プログラム索引道具群の作成経験

富士通株式会社　君島　浩
Hiroshi Kimijima

原始プログラムのキーワードの索引を印刷する道具群を作成した．我々は教室教育の負荷を減らすべく，実際のプログラムを教材として使えるようにすることを検討中してきた．索引道具群は実用プログラムであり，なおかつ教材でもある．索引道具群はシステムプログラムおよび事務処理プログラムの基本アルゴリズム・基本パターンを網羅している．またこの道具群は，小さいながらもプログラム群，プログラム，モジュールという3段階の構成を持つ情報処理システムである．そこにはデータ，プログラム，モジュールをできるだけ再利用する姿勢を見ることができる．教材という点を抜きにしても，プログラム索引道具群はソフトウェア開発・保守にとって，是非必要なものであり，もっと注目すべきものである．

1. プログラムの索引

システムプログラムの開発・保守の単位は，アセンブラのような単一のプログラム，またはコンパイラ＋実行時ライブラリのようなプログラム群である．その原始コードは複数のモジュールからなる一つの区分ファイルに代表される階層的ファイルの形を取る．我々の場合にはＧＥＭファイルを用いているが，ここでは区分ファイルという名で説明する．プログラムが大規模になるにつれて，あるキーワードが原始コードのどの位置に出現するかを示す索引の必要性が増す．位置を表すのには，ページ番号よりはモジュール名の方が便利である．キーワードとしては次のようなものがある．

・モジュール名
・共用データ名
・メッセージ識別子

ここでモジュール名は，あるモジュールを使っているモジュールを知るのに必要である．共用データ名は，共用データに値を設定しているモジュールを探して，障害の容疑モジュールをしぼったり，インタフェース変更の影響範囲を知るのに必要である．あるメッセージをどのモジュールが出しているのかを調べることは，保守・改造のときにしばしば行われる．モジュール内で特定の名標を探す苦労に対しては，ＧＯ　ＴＯなしコーディングや，流れ図をはじめとする図形表現など，いろいろな工夫がほどこされた．モジュール内の探索よりも，モジュールを超える探索の方が大変であるのに，索引などの工夫があまり普及していなかったのはどうかしている．そこでプログラム索引道具群を作ることにした．

2. 動機

この道具群を作る具体的な動機となったのは，以前紹介した部品集「ＳＰＬジャンク箱」<12>である．いくつかの部品が他の部品を使用しており，あるいはデータを共用している．ジャンク箱のプログラムリストに目次や索引を付けることを考えた．またジャンク箱は読ませる教材<１，５，13，17，18>として使うことを意図したものであったが，教材として次のような欠点があることが指摘された．

・サブルーチンなので，プログラム全体の構造の手本にはならない．
・アルゴリズムがあまり出てこない．配列や連鎖リストは，データ要素の内部形式がいろいろなので，共通サブルーチンで扱いにくいためである．部品で扱えるのは，文字列の操作アルゴリズムくらいであった．

以上のような問題があったために，部品（サブルーチン）に加えて道具（プログラム）を教材として開発することにした<16>．索引道具群には次のような利点もある．

・教材はおもちゃであってはならず，ある程度複雑な動作をする必要がある．一般に複雑な動作をするプログラムは入力や機能仕様が複雑で，機能を理解し，入力データをそろえるのが困難である．入力が単純で見ばえのある動作をするプログラムは，数値計算などに限られる．ところが索引道具群の場合には，機能は明快であり，入力データとしては既存プログラム（道具自身も含む）が，いくらでも存在する．
・計算機科学科で教えるようなアルゴリズムの他に，事務処理プログラムの定石も登場する<15>．システムプログラムには事務処理プログラム的な部分が結構ある．
・集合や関係など離散数学の基本概念が，身近な形で出てくる．

3. プログラム分割

プログラムのモジュール〔への〕分割は，すっかり常識になったが，システムのプログラム〔への〕分割は，重要な割に常識になっていない．ソフトウェア開発量をできるだけ増やさずに，各種の情報処理を行うには，情報処理システムの共通点を見抜いて，最小のプログラム群をそろえるとよい．そのようなプログラム群をツールセットとかツールキットと呼ぶこともある．我々は索引道具群を含むソフトウェア開発用の道具群を，ＳＰＬジャンク箱に対応させて「ＳＰＬ道具箱」と名付けた．たとえ当面一つの情報処理システムを作るのが目的であっても，汎用性の高いプログラムに分割することは，試験容易性，信頼性，保守性などの観点で非常に有利である．

プログラム索引印刷という情報処理システムを，どのようなプログラムに分割したらよいかをまず考える<２，３，10，19>．プログラム分割の他に，プログラムのモジュール分割も考えて<７>，共通点を部品にするようにする<４，６，11，12，14，15>．部品は既に存在するので，索引印刷という分野の専用部品を追加する形を取る．

システム全体を次のように分割する．

(1) 区分ファイルを入力して，キーワード辞書を作る道具群

(2) 区分ファイルとキーワード辞書を照合して，各モジュールのキーワード一覧を作る道具群

(3) 各モジュールのキーワード一覧を入力して，各種の索引を印刷する道具群

ここで(1)と(2)を分けるのは，(2)の道具をプログラム以外の文章ファイル一般に使えるようにするためである．また印刷という言葉は重要である．道具箱の扱うファイルは，カードイメージのレコード列に統一している．この標準形式ファイルをデータファイルと呼ぶ．データファイルは辞書などを表すのにほどのよい，Ａ４版に相当するレコード長を持っている．このファイルは再入力しやすいし，いざとなれば人間が端末で読むこともできる．これに対していわゆるラインプリンタイメージのファイルをリストファイルと呼ぶ．リストファイルはＢ４版に相当する長いレコード長を持つ．このファイルは人間の見やすさのためにヘディングを付加したり，行や桁をあけたりするので，再入力しにくい．データベースを例に取るまでもなく，同じような情報を含むファイルをいくつも作るのはよくない．一度作ったファイルは，できるだけ再利用すべきである．したがって我々はできるだけデータファイルを使用するようにしている．我々の道具群が「リストファイル」を「印刷」するのは，それ以上再入力する可能性のないファイルの場合に限られる．

4. キーワード辞書を作る道具群

4.1 モジュール名

モジュール名は区分ファイルのディレクトリに載っている．道具ＰＵＮＣＨＤは区分ファイルを入力して，モジュール名（メンバ名）の一覧を順次ファイルの形で出力する．これを辞書として，区分ファイルのデータ部と照合すれば，モジュール間の相互参照関係が分かる．辞書を使わない方法，すなわち構文解析をしてモジュール呼出しを検出する方法は，複雑であるし，言語に依存する．システムプログラムの場合，一つのプログラム中にＳＰＬモジュールの他にアセンブラモジュールを含むことが多い．また物によってはＣＯＢＯＬ，ＦＯＲＴＲＡＮ，ＰＬ／Ｉモジュールや更にはコマンドプロシジャなどを含むこともある．したがって構文解析による方法は益々不利である．

4.2 共用データ

共用データは一般に基底付データであり，その宣言を独立のモジュールにするのが一般的である．そしてそれを操作する手続きモジュールは％ＩＮＣＬＵＤＥ文かアセンブラのＣＯＰＹ命令で，その宣言を複写する．この場合も参照する文・命令を構文解析するのではなく，宣言する文・命令の方を構文解析する．その方が共用データ一覧ファイルが，それだけで有用な可視情報になったり，別の用途に使われたりするなど，いろいろと便利である．

道具ＰＵＮＣＨＩＤは区分ファイル中のＤＥＣＬＡＲＥ文で始まるモジュールを探して，宣言されている名標の一覧を，順次ファイルの形で出力する．

道具ＰＵＮＣＨＩＤＡはアセンブラ用の名標一覧出力道具である．区分ファイル中のＤＳＥＣＴ命令で始まるモジュールを探して，宣言されている名標の一覧を，順次ファイルの形で出力する．

メッセージ識別子は我々の場合は，Ｊで始まる３文字の英字に３～４文字の数字，そのあと必要なら英字１文字の重傷度表示で構成される．これはそれを参照しているモジュールを解析するしかないので，前もって一覧ファイルを作ることはしない．

モジュール名一覧ファイルや共用データ名一覧ファイルは，文章との照合に使う場合，辞書ファイルと呼ぶ．プログラム中の注釈やメッセージ文章の綴りミスを発見するときには，英単語辞書を使う．ＳＰＬ道具群は辞書ファイルを１単語／１レコード，各レコードの先頭１～１６文字が単語欄と標準化している．この標準化は内部辞書の操作を共用部品で済ませたり，道具を流用・改造で作るとき，入出力部分の変更なしで済ますのに役立つ．

5. 文章と辞書との照合

5.1 キーワードと全モジュールの照合

道具ＲＥＦＡＬＬは区分ファイルと辞書ファイルを入力して，全モジュールのキーワード参照状況を次のような形のファイルにして出力する．この形式を一般にモジュール参照ファイルと呼ぶ．

```
  モジュール名
〔参照キーワード　出現回数〕
  ・・・
  区切りレコード
〔 モジュール名
〔参照キーワード　出現回数〕
  ・・・
  区切りレコード 〕
・・・
```

ＲＥＦＡＬＬはまず辞書ファイルを入力して連鎖リスト形式の内部辞書を作る．それから区分ファイルを字句解析部品で走査しつつ，内部辞書と照合して，辞書にある単語なら出現回数を辞書に書き込む．字句解析部品は区分ファイルアクセス部品を使って，区分ファイルをレコードレベルで入力する．一つのモジュールの処理の最後に参照キーワードを出現回数とともに出力する．この道具は字句解析という計算機科学のアルゴリズムと，照合という事務処理の基本パターン<15>を合わせ持っている教材といえる．こ

の一つのプログラムを，字句解析，整列，重複単語の除去（いわゆるｕｎｉｑｕｅ処理<10>），照合の四つのプログラムに分割する方法もある．しかし我々の方針の一つに，教材として使われることを意識せずに，経験者が良いと思った設計をする，というのがある．この場合経験者として性能の良い，一つのプログラムにする方を選んだ．字句解析を前処理プログラムでなく，下請部品にするのは定石であるし，整列は連鎖リストの処理で自然に行われるので，分けなくても処理は複雑にはならない．ただしＳＰＬ道具箱には字句解析や重複単語除去をする独立プログラムもある．

5.2　ＳＰＬ用の照合

道具ＲＥＦＳＰＬはＲＥＦＡＬＬの字句解析部品をＳＰＬ用字句解析部品に替えたものである．プログラムの注釈の中には，ＣＨＥＣＫ，ＧＥＴ，ＰＯＩＮＴなどの動詞がよく使われるが，これがシステムマクロ名と一致するので，システムルーチンの参照と間違われる．ＲＥＦＳＰＬは注釈を無視するので，そのような不便はなくなる．ＳＰＬ用字句解析はかなりの行数になるが，いろいろな道具に使われるので，部品が既にある．したがって，ＲＥＦＳＰＬを作る手間は，テキストエディタでＲＥＦＡＬＬを複写し，プログラム名と字句解析部品のＣＡＬＬ文を変更することだけである．

以上の二つの道具は索引作成のためにだけ使うのではない．辞書ファイルとしてＳＰＬ文キーワード一覧を選ぶと，どのモジュールがどんな文をどれだけ（出現回数が出るので）使っているかが分かる．辞書ファイルを端末に割り当て，単語一つだけを入力すれば，会話型検索道具になる．

5.3　メッセージ識別子の参照状況

道具ＲＥＦＭＩＤは区分ファイルを入力し（辞書ファイルは使用しない），ＲＥＦＡＬＬなどの出力と同じモジュール参照ファイルの形式で，メッセージ識別子の出現状況を出力する．一般にはメッセージ番号を表す定数データに名標を付け，その名標をメッセージ識別子と一致させるので，メッセージ識別子のあるモジュールは，そのメッセージを出力しているモジュールか，またはそれについてのエラーを検出しているモジュールである．

6．索引を印刷する道具群

索引には参照遷移表，参照マトリクス，参照木構造図，記号参照表の四種類ある．これらの各種の表の選択は，パラメタではなく，道具の組合せ，ファイルの選択により行う．

最初の三種類の図表はモジュール名だけについてのモジュール相互参照図表である．これは
ＰＵＮＣＨＤ＋ＲＥＦＳＰＬ（またはＲＥＦＡＬＬ）＋三種の索引印刷道具により作る．記号参照表はモジュール名だけでなく，共用データ名，メッセージ識別子をも含めた総合索引である．これは
ＰＵＮＣＨＤ＋ＰＵＮＣＨＩＤ＋ＰＵＮＣＨＩＤＡ＋ＲＥＦＳＰＬ（またはＲＥＦＡＬＬ）＋
ＲＥＦＭＩＤ＋ＰＲＴＢ４ＲＦＩにより作る．四種類の索引印刷道具はモジュール参照ファイルを入力して，Ｂ４版の索引を印刷する．通常はこの出力をシステム出力ルーチンにより仲介処理して，Ｂ５版に縮小して印刷する．

6.1　参照遷移表

道具ＰＲＴＢ４ＩＭＤはモジュール間の参照関係を親・子・孫三代の単位で列挙する．まず中央の欄に辞書順に列挙されたモジュール名の中から，注目するモジュールを探し出す．その行の左側にはそのモジュールを参照する（呼び出す，または複写する）モジュールが列挙される．右側にはそのモジュールが参照するモジュールが列挙される．一般には後述するマトリクスや木構造図のような，プログラムの全体像を眺める必要のある機会はあまりない．モジュール間関係を知るのには，この参照遷移表を見る方が便利なことが多い（図１）．

```
EXTERNAL REFERENCES  DATE 82.10.23  TIME 11.23.59

                    REFERENCING MODULES    MODULE    REFERENCED MODULES

                                HELP        |ADDCH    |   SAVE
                                HELP        |ADDCHS   |   SAVE
                    HELP        SLINKS      |ADDVAR   |   CALL        PROCOPT    PROGSH
          HELP      SLINKS      TESTO17     |ALTDDNAM |   BASICDCL    DCBD       PROCOP
                                            |         |   PROGSHRD    REGISTER
ALTDDNAM  BUILDD    CHECKPS     CLOSEALL    |BASICDCL |
CLOSEF    CVBTODZ   CVBTOH      CVBTOLSD    |         |
CVBTOSD   CVDZTOB   FINDC       FINDD       |         |
GETD      GETINTL   GETINTLP    GETL        |         |
GETLPS    GETMO     GETM1       GETM2       |         |
```

図１　参照遷移表の例

6.2　参照マトリクス

道具ＰＲＴＢ４ＭＡＴはモジュール間の参照関係をマトリクスの形で印刷する．あるモジュールとその参照モジュールの交点には，参照モジュール名の出現回数（１０以上は９として）が表示される．このマトリクスはシステムルーチンの使用状況を概観するときなどに便利である（図２）．

```
EXTERNAL REFERENCE MATRIX  DATE 82.10.23  TIME 11.28.59

        REFERENCED MODULES

        AAAABBBBBBBBBBBBBBBCCCCCCCCCCCCCCCCCCCCCDDDDDDDDDEEFFFFFFFFFFFFFFGGGGGGG
        DDDLAILLLLLLLLLUUUUAAHHHHLLLRVVVVVVVVVVVVCCCCCCEEYNXGIIIRRRRRRRRRREEEEEEE
        DDDTSLDDDDDDDDDFIIILTEERROOOEBBBBBBBBDDHHBBBBBBCCNCLENNNEEEEEEEEEETTTTTTT
        CCVDIDDDDDDDDLLFLLLLACCCCSSSATTTTTTTTZTT DMMMMBOAOSTDDDEEEEEEEEEE BDDDDI
        HHADCDCCCCCFF NPDDD LKKLLEEETOOOOOOOTOO  RRRRPDLDTM CDMMMMMMMMMM U ABTN
REF.ING  SRNDCBBBBBDD OO DD O PAT AFEDDEHLLSBOBF  GGGGSELE 2   AAAAAO1234 F TLRT
MODULES    ACB LPP6CC UO  C G SSB L F ZB DSD B B   LPP  O      ILLLL        EKYL
             MLL   L4BP DL  B    SL L       D         L  C     N1234
ADDCH    .........:.........:.........:.........:.........:.........:.........:..
ADDCHS   .........:.........:.........:.........:.........:.........:.........:..
ADDVAR   .........:.........1.........:.........:.........:.........:.........:..
ALTDDNAM....1....:.........:.........:.........:.1.......:.........:.........:..
BILDDCBL..........:.........1.........:.........:11......:.........:.........:..
BLDDCB   .........:.........1.........:.........:5.1.....:1........:.........:..
BLDDCBL  .........:.........1.........:.........:5..1....:1........:.........:..
BLDDCBP  .........:.........1.........:.........:5...1...:1........:.........:..
BLDDCBPL..........:.........1.........:.........:5....1..:1........:.........:..
```

図２　参照マトリクスの例

6.3　参照木構造図

道具ＰＲＴＢ４ＴＲＥはモジュール間の参照関係を木構造図の形で印刷する．木は根を左，葉を右にした横倒し形式であり，むしろ目次を字下がりで示して，若干の縦罫線を加えたものといった方がよいかもしれない．縦罫線はコロンの並びで表す．行数を減らすために子孫のないモジュール（葉）は，改行せずに同じ行に詰める．また共用モジュール参照は２回目以降については前回の表示部の行番号で表す．これらの工夫についてはＩＮＴＥＲＬＩＳＰのモジュール木構造図を参考にした．

モジュール木構造図をよく手で書くことがある．しかしこれを正確に書き，正確に保守するのは馬鹿馬鹿しい作業である．しかも平均的システムプログラムにおいてはＰＲＴＢ４ＴＲＥを用いても，木構造図が１０ページを超えることが少なくない．手書きなら５～７倍のページ数になる．

プログラムの木構造表現は下降型設計の影響でもてはやされたが，この参照木構造図は他の表と比べると使う機会は少ない．使用頻度は記号参照表（総合索引），参照遷移表，参照マトリクス，参照木構造図の順のようである（図３）．

```
EXTERNAL REFERENCE TREES  DATE 82.10.23  TIME 11.31.56

  1 BLDLLIST JSPL PATCH128 PATCH192 PATCH256 PATCH320 PATCH384 PATCH448 PATCH51
  2 SCNTKSP SCNTWD SCOMPF SEXTALL SMERGE SPRTIMD SPRTMAT SPRTRFI SPRTTRE SPRTTR
  3 SPUNCHDC SPUNCHID SPUNCHLC SPUNCHT SREFALL SREFLINE SREFSPL SSORT SST SSUBT
  4 GEMPUTIN
  5   PUT
  6 GSYMGCHR
  7   BASICDCL CALL PROGSHRD RETURN SAVE SPLSYMBL
  8   CHRCLASS
  9   : CHRCLTBL
 10   MOVE
 11   : PROCOPT
 12   : : RETURN
 13   PUTEDIT
 14   : BASICDCL CALL PROCOPT->11 PROGSHRD
 15   : PUTEDITC
 16   : : BASICDCL CALL CHRCLASS->8 PROCOPT->11 PROGSHRD RETURN SAVE TABULATE
```

図３　参照木構造図の例

6.4　記号参照表

道具ＰＲＴＢ４ＲＦＩは親子二代の参照関係を表す表を印刷する．左端にキーワードを辞書順に列挙し，それぞれの右側にそのキーワードが出現するモジュール（通常の索引のページ番号の代り）を列挙する．一般に入力にはモジュール間の参照関係に加えて，共用データの参照状況およびメッセージ識別子の参照状況を含める．モジュール名に限らず記号全般を示すという意味で，この表を記号参照表とか総合索引と呼ぶ．この索引によりモジュール名，共用データ，メッセージ識別子がどのモジュールに出現するかを容易に調べることができる．入力情報の統合には，併合道具ではなく，ファイルの連結指定やファイルに対する追加出力指定など，オペレーティングシステムの通常のファイル処理機構を利用する（図４）．

```
SYMBOL REFERENCES  DATE 82.10.23  TIME 11.40.13

SYMBOL    REFERENCING MODULES

ADDCH    | HELP
ADDCHS   | HELP
ADDVAR   | HELP      SLINKS
ALTDDNAM | HELP      SLINKS    TEST017
BASICDCL | ALTDDNAM  BUILDD    CHECKPS   CLOSEALL CLOSEF    CVBTODZ   CVBTOH    CVBTOLS
         | FINDC     FINDD     GETD      GETINTL  GETINTLP  GETL      GETLPS    GETMO
JXX009   | GETSYMSP  GSYMGCHR  GSYMGCMT  GSYMGSP
JXX010   | GETSYMSP  GSYMGCHR  GSYMGCMT  GSYMGSP
K        | INITOUTF  S@TSTALL  TEST021
LBL      | BASICDCL
LINE     | PROGSHRD  PROGSHRV  PUTLP     PUTTITLE RECHEAD   REGHEAD   TEST001   TURNPAG
LINK     | SLINKS
LOAD     | JLINK     JLOAD     S@PRTASM  S@PRTJNK S@PRTLIB  S@PRTLKD  S@PRTPRT  S@PRTUT
```

図4　記号参照表（総合索引）の例

7. ジャンク箱と道具箱による学習

システムプログラマはジャンク箱と道具箱を，プログラム言語の構文を使って複写したり，呼び出したりする形で，まず利用する．使用手引書には原始コードも載せているので，次第にその中に使われているアルゴリズムを理解していく．理解できると，そのまま使うだけでなく，それを参考にして別な処理論理を設計するようになる．更にはテキストエディタを使って，原始コードを複写し，変更して，別のモジュールや道具を作るようにもなる．

建築設計士や音響機器製作マニアは，既存の設計集をよく見るが，同じようにこのジャンク箱と道具箱の使用手引書は，システムプログラムの設計集として気軽に読まれる．今までは適当な設計集がなかったが，これができたことにより，作業をする過程で設計のノウハウを知ることができる．一般的なプログラミング方法論の教科書は，具体性に欠けるので，作業をしながら勉強するのは困難である．既存の巨大なプログラムは，具体的ではあるが，読むだけでも時間がかかる．ジャンク箱・道具箱の使用手引書は，原始コード込みでもごく普通のプログラムのマニュアル程度のページ数なので，読みやすい．企業においては特別な教科書や講義なしで学習できるに越したことはない．ジャンク箱・道具箱は教室教育ではなく，作業中に読むということを意図して作った教材である．

8. 教室教育と宣伝

ジャンク箱・道具箱を使って，教室教育も行ってみた．次のような2.5日（正味１２時間）の講座である．これは教育講座というよりも，ジャンク箱・道具箱の存在を知らせるための，製品説明会的な会合といった方が当っている．

第１日　リスト標準とページ替え制御（２時間）　出力のデータ変換と書式制御（３時間）
第２日　字句解析と構文解析（２時間）　プログラム索引道具群のシステム構造（３時間）
第３日　表操作と記憶域管理（２時間：これは少し時間が不足した）

講義した感じでは，以前から行っている文体教育＜９＞と比較して，ずっと中味が濃かったようである．

何よりも話に出てくる語彙がはるかに豊富である．例えばページ替え制御については，用語としてページ，行，欄外見出し，表題，ベタ打ち，日付，時刻，版，レベル，などが出てくる．更に重要なのは，このような用語に対するコーディング名標までが，決められていることである．分かりやすいコーディング名標を付けよ，と一般論を言うよりも，具体的に洗練されたコーディング名標を示して，組織として標準化する方がずっとよい．このような標準化は事務処理プログラムで盛んに行われるようになったが，我々は同じことをシステムプログラムにおいても実施したのである．これらの教材は，アルゴリズムやコーディング名を教えるだけではなく，文体やモジュール分割などの良い例を示すことにもなる．

教室教育の一方で，道具箱の中のプログラム索引道具群を別途売り込んだ．各プロジェクトグループが開発・保守しているプログラムのファイル名を聞いて回り，片っ端から索引を印刷してプレゼントした．道具を使ってもらう前に，その出力結果をまず使ってもらったのである．プログラム索引は非常に有用なので，自分達でも出力したいという要求が生じた．それからプログラム索引道具群の使用方法を会議などで説明した．みんなの目を最終的に，道具箱の原始コード中のアルゴリズムに向けさせることを，この活動は狙っているのである．

9. まとめ

本稿ではプログラム索引道具群を紹介して，次のようないろいろな提案を行った．

プログラム文書化：プログラム索引はあまり注目を浴びていないが，有用な保守資料である．これは道具により，意外に簡単に作成できる．

プログラム構造論：情報処理システムを再利用可能な既存プログラム群を活用しつつ，慎重にプログラム分割することの重要性を訴える．

企業内教育：企業内教育においては，実務で使われている設計集で技術を伝授できるなら，教育の費用が安くて済む．部品集・道具集は，中味の濃い読みやすい教材になる．

参 考 文 献

1) 宇都宮：職業的プログラマ教育としてのプログラムの解読，情報処理学会，職業的プログラマの育成シンポジウム報告集，p．134 (1976)．

2) M. A. Jackson：Information Systems ―Modeling, Sequencing and Transformations, Proc. of 3rd International Conference on Software Engineering, pp. 72 81 (1978)．

3) J. P. Morrison：Data Stream Linkage Mechanism, IBM Systems Journal, Vol. 17, No.4, pp.383―408 (1978)．

4) L. A. Belady：Evolved Software for the 80's, IEEE Computer, pp. 79―82 (Feb. 1979)．

5) T. Kimura：Reading Before Composition, ACM SIGCSE Bulletin, Vol. 11, No. 1, pp. 162 ―166 (1979)．

6) R. Lanergan:Reusable Code — The Application Development Technique of the Future, Proc. of SHARE 53, pp. 696-703 (1979).

7) D. L. Parnas:Designing Software for Ease of Extension and Contraction, IEEE Transactions on Software Engineering, Vol. SE-5, No. 2, pp. 128-138 (1979).

8) 牛島：Fortranプログラミングツール，産業図書 (1979).

9) 君島：SPL／100の文体，情報処理学会，よいプログラムを作るにはシンポジウム報告集，pp. 33-39 (1979).

10) D. E. Hall, et al.:A Virtual Operating System, Communications of the ACM, Vol. 23, No.9, p.499 (1980).

11) 君島：根拠あるソフトウェア開発へのアプローチ — プログラム部品の整備，信学技報R80-8 (1980).

12) 君島：プログラムジャンク箱の現場への導入，情報処理学会，作譜工程の評価・改良・自動化シンポジウム報告集，pp. 15-20 (1980).

13) 日本電子工業振興協会：ソフトウェアエンジニアリングに関する調査 — ソフトウェアの生産性と品質の評価方法 — ソフトウェア技術者の効果的教育（ソフトウェアエンジニアリング専門委員会報告書），日本電子工業振興協会55-C-395 (1980).

14) 日本電子工業振興協会：ソフトウェアエンジニアリングに関する調査 — ソフトウェアの部品化（ソフトウェアエンジニアリング専門委員会報告書），日本電子工業振興協会56-C-416 (1981).

15) 藪田他：ソフトウェアの再利用 —パラダイム—，情報処理学会第23回全国大会予稿集，pp. 339-340 (1981).

16) B. W. Kernighan他，木村訳：ソフトウェア作法，共立出版 (1981).

17) 君島：開発・教育・研究におけるプログラムの読みと書き，情報処理学会，第23回プログラミング・シンポジウム予稿集，pp. 85-91 (1981).

18) 君島：ハードウェア技術者へのソフトウェア工学入門教育の経験，情報処理学会第23回全国大会予稿集，pp. 391-392 (1981).

19) W. P. Stevens:How Data Flow Can Improve Application Development Productivity, IBM Systems Journal, Vol. 21, No.2, pp.162-178 (1982).