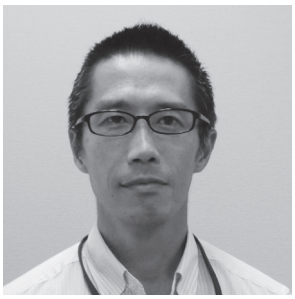
シルバー賞

Delphi/400でスプールファイル管理（WRKSPLFコマンドの活用）

ー i5コマンドを有効に活用する

三好 誠 様
ユサコ株式会社
システムグループ



ユサコ株式会社
http：//www.usaco.co.jp/

海外の学術雑誌、書籍の輸入販売を中心に事業を展開している。特に医学、薬学等の自然科学分野に強みを持つ。近年、学術情報媒体の多様化に伴い、電子ブック、データベース、各種ソフトウェアの取り扱いを強化。学術情報を通じ、知的情報の創造、蓄積、共有による社会貢献を目指している。

1.ユサコの事業内容、および当該システム関連部門・部署の事業内容

学術情報の提供を通じ、知的情報の創造、蓄積、共有のお手伝いをする会社

当社ユサコは創業以来、海外からの雑誌や書籍の輸入販売を行っており、近年では電子ジャーナル、データベースも取り扱う。雑誌や書籍等の輸入販売というと単なる書店の取次のように思われるが、医学や化学など学術研究に関連したものを中心に、知的情報の創造、蓄積、共有を通じて社会に貢献することを目指している。

この中で当方が所属するシステムグループは、少人数ながらも受発注を中心とした社内基幹システムの管理、開発から、インフラ整備やユーザーサポートまでトータルに会社を支えている。

2.アプリケーションの開発経緯、および概要

現行 System i5 のリプレースが急務

当社の基幹システムが稼働している System i5 は現在、リプレースの必要性が急務となっている。OS、ハードウェアともに導入から月日が経過しており、パフォーマンス、保守体制ともに、最新のものに切り替える時期が来ている。

現行システムを構築している CASE ツールがネック

単に OS やハードウェアを切り替えるだけなら、十分に検証を行い、ソースやオブジェクトを移行すればよい。しかし当社で一番問題となっているのは、現行システムの開発やコンパイル環境である CASE ツールだ。当時は 5250 画面やプログラムを簡単に自動生成してくれる便利なものであったが、このツールがなんと、新しい OS では動作が保障されない。

現行 System i5 の 5250 画面から、Delphi/400 による GUI への移行の準備

上記の問題があり 5250 画面を利用したままでは画面のメンテナンスができず、新しいハードウェアにも移行できない。かといって現在のプログラムを解析して同じ 5250 画面を再生成するのでは生産性がないし、今後の大量の CL や RPG を新たにメンテナンスするには、当社のシステムグループの人数では困難である。そこで、Delphi/400 による GUI 画面への移行を通じ、CASE ツールからの解放を目指しているところである。

GUI 化による OUTQ の管理、スプールファイルの見える化も必要

通常の対話画面はもちろんだが、System i5 の OUTQ を管理する機能も Delphi/400 に移行する上で必要だ。長年 System i5 を利用しているだけあって、コマンド自体は作成した画面から呼び出しているものの、WRKOUTQ コマンドや WRKSPLF コマンドの機能操作

図1　帳票サーバー利用の構成図

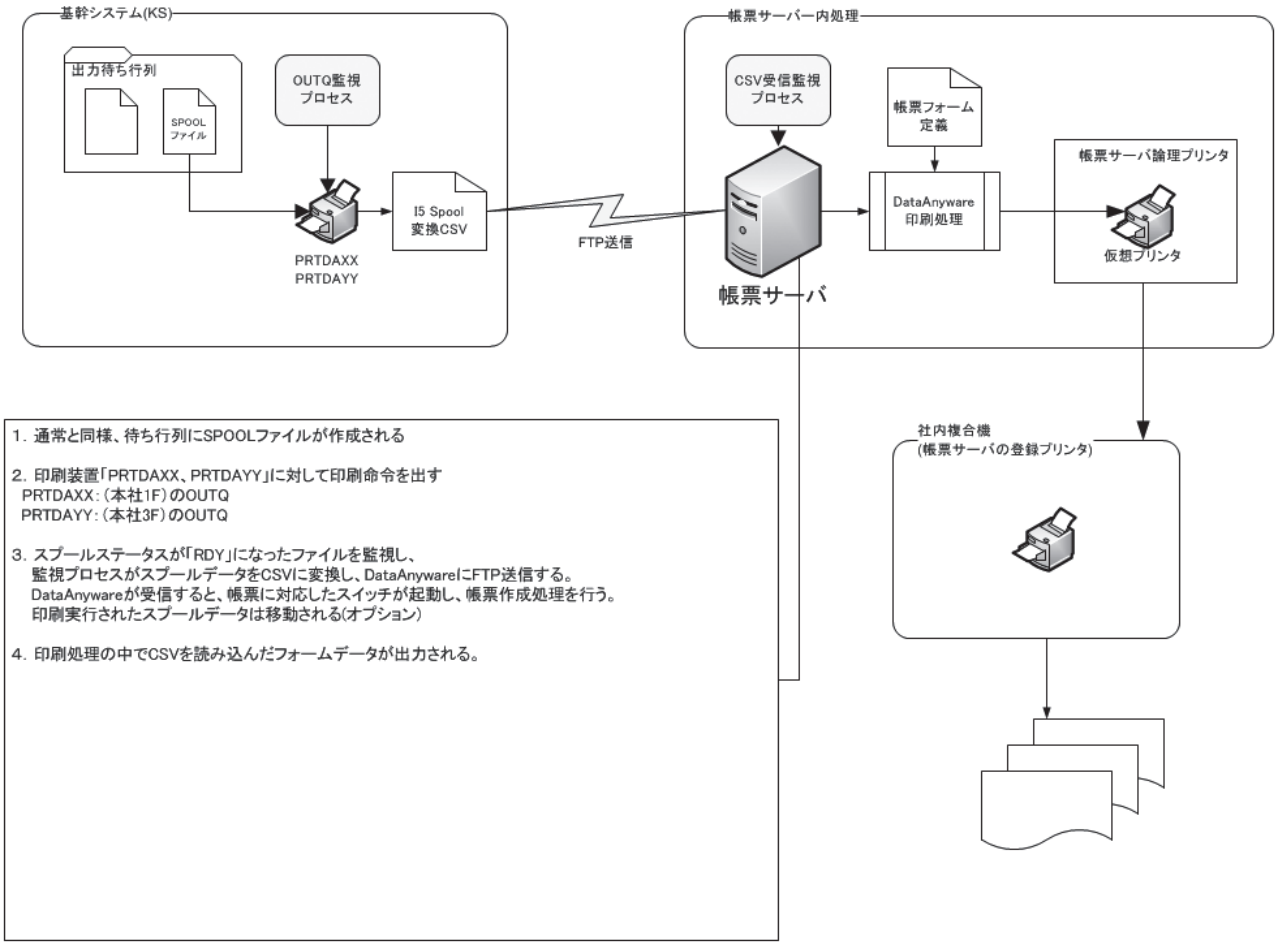
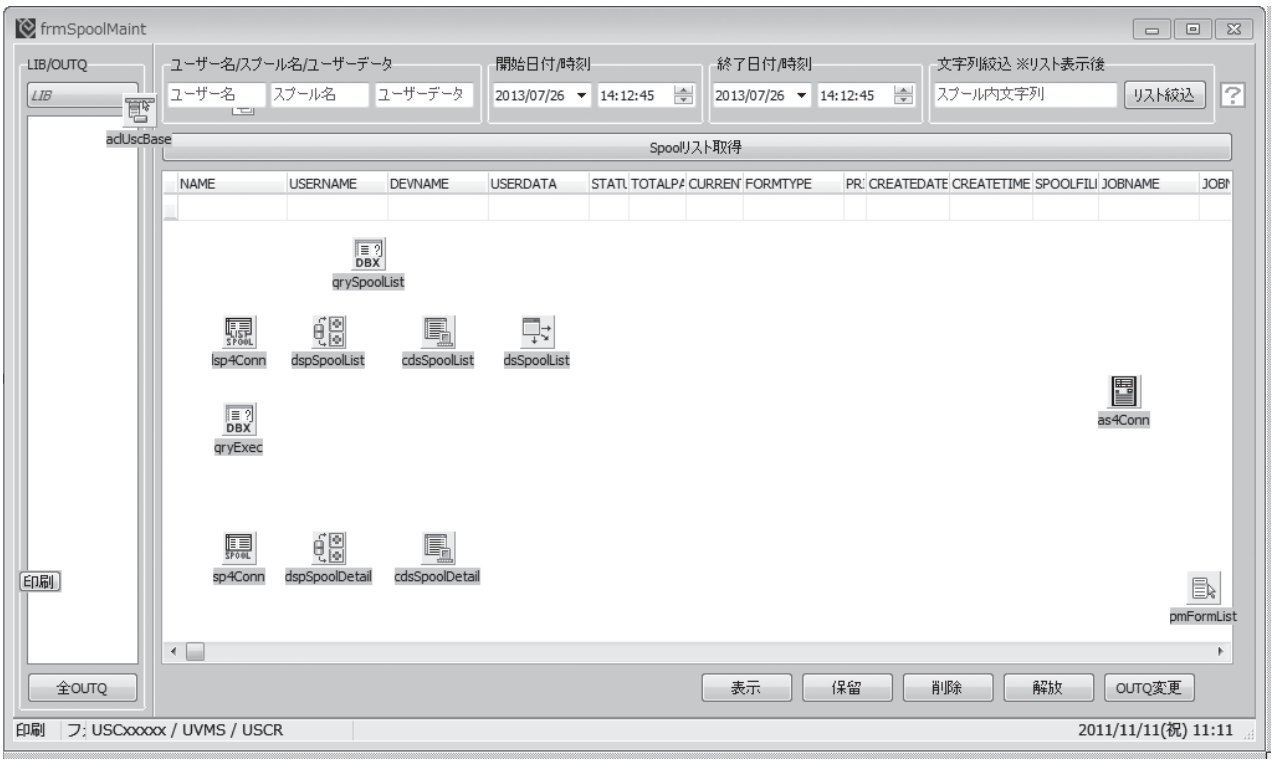


図2　ListSpool400利用時の開発画面



はユーザーも日々使用しており、熟練している。スプールファイルの移動､解放、表示などの機能は必須である。

多数のスプールファイル、OUTQ に対応できる機能要件

当社は帳票サーバーを介して一般プリンタに出力できる環境を整えており【図1】、プリンタ対応する OUTQ をそれぞれ準備しているが、そのほか保管用、出力済用に多くの OUTQ を用意している。また、当社の雑誌等商品の受注で、1 年で多くのトランザクションレコードが発生する。ゆえに注文書などを作成する際にはスプールファイルの数も多くなるし、繁忙期になると 1 つのスプールファイルに対する枚数も膨大になる。

5250 画面のようにスムーズな表示とアクションができることが必要だ。

3.取組内容（メインテーマ）

TListSpool400 ではライブラリーの指定と、スプールファイル数が多い時に課題

まず始めたのは、ツールとして提供されている TListSpool400 の活用である。非常に便利なコンポーネントであり、スプール一覧の取得に適しているため、当コンポーネントを採用して開発を始めた。スプール一覧を取得して、かつユーザー名、スプールファイル名、出力時間等で絞り込む機能を検討する。【図 2】

開発は順調に進み、いったんリリースを行ったが、検索対象が複数の OUTQ にまたがる時、またスプール件数が多い時、すべてのスプール一覧を取得してからの絞り込みしかできないため、非常に時間がかかることがわかった。複数の OUTQ を選択して OUTQ 横断検索を行える機能も備えていたが、時には数十分かかるため、実用に耐えない。

WRKSPLF の PRINT 出力を利用

i5 の最新 OS では簡単にできるのかもしれないが、スプールファイルの復元等を行う場合、CPYSPLF コマンドでスプールファイルをファイル化し、CPYF コマンドで復元印刷したりする。WRKSPLF コマンドは検索条件があらかじめ設定できることもあり【図 3】、条件さえ付加できれば結果を返すのが速い。PRINT オプションを付けることで、この速度を活かせないだろうか。

スプールの DB 化した結果を QTEMP に保存

上記 WRKSPLF コマンドを利用し、得られた結果のスプールファイルを CPYSPLF コマンドで DB 化し、QTEMP に保存する CL を作成する【図 4】。スプールファイルには当然 WRKSPLF のヘッダー等も含まれており、全体としては不定形の形だ。しかしフィールド分割したファイル【図 5】をスプールファイルのコピー先にしておけば、すべて文字列ながらエラーを発生させることなくファイルに取り込める。ヘッダーは取得時に取り除くこととする。

QTEMP に作成したファイルを Delphi から取得

ファイル化してしまえば、Delphi/400 から容易にアクセスできる。前述の通り、ファイル化したスプールファイルには不要なヘッダーデータも含まれているので、STATUS フィールドに必ず入る値で結果の絞り込み（RDY,HLD,SAV..）を行えば、スプール一覧だけに結果を絞り込める。WRKSPLF を PRINT 実行した時のスプールファイル（QPRTSPLF）も検索結果に出てきてしまうので、これも除外する SQL を準備しておく【図 6】。OUTQ だけは WRKSPLF コマンドのパラメータにないので、入力があれば条件に追加できるようにする。【図 7】

Delphi/400 から WRKSPLF コマンドの実行と、ClientDataSet への取得

SQL のベースができたら、検索実行時に WRKSPLF コマンドの実行と、取得した結果を ClientDataSet への取得を行う。【図 8】

スプール管理の一般的な機能に対応

ClientDataSet に取得さえできれば、スプールの保留から解放、移動まで容易に可能だ。取得したスプール情報から、それぞれコマンドを呼出し、実行できる機能を実装する。【図 9】

スプール文字列絞込の機能まで対応

絞り込んだスプール一覧から、印字されている文字列でフィルタリングできる機能があるとよい。スプール内容を読み取るには Spool400 コンポーネントがあるが、同様に印字ページが多いと、ClientDataSet に展開してしまうと時間がかかってしまう。文字列があるかどうかを知るだけなら、これも i5 上で DB 化したスプールファイルから文字列検索するのでよい。QTEMP にスプールデータを展開する CLP を作成し、SQL でこのファイルにアクセスする。【図 10】【図 11】【図 12】

スプール表示も CPYSPLF を使用して ClientDataSet に表示

スプール表示についても CPYSPLF を活用して代用する。ListSpool400、Spool400 は BDE を使用するが､この際、BDE 不要で Delphi/400 を動作できるようにした【図 13】。スプール文字列絞込で活用した CLP からスプールを SQL で取得する。CLP から呼び出す RPG でページ番号を振ってあるので【図 14】、この番号でフィルタ機能を利用し、疑似ページめくり機能を有効にする【図 15】。表示したスプールからも、Windows で行うような、文字列検索をできるようにした。【図 16】

検索結果の文字列の場所を覚えておき、次に検索が実行された時、次に見つかる文字列からスタートする。

4.ノウハウ、教訓、エンドユーザからの評価、今後の予定･計画

GUI 画面へは移行の途中であり、スプール管理画面もまだ、積極的には利用されておらず、ユーザーからの反応はまだ多くは得られていない。しかしながら、出力したスプールを検索できる機能については、今後大いに効率を上げられる見込みだ。GUI ベース画面への移行が進んでいけば現行のスプールを使用する必要もなくなり、Delphi/400 で利用できるレポートツールに切り替えていくことになるだろうが、当面はこの機能を役立てることになるだろう。

M

図3　WRKSPLFのパラメータ

図4　WRKSPLFの結果をQTEMPに保存するCLP

```
        PGM       PARM(&USER &USRDTA &SPLF &STDATE &STTIME &EDDATE +
                    &EDTIME &CURSS &MSGID &MSG)

/*****************************************************************/
/*表題:WRKSPLFの結果をQTEMPにファイル保存する                    */
/*****************************************************************/

        /*パラメータ*/
        DCL       VAR(&USER) TYPE(*CHAR) LEN(10)
        DCL       VAR(&USRDTA) TYPE(*CHAR) LEN(10)
        DCL       VAR(&SPLF) TYPE(*CHAR) LEN(10)
        DCL       VAR(&STDATE) TYPE(*CHAR) LEN(8)
        DCL       VAR(&STTIME) TYPE(*CHAR) LEN(6)
        DCL       VAR(&EDDATE) TYPE(*CHAR) LEN(8)
        DCL       VAR(&EDTIME) TYPE(*CHAR) LEN(6)
        DCL       VAR(&CURSS) TYPE(*CHAR) LEN(1)
        DCL       VAR(&MSGID) TYPE(*CHAR) LEN(7)
        DCL       VAR(&MSG) TYPE(*CHAR) LEN(132)
        /* RTVJOBA */
        DCL       VAR(&JOBNAME) TYPE(*CHAR) LEN(10)
        DCL       VAR(&JOBUSER) TYPE(*CHAR) LEN(10)
        DCL       VAR(&JOBNBR) TYPE(*CHAR) LEN(6)
        /* OUTPUT LIB/FILE */
        DCL       VAR(&TPLIB) TYPE(*CHAR) LEN(10)
        DCL       VAR(&SPFILE) TYPE(*CHAR) LEN(10)

@START:
        CHGVAR    VAR(&TPLIB) VALUE('QTEMP') /* QTEMP */
        CHGVAR    VAR(&SPFILE) VALUE('WRKSPLFL') /*SPLF TO FILE*/

        RTVJOBA   JOB(&JOBNAME) USER(&JOBUSER) NBR(&JOBNBR)
        OVRPRTF   FILE(QPRTSPLF) OUTQ(PRT01) /* SPLFNAME(&SPFILE) */
        MONMSG    MSGID(CPF0000) EXEC(GOTO @ERR)

        /*ユーザーセッション'Y'のとき、ジョブ情報で検索*/
        IF        COND(&CURSS *EQ 'Y') THEN(DO)
          WRKSPLF   SELECT(*ALL *ALL *ALL &USRDTA *ALL &SPLF) +
                      JOB(&JOBNBR/&JOBUSER/&JOBNAME) +
                      PERIOD((&STTIME &STDATE) (&EDTIME &EDDATE)) +
                      OUTPUT(*PRINT)
          MONMSG    MSGID(CPF0000) EXEC(GOTO @ERR)
        ENDDO
        ELSE      DO
          WRKSPLF   SELECT(&USER *ALL *ALL &USRDTA *ALL &SPLF) +
                      PERIOD((&STTIME &STDATE) (&EDTIME &EDDATE)) +
                      OUTPUT(*PRINT)
          MONMSG    MSGID(CPF0000) EXEC(GOTO @ERR)
        ENDDO

        DLTF      FILE(&TPLIB/&SPFILE)
        MONMSG    MSGID(CPF0000)

        CRTPF     FILE(&TPLIB/&SPFILE) SRCFILE(COMMONLIB/QDDSSRC) +
                    SRCMBR(SPFILED) IGCDTA(*NO) OPTION(*NOLIST +
                    *NOSOURCE) MAXMBRS(*NOMAX) SIZE(*NOMAX)
        MONMSG    MSGID(CPF0000) EXEC(GOTO @ERR)

        CPYSPLF   FILE(QPRTSPLF) TOFILE(&TPLIB/&SPFILE) +
                    JOB(&JOBNBR/&JOBUSER/&JOBNAME) SPLNBR(*LAST) +
                    CTLCHAR(*FCFC)
        MONMSG    MSGID(CPF0000) EXEC(GOTO @ERR)
        GOTO      CMDLBL(@END)
@ERR:
        RCVMSG    MSGTYPE(*LAST) MSGID(&MSGID) MSG(&MSG)
@END:     ENDPGM
```

図5　分割ファイルのDDS（SPFILED）

No	PK	内部名	英字名	表記	型	位置	長	桁	小数点	DEF	Null可
1		SPCTRL		印刷制御文字	A	1	1	0	0		N
2		SPBLANK			A	2	1	0	0		N
3		BLANK01			A	3	1	0	0		N
4		SPLNAM	Name	スプール名	A	4	10	0	0		N
5		BLANK02			A	14	1	0	0		N
6		USRNAM	UserName	ユーザ名	A	15	10	0	0		N
7		BLANK03			A	25	1	0	0		N
8		DEVNAM	DevName	装置名	A	26	10	0	0		N
9		BLANK04			A	36	1	0	0		N
10		USRDTA	UserData	ユーザーデータ	O	37	10	0	0		N
11		BLANK05			A	47	1	0	0		N
12		STATUS	Status	ステータス	A	48	3	0	0		N
13		BLANK06			A	51	2	0	0		N
14		TTLPAG	TotalPages	総ページ数	A	53	5	0	0		N
15		BLANK07			A	58	6	0	0		N
16		CURPAG	CurrentPage	現在ページ	A	64	5	0	0		N
17		BLANK08			A	69	1	0	0		N
18		FRMTYP	FormType	フォームタイプ	O	70	10	0	0		N
19		BLANK09			A	80	2	0	0		N
20		PRIOTY	Priority	優先順位	A	82	1	0	0		N
21		BLANK10			A	83	2	0	0		N
22		CRTDAT	CreateDate	作成日	A	85	8	0	0		N
23		BLANK11			A	93	1	0	0		N
24		CRTTIM	CreateTime	作成時刻	A	94	8	0	0		N
25		BLANK12			A	102	1	0	0		N
26		SPLNO	SpoolFileNumber	スプールNo	A	103	6	0	0		N
27		BLANK13			A	109	1	0	0		N
28		JOBNAM	JobName	ジョブ名	A	110	10	0	0		N
29		BLANK14			A	120	1	0	0		N
30		JOBNO	JobNumber	ジョブNo	A	121	6	0	0		N
31		BLANK15			A	127	1	0	0		N
32		OUTNAM	OutqName	OUTQ名	A	128	10	0	0		N
33		BLANK16			A	138	1	0	0		N
34		OUTLIB	OutqLibraryName	OUTQライブラリー	A	139	10	0	0		N
35		BLANK17			A	149	2	0	0		N
36		ASP	ASP	ASP	A	151	3	0	0		N
37		BLANK18			A	154	1	0	0		N
38		LSTDAT	LastUseDate	最終使用日	A	155	8	0	0		N
39		BLANK19			A	163	1	0	0		N
40		SPLSIZ	SpoolFileSize	スプールファイルサイズ	A	164	9	0	0		N
41		BLANK20			A	173	30	0	0		N

※CPYSPLF実行時に追加される
※CPYSPLF実行時に追加されるブランク

図6　QTEMPのWRKSPLF結果（WRKSPLFL）にアクセスするSQLベース

```
//スプールファイル取得用SQL文のベース
SQLSPBase:='SELECT ';
SQLSPBase:=SQLSPBase+' TRIM(SPLNAM) AS Name, ';
SQLSPBase:=SQLSPBase+' TRIM(USRNAM) AS UserName, ';
SQLSPBase:=SQLSPBase+' TRIM(DEVNAM) AS DevName, ';
SQLSPBase:=SQLSPBase+' TRIM(USRDTA) AS UserData, ';
SQLSPBase:=SQLSPBase+' TRIM(STATUS) AS Status, ';
SQLSPBase:=SQLSPBase+' TRIM(TTLPAG) AS TotalPages, ';
SQLSPBase:=SQLSPBase+' TRIM(CURPAG) AS CurrentPage, ';
SQLSPBase:=SQLSPBase+' TRIM(FRMTYP) AS FormType, ';
SQLSPBase:=SQLSPBase+' PRIOTY AS Priority, ';
SQLSPBase:=SQLSPBase+' TRIM(CRTDAT) AS CreateDate, ';
SQLSPBase:=SQLSPBase+' TRIM(CRTTIM) AS CreateTime, ';
SQLSPBase:=SQLSPBase+' TRIM(SPLNO) AS SpoolFileNumber, ';
SQLSPBase:=SQLSPBase+' TRIM(JOBNAM) AS JobName, ';
SQLSPBase:=SQLSPBase+' TRIM(JOBNO) AS JobNumber, ';
SQLSPBase:=SQLSPBase+' TRIM(OUTNAM) AS OutqName, ';
SQLSPBase:=SQLSPBase+' TRIM(OUTLIB) AS OutqLibraryName, ';
SQLSPBase:=SQLSPBase+' TRIM(ASP) AS ASP, ';
SQLSPBase:=SQLSPBase+' TRIM(LSTDAT) AS LastUseDate, ';
SQLSPBase:=SQLSPBase+' TRIM(SPLSIZ) AS SpoolFileSize ';
SQLSPBase:=SQLSPBase+' FROM QTEMP.WRKSPLFL ';
SQLSPBase:=SQLSPBase+' WHERE TRIM(STATUS)
    IN (''RDY'',''OPN'',''DFR'',''SND'',''CLO'',''HLD'',''SAV'',''WTR'',''FIN'',''PND'',''PRT'',''MSGW'') ';
   //ヘッダー等も含まれてくるため
SQLSPBase:=SQLSPBase+' AND TRIM(SPLNAM) <> ''QPRTSPLF'' '; //WRKSPLF印刷原本は除く
```

WHERE条件でSTATUSでの値絞込と、WRKSPLF印刷の原本を除外する。

図7　動的にWHERE文の追加

```
{-------------------------------------------------------------------------------
* 表題:Where文以下の条件式を返す
* 引数1[i]:PrmName:条件式で使用されるパラメータ名
* 引数2[i/o]:FstCon:条件式の先頭かどうか ※参照渡し
* 戻値:条件式
*-------------------------------------------------------------------------------}
function TfrmSpoolMaint.fGetWhereCon(PrmName: string; var FstCon: Boolean): string;
var
  strSQL: string;
begin

    strSQL:=' AND ';

  if PrmName = 'pOUTQ' then begin
//    strSQL:=strSQL+' UPPER(TRIM(SUBSTRING(SPTEXT,126,10))) = :pOUTQ '; //フィールド分割前
    strSQL:=strSQL+' UPPER(TRIM(OUTNAM))) = :pOUTQ '; //ベンダーCD
  end;
```

図8　WRKSPLF結果の取得

```
{-------------------------------------------------------------------------------
* 表題:検索クエリの実行
*-------------------------------------------------------------------------------}
procedure TfrmSpoolMaint.fOpenQuery;
var
  strSQL: string;
  fstCondition: boolean; //条件式の先頭かどうか
  intRecCnt: Integer;
  strCmd: string;
  strUserName, strSpoolName, strUserData: string;
  strStartDate,strStartTime: string;
  strEndDate,strEndTime: string;
  strCurSession: string;
  SaveCursor: TCursor; //現在のマウスカーソル
begin
  { SQL文の作成 }

  if (edtUserName.Text='') and (edtUserData.Text='') and (edtSpoolName.Text='') and (edtOUTQ.Text='') then begin
    if cfMessageDlgEx('ユーザー名、ユーザーデータ、スプールファイル名、
    OUTQがブランクの場合時間がかかる場合がありますが、実行しますか？',
    mtConfirmation, [mbYes,mbNo], 2) <> mrYes then Abort;
  end;

  //WRKSPLF→QTEMP/WRKSPLFTOFへ。画面入力がないとき*ALLで取得
  if edtUserName.Text='' then strUserName:='*ALL' else strUserName:=edtUserName.Text; //ユーザー名
  if edtUserData.Text='' then strUserData:='*ALL' else strUserData:=edtUserData.Text; //ユーザーデータ
  if edtSpoolName.Text='' then strSpoolName:='*ALL' else strSpoolName:=edtSpoolName.Text; //スプールファイル名
  if chkCurSession.Checked then strCurSession:='Y' else strCurSession:=''; //現在のセッション

  DateTimeToString(strStartDate, 'yyyyMMdd',dtpStartDate.DateTime);
  DateTimeToString(strStartTime, 'HHmmss',dtpStartTime.DateTime);
  DateTimeToString(strEndDate, 'yyyyMMdd',dtpEndDate.DateTime);
  DateTimeToString(strEndTime, 'HHmmss',dtpEndTime.DateTime);
  strCmd:='CALL PGM(COMMONLIB/WRKSPLFTOF) ';
  strCmd:=strCmd+' PARM('''+strUserName+''' '''+strUserData+''' '''+strSpoolName+''' ';
  strCmd:=strCmd+' '''+strStartDate+''' '''+strStartTime+''' '''+strEndDate+''' '''+strEndTime+''' '''+strCurSession+''' ''''''')';

  fdmCommon.as4Main.RemoteCmd(strCmd); //WRKSPLF実行

  //ここから結果FILEをSQL
  strSQL:=SQLSPBase;
  fstCondition:=True;
  if edtOUTQ.Text <> '' then  strSQL:=strSQL+fGetWhereCon('pOUTQ',fstCondition); //OUTQ条件の追加
  //--------------------------------

  SaveCursor:=Screen.Cursor;
  Self.Enabled:=False;
  try//マウスカーソル操作
    Screen.Cursor:=crHourGlass; //砂時計
    { 件数の事前確認 }
    qryCount.Close;
    qryCount.SQL.Clear;
    qryCount.SQL.Add('SELECT COUNT(*) FROM ( '+strSQL+' ) src');
    fSetSQLParam(qryCount); //パラメータ設定
    qryCount.Open;
    Application.ProcessMessages; //応答なしメッセージ回避
    intRecCnt:=qryCount.Fields[0].AsInteger;
    qryCount.Close; //セッションの解除
    if intRecCnt >= 200 then begin
      if cfMessageDlgEx('検索結果が200件以上('+IntToStr(intRecCnt)+'件)ありますが、実行しますか？',
      mtConfirmation, [mbYes,mbNo], 2) <> mrYes then Abort;
    end;
    //--------------------------------

    { 検索処理 }
    cdsSpoolList.Close;
    cdsSpoolList.IndexName:=''; //並び替えクリア
    qrySpoolList.SQL.Clear;
    qrySpoolList.SQL.Add(strSQL);
    fSetSQLParam(qrySpoolList); //パラメータ設定
    cdsSpoolList.Open;
    Application.ProcessMessages; //応答なしメッセージ回避
    //--------------------------------

    if (cdsSpoolList.RecordCount>0) then begin //対象リストがあるとき
      edtStrings.Enabled:=True;
      btnStringsFilter.Enabled:=True;
      btnStringsFilterOff.Enabled:=True;
    end
    else begin
      edtStrings.Enabled:=False;
      btnStringsFilter.Enabled:=False;
      btnStringsFilterOff.Enabled:=False;
    end;

    lblRecordCount.Caption:=cfGetRecCnt(cdsSpoolList,''); //レコード数の表示
  finally
    Screen.Cursor:=SaveCursor; //保存していたカーソルに戻す
    Self.Enabled:=True;
  end;

end; ┗
```

図9　スプール管理コマンド

```
{-------------------------------------------------------------------------------
* 表題:Spool解放ボタン(RLSSPLF)
*-------------------------------------------------------------------------------}
procedure TfrmSpoolMaint.btnRelease_OnClick(Sender: TObject);
var
  i: Integer;
begin
  inherited;

  if cdsSpoolList.Active=False then Exit;
  if cdsSpoolList.RecordCount=0 then Exit;

  try
    //選択状態になっているか
    if dbgMain.SelectedRows.Count > 0 then begin
      for i:=0 to dbgMain.SelectedRows.Count - 1 do begin
        //解放実行
        cdsSpoolList.Bookmark:=dbgMain.SelectedRows[i];
        fReleaseSplF;//RLSSPLF実行
        if not cdsSpoolList.Modified then cdsSpoolList.Edit;
        cdsSpoolList.FieldByName('STATUS').AsString:='*RELEASE';
      end;
    end else begin
    end;
  except
    on e: Exception do begin //Exceptionクラスは全例外クラスの基本クラスであり、全例外をトラップ出来る
      ShowMessage('Exception:'+e.ClassName+'/'+e.Message);
    end;
  end;

end;

{-------------------------------------------------------------------------------
* 表題:Spool削除ボタン(DLTSPLF)
*-------------------------------------------------------------------------------}
procedure TfrmSpoolMaint.bbtnDelete_OnClick(Sender: TObject);
var
  i: Integer;
begin
  inherited;

  try
    //選択状態になっているか
    if dbgMain.SelectedRows.Count > 0 then begin
      for i:=0 to dbgMain.SelectedRows.Count - 1 do begin
        //削除確認
        if cfMessageDlgEx('削除しますか？', mtConfirmation, [mbYes,mbNo], 2) <> mrYes then Abort;
        //削除実行
        cdsSpoolList.Bookmark:=dbgMain.SelectedRows[i];
        fDeleteSplF;//DLTSPLF実行
        if not cdsSpoolList.Modified then cdsSpoolList.Edit;
        cdsSpoolList.FieldByName('STATUS').AsString:='*DELETE';
      end;
    end else begin
    end;
  except
    on e: Exception do begin //Exceptionクラスは全例外クラスの基本クラスであり、全例外をトラップ出来る
      ShowMessage('Exception:'+e.ClassName+'/'+e.Message);
    end;
  end;

end;
```

```
{-----------------------------------------------------------------------------
* 表題:CHGSPLFA実行
*-----------------------------------------------------------------------------}
procedure TfrmSpoolMaint.btnOUTQChange_OnClick(Sender: TObject);
var
  i: Integer;
  strLIB: String;
  strOUTQ: String;
begin
  inherited;

  if cdsSpoolList.Active=False then Exit;
  if cdsSpoolList.RecordCount=0 then Exit;
  if edtChgOUTQ.Text='' then begin
    ShowMessage('変更先OUTQを指定して下さい。');
    Abort;
  end;

  strLIB:='*LIBL';
  strOUTQ:=edtChgOUTQ.Text;

  try
    //選択状態になっているか
    if dbgMain.SelectedRows.Count > 0 then begin
      for i:=0 to dbgMain.SelectedRows.Count - 1 do begin
        //解放実行
        cdsSpoolList.Bookmark:=dbgMain.SelectedRows[i];
        fChangeSplF(StrLIB,StrOUTQ);//CHGSPLFA実行
        if not cdsSpoolList.Modified then cdsSpoolList.Edit;
        cdsSpoolList.FieldByName('STATUS').AsString:='*CHANGE';
        cdsSpoolList.FieldByName('OUTQLIBRARYNAME').AsString:=strLIB;
        cdsSpoolList.FieldByName('OUTQNAME').AsString:=strOUTQ;
      end;
    end else begin
    end;
  except
    on e: Exception do begin //Exceptionクラスは全例外クラスの基本クラスであり、全例外をトラップ出来る
      ShowMessage('Exception:'+e.ClassName+'/'+e.Message);
    end;
  end;
end;
{-----------------------------------------------------------------------------
* 表題:HLDSPLF実行
*-----------------------------------------------------------------------------}
procedure TfrmSpoolMaint.fHoldSplF;
var
  strCmd: String;
  strSplF,strJobName,strJobUser,strJobNumber,strSplNumber: String;
begin
  with cdsSpoolList do begin
    strSplF:=Trim(FieldByName('NAME').AsString);
    strJobName:=Trim(FieldByName('JOBNAME').AsString);
    strJobNumber:=Trim(FieldByName('JOBNUMBER').AsString);
    strJobUser:=Trim(FieldByName('USERNAME').AsString);
    strSplNumber:=Trim(FieldByName('SPOOLFILENUMBER').AsString);
  end;

  strCmd:='HLDSPLF FILE('+strSplF+') '+
          'JOB('+strJobNumber+'/'+strJobUser+'/'+strJobName+') '+
          'SPLNBR('+strSplNumber+')';
  fdmCommon.as4Main.RemoteCmd(strCmd);
end;
```

図10　スプールデータのQTEMP取得

```
            PGM         PARM(&SPLNAME &JOBNBR &JOBUSER &JOBNAME &SPLNMBR +
                          &MSGID &MSG)

/*************************************************************************/
/*表題:CPYSPLFの結果をQTEMPにファイル保存する                            */
/*************************************************************************/

            /*パラメータ*/
            DCL         VAR(&SPLNAME) TYPE(*CHAR) LEN(10)
            DCL         VAR(&JOBNAME) TYPE(*CHAR) LEN(10)
            DCL         VAR(&JOBUSER) TYPE(*CHAR) LEN(10)
            DCL         VAR(&JOBNBR) TYPE(*CHAR) LEN(6)
            DCL         VAR(&SPLNMBR) TYPE(*CHAR) LEN(10)
            DCL         VAR(&MSGID) TYPE(*CHAR) LEN(7)
            DCL         VAR(&MSG) TYPE(*CHAR) LEN(132)
            /* RTVJOBA */
            /* OUTPUT LIB/FILE */
            DCL         VAR(&TPLIB) TYPE(*CHAR) LEN(10)
            DCL         VAR(&SPFILE) TYPE(*CHAR) LEN(10)

 @START:
            CHGVAR      VAR(&TPLIB) VALUE('QTEMP') /* QTEMP */
            CHGVAR      VAR(&SPFILE) VALUE('CPYSPLFL') /*SPLF TO FILE*/

            DLTF        FILE(&TPLIB/&SPFILE)
            MONMSG      MSGID(CPF0000)

            CRTPF       FILE(&TPLIB/&SPFILE) SRCFILE(COMMONLIB/QDDSSRC) +
                          SRCMBR(SPFILEC) IGCDTA(*NO) OPTION(*NOLIST +
                          *NOSOURCE) MAXMBRS(*NOMAX) SIZE(*NOMAX)
            MONMSG      MSGID(CPF0000) EXEC(GOTO @ERR)

            CPYSPLF     FILE(&SPLNAME) TOFILE(&TPLIB/&SPFILE) +
                          JOB(&JOBNBR/&JOBUSER/&JOBNAME) SPLNBR(&SPLNMBR) +
                          CTLCHAR(*FCFC)
            MONMSG      MSGID(CPF0000) EXEC(GOTO @ERR)

            /*ページ番号採番*/
            CALL        PGM(COMMONLIB/SPPAGEUP)
            MONMSG      MSGID(CPF0000) EXEC(GOTO @ERR)
            GOTO        CMDLBL(@END)
 @ERR:
            RCVMSG      MSGTYPE(*LAST) MSGID(&MSGID) MSG(&MSG)
 @END:       ENDPGM
```

図11　DB化スプールファイルのDDS（SPFILEC）

No	PK	内部名	英字名	表記	型	位置	長	桁	小数点	DEF	Null可	
1		SPCTRL		印刷制御文字	A	1	1	0	0		N	
2		SPBLANK			A	2	1	0	0		N	
3		SPTEXT		TEXT	O	3	200	0	0		N	※スプール文字列
4		SPPAGE		PAGE	A	203	9	0	0		N	※ページ番号

※DDSは上記の名前となっているが、初めからSPYSPLFLでも良い。

図12　取得一覧から印字文字列で絞込み

```
{-------------------------------------------------------------------------------
* 表題:リスト絞込ボタン押下
*-------------------------------------------------------------------------------}
procedure TfrmSpoolMaint.btnStringsFilter_OnClick(Sender: TObject);
var
  strCmd: string;
  intRecCnt: Integer;
  SaveCursor: TCursor; //現在のマウスカーソル
begin
  inherited;

  if cdsSpoolList.Active=False then Exit;
  if cdsSpoolList.RecordCount=0 then Exit;

  cdsSpoolList.First;

  prbProgressBar.Visible:=True;
  prbProgressBar.Position:=0;
  prbProgressBar.Max:=cdsSpoolList.RecordCount;

  SaveCursor:=Screen.Cursor;
  Self.Enabled:=False;
  try//マウスカーソル操作
    Screen.Cursor:=crHourGlass; //砂時計
    try

      while not cdsSpoolList.Eof do begin
        strCmd:='CALL PGM(COMMONLIB/CPYSPLFTOF) ';
        strCmd:=strCmd+' PARM(';
        strCmd:=strCmd+' ""+cdsSpoolList.FieldByName('NAME').AsString+""; //スプール名
        strCmd:=strCmd+' ""+cdsSpoolList.FieldByName('JOBNUMBER').AsString+""; //スプール名
        strCmd:=strCmd+' ""+cdsSpoolList.FieldByName('USERNAME').AsString+""; //ユーザー名
        strCmd:=strCmd+' ""+cdsSpoolList.FieldByName('JOBNAME').AsString+""; //ジョブ名
        strCmd:=strCmd+' ""+cdsSpoolList.FieldByName('SPOOLFILENUMBER').AsString+""; //スプール番号
        strCmd:=strCmd+' "" "" )';
        fdmCommon.as4Main.RemoteCmd(strCmd); //WRKSPLF実行

        { 件数の事前確認 }
        qryCount.Close;
        qryCount.SQL.Clear;
        qryCount.SQL.Add('SELECT COUNT(*) FROM QTEMP.CPYSPLFL WHERE UPPER(SPTEXT)
          LIKE '+cfQStr('%'+UpperCase(edtStrings.Text)+'%'));
        fSetSQLParam(qryCount); //パラメータ設定
        qryCount.Open;
        Application.ProcessMessages; //応答なしメッセージ回避
        intRecCnt:=qryCount.Fields[0].AsInteger;
        qryCount.Close; //セッションの解除

        if intRecCnt = 0 then cdsSpoolList.Delete
        else cdsSpoolList.Next;

        if prbProgressBar.Position<prbProgressBar.Max then prbProgressBar.StepIt//進捗進行
        else cdsSpoolList.Last;//終了

        prbProgressBar.Repaint;
        dbgMain.Repaint;

      end;

      lblRecordCount.Caption:=cfGetRecCnt(cdsSpoolList,''); //レコード数の表示
    except
      on e: Exception do begin //Exceptionクラスは全例外クラスの基本クラスであり、全例外をトラップ出来る
        ShowMessage('Exception:'+e.ClassName+'/'+e.Message);
      end;
    end;
  finally
    Screen.Cursor:=SaveCursor; //保存していたカーソルに戻す
    Self.Enabled:=True;
    prbProgressBar.Visible:=False;
  end;

end;
```

図13　スプール表示画面

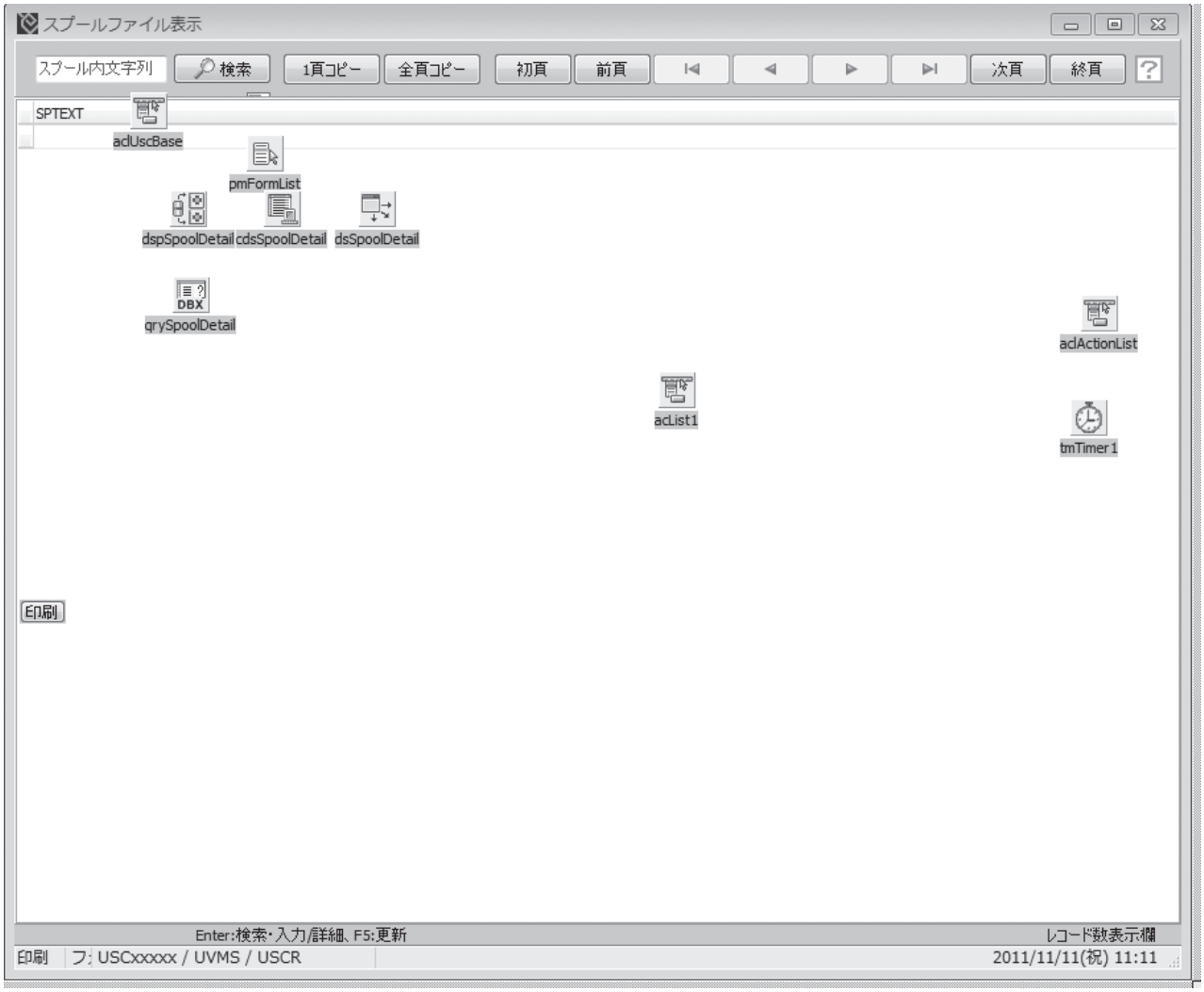


図14　スプールデータのページカウントRPG(LE)

```
FCPYSPLFL  UP   E           DISK
D* SQL RETURN CD
D CNT             S             5S 0
 /FREE
     IF SPCTRL='1';
       CNT = CNT + 1;
     ENDIF;
     SPPAGE=CNT;
     UPDATE    SPRECD;
 /END-FREE
```

SPCTRLに制御文字'1'があればページカウントして更新

図15　スプール表示のページ移動

```
{-----------------------------------------------------------------------------
* 表題:初頁ボタン実行
*-----------------------------------------------------------------------------}
procedure TfrmSpoolDisp.btnFirst_OnClick(Sender: TObject);
var
  intCurPage: Integer;
begin
  inherited;

  intCurPage:=1;

  //開始頁
  cdsSpoolDetail.Filter:='SPPAGE='+IntToStr(intCurPage);
  fPageChange;
  lblRecordCount.Caption:=fGetPageCnt(intCurPage,intMaxPage);

  fSelectInfo_Clear;//選択文字列情報のクリア

end;

{-----------------------------------------------------------------------------
* 表題:終頁ボタン実行
*-----------------------------------------------------------------------------}
procedure TfrmSpoolDisp.btnLast_OnClick(Sender: TObject);
begin
  inherited;

  //最終頁
  cdsSpoolDetail.Filter:='SPPAGE='+IntToStr(intMaxPage);
  fPageChange;
  lblRecordCount.Caption:=fGetPageCnt(intMaxPage,intMaxPage);

  fSelectInfo_Clear;//選択文字列情報のクリア
end;

{-----------------------------------------------------------------------------
* 表題:次頁ボタン実行
*-----------------------------------------------------------------------------}
procedure TfrmSpoolDisp.btnNext_OnClick(Sender: TObject);
var
  intCurPage: Integer;
begin
  inherited;

  intCurPage:=cdsSpoolDetail.FieldByName('SPPAGE').AsInteger+1;

  //現ページより+1
  cdsSpoolDetail.Filter:='SPPAGE='+IntToStr(intCurPage);
  fPageChange;
  lblRecordCount.Caption:=fGetPageCnt(intCurPage,intMaxPage);

  fSelectInfo_Clear;//選択文字列情報のクリア

end;

{-----------------------------------------------------------------------------
* 表題:前頁ボタン実行
*-----------------------------------------------------------------------------}
procedure TfrmSpoolDisp.btnPrior_OnClick(Sender: TObject);
var
  intCurPage: Integer;
begin
  inherited;

  intCurPage:=cdsSpoolDetail.FieldByName('SPPAGE').AsInteger-1;

  //現ページより-1
  cdsSpoolDetail.Filter:='SPPAGE='+IntToStr(intCurPage);
  fPageChange;
  lblRecordCount.Caption:=fGetPageCnt(intCurPage,intMaxPage);

  fSelectInfo_Clear;//選択文字列情報のクリア

end;

{-----------------------------------------------------------------------------
* 表題:頁変更時のボタン利用変更
*-----------------------------------------------------------------------------}
procedure TfrmSpoolDisp.fPageChange;
begin

  if cdsSpoolDetail.FieldByName('SPPAGE').AsInteger=1 then begin
    btnPrior.Enabled:=False;
    btnFirst.Enabled:=False;
  end else begin
    btnPrior.Enabled:=True;
    btnFirst.Enabled:=True;
  end;

  if cdsSpoolDetail.FieldByName('SPPAGE').AsInteger=intMaxPage then begin
    btnNext.Enabled:=False;
    btnLast.Enabled:=False;
  end else begin
    btnNext.Enabled:=True;
    btnLast.Enabled:=True;
  end;

end;
```

図16　ブラウザで文字列検索するような機能を搭載する

```
{-------------------------------------------------------------------------------
* 表題:文字列検索実行
*-------------------------------------------------------------------------------}
procedure TfrmSpoolDisp.bbtnSearch_OnClick(Sender: TObject);
var
  i: Integer;
  SaveCursor: TCursor; //現在のマウスカーソル
begin
  inherited;

  if edtStrings.Text='' then Exit;

  if dbgSpoolDetail.SelectedIndex=0 then dbgSpoolDetail.SelectedIndex:=1 ;//選択状態
  if intSelectedRow<>dbgSpoolDetail.SelectedIndex then intOffSet:=1 ;//

  SaveCursor:=Screen.Cursor;
  Self.Enabled:=False;
  i:=0;
  try//マウスカーソル操作
    Screen.Cursor:=crHourGlass; //砂時計

    while (not cdsSpoolDetail.Eof) do begin

        i := PosEx(UpperCase(edtStrings.Text),UpperCase(dbgSpoolDetail.SelectedField.Text),intOffSet);
      if i>0 then begin

        Break;
      end
      else begin
        intOffSet:=1 ;//初期化
          cdsSpoolDetail.Next;
      end;

      if cdsSpoolDetail.Eof then
        if cdsSpoolDetail.FieldByName('SPPAGE').AsInteger<intMaxPage then btnNext_OnClick(nil)
        else ShowMessage('見つかりませんでした。');

    end;
  finally
    Screen.Cursor:=SaveCursor; //保存していたカーソルに戻す
    Self.Enabled:=True; //オフにすると選択状態が表示されない
  end;

  if i>0 then begin
    //入力状態にする。
    dbgSpoolDetail.SetFocus;
    dbgSpoolDetail.EditorMode := True;

    TDummyDBG(dbgSpoolDetail).InplaceEditor.SelStart := i - 1;
    TDummyDBG(dbgSpoolDetail).InplaceEditor.SelLength := Length(edtStrings.Text);
    intOffSet:=TDummyDBG(dbgSpoolDetail).InplaceEditor.SelStart+TDummyDBG(dbgSpoolDetail).InplaceEditor.SelLength+1;
    intSelectedRow:=dbgSpoolDetail.SelectedIndex;
  end

end;
```

検索結果の文字列の場所を覚えておき、次に検索が実行されたとき、次に見つかる文字列からスタートする

図17　完成図（WRKSPLFを利用した場合の開発画面）

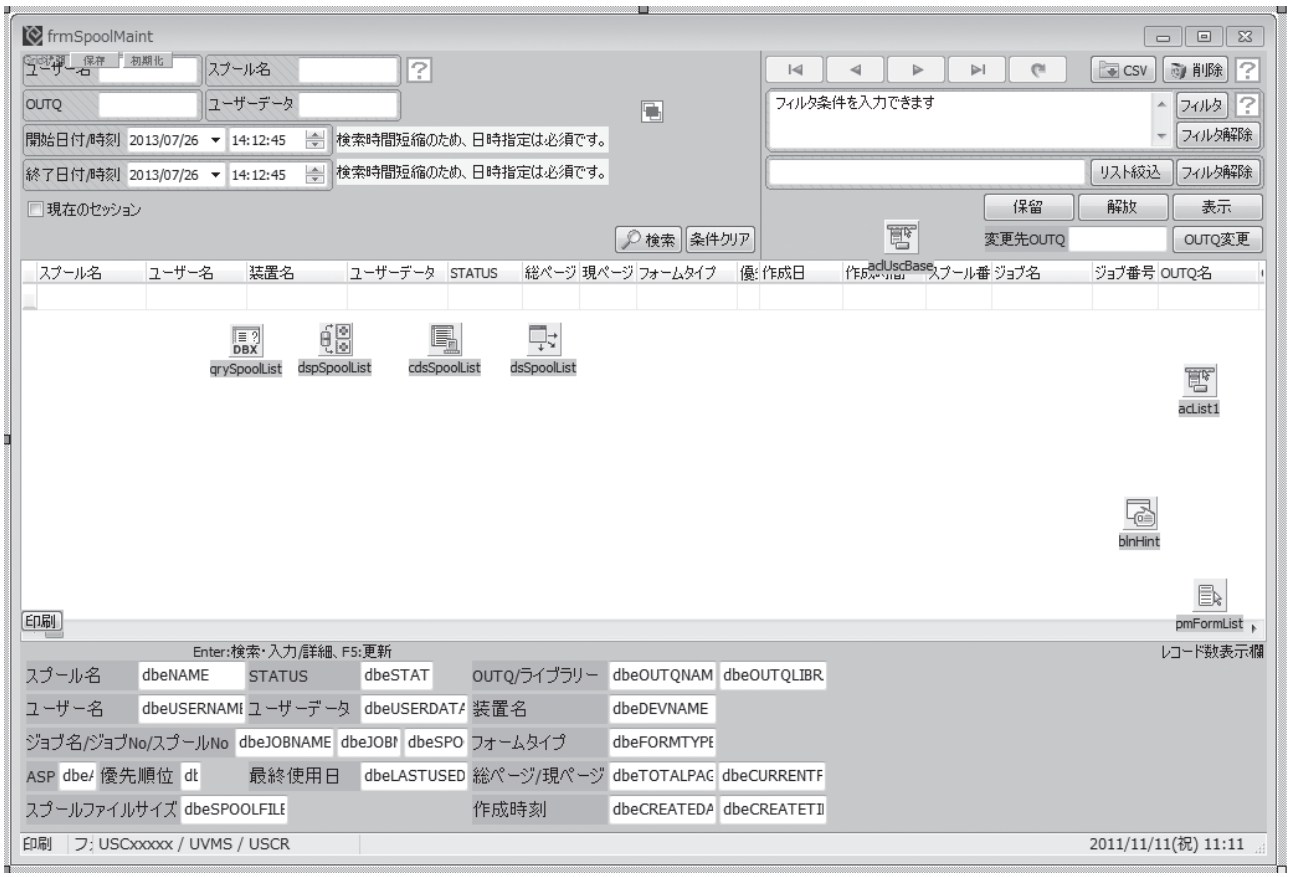


図18　完成図（実行画面）