研究ノート

# パソコンで稼働する西暦2000年問題対応ツールの開発

大林久人*

筆者は、東京情報大学の抱える約３千本のプログラムの西暦２千年問題への対応にパソコンが利用できるよう、プログラムの棚卸しを含む調査段階向けと修正段階、テスト段階向けの支援ツール群をとりあえず開発した。

対応作業のためのマシン時間やファイルスペースの確保は大規模ユーザほど深刻な問題であるが、小規模ユーザでも同様のことがいえる。パソコンの利用が可能ということになれば、複数のマシン、無限のファイルスペースを一挙に確保できるのに近い効果がある。

冷静に考えれば、現在世界の潜在コンピューティングパワーの５割以上を占めるのはパソコンである。ビジネスアプリケーションの中ではデータの入力端末としての役割しか与えられないことが多かったが、計算能力にしぼれば20年前の技術計算むけ大型コンピュータなみの能力を発揮できる。ファイル媒体としても、30年前の磁気テープ５巻に実質的に格納できた情報は、ＭＯ１枚分に満たない。

これらのツールは、以下のような点に留意して作成した。

a．汎用コンピュータでは難しい連続実行
b．現在使用中の正しいデータの再利用
c．ユーザごとのカスタマイズの容易性
d．プログラム実行時の操作方法の統一
e．汎用パソコン別のファイル編成への対応

## 1. はじめに

西暦２０００年問題の原因は、年を表すのに西暦下２桁だけを使ってきた扱い方がまずかったというだけのことである。それが世界中のコンピュータによるデータ処理を大混乱に陥れようとしている。予想される問題などについては、すでに「コンピュータ西暦２０００年問題への対応」（経営情報科学 第9巻第3号）に掲載したのでここでは説明を省略するが、この問題を解決する方法には大別して２つの方法があること、いずれを選択したとしても作業のためにどれだけの時間がかかるか、あと３年足らずとなった期間のうちどれだけの時間が残されているか、その間に完全に対応しきれるか、対応しきれなかった場合はどうするかのリスクマネジメントを含めて、対応時期が遅れるほど深刻化する問題であると提言しておきたい。

基本的な解決法である年の４桁表記を実現するには、ビジネスアプリケーションに使用するすべてのデータファイル、データベース中の日付項目の桁数引き延ばしの作業とともにそれに対応して、プログラム中の該当項目の PIC句の変更、レコード長の変更、JCL やVSAMファイルのDEFINEの変更、ソートパラメータの書き変えなどが必要になる。日付の大小比較や期間計算の算式は基本的に変更する必要がないので、ロジック誤りのテストなどの手数はかからないのが救いといえるが、期間の計算などの必要に応じて、すでに上２桁に19を補って処理していたようなプログラムは、余分なコードを取り除くなど細心の注意を必要とする。

*東京情報大学教授

1997年6月16日受理

また、ファイルの形式変換が行われた時は、新形式のファイルを使うプログラムすべてを使ってひととおりのテストをすませ一斉に切り換えることが要求される。

データファイル中の日付項目は6桁のままにしておいて、問題のある処理の部分だけを内部で対応できるように修正すれば、対象となるプログラムの数は3割から5割程度に減少するかもしれない。出力リポート上の日付の表記や西暦下2桁を要素に含む学籍番号順では年代順の配列に不都合が生じるといった問題をささいなこととして目をつぶれば、とりあえず月次、年次の処理などほとんどのバッチ処理は手を着けずにすむかもしれないからである。

ただし、この方法は、問題を先送りするだけのものであること、プログラム中の処理を詳細に調べて、必要な部分すべてに対応措置を施しておく必要があるため、必要とする要員の質が高いこと、あるいは対象とするシステムとそこで使われているプログラムについての理解が深いことが要求されることを認識しておいていただきたい。

## 2. 対応作業とツール

いずれの方法を選択するにしても、対応作業にとりかかるときは、現在プログラム資産として保有するすべてのプログラムについての棚卸しが必要である。

過去に使用していたが現在使われていないプログラム、現在使用中であるがソースプログラムの所在が不明になっていてるものなど、その整理に思わぬ時間を費やしたユーザも多いといわれるが、本学のように開校いらい十年程度しか経過していないユーザの場合は、開発途上のサブシステムが多く、所在不明のソースプログラムなどの問題が少ないことは不幸中の幸いといえるが、その代わり、対応作業と開発の凍結のスケジュールを整合させるため、対象範囲をできるだけ縮小し、サブシステムごとの対応作業を短期間に集中して終了させていくようなスケジュール管理が重要になる。

棚卸しの資料は JCL中で指定されているプログラムのソース（原文）とプログラム中に記述されているCOPY句の使用するコピーファイルとCALL文で呼び出す副プログラムの同じくソースの所在を確認するところから始まる。次の調査段階の主要なステップが、全プログラムを対象とする日付の扱いに関する調査と修正方法を決定するためのパイロット業務の先行処理だからである。

プログラムリストから日付などの関連項目を洗い出すときの問題点は文字列検索をするとき、日付項目にどんなデータ名をつけているかわからないことであろう。英語を話す国のプログラマなら、日付を示すのに使いそうな単語や略語は限られてこようが、日本の場合は英語あり、日本語あり、ローマ字ありの状態である。したがって、ユーザごとに使用しているデータ名をいちどに検索できるツール、それもカスタマイズ容易なツールが必要になる。

また、対応作業のマシン時間やファイルスペースの確保のためにパソコンの利用を進めたい。複数のマシン、無限のファイルスペースを確保できるからである。

冷静に考えれば、現在世界の潜在コンピューティングパワーの5割以上を占めるのはパソコンである。ビジネスアプリケーションの中ではデータの入力端末としての役割しか与えられていないことが多かったかもしれないが、計算能力にしぼれば20年前の技術計算むけ大型コンピュータなみの能力を発揮できる。ファイル媒体としても、30年前の磁気テープ5巻に実質的に格納できた情報は、ＭＯ１枚分に満たない程度であった。

少なくとも、調査段階でのホストにあるライブラリの内容の棚卸し、修正後のプログラムのリ

コンパイルとテストなどホストを使用しなくては実施できない作業を除いて、プログラムのソースから日付項目らしきものを洗い出すための文字列処理やプログラム原文中のデータの桁数引き延ばし、データファイル（データベースの場合はいったん順ファイルに吸い上げる）の桁数引き延ばしなどの作業、テストデータの準備作業などはパソコンでも十分実行できる。

パソコンとのダウンロード・アップロードの手数はかかるが、その後の処理はパソコンのほうが迅速で、手数もかからない。

実際のところ、パソコンのほうが作業を進めるのに楽な点はいろいろある。プログラムのファイル名をひとつのファイルにまとめて記録しておいて、そこからひとつずつ読みだしながら対象ファイルのOPEN CLOSEを繰り返す。こうしておけば、作業は人がいなくても実行できる。

また、パソコンを使用することで現在、パソコン上で稼働する多くのアプリケーションソフトを利用できることも大きい強みとなる。検索されたコピーファイルとプログラムの関係や日付項目名のデータ型とそれを含むプログラムとの関係などを整理し、検索する簡易なシステムをパソコンで稼働するデータベースソフトを利用して作ることは容易に行える。少なくともプログラムの棚卸し段階、日付項目が検索されたあとの検討段階にはこうした検索手段があるかないかでは、作業の進捗に大きく影響しよう。

さらに、汎用機種やオフコンでは JCLやプロシーデュアの作成など操作のための準備に面倒が多く、学生バイトや未熟練者を補助者として同時並行的に作業を進めることには困難がともなう。その点、操作の容易なパソコンでは複数の未熟練者を作業に動員することも可能である。

とはいうものの、一般の汎用ユーザには、パソコンの性能に関する不安も多い．それを解消するため今回開発したパソコン用のプログラムの実行時間の計測を行った。もし不満であれば、パソコンを複数台使うことでスループットの向上をはかればよい。

計測に使用したプログラムは JCLの解析とプログラムソースの文字列検索のプログラムでいずれも、複数のデータファイルのOPEN/CLOSEを繰り返して計測した．はじめ、25Mhz のパソコンでFDD(フロッピディスク) を使用したがかなり時間がかかるので、データをラムテディスクにコピーしてから実行するとどのくらい時間が短縮できるかを計測してみたところほとんど効果がないことがわかった。

ただし、入出力時間がネックとなるその他のプログラムの実行時間は、当然のことながらアクセス速度の早い装置を使うと一挙に処理時間が短縮する。下の計測結果はJCL ファイル10本、492行、 23314バイト分の処理にかかる時間で 25Mhz程度のマシンでも本学にある JCLファイル約1100本の処理に実質20分くらいの所要時間と推定できる。

プロセスネックの文字列処理、ZPRO4 の実行時間は内部速度の速いパソコンに切り換えると、予想通り効果的に改善できることが判明した。

結局、実用するにはできるだけ内部処理の速いパソコンを使用することと対象プログラムの本数しだいでMOなど大容量の外部記憶装置を接続してダウンロードし、実行時にはできるだけラムディスクにコピーして使うことがFDD などの装置の安全性を保つためにも必要と考えられる。

**入出力時間がネックとなるZPROJCLの実行時間**

| | PC9821 5Mhz | | |
|---|---|---|---|
| 10本 | FDD | HDD | RAMディスク |
| 492行 23314バイト | 5秒 | 1秒 | 0.5秒 |

**文字列検索 ZPRO4 の実行時間**

| | 行数 | バイト数 | | PC9821 CE2 25Mhz FDD 分 秒 | PC9821 CE2 25Mhz RAMディスク 分 秒 | FMV 5DH 133Mhz FDD 分 秒 |
|---|---|---|---|---|---|---|
| B100 | 393 | 15976 | +8 | 1:58 | 1:53 | 0:17 |
| B270 | 916 | 39819 | +11 | 4:11 | 4:02 | 0:56 |
| B290 | 417 | 16659 | +10 | 1:57 | 1:53 | 0:42 |
| B300 | 240 | 10783 | +3 | 0:47 | 0:44 | 0:06 |
| B030 | 750 | 34035 | +9 | 3:43 | 3:35 | 0:52 |
| 計 | 2716 | 117272 | | | | |

＋は二次検索の語数

## 3. パソコンでの対応作業のためのツール群

本学のプログラムはすべて COBOL 言語を使用して作成されており、また、日本のユーザアプリケーションに使用されているプログラムの 7 割以上は COBOL 言語を使用していると推定される. したがって今回開発したツールは COBOL 言語で作成されているプログラムのみを対象として扱う.

ツールの開発言語としては COBOL を使用したが、汎用機種やオフコンへの移植可能性を考慮して主要な部分で使用する言語は、JIS COBOL 言語仕様に掲載されているものに限定した。ただし、画面でのやりとりなどパソコン固有の機能と密接している部分は特定のモジュールにまとめ、移植のさいまとめて置換できるようにした。

なお、これらツールの開発に C 言語を使用すると、順ファイル・行順ファイルいずれの入力にも使用できる関数、COBOL の STRING UNSTRING INSPECT などの命令に対応する多機能の関数などを新たに作成する必要があるので、とり急いで開発しておきたいツール群は COBOL で開発し、すべてが終了したあと学生の演習課題として C 言語への変換を手がけることとした。

STRSTR 関数と IF AREA ( K : PT ) の形式で書く部分参照も機能的には同様の結果を得られるものの、参照された後の値の再利用という点で検討の余地が残っている。

今回、開発したパソコンで稼働するツール群は、棚卸しを含む調査段階向けと修正段階、テスト段階向けに別れるが、本学では次のような手順に従って対応作業を進めることを前提としている。

**棚卸し段階向けのツール**

| | |
|---|---|
| ZPROJCL | JCL からのプログラム･ファイル･ソートパラメータなどの抽出用 |
| ZPJCLST | ファイルと使用プログラムの一覧を作る |
| ZPJCLPRG1 | プログラム名を JCL 順に並べた検索指示用のデータを作る |
| ZPJCLPRG2 | ソートフィールドを並べたリストを作る |
| ZPFILE | プログラム中の COPY CALL 文およびファイル名と OPEN モードの検索用データを作る |

| | |
|---|---|
| ZPCPLST | COPY CALL の対象ファイルと入出力ファイルのデータを作る |
| ZPCPYPRG | COPY ファイル名を並べた登録ファイルを作る |

**調査段階向けのツール**

| | |
|---|---|
| ZPRO4 | 文字列検索 COBOL プログラム用 |
| ZPRO1 | 文字列検索 COPY 句で使うファイル用 |
| ZPLOGMK | 検索された行にマークをつける |
| ZPDTNAM | 検索された語と PIC 句の一覧を作る |
| ZPDTCHCK | レコード記述をもとに日付らしい値を持つフィールドを探す |

**修正段階向けのツール**

ツールとして開発したプログラムは次のとおりである。日付に関しては内部対応を主体に進める予定を立てているが、郵便番号や電話番号については物理的な桁数の引き伸ばしに対応せざるを得ないので下のツールを別途スケジュールにしたがって使用することとした。

| | |
|---|---|
| ZPROLNG | プログラム中の字数書換え |
| ZPRODAT | データファイルの形式変換 |
| DATDIV1 | 同上 ファイルごとの処理プログラム作成 |
| ZPWAREKI | データファイルの和暦への変換 |
| ZPROINS | 標準コードの引き込み用 |
| ZPINSCHK | 標準データ名との重複チェック |
| ZSTRCHNG | データ名などの変更 文字列の書換え |

**テスト段階向けのツール**

現在使用中のプログラムと同じ結果が、1999年末まで保証できること、2000当初およびそれ以降も同様の結果が保証できるかをテストするために、現在使用中のマスタおよびデータファイルの日付をそれぞれスライドさせる。

| | |
|---|---|
| ZPTESTDT | テスト用に日付を先にスライドしたデータファイルを作る |

---

**棚卸し段階の進め方**

本学の場合も現用のプログラム、ソース、コピーファイルなどすべてがプロダクションライブラリに存在することを先に確認する必要はある。また、印刷する資料のレイアウト、画面のレイアウトなどを基本的に変更しなければ作業量の縮小がはかれる。一般的な手順は次の通りとなろう。

1) サブシステム単位に JCLﾌｧｲﾙ名を一覧できるファイルをホスト側で作成する
並行して、サブシステムごとにドキュメント、入力原票、出力リポートのサンプルなど

を収集する
2) このファイルにあるJCLをすべてパソコンへ転送する
3) ＪＣＬからPROG名などを抽出。
4) サブシステム単位にｿｰｽﾌﾟﾛｸﾞﾗﾑとﾛｰﾄﾞﾓｼﾞｭｰﾙ名を一覧できるファイルをホスト側で作成する
5) ＪＣＬから抽出したﾌﾟﾛｸﾞﾗﾑ名とつきあわせて確認する
6) それに含まれるｿｰｽﾌﾟﾛｸﾞﾗををすべてパソコンへ転送する
7) JCLファイルからプログラム検索用のパラメータファイルを作成する
8) プログラムからCOPY CALL I/O論理ファイル名を抽出。
9) それに含まれるｺﾋﾟｰﾌｧｲﾙをすべてパソコンへ転送する
10) このファイルからコピー句検索用のパラメータファイルを作成する

**調査段階の進め方**

11) プログラム単位の文字列検索
12) ソースプログラムの検索された部分へマークづけ
13) コピー句単位の文字列検索
14) 検索結果の印刷、チェック、整理保管
人手による検索結果の検討　（始めはサンプルをいくつか選んで検討）
入力原票上の日付項目とデータの入力チェックを行うプログラムで使用する
変数名、チェック方法の確認
出力リポート上の日付の表記方法とソースの確認　　など

**修正段階の進め方**

15) ソースプログラムの印刷、挿入ルーチンと引数の決定、ソースの編集、確認
16) ソースプログラム中の重複データ名のチェック
17) 標準ルーチンの挿入または2重の入力えエリアの設定
18) ホストに戻してリコンパイル　　本番稼働中に並行処理

**テスト段階の進め方**

テストの実施は、コンパイラの関係からパソコンでなくホストを使用せざるを得ないであろう。時期としては本学のように学生休暇期間のある場所では、それをあてる。

19) テストデータの準備　マスタファイル
20) 日付を変更してテスト

## 4. ツールの開発にあたっての留意点

汎用機種やオフコンと違って、パソコンでのプログラム言語では、C言語でもCOBOLでも、これからOPENするファイルを、そのファイルの存在する装置・ディレクトリ・ファイル名・拡張子を変数として標準入出力ルーチンに引き渡すことにより、自由に指定することができる。キーボードから入力した変数でも、プログラム内部の定数でもよいし別のファイルから入力したデータでもよい。

また、パソコンのMS/DOS標準形式である行順ファイルは、レコードごとの区切りを復帰・改行符号（CR,LF）で示しているので、4000バイトと指定したバッファエリアにそれ以下の長さのレコードであれば、問題なく１レコードずつ呼び出せるという特性がある。汎用機種で主流を占めるブロック化した固定長レコードのように、プログラム中のレコード記述に実際に扱うデータの論理レコード長を書いておかなくても処理は正しく行える。

これらの特性を利用すると、実際のレコード長とは無関係にプログラムを作成しておき、使用するデータファイルは、別のファイルに記録しておいて、一つずつ読みだしてOPEN/CLOSEを繰り返し、リユーザブルなプログラムを自動的に連続して実行させることが容易に実現できる。今回、２０００年問題に対応するために開発したパソコン用のツールは、短期間に対応作業を進捗させることを目標として、可能な限り連続的に実行できることに留意した。

**a．連続実行**

自動的に連続実行させることで可能な限りの省力化を図ることと同時に、調査漏れになるプログラムの発生を防ぐ。サブシステム（業務）単位にチェック対象とするプログラム名を順に記録し、それに業務名をつけたファイルを作る。

登録ファイルの例

GMGK　<--- ファイル名

¦<-- 16ﾊﾞｲﾄ -->¦<- 12ﾊﾞｲﾄ ->¦

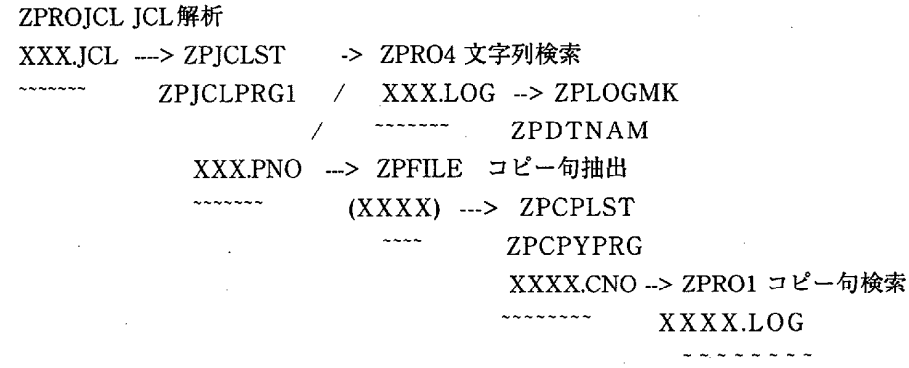
GMGKM010　----------
GMGKY010　----------
GMGKY020　----------
GMGKY030　----------
GMGKY040　---------

このファイルを使用し、文字列の検索(ZPRO4) COPY句の抽出など(ZPFILE)を処理した日付を記録し、調査資料の作成済みを確認できるようにする。

棚卸しや調査段階ではとくに漏れの発生を防ぎたいため、本学ではＪＣＬからプログラム名を抽出して登録ファイルを作成し、その順に処理したプログラムからＣＯＰＹファイル名の登録ファイルを自動的に作成する。

ただし、最初のＪＣＬファイル名だけはホストから一覧形式でダウンロードし、それをパソコン側で加工して使用する。

プログラムと連続実行指示用データの関連を簡単に図示してみる。下線を引いたファイルが、その上に書いたプログラムの出力ファイルで、それが次のどのプログラムに対する実行指示に使われるかを示した。




ZPROJCL, ZPFILE, ZPRO4, ZPRO1 など個別のＪＣＬまたはプログラムファイルを処理対象とするツールは、共通の登録ファイルを使用し、そこへ処理日付を書き戻す。作業の進捗管理にも役立てたいからである。

なお、登録したプログラム名の誤りのため連続実行が中断しないよう、エラーログをとるファイルに記録を残し次のファイルの処理に進む機能も組み込んである。

ZPERRLOG.ERR のファイル名でﾙｰﾄﾃﾞｲﾚｸﾄﾘ中に作成するエラーログ

```
SHT         SHJM06        オープン不能    970301
```

**b．現在使用中の正しいデータの再利用**

現在の処理に使用している正しい素材は、できるだけ活用することとした。たとえば、プログラム中のデータの記述をファイル形式変換の指示用のデータに使用するというようなことである。人手による誤りを可能な限り防ぐことが目的であるが同時に、省力化も図れる。コピー句単位でレコード形式を記述する場合は、そのファイルを利用して桁数を増加する項目、減少させる項目などの指示データを作成できる。

次の例はFDにあるレコード記述を利用したデータファイルに対する形式変換の指示である。プログラム中の PIC句の字数の変更のための指示も、同様の方法で作成できる。

```
**ZP7DATA           <-- ﾍｯﾀﾞﾚｺｰﾄﾞ 実際のファイル名がよい
    75  01  MEIBOREC.
    76      03 KNO    PIC 9(4).
    77      03 KSH    PIC X(4).
    78      03 KCAP   PIC 9(8).
    79      03 KDOM   PIC X(4).
/D  80      03 RDATE  PIC X(6).  <-- 日付の先頭に 19 を追加する
/J  81      03 JIP    PIC X(6).  <-- 郵便番号は後に 2ｽﾍﾟｰｽ を追加する
    88      03 KADR   PIC X(40).
```

**c．カストマイズの容易性**

日付項目の検索のさいには前述どおりどのような名称が用いられているかわからないので、サンプルをいくつか検索したうえで検索に使う部分文字列を指定できるように考慮しておく必要がある。ZPRO4（COBOL プログラム用）ZPRO1（COPY句用）での文字列は既定値として COPY CALL ENTER DATE などをプログラム定数として持たせ、YEAR NEN YY HIZ TUK などの文字列をパラメータファイル PARAMF.NEN に持たせておいてユーザがそれらを取捨選択できることと、実行時に追加した文字列をこのファイルに追加していけるようにした。

また、プログラム内部対応のためには標準的な手続きとの置き換えを行う方法をとることが望ましいが、日付項目に関する演算や処理方法についてはユーザごとの特殊事情による多様化が避けられない。それらを考慮して標準ルーチンを多数用意するとかえってその選択を誤りやすい弊害を生ずる恐れが多いので、ZPROINS では、標準ルーチンをユーザ自身が簡単に作って追加できるように配慮することとした。

その方法は次のようなものである。

```
P1100  03 <-- P11 は手続き番号  行番 00 のところに引数の数を書く
P1101Y    <--  ?1 ?2 ?3 のように引数と置換する部分を含む行は Y と書く
P1102
```

例 経過年数の計算

```
P1100  03
P1101Y    MOVE  ?1      TO   WWY4NEN1.
P1102     IF    WWY4NEN1  <   WWKIJYUN  ADD  100  TO  WWY4NEN1.
P1103Y    MOVE  ?2      TO   WWY4NEN2.
P1104     IF    WWY4NEN2  <   WWKIJYUN  ADD  100  TO  WWY4NEN2.
P1105Y    COMPUTE   ?3  =  WWY4NEN1  -  WWY4NEN2.
```

手続き部は Y2KSTD.DAT の最後の部分に追加する。手続き番号は最後の番号に続く番号を適当に使用できる。

**d．プログラム実行時の操作方法の統一**

関連するプログラムの実行時の操作はできるかぎり統一して、パソコン操作に不慣れな人の混乱をふせぐようにした。

スタート時のメッセージはほとんどすべてのツールで次のようになる。

プログラム名を登録したファイルを使いますか?(Y OR N)

業務別にチェック対象のプログラムを登録しているときは

Ｙと打つと　　続けて
プログラムを登録したファイル名?　の入力を求める
Ｎとしたときは
　プログラム原文のファイル名?　の入力を求める

原文ファイルの編成法（業務単位のときは全部がここで指定した編成法とみなす）
レーベルなし順ファイル　--Ｓ <-- 汎用 オフコンでフロッピに書きこんだデータはふつう固定長の順ファイル
MS/DOS 標準の行順ファイル --Ｌ <- いわゆるテキスト形式の行順ファイル

ただし、サーバを経由してダウンロードするとき、MS/DOS 標準のテキスト形式に変換できることがわかれば、汎用機種に対する考慮は無用となる。

## 5. 各ツールの概要

### a． ZPROJCL ＪＣＬ文の解析プログラム

このプログラムは指定したＪＣＬ文にある、JOBNO STEPNO プログラム名およびファイル名、SORT プログラムの入出力ファイル名とFIELD の指定などを抽出する。
記録用ファイルは業務単位にその中で使用している全ＪＣＬからの検索結果をひとつのファイルにまとめて作る。

プログラムを実行するためには、まず、ＪＣＬ名を登録したファイルの作成が必要である。

例　GMGKJCL　<--- サブシステムの略名をファイル名とする

内容

|<-- 16ﾊﾞｲﾄ -->|<- 12ﾊﾞｲﾄ ->|　　ZPROJCL の実行後　日付を記録する

| | | | |
|---|---|---|---|
| GMGKM010 | ---------- | GMGKM010 | 970219----- |
| GMGKY010 | ---------- | GMGKY010 | 970219----- |
| GMGKY020 | ---------- | GMGKY020 | 970219----- |
| GMGKY030 | ---------- | GMGKY030 | 970219----- |
| GMGKY040 | ---------- | GMGKY040 | 970219----- |
| GMGKV010 | ---------- | GMGKV010 | 970219----- |

<---- 最終行のあとに END のようなレコードは不要

ＪＣＬの内容は次のようなものである。

```
//JST005#  JOB CLASS=A,MSGCLASS=Y,MSGLEVEL=(1,1,1),REGION=4096K,
//        SPARM=' LANG=J'
//*****************************************************************
//*                                                               *
//*    SYSTEM        XXXXXX              GM            *
//*    JOB           YYYYYY              GMGKV020      *
//*                                                               *
//*****************************************************************
//JOBLIB   DD DSN=JGS.LMD.T00,DISP=SHR
//        DD DSN=SYS1.COBLIB,DISP=SHR
//IMAGELIB DD DSN=JGS.IMG.T00,DISP=SHR
//*****************************************************************
//*   XXXXXX                              GMGKB095      *
//*****************************************************************
//STEP010 EXEC PGM=GMGKB095
//AIMPED   DD SUBSYS=(AIM,GTPED002,GMAPG2),DISP=SHR
//F004     DD DSN=&&F004IN,UNIT=SYSDA,DISP=(NEW,PASS),
//        DCB=(RECFM=FB,LRECL=80,BLKSIZE=4000),
//        SPACE=(TRK,(10,1),RLSE)
//SYSOUT   DD SYSOUT=*
//SYSDBOUT DD SYSOUT=*
/*
//*****************************************************************
//*   ソート F004                         SORT          *
//*****************************************************************
//STEP020 EXEC PGM=SORT,COND=(8,LE)
//SORTIN   DD DSN=&&F004IN,DISP=(OLD,DELETE)
//SORTOUT  DD DSN=&&F004OUT,UNIT=SYSDA,DISP=(NEW,PASS),
//        DCB=(RECFM=FB,LRECL=80,BLKSIZE=4000),
//        SPACE=(TRK,(10,1),RLSE)
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(5))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(5))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(5))
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
  SORT FIELDS=(33,1,CH,A,34,1,CH,A,1,6,CH,A)
/*
```

プログラムの実行結果として得られる記録用ファイルの出力内容は次のようになる。

```
-----------------------------------------------

JCL番号   STEPNO. ﾌﾟﾛｸﾞﾗﾑ NO.  ﾌｧｲﾙ名      DSN        ｿｰﾄ ﾊﾟﾗﾒｰﾀ


GMGKM010  STEP010 GMGKB100    F014       &&F014
GMGKM010  STEP020 GMGKB115    F014       &&F014
GMGKM010  STEP020 GMGKB115    L047       SYSOUT
GMGKY010  STEP010 SORT        JGM.F310.DAT JGS.F002.DAT(14,1,CH,A,1,6,CH,A)
GMGKY010  STEP020 GMGKB010    F002       JGS.F002.DAT
GMGKY010  STEP020 GMGKB010    L016       SYSOUT
GMGKY020  STEP010 SORT        JGS.F002.DAT &&F002     (1,6,CH,A)
GMGKY020  STEP020 GMGKB030    FB00       JST.FB00.VSA
GMGKY020  STEP020 GMGKB030    FD21       JGS.FD21.DAT
GMGKY020  STEP020 GMGKB030    F002       &&F002
GMGKY020  STEP020 GMGKB030    L017       SYSOUT

-----------------------------------------------
```

○ ZPJCPRG1 は上の内容をデータとして次のプログラム登録ファイルを作成する。このファイルは ZPFILE ZPRO4 ZPLOGMK などの処理にそのまま使用できる。

```
GMGKB100    ----------
GMGKB115    ----------
GMGKB010    ----------
GMGKB030    ----------
GMGKB060    ----------
GMGKB065    ----------
```

○ ZPJCPRG2 は上の内容をデータとして次のようにソートフィールドだけを抽出したリストを作成する。

```
GMGKY010    STEP010    (14,1,CH,A,1,6,CH,A)
GMGKY020    STEP010    (1,6,CH,A)
GMGKY030    STEP020    (4,6,CH,A)
GMGKY040    STEP020    (1,2,CH,A,5,2,CH,D,7,3,CH,A)
GMGKV010    STEP020    (1,10,CH,A)
```

IBM 社と同様に2桁年の表記のまま年度順にデータをソートできる DFSORTと同等の機能がメーカから提供されたときは、このソートフィールドの一部を 1,2,Y2C,A のように書き換えることが必要になる。そのためのツールも準備する予定である。

**b．ZPFILE プログラム原文中の　COPY CALL　対象ファイル名と**
**入出力ファイルの抽出プログラム**

このプログラムは指定したプログラム原文ファイルの中にある、SELECT文からプログラム中のファイル名とASSIGN TO で指定する論理ファイル名を検索するとともに、COPY CALL する内容を抜きだして記録用ファイルに出力するプログラムである。COPY CALLの対象は JCLから抽出することができない（単独で実行できるものではない）からこのようなプログラムが必要になる。

この記録ファイルを使うと、同じものをCOPYするプログラム、同じルーチンをCALLするプログラムなどの一覧表が、業務ごとに作れる。

また、JCL ステートメントとつきあわせることでプログラムとJCL の整合関係をチェックすることもできる。

連続実行するにはプログラム登録ファイルにチェックする対象のプログラム名を順に登録し、それに業務名をつけたファイルを作っておくだけでよい。

JCL を解析する ZPROJCLを使う機種では ZPJCLST ZPJCLPRG の2つのプログラムを使うことで自動的に登録ファイルを作ることができる。

例　SHJM　<-- ファイル名

内容

¦<-- 16ﾊﾞｲﾄ -->¦<- 12ﾊﾞｲﾄ ->¦

GMGKB100　　---------- <-- ﾊｲﾌﾝ 以外の記号でもよい
GMGKB270　　----------
GMGKB290　　----------
GMGKB300　　----------
GMGKB030　　----------
<----- 最終行のあとに END のようなレコードは不要

----- の部分には処理した日付を格納す 。

記録用のファイルに出力される内容は、次のようになる。

---

プログラム *** 論理ファイル 用途 内部ファイル名

GMGKB100 MFS FD23 IN FD23-CYHSTUJKEPAR
GMGKB100 MFS F014 OUT F014-CYHFIL
GMGKB100 MY CFD23 COPY
GMGKB100 MY CF014 COPY
GMGKB100 MY CE001 COPY
GMGKB100 MY CC901 COPY
GMGKB100 MC " CMCM8011" CALL
GMGKB270 MFS FD22 IN FD22-SOTYMDPARFIL
GMGKB270 MFS F026 IN F026-FIL

---

○ ZPCPLST は上のデータをソートして次のような、COPY CALL 対象の一覧と次のファイル別の使用プログラムと OPENモードの一覧データを作成する。

COPY CALL 対象

| | | |
|---|---|---|
| "CMCM8011" | GMGKB030 | CALL |
| | GMGKB100 | CALL |
| | GMGKB270 | CALL |
| "JCVEBCC" | GMGKB270 | CALL |
| CC901 | GMGKB030 | COPY |
| | GMGKB100 | COPY |
| | GMGKB270 | COPY |
| CE001 | GMGKB030 | COPY |
| | GMGKB100 | COPY |

---

ファイル一覧

| | | |
|---|---|---|
| F002 | GMGKB030 | IN |
| F010 | GMGKB290 | OUT |
| | GMGKB300 | IN |
| F014 | GMGKB100 | OUT |
| F026 | GMGKB270 | IN |

---

### c．ZPRO1 ZPRO4　文字列の部分参照による１次検索と受け取り側データ名による２次検索を行うプログラム

ZPRO4.CBL は COBOLプログラムを対象とするもので、ZPRO1.CBL のほうは COBOLで使うCOPYライブラリの内容、 RPG や ｱｾﾝﾌﾞﾗ など、たいていの形式のプログラムを先頭から１行ずつ検索して指定文字列を探すように作ったものである。

ZPRO4 は、指定したプログラム原文ファイルの中にある、指定文字列のある行を検索するとともに、FD句の全内容を抜きだして .LOG ファイルに出力する。 .LOG ファイルは、プログラム原文のファイルごとに同じファイル名に拡張子.LOGをつけて作成する。

このフログラムを使うときは、まず、検索したい文字列の登録が必要である。パラメータファイル PARAMF.NEN ファイルにはあらかじめ次の文字列が登録されているので、ユーザはそれに追加すればよい。

例　　<--- 登録できる語は50以内。

YEAR　YY NEN HIZ BI YMD HZK MM DD TUK

チェックの対象とするプログラムの登録ファイルは ZPFILE の実行のために作成したものを共通に使用する。

```
GMGKB100      970219970219    <-- ZPFILE  ZPRO4 を実行した日付
GMGKB270      970219970219
GMGKB290      970219970219
GMGKB300      970219970219
GMGKB030      970219970219
```

作成されるログ .LOG の例を示す。日付の大小を比較するような命令があったら要注意ということで行番号のあとに ?? を付けて表示した。

```
 84 #        ACCEPT  SHIZUKE  FROM  DATE.
 85          MOVE SHIZUKE TO HIZUKE. DISPLAY   HIZUKE  AT  0301.
 86          MOVE  SHIZUKE  TO  YOKUNEN.
 87          ADD  1  TO  YYY.
 88          MOVE YOKUNEN TO HIZUKE. DISPLAY   HIZUKE  AT  0501.
105          MOVE  STDATE  TO  WHIZUKE.
106 ??       IF  WHIZUKE  NOT >  SHIZUKE  MOVE  "**2  "  TO ERRMRK GO TO
```

```
107 ??      IF STDATE  NOT < RTDATE  MOVE "**3 " TO ERRMRK GO TO
108 ??      IF STDATE   >  YOKUNEN MOVE "**4 " TO ERRMRK GO TO
112         MOVE STDATE TO XSTDATE
113         MOVE RTDATE TO XRTDATE
    RECIEVER ----
    SHIZUKE
    HIZUKE
```

RECEIVER ---- 受け側にある項目名は再度検索して波及する先を検索する。上の例ではそれ以上波及していないことがわかる。

○ ZPDTNAM 検索された語とPIC句の一覧を作る

.LOGファイルにある項目名と PIC句、使用されているプログラムとその中の行番を一覧できるデータファイルを累積していく。このファイルをパソコンのデータベースソフトや表計算ソフトなどに読みこんで加工すれば索引表や簡易データベースを直ちに作成することができる。

```
01 CYSYMD       X(06)    GMGKB100    45
01 KYVYMD       X(06)    GMGKB100    52
01 NUGYMD       X(06)    GMGKB100    54
01 SOTYMD       X(06)    GMGKB100    55
03 WORK-YMD              GMGKB100    73
05 WORK-NEN     9(02)    GMGKB100    74
05 WORK-TUK     9(02)    GMGKB100    75
01 CTL-REC      X(61)    GMGKB270    64
```

○ ZPLOGMK 原文ファイルへのマークづけ

検索処理が終わった後、出力された .LOG ファイルと原文ファイルを使って、検出された部分の先頭に ## をマークとして付加するプログラム。先頭の行番号の部分を ## に置換する。このマークがあるとエディタなどで検索しながらソースプログラムを読むことができる。マークに行番号をあてた理由は、この部分がかっての重要な機能を果たさなくなっているにも拘らず、形式的に残っているのに対し、73桁目以降の始めから使用しない領域はコメントの記入用にあてているケースが多かったからである。

マークした結果は、次のようになる。

```
200100**************************************************************
200400 PROCEDURE DIVISION.
200500 MAIN-CONTROL.
```

```
200600    PERFORM  INITIAL-RTN      THRU   EX-INITIAL-RTN.
200800    PERFORM  REPEAT-RTN       THRU   EX-REPEAT-RTN
200900             UNTIL   ENDIDT  =  VENDIDT.
201100    PERFORM  END-RTN          THRU   EX-END-RTN.
201200       STOP   RUN.
201300*
201400 INITIAL-RTN.
       DISPLAY   SPACE.
##      ACCEPT  SHIZUKE  FROM  DATE.
##      MOVE  SHIZUKE   TO   WWSYSDATE6.
##      IF  WWYY < WWKIJYUN  MOVE 20 TO WWCC
       ELSE              MOVE 19 TO WWCC.
##      MOVE WWSYSDATE TO HIZUKE  DISPLAY  HIZUKE AT 0311.
```

---

### d．ZPROINS 標準ルーチンの引き込みプログラム

プログラム中の引き込み指示にあわせて、標準コードとして用意されている手続きをそれぞれの場所に挿入するプログラム。

標準手続きで使用するデータは、とりあえず COPY 句でまとめて引き込む。

挿入する標準手続き中の ?1 ?2 ?3 などの部分は引き込み行に書いた引数の文字列に置換される。この手続きはユーザが簡単に追加できる。

例

```
P01  6桁のシステム日付を8桁の日付に変換する
     システム日付から和暦への換算、年度の設定
       も行っておく

   MOVE  ?1   TO   WWSYSDATE6.              <-- ?1  ACCEPT したシステム日付
   IF  WWYY < WWKIJYUN  MOVE 20 TO WWCC
   ELSE              MOVE 19 TO WWCC.
   MOVE   'H'   TO  WWYNENGO.
   COMPUTE  WWYWANEN = WWCCYY  -  1988.
   IF  WWMM  <  4   COMPUTE  WWNENDO = WWCCYY - 1
   ELSE    MOVE   WWCCYY   TO   WWNENDO.
```

引き込み行の書き方

基本的に、引き込みたい場所に次のような引き込み行を１行ずつ書き込む。

&PXX( , )　() 内は引数――２つ以上のときは ,ｺﾝﾏ で区切る
　　引数のないものは()とも省略
XX は引き込みﾊﾟﾀｰﾝ番号

例

```
&P01(SDATE)
&P02
&P03(YOTEIBI,<=)
&P04(STARTDATE,NOT<,RETURNDATE)
```

プログラムへの引き込みの例

（引き込み前）

```
201400 INITIAL-RTN.
           DISPLAY  SPACE.
           ACCEPT  SHIZUKE  FROM  DATE.
         &P01(SHIZUKE)
        *   MOVE  SHIZUKE  TO  YOKUNEN.
         &P02
        *    ADD  1  TO  YYY.
201800     OPEN  INPUT   I1-FILE   OUTPUT  O1-FILE.
202800     PERFORM  I1-READ          THRU   EX-I1-READ.
201900 EX-INITIAL-RTN.       EXIT.
203900*****************************************************************
```

（引き込み後）

```
201400 INITIAL-RTN.
           DISPLAY  SPACE.
           ACCEPT  SHIZUKE  FROM  DATE.
991231     MOVE  SHIZUKE  TO  WWSYSDATE6.              <-  &P01
991231     IF  WWYY < WWKIJYUN  MOVE 20 TO WWCC
```

```
991231    ELSE              MOVE 19 TO WWCC.
991231    MOVE  'H'  TO  WWSYSNENGO.
991231    COMPUTE  WWSYSWANEN = WWCCYY  -  1988.
    *  MOVE  SHIZUKE  TO  YOKUNEN.
991231    MOVE  WWSYSDATE  TO  WWYOKUNEN.          <-- &P02
991231    ADD  1   TO   WWYYK.
    *   ADD  1  TO  YYY.
201800    OPEN  INPUT   I1-FILE   OUTPUT  O1-FILE.
202800    PERFORM  I1-READ          THRU   EX-I1-READ.
201900 EX-INITIAL-RTN.      EXIT.
203900*****************************************************************
```

**e．ZPRODAT データファイルの桁数調整プログラム**

このプログラムは、データファイルの中の指定する部分を、西暦年の場合は先頭に 19 をつぎたし、郵便番号の場合は、後尾に 2 スペースをつぎたすプログラム。つぎたし部分の指定には、ZPRO1, ZPRO4 で作成し、ZPROLNG でプログラムの書き換えに使用した .LOG ファイルと同じ形式のファイルを使うことができる。また、原文のレコード定義またはCOPY句を使うこともできる。

なお、ZPRODAT プログラムは、DOS 標準形式に変換されているデータファイルが対象で、1 レコードの長さが 4000 バイト以内に制限した。レコードの中の繰返し項目については、1 次元までしか対応していない。2 次元以上の繰り返しは例外的なサマリファイルしかないとの判断による。

データレコードの形式と桁数追加を指定するパラメータファイルの内容

```
**ZP7DATA
          01   MEIBOREC.
          03   KNO    PIC  9(4).
          03   KSH    PIC  X(4).
          03   KCAP   PIC  9(8).
          03   KDOM   PIC  X(4).
/D        03   RDATE  PIC  X(6).
/J        03   JIP    PIC  X(6).
          03   KADR   PIC  X(40).
   -- <-- プログラムから抽出したときの行番号
          または原文の行番号      なくてもよい。
```

形式は次のとおり。

1-2 桁目 **　　ファイル名　　注釈扱い

/D　年数の継ぎ足し対象部分
/J　郵便番号の引き延ばし対象部分
/M　縮める項目
ｽﾍﾟｰｽ なにもしない

3 - 4 桁目　数字　減少させる桁数
なお、もとの PIC の桁数を減少しないことに注意

/M のとき減少させる桁数を指定していないとエラー

5 - 8 桁目　行番号

10 桁目以降　なにが書かれていても無視する

ZPRODAT の実行結果

変更前のデータ

```
0001111100560000811093070127 2    市川市市川1-25-20 ﾒｿﾞﾝ市川
0002111100680000811093070156 6    摂津市三島2-16-39
0003113102530000811093070165 8    神戸市東灘区岡本2-13-12
0004113108650000811093063060 5    京都市東山区問屋町通り五条下ﾙ
0005113106500000811093063065 8    神戸市東灘区住吉山手8-21-1
0006121007800000412093063050 1-31 岐阜市北山11-132-7
0007823115480000121093063067 1-31 兵庫県佐用郡南光町河崎
                    |  |  |
```

変更後のデータ

```
                    |  |  |        .SIZ の拡張子をつけたファイル
                    | DATE | JIP |
0001111100560000811019930701272    市川市市川1-25-20 ﾒｿﾞﾝ市川
0002111100680000811019930701566    摂津市三島2-16-39
0003113102530000811019930701658    神戸市東灘区岡本2-13-12
0004113108650000811019930630605    京都市東山区問屋町通り五条下ﾙ
0005113106500000811019930630658    神戸市東灘区住吉山手8-21-1
0006121007800000412019930630501-31 岐阜市北山11-132-7
0007823115480000121019930630671-31 兵庫県佐用郡南光町河崎
```

---

### f．ZPROLNG プログラム原文中の PIC 句の書き換えプログラム

プログラム原文ファイルの中にある、指定文字列のある行を検索して作った .LOG ファイルの中で / または * を先頭につけた行の PIC句を 2 桁ずつ、または指定した桁数を増減して書き換えるプログラム。FILLER の行でもマークできる。

ただし、行番を頼りに書き換えを行なうので、プログラム原文のファイルは .LOGファイルを作ったときと同じものを使用する必要がある。

桁数追加を指定するパラメータファイルの形式と内容は、ZPRODAT と共通とする。
データの物理的な形式変更は、プログラム中のデータ記述の変更を伴うから、できるだけ、同じパラメータを使用することが望ましい。

形式は次のとおり。

1 - 2 桁目 **　ファイル名　　注釈扱い

/D　年数の継ぎ足し対象部分
/J　郵便番号の引き延ばし対象部分
/M　縮める項目
ｽﾍﾟｰｽ なにもしない

3 - 4 桁目　数字　減少させる桁数
なお、もとの PIC の桁数を減少しないことに注意

/M のとき減少させる桁数を指定していないとエラー

5 - 8 桁目　行番号

10 桁目以降　なにが書かれていても無視する

例

```
 43      01  DATE-REC.
/D  44          03  NEN        PIC 99.
/D  45          03  TSUKI      PIC BB99.
/D  46          03  HINICHI    PIC 99BB.
/D  65          05  HIZUKE     PIC X(06)BB.
/D4 66          05  DENBAN     PIC BB9(04).
/D  123             05  NEN-SHUKKO-SURYO    PIC  ZZ,ZZ,ZZ.
/J  124             05  NEN-SHUKKO-KINGAKU  PIC  Z(08).
/D  126         03  IDOBI      PIC     B(2)XXXXXX.
```

```
/M4 140        03  FILLER    PIC  X(24).
/D  174        03  LHIZUKE   PIC  B(2)X(6).
197 201800     OPEN  INPUT I1-FILE I2-FILE I3-FILE DATE-FILE.
199            READ  DATE-FILE AT END
```

テスト結果

| 引き延ばし前 | | 引き延ばし後 |
|---|---|---|
| 01 DATE-REC. | | |
| /D PIC 99. | 03 NEN | PIC 9999. |
| /D PIC BB99. | 03 TSUKI | PIC BB9999. |
| /D PIC 99BB. | 03 HINICHI | PIC 9999BB. |
| /D PIC X(06)BB. | 05 HIZUKE | PIC X(08)BB. |
| /D4 PIC BB9(04). | 05 DENBAN | PIC BB9(08). |
| /D PIC X(01). | 05 HENKO | PIC X(03). |
| /D PIC B(2)X(4). | 05 BUMON | PIC B(2)X(6). |
| /D PIC 9(06). | 05 NEN-NYUKO-SURYO | PIC 9(08). |
| /D4 PIC S9(8). | 05 NEN-NYUKO-KINGAKU | PIC S9(12). |
| /D PIC ZZ,ZZ,ZZ. | 05 NEN-SHUKKO-SURYO | PIC ZZZZ,ZZ,ZZ. |
| /D PIC Z(08). | 05 NEN-SHUKKO-KINGAKU | PIC Z(10). |
| /D PIC B(2)XXXXXX. | 03 IDOBI | PIC B(2)XXXXXXXX. |
| /M4 PIC X(24). | 03 FILLER | PIC X(20). |
| /D PIC B(2)X(6). | 03 LHIZUKE | PIC B(2)X(8). |

---

**参考文献**

情報システムの西暦2000年対応の実務資料集　日本情報システムユーザ協会 ( 1996.6 )
IBM 西暦2000年対応　計画・導入ガイド　第4版 ( 1996.9 )
コンピュータ西暦2000年問題の衝撃　　足立　晋著　実業之日本 ( 1996.9 )
2000年問題の傾向と対策　大林　久人・越智　洋之著　AI出版　( 1997.3 )