ソフトウェア工学 71－6
（1990．2．8）

実行時エラーに関するプログラム相談システム Consult:R

大西 淳[†]、佐藤真木彦[††]、島崎真昭[†††]
[†]京都大学、[††]富士通株式会社、[†††]九州大学

Fortranプログラムの実行時に生じるエラーに関する相談システム「**Consult:R**」を開発しているので紹介する。相談システムの特長として、

1)エラーが検出されたソースコードを解析することによって、あらかじめ用意したエラー原因の中から、そのエラーの原因を特定し、正確な対処法を示す、

2)それ自体は誤りでないのに、他のエラーが原因となって生じるエラーメッセージに対して、根本となっているエラーを指摘する、

3)個々の利用者の習癖を利用して、効率良くエラー解析を行う

といった3点があげられる。

相談手法とシステムの実行例について述べると共に、コンパイル時のエラーに関する相談システムについても言及する。

Program Consultation System for run-time errors, Consult:R

by Atsushi OHNISHI [†], Makihiko SATO[††], Masaaki SHIMASAKI [†††]
[†] Kyoto University, [††] Fujitsu Ltd., [†††] Kyushu University
[†]Sakyo, Kyoto 606, JAPAN

A program consultation system for run-time errors of Fortran programs, named "Consult:R," is presented. The features of the system are as follows.

1) Causes of error are closely analyzed by scanning the source program where the errors are detected.
2) Ripple effects from other errors can be detected.
3) By providing user model which reflects user's habit in making a program,consultation can be effectively assisted.

Our consultation method and system's behaviors are described. The other consultation system for compile-time errors is also described.

1. はじめに

京都大学大型計算機センター(以下、京大センターと略す)では、利用者のプログラミング技術の向上と計算機システムの効率的な利用の促進を目指して、システムの利用法やメッセージの意味に関する問い合わせなど利用者からの種々の質問や相談に応えられるようにプログラム相談サービスを行っている。近年の計算サービスの多様化に伴って、相談件数も年間でおよそ8,000件と10年前の3倍に増加するとともに相談内容も言語、統計パッケージ、ネットワーク、データ利用など多岐に渡る傾向が見られ、相談員の負担が大きくなってきている。

相談員の負担の軽減のための一つの方策として、計算機支援によるプログラム相談が考えられるが、これについては、国内でもかなり以前から研究されてきたものの[1-6]、実用化には到っていない。この理由としては

ア)相談システムの利用者が自分で判断しなければならない場合が多い。初心者にとっては判断すら下せない場合があり、また熟練者にとっては応答するのが面倒であり、使い勝手が良くない。

イ)実際には間違っていないのに、他で生じたエラーが原因でエラーメッセージが出力される場合がよく起こるが、そのようなエラーを区別できない。このため、本来は間違いではないのに、その対処を指摘することがあり、利用者は混乱してしまう。

ウ)利用者の習癖(例えば打鍵の際に隣接する文字が反転し易いなど)が考慮されていない。実際には、ある原因でエラーが生じる場合に、その原因の生起する確率は利用者によって異なる場合があるが、それが考慮されていないために、効率的な対処ができない。

などが考えられる。

我々は、上に示した従来のプログラム相談手法や相談システムの問題点を解決するようなプログラム相談の自動化手法の研究を進めており、手法に基づいてFortranプログラムのコンパイル時のエラーと実行時のエラーに対処するシステム[7, 8]を開発中であるが、本稿では手法と実行時のエラーに対する相談システムについて紹介する。

2. プログラム相談の実態とシステム化

京大センターのプログラム相談室には、運用しているプログラミング言語、データベース、統計パッケージ、グラフィックス等のソフトウェアから、パソコン、汎用機、スーパーコンピュータ、画像処理システムなどのハードウェアまで、様々な質問や相談が寄せられるが、そこで扱う相談のすべてを対象としたシステム化は困難であると予想されるため、比較的件数が多く、かつシステム化の容易なものを対象とすることにした。

まず最近1年間の相談内容について分類した結果、プログラミング言語に関する相談が一番多いことが判明した[9]。センター利用者のほとんどはプログラム言語としてFortranを利用しており、プログラミング言語の相談件数の8割以上はFortranである。次にFortran関係の相談と原因について分析したところ、1/3は実行時エラーに対する相談であり、コンパイルとリンク時の相談を合わせると全体の半分を占めることが判明した。残りは、言語というよりOSやジョブ制御言語に関するものであった。

一般にプログラムが正しく動作するかは、プログラム仕様(プログラマの意図)をプログラムが満足するかどうかを調べることによって明らかになる。プログラム相談システムにおいて、仕様やプログラマの意図を的確に捉えることは難しい。一方、コンパイルやリンク時では、誤りはコンパイルエラーやリンクエラーとして認識できる。別の言い方をすると、例えばコンパイラにとっての「正しさ」はFortranの文法通りにプログラムが記述されていることであり、「正しさ」の基準が明確である。このため、プログラムの仕様なしでも、ソースプログラムとエラーメッセージから、ある程度エラー現象を把握できる。同様に、実行時においても、実行時エラーのメッセージ(例えば「配列の上限を越えてアクセスした」、「演算オーバーフロー」など)が得られる場合は、メッセージとソースプログラムや入力データを解析することによって、ある程度エラー現象を把握

できる。

以上述べてきたことを考慮して、Fortranプログラムのコンパイル時と実行時(リンク時を含む)のエラーに対する相談の自動化について研究を進めている。

3. プログラム相談手法:Consult

2章で述べたように、我々はコンパイル時と実行時のエラーに対する相談の自動化を進めているが、両方の手法/システムを合わせてConsultと呼んでいる。特にそれぞれを区別したい場合は、コンパイルエラーに対する相談手法/システムをConsult:C、実行時エラーに対する相談手法/システムをConsult:Rと呼ぶ。ここでは、Consult手法の特長と手順について述べる。

3.1 特長

相談手法Consultの特長は以下の4点である。

1) エラーメッセージだけでなく、エラーが生じたソースプログラムを解析することによって、エラーの原因を同定する。
2) 他のエラーが原因となって、見かけ上、生じたエラー(このようなエラーを以下、「波及エラー」と呼ぶ)かどうかを判定する。
3) 利用者の誤り方について一般的なモデルと各自に固有なモデルを用意することによって、より効率的にプログラム相談を支援する。
4) 解析の結果、エラーの修正方法が明らかな場合、利用者の確認を得た後にソースプログラムを自動修正する

従来の計算機支援によるプログラム相談は、ソースプログラムを解析しないで、エラーメッセージだけから原因を推定するものがほとんどであり、原因を的確に指摘できなかったり、利用者の判断を仰ぐ場合が多く、さらに、波及エラーに対処できないところから、実用には至らなかったと思われる。Consultでは特長の1と2により、これらの問題点を解決している。また、特長の3の利用者のバグモデルと、4の自動修正機能によって、使い勝手を良くしている。

3.2 相談の手順

前提として、Fortran処理系からのエラーメッセージに対して、考えられる症状とその原因が用意されているものとする。従って、原因が用意されていないエラーコードについては対処不能となる。現在のところ、Consult:Cでは約60種類、Consult:Rでは20種類のエラーコードに対して原因を用意している。用意した原因に抜けがある場合は、原因不明となる。

相談手順は、

(1) 相談対象のソースプログラムを読み込む
(2) コンパイラを起動し、クロスリファレンス情報やエラーコードがあれば読み込む
(3) エラーコードがある場合は、Consult:Cに制御が渡され、ソースプログラムを走査することによって、エラー原因を絞り込み同定する。エラーの対処法を示すとともに、可能な場合はエラーを修復する。
(4) コンパイル時のエラーがない、もしくはあっても相談対象プログラムを実行可能な場合は、(3)をスキップして、Consult:Rに制御を渡すことができる。この場合は、必要に応じて入出力ファイルを割当てるよう利用者に指示する。
(5) リンクし実行する。
(6) 実行時に生じるエラーコードに対して、ソースプログラムを走査することによって、エラー原因を絞り込む。場合によってはコンパイルオプションでデバッグ指定を行い、再コンパイル、再実行する。
(7) エラー原因を同定すると、その対処法を示す。可能な場合はエラーを修復する

なお、波及エラーと判断した場合は、波及エラーの原因となっているエラーの解析に取り掛かる。以下ではConsult:Rの場合についてさらに詳しく説明する。

[前提1]システムからのエラーメッセージに対して、考えられる症状をあらかじめ用意する

例えば、システムからのメッセージとして、

> **JZL186I-W name HAS AN UNDEFINED VALUE.** (nameの値が未定義である)

が、得られた場合、症状としては、

(a)定義文がない
(b)定義文、または引用文での英字名のタイプのミス

(c)定義文はあるが、それを実行する前に参照文を実行した

が考えられる。

[前提2]さらに症状のもととなっている原因を用意する

例えば上の(a)の「定義文の欠如」の原因として

(a1)プログラマは定義の必要性を知らない

(a2)定義が必要なことは知っているが、たまたま定義文を忘れた

(a3)定義が必要なことは知っているが、省略時解釈でよいと考え定義文を省略した(ゼロクリアを期待して定義文を明記しなかった)

(a4)定義文がコメントになっている

(a5)定義文が他のエラーにより無視された

が考えられる。(b)のタイプミスには、

(b1)文字列中の二つの隣接した文字が置き換わる。

(b2)ある文字が抜け落ちる。

(b3)ある文字が誤って重なる。

(b4)キーボードの隣のキーと置き換わる。

などが、考えられる。(c)には、

(c1)論理ミスやデータミス

が考えられる。

このように、エラーメッセージ1つ1つに対して、『ユーザはそのエラーをどうやって起こすか』という原因をあらかじめ用意しておく。相談システムは、用意した原因のうち1つを選んで、それによってエラーが生じているかどうかをソースプログラムを走査して探す。エラーが見つかったならば、ユーザにその原因と対処法を提示できる。さらに、場合によっては、エラーを自動的に修正することも可能である。見つからなかった場合は、用意した別の原因について同様の操作を繰り返す。具体的な手順を以下に示す。

手順1) 相談対象のソースプログラム名を利用者に指定させ、プログラムを読み込む

手順2) ソースプログラムに対してコンパイラを起動させ、クロスリファレンス情報やコンパイルエラーがあればそのコードを読み込む

手順3) プログラムを実行するには支障のあるエラーコードがある場合は、Consult:Cに制御が渡される。(が、ここではそのようなエラーはなかったと仮定する)

手順4) Consult:Rに制御が渡たされる。最初に、実行に必要な入出力ファイルがあれば、それらを割当てる。

手順5) リンカを起動させ、ロードモジュールを作成し実行させる。

手順6) 実行時に生じるエラーコードに対して、ソースプログラムを走査することによって、エラー原因を絞り込む。場合によってはコンパイルオプションでデバッグ指定を行い、再コンパイル、再実行する。

先に示したJZL186Iの場合の手順6では、まず、コンパイル時にエラーメッセージJZK524I『この英字名はこのプログラム単位内で未定義である』が生じているかどうかを調べ、もし生じている場合は、次にコメント文の中に英字名の定義がないか走査する。もし見つかればa4が原因であると判断する。見つからなければ、他のエラーによってコンパイラに一行削除され無視された文に注目して英字名の定義を探す。見つかればa5が原因であると判明し、波及エラーと判る。

それもなければ、クロスリファレンスから上に示したタイプミスによる類似した文字列が存在するかどうか調べ、類似文字列の英字名の定義文があり、しかもエラーが出た名前と同じ文字列の英字名の参照がある場合は原因b「定義文でのタイプミス」と見なす。類似文字列の英字名の定義文はあるが、同じ文字列の英字名の参照がない場合は原因b「引用文でのタイプミス」と見なす。類似文字列がない場合は、原因a1,a2,a3と見なす。

もし、コンパイルエラーJZK524Iが生じていない場合は、タイプミスによる類似した文字列が存在するかどうか調べ、類似文字列の英字名の定義文があり、しかもその文字列の英字名の参照がない場合は原因b「定義文でのタイプミス」と判断する。類似文字列の英字名の定義文があり、しかもその文字列の英字名の参照もあり、エラーの出た文字列より類似文字列の方が出現する頻度が多い場合は原因b「引用文でのタイプミス」と見なす。そうでなければ、原因a1,a2,a3と見なす。

これらの原因すべてが該当しない場合は、原因cと判断する。

手順7) エラー原因を同定すると、その対処法を示す。可能な場合はエラーを修復する

例えば出現頻度と類似性から明らかにタイプミスと判明した文字列は利用者の確認を得た上で修正する。波及エラーの場合は、利用者の確認を得た上で原因となったエラーについて、さらに解析を続ける。

なお、a1とa2とa3の差異はソースプログラムの走査だけでは特定できないが、用意する利用者モデルから利用者が初心者の場合はa1の可能性があることがわかる。

利用者モデルには利用者の熟練度(初心者や上級者)といった情報や習癖(例えば、よく犯すタイプミスのパターン)、相談の履歴(過去に相談したエラーごとに、どのような原因から、そのエラーを起こしたか)が利用者ごとに用意されている。熟練度は原因の特定に用い、習癖は原因の解析時の参考にし、相談の履歴によって原因を解析する順序を決める。

4. 相談システム:Consult

Consultはコンパイル時のエラー対処用のConsult:Cと実行時のエラー対処用のConsult:Rから成る。本稿ではConsult:Rについて述べるが、双方で共通している点も多い。

Consult:Rは実行時のエラーメッセージとソースプログラム、クロスリファレンス、(入力データ)を外部からの入力として、

(1)エラーメッセージの説明、

(2)エラー原因の説明、

(3)エラーの対処方法の提示

を行うシステムであり、利用者からの指示によってエディタを起動したり再コンパイルできるようになっている。また、エラーコードによっては、デバッグオプションを指定して解析すべきであると判断し、利用者の確認を得た後に、自動的にオプションを付加して再コンパイルし、再実行する機能を備えている。さらに、実行時に必要な入出力ファイルがある場合は、システムからファイルの割当てを行ったり、割当てられているファイルの一覧を表示する機能も備えている。図1にConsult:Rのシステム構成を示す。

Consultの構築に当っては、京大センターで一番利用の多い言語である富士通のFortran77で書かれたソースプログラムを相談の対象とした。同じFortran77であっても他メーカのものになるとエラーのコードは一致しないが、IBM系の場合はエラーコードが番号も含めてかなり類似しており、若干の修正によってConsultが適用できると思われる。もちろんFortran77以外の言語になるとエラーコードと原因の対応を新たに構築する必要がある。

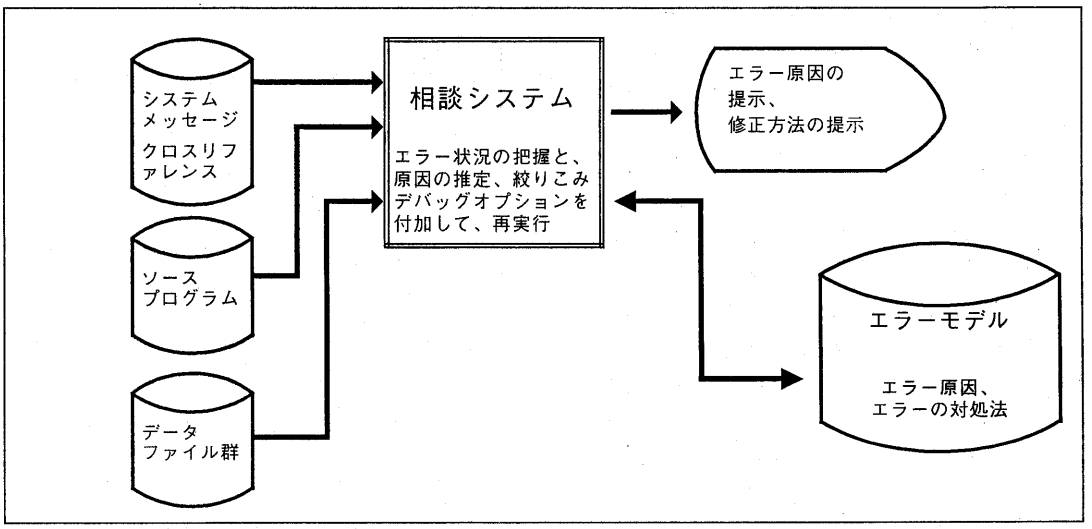

図1 Consult:Rのシステム構成

5. Consult:Rの実行例

図2にConsult:Rの起動時の画面を示す。図で太線が利用者からの入力を示している。この画面で相談の対象となるソースプログラムのデータセット名を指定できる。

```
--- プログラム相談システム(実行時エラー編) ---

相談するデータセット名を指定して下さい。

相談対象データセット:
  プロジェクト名 ===> W55010
  ライブラリ名   ===> RUNTIME
  タイプ名       ===> FORT77
  メンバ名       ===> S0C4 (省略するとメンバ名表示)

上記以外の相談対象データセット:
  データセット名 ===>

日本語エラーメッセージ ===> YES(YES又は、NO)
相談済エラーの表示消去 ===> NO(YES又は、NO:低輝度表示)

終了する時は、PF4キーを押して下さい。
```

図2　Consult:R起動時の初期画面

データセット名の入力後、図3のように、コンパイルオプションを指定する。引き続いて、コンパイラが呼び出される(図4)。図4では、コンパイルエラーが検出されなかったため、実行に移ることを利用者に知らせている。実行に先だって、ファイル割当ての状況が表示され、必要ならば、新たに割当てたり、修正できる(図5)。図5では入力ファイルを新たに割当てている。

次にプログラムが実行され、エラーがあれば図6のように表示される。この例では、1つのエラーJZL240I-Uしか生じていないが、複数生じた場合は、相談したいエラーコードを選択することによって、そのエラーコードに対する相談が行われる。

図7は図6のエラーコードの相談結果である。プログラムの3行目のエラーの生じた部分が、実際にはブリンクして、エラー箇所がわかる仕組みになっている。もし、エラーの原因とエラーコードの生じた箇所がプログラム上で離れており、1つの画面に収まらない場合は、画面が分割されて両方の箇所を表示するようになっている。エラーの相談結果は上から3行目に示される。この例では、エラー原因をより詳細に解析するために、デバッグオプションを付加して再実行することを勧めている。

```
--- コンパイル・オプション指定 ---
コマンド ===>

 設 定  コンパイルオプション及びオプションの意味
非選択 AE  :データの拡張リージョンへの割付け
非選択 ALIGNC:共通ブロック内のデータに対する
              記憶領域の割付け
非選択 ASTER:アスタ行の扱い
非選択 DOUBLE:精度拡張機能
非選択 IHALF:整数型の標準長を2バイトにする
非選択 ITR  :DOループ回転数の半減の最適化
非選択 LANGLVL(66):原始プログラムの言語仕様
非選択 QUAD :精度拡張機能
非選択/ NOXI :オブジェクト強化命令の不使用

選択/非選択を変更したい欄に"S"を入力して下さい。

その他オプション:
```

図3　コンパイルオプションの指定画面

```
**************************
*                        *
*    只今、コンパイル中です。    *
*                        *
**************************
最大のエラーレベルは 0 です。
コンパイルは正常に行われました。実行に移ります。
***
```

図4　コンパイラの起動

```
--- UNIT/データセット割付(状況表示) ---
コマンド ===>

UNIT データセット名                    --DISP--
 06   TERM                          NEW   DELETE
 05   W55043.RUN.DATA(D02)          SHR   KEEP
```

図5　ファイル割当て表示画面

```
--- 相談対象エラーメッセージ選択 ---
コマンド ===>          エラーレベル ===> ESWI

 エラー番号     エラーメッセージ
JZL240I-U  ISN= 3
 (ABEND) CODE=S00C4-U0000 PSW=FF850004A020EB0A
 SDWA=00217820

S:選択 低輝度表示は、相談済みです。  ( 1~ 1/ 1件)
```

図6　実行結果の表示

```
相談:240I-U  ISN= 3
コマンド ===> Y                確認:Y 別解答要求:N

メモリーが破壊されたと考えられます。
デバッグオプションを付けて再実行して調べますか?

 1    INTEGER  A(10)
 2    DO 100 I=1,100,5
 3      WRITE(6,*) 'ABEND TEST'
 4      A(I) = I
 5  100 CONTINUE
 6    END
```

図7　相談結果の表示

図8はSUBCHKというデバッグオプションを自動的に付加して再実行した時に生じたエラーコードの一覧である。SUBCHKは配列要素や文字部分列の引用において、その添字式や部分列の値が宣言された範囲内にあるかどうかを調べるものである。オプションを付加しない場合に比べると、JZL184I-Wというエラーが新たに2つ生じている。

図9は図8の最初のエラーコードを選択して、相談した結果である。プログラムの4行目が太文字になっており、実際にはその行がブリンクする。エラーの原因は宣言された配列の上限を越えたアクセスであると指摘している。

なお、波及エラーが生じているとシステムが判断した場合は、自動的に根本の原因を解析するようになっている。

Consult:Rで利用可能なコマンドを図10に示す。相談を中断して、他のソースプログラムを選択して相談を始めたり(ASSIGN)、ソースプログラムの内容を表示したり(BROWSE)、相談結果をもとにプログラムを修正して、再実行したり(EDIT)できる。また、ソースプログラムの内容を表示した後で、再びもとの相談に戻る(CONSULT)こともできる。

6. 終りに

相談システムConsult:Rは現在のところ2種類のエラーコードにしか対処できないが、徐々に対処できるエラーを増やしていく。現在作成中のプロトタイプについては、東京大学大型計算機センターでまとめられた資料[10]を参考にして、発生する頻度が高いと予想される20タイプの実行時エラーに対処する予定である。

相談システムの有効性については、Consult:Cの場合ではあるが、ある程度確認している。具体的には、富士通電算機専門学院において初心者の作成した60種類ほどの異なるプログラム(1プログラム単位当り平均して30行程度)に対して、Consult:Cを試用して頂いた結果を評価したところ、エラー全体の1/3については原因を用意していないために対処不能であり、1/6に対しては不適切な指摘をするものの、エラー全体の半分には正

```
---相談対象エラーメッセージ選択---
コマンド===>                    エラーレベル ===> ESWI

 エラー番号     エラーメッセージ
JZL184I-W  ISN= 4
  00004(デバッグ)配列要素又は文字部分列の引用で,
  添字式又は部分列式の値が宣言された範囲を超えてい
  ます
JZL184I-W  ISN= 4
  00004(デバッグ)配列要素又は文字部分列の引用で,
  添字式又は部分列式の値が宣言された範囲を超えてい
  ます
JZL240I-U  ISN= 3
 (ABEND) CODE=S00C4-U0000 PSW=FF850004A020EB0A
 SDWA=00217820

S:選択 低輝度表示は、相談済みです。  ( 1~ 3/ 3件)
```

図8 デバッグオプションを付加した場合の実行結果

```
相談:184I-W  ISN= 4
コマンド===> Y                  確認:Y 別解答要求:N

配列Aの引用が宣言された範囲を超えてなされました。
宣言された範囲はA(1)~A(10)で,引用されたのは
A(11)です。アルゴリズムを見直して下さい。

 1   INTEGER A(10)
 2   DO 100 I=1,100,5
 3     WRITE(6,*) 'ABEND TEST'
 4     A(I) = I
 5 100 CONTINUE
 6   END
```

図9 図8のエラーJZL184I-Wに対する相談結果

```
---プログラム相談システム(実行時エラー編)---

オプション ====>

------------------------------------------ ユーザID - W55043
                                           時刻  - 10:18

 0  ASSIGN  - 相談対象データセットの変更
 1  BROWSE  - 相談対象データセットの内容の表示
 2  EDIT      - 相談対象データセットの編集
                (終了後、再コンパイル)
 3  CONSULT   - 実行時エラーの相談
 H  HELP    - 相談システムの操作説明
 X  EXIT    - 相談システムを終了する

-------------------------------------------------------------------------
相談対象のデータセットは、
 'W55010.RUNTIME.FORT77(S0C4)'です。

終了する時は、PF4キーを押して下さい。
```

図10 Consult:Rのコマンド選択画面

確に答えられることが判明した。従って、Consultで対処できるエラーコードを徐々に充実させることによって、相談員の負担の軽減という目標を達成できると考えている。なお、試用の結果を参考にして、Consult:Cの実用版を作成中である。

Consult:Rについても、初心者のプログラムを用いてプロトタイプを試用頂く予定であり、その評価と強化が今後の当面の課題である。

最後になるが、システムからのエラーコードに対して、エラーの原因を用意し、種々の情報から原因を特定する手法は、Fortran処理系以外の相談にも応用できると思われる。他の言語でのエラーの対処、並びにコンパイルや実行時以外の過程でのエラー相談にも対処できるように強化していき、最終的には、かなり高度な相談の支援を可能としたい。

謝　辞

本研究は京都大学と富士通株式会社の共同研究として進めており、共同研究関係者各位に感謝する。特に、小林正和氏、岡本匡人氏、斉藤　哲氏、ならびに久楽　匡氏に感謝する。

参考文献

[1]田中　一:プログラム相談の機械化報告書
文部省科学研究費補助金による特定研究「広域大量情報の高次処理」(1976)

[2]磯本征雄他:計算機援助型相談システム
"CONSULTANT"の構成とその設計思想
情報処理学会論文誌**24**巻5号(1983)

[3]磯本征雄他:Fortranデバッグ助言システム
Advisorにおける知識表現と画面表示
電子通信学会技術研究報告**ET84** – 5(1984)

[4]松本範久他:初心者を対象とするFortran教育支援システムCalen
電子通信学会技術研究報告**ET85** – 10(1986)

[5]杉岡一郎:エキスパートシステム構築ツールを用いたFORTRANエラー相談システム
電子情報通信学会技術研究報告**ET87** – 2(1987)

[6]野村研仁他:実行時エラーの原因診断エキスパートシステムの試作
情報処理学会研究報告**SE54** – 3(1987)

[7]大西　淳他:CONSULT/C:コンパイル時のエラーに関するプログラム相談手法/システム
情報処理学会研究報告ソフトウェア工学
64-24(1989)

[8]佐藤真木彦他:実行時のプログラム相談システム「Consult:R」
情報処理学会第**39**回全国大会発表**6N**-2(1989)

[9]京都大学大型計算機センター:「プロ相すくらんぶる」(その1-5)
広報第**21**巻5号から第**22**巻4号(1988、1989)

[10]東京大学大型計算機センター:FORTRANプログラムデバッグの手引第**3**版(1988)