコードテキスト設定
    sREC := '''Check_Out','''' + FieldByName('プログラム名').AsString + ''','''' +
          FieldByName('日時').AsString + ''','''' + FieldByName('端末').AsString + ''','''' +
          FieldByName('ユーザー').AsString + '''';
    //ログファイル追記
    WriteText(SV_Path + 'DSMGR.LOG',sREC);
    //cdsクローズ
    Close;
  end;

  Dialogs.MessageDlg('チェックアウトしました。', mtInformation, [mbOK], 0);

end;
```

排他フラグセット

ソース2　チェックイン処理

```
{*****************************************************************************
 目的: チェックイン処理
 引数:
 戻値:
*****************************************************************************}
procedure TfrmDSMGR.Check_In;
var
  sInfolder,sInDir,sPrgNm:string;
  sYYYYMMDD,sHHMMSS:string;
  CurDir : String;      //カレントディレクトリを格納する変数
  SelectFolder: String;  //フォルダのパスを格納する変数
  SHFILEOPSTRUCT : TSHFileOpStruct;
  sExeName : string;
  sREC:string;
begin
  with dmMain.cdsAccess do
  begin
    //カレントレコードのプログラム名の変数格納
    sPrgNm := FieldByName('プログラム名').AsString;
    //年月日の変数格納
    sYYYYMMDD := FormatDateTime('yyyymmdd', Now());
    //時分秒の変数格納
    sHHMMSS := FormatDateTime('hhmmss', Now());

    //EXE展開の場合は、EXEが圧縮してあるか応答画面を表示させる
    //「はい」と「いいえ」で選択させる形式とし、「いいえ」だとこれ以降の処理をさせない
    if EXEUP.Checked then
    begin
      if MessageDlg('EXEは圧縮してますか？', mtConfirmation, [mbYes, mbNo], 0) = mrNo then
      begin
        Dialogs.MessageDlg('圧縮後、再度チェックイン作業を行ってください。', mtInformation, [mbYes], 0);
        //procedure抜ける
        exit;
      end;
    end;

    //選択したプログラムが同端末かチェックする
    if FieldByName('端末').AsString <> GetComputerNameStr() then
    begin
      Dialogs.MessageDlg('チェックアウトした端末ではない為、チェックインできません。', mtWarning, [mbYes], 0);
      exit;
    end;
    //選択したプログラムが同ユーザーかチェックする
    if FieldByName('ユーザー').AsString <> GetLoginNameStr() then
    begin
      Dialogs.MessageDlg('チェックアウトしたユーザーではない為、チェックインできません。', mtWarning, [mbYes], 0);
      exit;
    end;

    //プログラム分類名格納
    sInfolder := StringReplace(FieldByName('PATH').AsString,SV_Path,'', [rfReplaceAll, rfIgnoreCase]);
    //プログラムパス格納
    sIndir := CT_Path + sInfolder + sPrgNm;

    //自端末に同じフォルダがあるかチェックを行う
    if DirectoryExists(sIndir) = False then
    begin
      MessageDlg('コピー元にチェックアウトしたフォルダがありません。', mtInformation, [mbOK], 0);
      exit;
    end;

    //バックアップフォルダに変更前のソースをコピーする
    //バックアップ用フォルダがない場合は作成する
    if DirectoryExists(SVBK_Path + sYYYYMMDD + '¥' + sInfolder) = False then
    begin
      ForceDirectories(SVBK_Path + sYYYYMMDD + '¥' + sInfolder);
    end;
```

ソース2　チェックイン処理-2

```
//変更前のソース格納フォルダをコピーする。
CurDir := SV_Path + sInfolder + sPrgNm + '¥';   //from

//バックアップ先に同プログラム格納フォルダがあった場合は、時分秒を付けたフォルダを別途作成し、そこに格納する。
//【理由】同日中にチェックアウト＆チェックインした場合に、バックアップフォルダ自体を上書きされるのを防ぐため。
if DirectoryExists(SVBK_Path + sYYYYMMDD + '¥' + sInfolder + sPrgNm + '¥') = False then
begin
  //同日バックアップ先ダブりなしの場合
  SelectFolder := SVBK_Path + sYYYYMMDD + '¥' + sInfolder + sPrgNm + '¥'; //to
end
else
begin
  //プログラム格納フォルダの後ろに時分秒を付加したフォルダにコピーする
  SelectFolder := SVBK_Path + sYYYYMMDD + '¥' + sInfolder + sPrgNm + '_' + sHHMMSS + '¥'; //to
end;

//構造体の初期設定
With SHFILEOPSTRUCT do
begin
  wnd    := Handle;
  wFunc  := FO_COPY;
  pFrom  := PChar(CurDir + #0#0) ;
  pTo    := PChar(SelectFolder + #0#0);
  fFlags := FOF_ALLOWUNDO or FOF_FILESONLY;
  fAnyOperationsAborted := False;
  hNameMappings := nil;
  lpszProgressTitle := nil;
end;
//コピー実行
SHFileOperation(SHFILEOPSTRUCT);

//h222の変更前ソースを削除する
//誤ってscreenrealフォルダを削除してしまった。修正すること。
DeleteDirectory(SV_Path + sInfolder + sPrgNm);

//端末の変更後ソースをh222にコピーする
CurDir := sIndir + '¥';         //from
SelectFolder := SV_Path + sInfolder + sPrgNm + '¥'; //to
//構造体の初期設定
With SHFILEOPSTRUCT do
begin
  wnd    := Handle;
  wFunc  := FO_COPY;
  pFrom  := PChar(CurDir + #0#0) ;
  pTo    := PChar(SelectFolder + #0#0);
  fFlags := FOF_ALLOWUNDO or FOF_FILESONLY;
  fAnyOperationsAborted := False;
  hNameMappings := nil;
  lpszProgressTitle := nil;
end;
//コピー実行
SHFileOperation(SHFILEOPSTRUCT);

//EXE展開の場合⇒h222へコピー
if EXEUP.Checked then
begin
  //exe名称作成
  sExeName := StringReplace(FieldByName('DPROJ').AsString,'.dproj','', [rfReplaceAll, rfIgnoreCase]);

  if FileExists(CT_Path + 'Exe¥' + sExeName + '.exe') then
  //コピーするexeがある場合⇒コピー実行
  begin
    CopyFile(PChar(CT_Path + 'Exe¥' + sExeName + '.exe'),
            PChar(SVEXE_Path + sExeName + '.exe'),false);
  end
  else
  //コピーするEXEがない場合⇒警告メッセージ表示
```

ソース2　チェックイン処理-3

```
    begin
      Dialogs.MessageDlg('コピーするEXEがありませんでした。', mtWarning, [mbYes], 0);
    end;
  end;

  //管理用ACCESSの完了マークを'1'にする(カレントレコード)
  //編集モードオン
  Edit;
  //編集レコードのパラメータ設定
  FieldByName('完了').AsString := '1';
  //レコードテキスト設定
  sREC := '''Check_In'','''' + FieldByName('プログラム名').AsString + ''','''' +
        FieldByName('日時').AsString + ''','''' + FieldByName('端末').AsString + ''','''' +
        FieldByName('ユーザー').AsString + '''';
  //運用管理登録用ファイル追記
  WriteText(SV_Path + 'DSMGR.TXT',sREC);
  //ログファイル追記
  WriteText(SV_Path + 'DSMGR.LOG',sREC);
  //cds確定
  Post;
  //cdsとAccessの同期
  ApplyUpdates(-1);

  //端末側のソースを削除する_しないの選択肢から判断し、ソース削除処理を実行する
  if DEVELOP.Checked then
  begin
    DeleteDirectory(sIndir);
  end;

  Dialogs.MessageDlg('チェックインしました。', mtInformation, [mbOK], 0);
 end;

end;
```