情 報 シ ス テ ム 10－2
（1986. 5. 20）

プログラミング不要の技術計算用ソフト（ＥＱＵＡＴＲＡＮ－Ｍ）の設計思想と活用事例

小口　梧郎
三井東圧化学（株）　システム部

ＥＱＵＡＴＲＡＮ－Ｍは汎用の方程式解法ソフトウェアであり、もともと大型計算機用に開発され、化学工業の分野で長年実用されてきたもを、パーソナルコンピュータ用に改良の上移植したものである。ＥＱＵＡＴＲＡＮ－Ｍのユーザーは、線形・非線形を問わず解こうとする連立方程式を通常の数学的記法に近い形で入力することにより、何らプログラミングを行うことなく、その数値解を得ることができる。本報告では、ＥＱＵＡＴＲＡＮ－Ｍの機能と設計思想とをいくつかの実用例に基づいて解説・紹介している。

"Functions and Design Concepts of EQUATRAN-M ——
A General Purpose Equation Solver" (in Japanese)

by Goro OGUCHI (Systems and Computing Department,
Mitsui Toatsu Chemicals Inc., 3-2-5 Kasumigaseki,
Chiyodaku, Tokyo, 100, Japan)

EQUATRAN-M is a computor software for personal computors, to solve a set of linear and non-linear equations. This software was developed originaly on the main-frame computor, and has been utilized over a decade in chemical industries. The user of EQUATRAN-M can get a numerical solution of the equations only by inputting them in normal mathematical form, without any programming efforts. This paper describes the functions and the design consepts of EQUATRAN-M with its applied examples.

１．はじめに（開発の背景と経緯）

化学工業におけるプロセスの物質・熱収支計算は古くから計算機利用の主要分野の一つであり、プラントの設計にあたっては、プロセス全体にわたる詳細な物質・熱収支計算が不可欠となっている。そしてその道具として、いわゆるフローシートシミュレータと呼ばれるプログラムが数多く開発され実用されている。

一方、全く新しいプロセスの研究・開発の段階では、プロセスのフィージビリティや経済性を評価するための予備的な物質・熱収支計算が必要となる。この段階の計算では、各機器の詳細なモデル化は不要であるが、一方、計算結果が早急に得られることやまた数式モデルやプロセスの構成の変更、計算条件の変更に対して高いフレキシビリティが要求される。また時として、いくつかのパラメータについて最適化された物質収支計算が必要となる。このような目的に見合った道具を、一般的な連立方程式解法プログラムとして大型汎用計算機上で実現したものが「ＥＱＵＡＴＲＡＮ」（EQUAtion TRANslator）であった（1975年）(1)。

ＥＱＵＡＴＲＡＮは開発後、１０年以上にわたって主としてプロセスの物質収支計算に活用されているが、方程式をそのまま書けば解が得られるという簡便さから、他の一般的な技術計算にも広く利用された。８ビットマイクロコンピュータの登場時に、ＥＱＵＡＴＲＡＮのこのような性格に着目して、これを載せることによりいわば超電卓として使うことを検討したが、メモリ容量や計算能力の制限から実現することができなかった。その後１６ビット機の出現によってメモリ容量、計算能力とも充分実用可能と判断し、プログラミング不要の技術計算用ソフト「ＥＱＵＡＴＲＡＮ－Ｍ」（EQUAtion TRANslator for Micro-computor）として完成した（1985年）。

以下に、ＥＱＵＡＴＲＡＮ－Ｍを中心にその機能と設計法を実用例を含めて報告する。

２．機能と設計

化学プロセスの物質・熱収支計算に限らず技術計算の分野において、方程式を解く（その数値解をもとめる）ことに帰着される問題は少なくない。ＥＱＵＡＴＲＡＮ－Ｍはそのような問題を方程式をそのまま入力するだけで自動的に解くことを目的として作成したソフトウェアである。方程式をそのまま入力するとは言っても、方程式の中には単純な式では表現できない関数関係なども含まれており、したがってＥＱＵＡＴＲＡＮ－Ｍはこれらを記述するための１つの言語を持っている。「ＥＱＵＡＴＲＡＮ－Ｍ」はプログラムの名前であると同時にこの言語の名前でもある。

（１）方程式の記述法（言語の設計）

以下にＥＱＵＡＴＲＡＮ－Ｍの方程式記述機能について述べるが、後出の使用例を同時に参照して頂きたい。

解くべき方程式をＥＱＵＡＴＲＡＮ－Ｍで記述したものを「ソーステキスト」と呼んでいる。言語の設計にあたっては、記述性の良さとともに、ドキュメント性の良いソーステキストが得られることを重要なポイントと考えた。

【一般の方程式】　代数方程式や初等関数を含む方程式は、通常の数学的記法と同様に記述する。演算子はＦＯＲＴＲＡＮやＢＡＳＩＣに準じたものを採用した。ただし等号は本来の意味（左辺と右辺が等しい）で用いる。式の変形や式と式の順序の並べ換えは不要である。方程式中の変数は全て実数値をとるものとし、複素数は扱わない。また、微分方程式は差分等により離散化する必要がある。

【配列変数とその演算】　２次元までの配列変数（ベクトルとマトリックスに相当）を含む方程式をそのまま記述できるようにしている。化学プロセスの物質収支計算に限らず、実用規模の問題は多くの場合配列変数を含んでおり、配列変数を含む方程式が一括して書けないと非常に記述性の悪いものになってしまう。配列を含む方程式が簡潔に記述でき、また、ＦＯＲＴＲＡＮのＤＯループのような手続き的な記述を用いなくても済むように、配列変数の演算規則と配列の添字の記述方法に次のような工夫を行った（表－１参照）。

表－１　配列変数の演算規則と部分配列の表記法

項　目	EQUATRAN-M での表記例	内　容
変数の定義	VAR　a(5), b(2,5), c, d(5)	次元と要素数を定義する
配列の演算		
関数の適用	EXP(a)	$(e^{a_1}\ e^{a_2}\ e^{a_3}\ e^{a_4}\ e^{a_5})$
２項演算	a＊d	$(a_1d_1\ a_2d_2\ a_3d_3\ a_4d_4\ a_5d_5)$
次元の拡張	a＋c	$(a_1+c\ a_2+c\ a_3+c\ a_4+c\ a_5+c)$
	b／a	$\begin{pmatrix} b_{11}/a_1 & b_{12}/a_2 & b_{13}/a_3 & b_{14}/a_4 & b_{15}/a_5 \\ b_{21}/a_1 & b_{22}/a_2 & b_{23}/a_3 & b_{24}/a_4 & b_{25}/a_5 \end{pmatrix}$
部分配列		
一要素	b(1.4)	b_{14}
添字の結合	a(2.4)	$(a_2\ a_4)$
	a(2:4)	$(a_2\ a_3\ a_4)$
添字の省略	a	$(a_1\ a_2\ a_3\ a_4\ a_5)$
	b(2)	$(b_{21}\ b_{22}\ b_{23}\ b_{24}\ b_{25})$

①単項演算子および引数が１つの関数は配列変数の各要素にそれぞれ適用される。

②２項演算子（等号も含む）および引数が２つ以上の関数については
・同一次元の配列変数間の演算は対応する要素間の演算とする。
・次元の異なる変数間の演算は、次元の低い方の変数を並べることによって次元を拡張し、次元を揃えてから演算する。

③配列中の１要素は添字をつけて表す。添字は結合記号（　.　および　:）を用いて添字の集合とすることができ、これによって配列変数の部分配列を表すことができる。

④添字を省略することによって添字全体の集合を表す。

以上の規則の他に配列の要素を加算する関数　ＳＵＭが用意されており、これを使うとベクトルの内積あるいはマトリックスと縦ベクトルの積は、

ＳＵＭ（Ａ＊Ｂ）

の形で表現することができる。

【数表】　線図や表によって変数間の関係が与えられることは多い。ＥＱＵＡＴＲＡＮ－Ｍではこれらを数表として直接定義して、一種の関数として用いることができる。数表は引数の配列と数表値の配列との関係として定義し、２次元の表まで使うことができる（使用例３　参照）。

【条件付の式】　例えば

$$C_D = \begin{cases} 24/Re & Re \leq 2 \text{ のとき} \\ 10/\sqrt{Re} & 2 < Re \leq 500 \text{ のとき} \\ 0.44 & 500 < Re \text{ のとき} \end{cases}$$

のように、式の形が条件によって異なる式を条件付の式と呼んでいる。ＥＱＵＡＴＲＡＮ－Ｍでは条件付の式をそのままの形で書けるようにすることによって、ＩＦ……ＴＨＥＮ……のような構文を用いずに複雑な関係式を表すことができる（使用例３）。

【論理演算】　論理演算は条件付の式の条件項や、後出の最適化計算の制約条件の記述に必要となるが、一般の方程式の中の演算としても用いることができる。論理演算の結果は真が１、偽が０として扱われる。

【最適化問題】　1つあるいは複数の未知変数（独立変数）の値を別の1つの変数（評価変数）の値を最大あるいは最小になるように決定する最適化問題を記述できるようにしている。最適化には論理演算式を使って不等号制約条件を附することができる。

【マクロ】　同一構成の方程式群が繰り返し使用される場合にはこれらを一まとめにしてシンボル化して用いることが望ましい。ＥＱＵＡＴＲＡＮ－Ｍでは、方程式の原形をテキストの形でまとめたものを「マクロ」として登録しておき、使用時にこれを取り込んで（「マクロコール」して）同時に原形の一部を修正して用いることによってこれを実現している。原形の修正（主として変数名の変更）はマクロコール時のパラメータとして指定する（使用例２）。

マクロを用いる方法の他に、例えばＦＯＲＴＲＡＮのサブプログラムのように、独立した計算ブロックを構成してこれをリンクする方法も考えられるが、マクロによる方が計算のフレキシビリティが大きくなること、解法についての自由度が増す（たとえば異なるマクロにまたがる連立方程式を解く場合など）ことからこれを採用している。

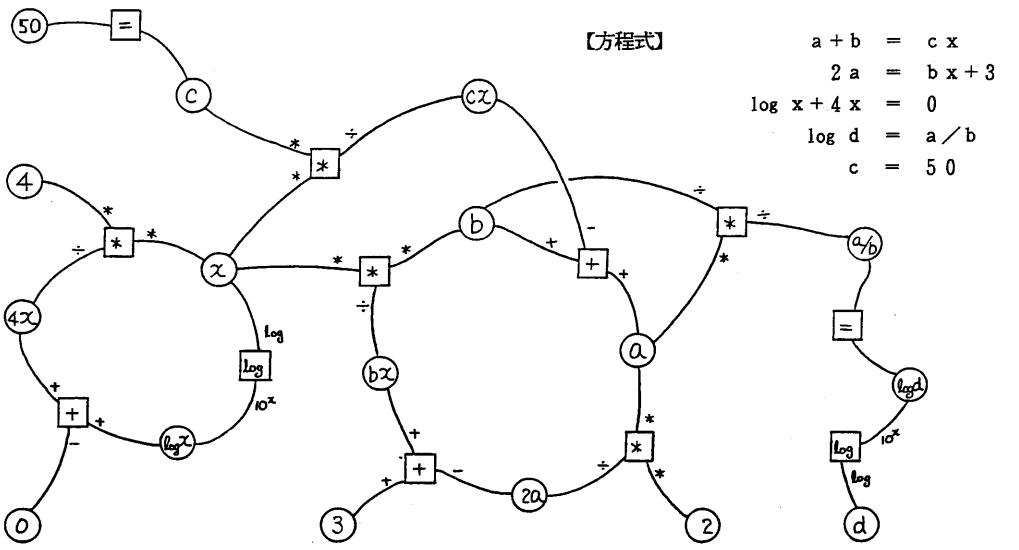
（２）方程式の解法

ソーステキストに記述された方程式を解読し計算の手順を決定し、これを逐次計算の実行が可能な中間コードの形に直す作業をＥＱＵＡＴＲＡＮのコンパイルと呼び、この処理プログラムをＥＱＵＡＴＲＡＮコンパイラという。コンパイラは以下のような処理を行う（詳細は文献(2) 参照）。

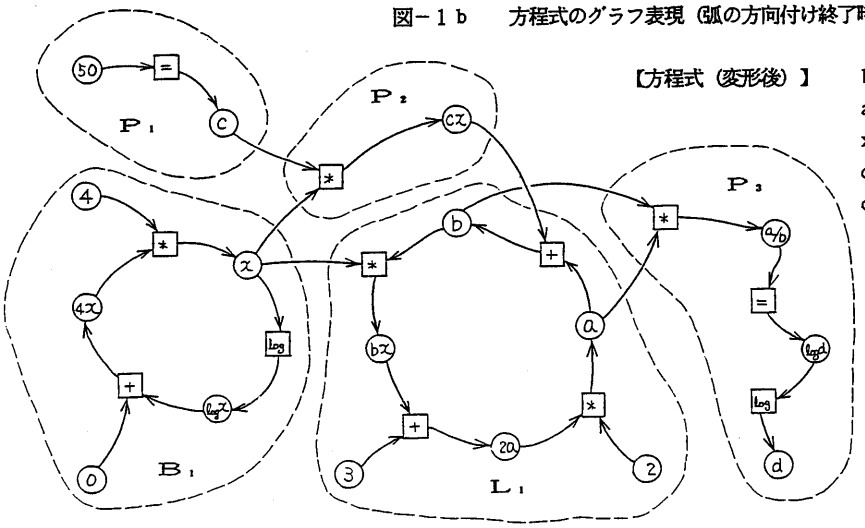
【構文解釈】　ソーステキストの構文解釈を行い文法上のエラーを検査するとともに、全ての方程式を変数間の関係を表す一つのグラフとして表現する。ＥＱＵＡＴＲＡＮ－Ｍではこのグラフとして、変数を表す頂点の集合と、式（演算）を表す頂点の集合とからなる２部グラフを用いている。図－１ａにグラフ表現の例を簡単な方程式について示した。

【式の変形と可解性のチェック】　全ての変数が１回かつ１回だけ計算されるような形に方程式を変形する。このような変形が可能であればこの方程式は可解である（正確には「構造的に可解」(5)）と判断される。この方程式の変形は上述の２部グラフにおいて、計算の方向に従って弧を方向付けする操作として比較的容易に実現することができる。図－１ｂは図－１ａに示した方程式に対してこの変形の操作を行った結果を示している。

図－１ａ　方程式のグラフ表現（グラフ作成時）



図－１ｂ　方程式のグラフ表現（弧の方向付け終了時）



【線形連立方程式の判別】　方程式の中には逐次的に計算可能な部分と連立して解かねばならない部分が含まれているが、後者についてさらに線形な連立方程式と非線形な連立方程式とに分けることができる。線形連立方程式を判別してこれを直接法で解くことにより不要な収束計算を避けることができる。図－１ｂにおいて、L_1 によって示された部分は線形、B_1 によって示された部分は非線形の連立方程式になっている。

【非線形連立方程式の処理】　非線形連立方程式に対しては繰り返し収束計算による解法のロジック（ニュートンラフソン法を採用している）を組み込むことが必要である。この場合問題になるのは、どのような収束計算をさせるか、すなわち値を仮定する変数と収束の判定を行う式との組み合わせをどう選ぶかである。この選択の良し悪しは収束の確かさや能率に大きな影響を持つ。一般にはなるべく少数で済むような組み合わせを選ぶのが良いが、一方、ある程度冗長な組み合わせを選ぶことによって収束性が改善されることもよく経験するところである。また、値を仮定する変数の初期推定値の良し悪しは重要でありしたがって良い初期値を与えうる変数を選ばなければならない。ＥＱＵＡＴＲＡＮ－Ｍでは次のようにしてこの組み合わせを選ぶことにした。

①ユーザーがソーステキスト中に収束計算の方法を指定する。

②ユーザーの指定がないか、あるいは指定が不足している場合にはコンパイラが自動的に追加の組み合わせを選択する。

これによってユーザーはシステムと対話をしながら確実な収束計算の手順を作り出すことができる。コンパイラはなるべく少数の組み合わせ（最少ではない）で済むような選択を行っている。

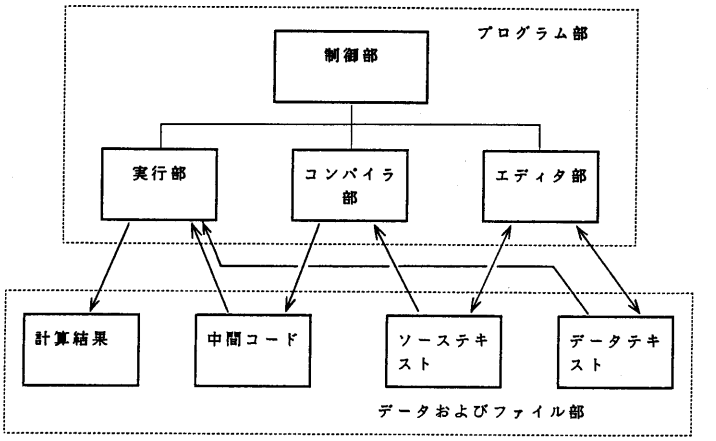
【中間コードの生成】　以上の処理の結果決定された計算手順は中間コードの形で生成され、後にインタープリーティブに実行される。

（３）プログラムの構成

ＥＱＵＡＴＲＡＮとＥＱＵＡＴＲＡＮ－Ｍとではプログラムの構成方法が異なる。前者は大型計算機でのバッチ使用を前提としていたのに対し、後者はパーソナルコンピュータでの対話形使用を前提として設計したためである。両者の最も大きな相違点はＥＱＵＡＴＲＡＮではソーステキストを解析して生成される中間コードが一部ＦＯＲＴＲＡＮのソースコードになっており、これをＦＯＲＴＲＡＮのコンパイラでコンパイルしてから他の数値計算プログラムとリンケージして実行プログラムが生成されるのに対し、ＥＱＵＡＴＲ

ＡＮ－Ｍでは実行プログラムが最初から完成した形で用意されており、中間コードが直ちに実行されることである。前者の方法の利点は、実行プログラムにユーザーが作成したプログラムを直接つなぎこむことができることで、実際、ＥＱＵＡＴＲＡＮではＦＯＲＴＲＡＮのサブルーチンを方程式の一部として組み込むことができる。一方この方法では、ソーステキストを修正するたびにＦＯＲＴＲＡＮのコンパイルとリンケージが必要となるため、特にパーソナルコンピュータでの対話的利用では操作環境が著しく悪化してしまうという欠点がある。

図－２　　ＥＱＵＡＴＲＡＮ－Ｍの構成



ＥＱＵＡＴＲＡＮ－Ｍでは、エディタ、コンパイラ、実行プログラムを一体とする構成によって、大変良い操作環境を実現することができた。図－２に構成を示した。

３．使用例

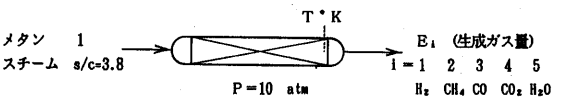
次に、ＥＱＵＡＴＲＡＮ－Ｍの使用例を３つ紹介する。なお、文献（3）(4) にはさらに多くの使用例が紹介されている。

（１）スチームリフォーマの検討

天然ガス（メタン）とスチームとを触媒を用いて反応させ、水素と一酸化炭素および二酸化炭素を得るスチームリフォーマの計算を最初の例として取り上げる。スチームリフォーマは古くからメタノールやアンモニア製造に使用されているが、近年では燃料電池発電のための燃料製造用としてしばしば検討の対象となっている。図－３に問題の概要と計算に使用する方程式を示した。説明の都合上実際のモデルより幾分か簡略化してある。反応生成ガスの組成はいずれも反応器出口における化学平衡によって決定されるが、メタンの分解反応（反応１）は平衡にまで至らないため、アプローチ温度（反応温度と平衡温度との差）を用いた擬似平衡により計算している。計算は次の３つの条件について行う必要があった。

①設計モード　……　メタンの分解率とアプローチ温度を指定して必要な反応温度を算出する。

図－３　　使用例１　スチームリフォーマーの検討



反応量　平衡定数

反応　1　$CH_4 + H_2O \rightleftharpoons CO + 3H_2$　r_1　K_1

2　$CO + H_2O \rightleftharpoons CO_2 + H_2$　r_2　K_2

物質収支式　(kgmol/h)

H_2 : $E_1 = 3r_1 + r_2$　　CO_2 : $E_4 = r_2$

CH_4 : $E_2 = 1 - r_1$　　H_2O : $E_5 = s/c - r_1 - r_2$

CO : $E_3 = r_1 - r_2$

生成ガス組成

$x_i = E_i / \Sigma E_i$　　i=1,2・・5

$xd_i = E_i / \Sigma E_i$　　i=1,2・・4　(ドライベース)

平衡定数

$K_1 = (x_3 x_1^3)/(x_2 x_5) \cdot P^2 = \exp(-27464/(T-Tap)+3071)$

$K_2 = (x_1 x_4)/(x_2 x_5) = \exp(4084/T-3.77)$

②運転モード …… 反応温度とアプローチ温度を与えて生成ガス組成を算出する。

③解析モード …… 出口ガス組成の分析値（ドライベースのメタン濃度とCO/CO_2比）を与えて反応温度とアプローチ温度を算出する。

図－４にＥＱＵＡＴＲＡＮ－Ｍで書いたソーステキストのリストを示す。ソーステキストは完全なフリーフォーマットで、／＊　＊／で囲まれた部分はコメントである。３行目のＶＡＲ文は変数を定義する文で、配列変数の大きさはここで指定する。２６行目のＩＮＰＵＴ　文は、この文で指定した変数の値を計算実行時に読み込むことを指定している。２７行目の　ＲＥＳＥＴ　文は収束計算の方法を指定する文であり、値を仮定する変数（r2）、その初期値、変域および収束をチェックする式の名前（e2）などが指定できる。３８行目の　ＯＵＴＰＵＴ　文は計算結果を表示する変数を指定するものである。３０行目以降には、運転モードと解析モードの場合のＩＮＰＵＴ文とＲＥＳＥＴ文の指定がある（現在はコメントになっている）。この部分を取り替えるだけで（方程式の記述には全く手を入れずに）各モードの計算を行うことができる。図－５に実行時の入力データの例と計算結果を示した。

図－４　使用例１のソーステキスト

```
 1:  /* スチームリフォーマの検討 */
 2:
 3:  VAR  E(5),X(5),Xd(4)
 4:
 5:       E(1)  = 3*r1 + r2                    /* H2   */
 6:       E(2)  = 1 - r1                       /* CH4  */
 7:       E(3)  = r1 - r2                      /* CO   */
 8:       E(4)  = r2                           /* CO2  */
 9:       E(5)  = sbyc - r1 - r2               /* H2O  */
10:
11:       X = E/SUM(E)                        /* 反応ガス組成 */
12:       Xd = E(1:4)/SUM(E(1:4))             /* ドライベース */
13:  ec:  Rc = Xd(3)/Xd(4)                          /* CO/CO2 */
14:
15:  e1:  K1 = (X(3)*X(1)^3)/(X(2)*X(5)) * P^2
16:  e2:  K2 = (X(1)*X(4))/(X(3)*X(5))
17:
18:       K1 = EXP(-27464/(T-Tap)+30.71)  /* 平衡定数 (1) */
19:       K2 = EXP(4084/T-3.77)           /*          (2) */
20:       TC = T-273
21:
22:       P = 10                                /* 系の圧力 [atm] */
23:       sbyc = 3.8                    /* スチーム・カーボン比 */
24:
25:                                             /* 設計モード */
26:     INPUT   r1,Tap
27:       RESET r2#0.5[0,1] BY e2
28:
29:                                             /* 運転モード */
30:  /* INPUT   TC,Tap
31:       RESET r1#0.8[0,1] BY e1
32:       RESET r2#0.5[0,1] BY e2  */
33:
34:                                             /* 解析モード */
35:  /* INPUT   Xd(2),Rc
36:       RESET r2#0.5[0,1] BY ec  */
37:
38:  OUTPUT r1,r2,Xd,TC,Tap
```

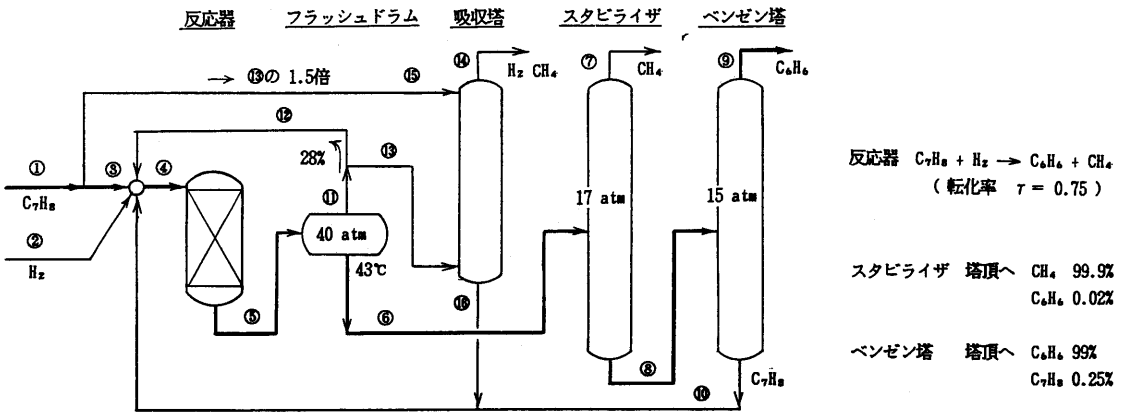
図－５　使用例１の入力例と計算結果

```
[  入力値(=) または 初期値(#)  ]
r1                = 0.92
Tap               = 15
r2                # 0.5

[  計算結果  ]
r1 =   0.92
r2 =   0.39732
Xd =
  1) 0.75946        2) 0.019243        3) 0.125725        4) 0.095571
TC =   820.107061
Tap =   15
```

（２）トルエン脱アルキルプロセスの物質収支

図－６　使用例２　トルエン脱アルキルプロセスの物質収支



トルエンを原料とし、脱アルキル反応によってベンゼンを製造するプロセスであり、図－６にそのフロー図と主な計算条件を示した。反応器では１パスの転化率が与えられている。フラッシュドラムでは気液平衡計算によって分離の計算を行う。２つの蒸留塔（ベンゼン塔とスタビライザ）ではキー成分の分離率を与え、他の成分の分離はＦｅｎｓｋｅの式を用いた簡易蒸留計算によって決定される。

図－７にソースリスト、図－８に計算結果を示した。ソースリストの２行目のＧＬＯＢＡＬ文はグローバルパラメータと呼ばれるものの定義で、これによってソーステキスト中のＮＣというシンボルが全て４（すなわち成分数）に置き換えられる。３行目のＶＡＲ文で全ストリームの成分流量を表す２次元の配列変数Ｆが定義されている。５行目～３５行目はマクロを定義している部分であり、ここでフラッシュ計算のマクロ（Ｆｌａｓｈｎ）、蒸留計算のマクロ（Ｆｅｎｓｋｅ）および気液平衡定数を計算するマクロ（Ｋｃａｌｃ）が定義されている。Ｋｃａｌｃは他のマクロ中（１２行目、２４行目）の　ＣＡＬＬ　文によりコールされている。このようにマクロはネスティングもできる。実際のプロセスのモデルは３７行目～６３行目までであり、マクロの利用によって大変コンパクトに記述することができた。

（３）コーナタップオリフィスの設計計算

つぎに、日本工業規格（Ｚ８７６２）によるコーナタップオリフィスの設計計算の例を取り

図－７　使用例２のソーステキスト

```
 1: /* トルエン脱アルキルプロセス */
 2: GLOBAL NC=4
 3: VAR  F(16,NC)     /* 1:水素  2:ﾒﾀﾝ  3:ﾍﾞﾝｾﾞﾝ  4:ﾄﾙｴﾝ  */
 4:
 5: MACRO Flashn                  /* フラッシュ計算のマクロ */
 6:    LOCAL vbyfi = 0.5
 7:    VAR K(NC),BUNBO(NC)
 8:      feed = vap + liq
 9:      vap = liq*K*(vbyf/(1-vbyf))
10: eq:  SUM(feed/BUNBO) = SUM(feed*K/BUNBO)
11:           BUNBO = 1-vbyf+K*vbyf
12:      CALL Kcalc(T=T, P=P, K=K)
13:    RESET vbyf#vbyfi[0,1] BY eq
14: END Flashn
15:
16: MACRO Fenske                  /* 簡易蒸留計算のマクロ */
17:    VAR alp(NC),eta(NC),K(NC)
18:
19:       feed = top + btm
20:       top = feed*eta
21:       Nm+1 = LOGE(etal/(1-etal)*(1-etah)/etah)/LOGE(alp(lkey))
22:       Nm+1 = LOGE(eta(nkey)/(1-eta(nkey))*(1-etah)/etah) ..
23:            /LOGE(alp(nkey))
24:       CALL Kcalc(T=T, P=P, K=K)
25:       alp = K/K(hkey)
26:       eta(hkey) = etah ;  eta(lkey) = etal
27: END Fenske
28:
29: MACRO Kcalc                   /* K値計算のマクロ */
30:    VAR AK(4),BK(4),CK(4)
31:       AK=(10.361, 8.5937, 9.5007, 9.3802 )
32:       BK=(-426.41,-898.01,-2789.0,-3059.1)
33:       CK=(251.7,  266.0,  220.8,  219.2  )
34:    LOGE(K*P) = AK+BK/(T+CK)
35: END Kcalc
36:
37: /* フィード */
38:    F(1) = (0,0,0,276.2)
39:    F(2) = (924,50,0,0)
40: /* 反応器 */
41:    F(5,1.4) = F(4,1.4)-R
42:    F(5,2.3) = F(4,2.3)+R
43:       R = F(4,4)*gamma
44: /* フラッシュドラム */
45: F101: CALL Flashn(feed=F(5), vap=F(11), liq=F(6), T=43, P=40)
46: /* 吸収塔 */
47:    F(13) + F(15) = F(14) + F(16)
48:    F(14,1.2) = F(13,1.2)
49:    F(14,3.4) = 0
50: /* スタビライザ */
51: D102: CALL Fenske(feed=F(6), top=F(7), btm=F(8), P=17, T=230 ..
52:           ,lkey=2, hkey=3, nkey=1.4, etal=0.999, etah=0.0002)
53: /* ベンゼン塔 */
54: D103: CALL Fenske(feed=F(8), top=F(9), btm=F(10), P=15, T=240 ..
55:           ,lkey=3, hkey=4, nkey=1.2, etal=0.99, etah=0.0025)
56: /* 分岐と混合 */
57:    F(1) = F(3) + F(15);   F(15) = r15*F(1)
58:    F(4) = F(2) + F(3) + F(10) + F(12) + F(16)
59:    F(11) = F(12) + F(13);  F(12) = F(11)*r12
60: /* スペック */
61:    r12 = 0.28
62:    F(15,4)/F(13,3) = 1.5
63:    gamma = 0.75
64: OUTPUT F
```

図－８　使用例２の計算結果

```
[  計算結果  ]
F =
      (  1)        (  2)        (  3)        (  4)
 1) 0            0            0            276.2
 2) 924          50           0            0
 3) 0            0            0            267.328612
 4) 1175.465335  170.194317   11.001275    367.961801
 5) 899.493984   446.165667   286.972625   91.99045
 6) 1.403504     16.90025     278.758377   91.193242
 7) 1.403504     16.88335     0.055752     0.000668
 8) 8.888455E-009 0.0169      278.702625   91.192573
 9) 8.888455E-009 0.0169      275.915599   0.227981
10) 0            0            2.787026     90.964592
11) 898.090481   429.265417   8.214248     0.797208
12) 251.465335   120.194317   2.29999      0.223218
13) 646.625146   309.0711     5.914259     0.57399
14) 646.625146   309.0711     0            0
15) 0            0            0            8.871388
16) 0            0            5.914259     9.445378
```

上げる。基本的な関係式は図－９に示すとおりであるが、式中のα_0はレイノルズ数R_dと絞り直径比βの関数として、r_0はレイノルズ数R_dと管内壁の相対粗さｋ／Ｄの関数として別に２次元の表で与えられている。図－１０に示したソーステキストにおいて、３行目～３３行目までがこの数表の定義になっている。５行目と２３行目の　ＴＡＢＬＥ文がそれぞれ数表　ａｌｐｈ０とｒ０を定数配列間の関係として定義づけている。(4)式のr_{Rd}はレイノルズ数R_dの値によって式の形が分かれるが、条件付の式としてそのまま書くことができる（５４～５６行目）。図－１１に計算例を示した。

オリフィスの設計は通常、管と流体の条件と、設計差圧を与えて、使用するオリフィスの孔径を求めるように行うが、既に設置されているオリフィスを測定範囲を変更して使用する場合には（レンジ変更という）逆にオリフィスの孔径を与えて設計差圧を計算することが必要になる。この場合にもリストの６５～６７行目を６９～７１行目（現在コメントになっている）に変更するだけで簡単に対応できる。

図－９　　使用例３

コーナータップオリフィスの設計

計算式

$$Q=\alpha\varepsilon\frac{\pi}{4}d^2\sqrt{\frac{2\,dP}{\gamma_1}} \tag{1}$$

$$W=\gamma_1 Q \tag{2}$$

$$\alpha=\alpha_0\,\gamma_{Rd} \tag{3}$$

$$\gamma_{Rd}=\begin{cases}(r_0-1)\left(\dfrac{\log_{10}R_d}{6}\right)^2+1 & (R_d<10^6)\\ r_0 & (R_d\geq 10^6)\end{cases} \tag{4}$$

$$\varepsilon=1-(0.3707+0.3184\beta^4)\times\left(1-\left(\frac{P_2}{P_1}\right)^{1/\kappa}\right)^{0.935} \tag{5}$$

$$R_d=\frac{uD\gamma_1}{\mu} \tag{6}$$

$$\beta=\frac{d}{D}\qquad(0.22<\beta<0.8) \tag{7}$$

図－１１　　使用例３の入力例と計算結果

```
[   入力値(=) または 初期値(#)   ]
W                = 30
P1               = 4
delP             = 1000
gam1             = 7.44
mu               = 0.014
kappa            = 1.4
D                = 19.4
k                = 0.08
b                = # 0.5

[   計算結果   ]
d =    6.789521
b =    0.349975
alph =    0.606919
eps =    0.992955
Rd =    39065.923152
```

図－１０　　使用例３のソーステキスト

```
 1:  /*       コーナタップ　オリフィス　設計計算          */
 2:  /*                  日本工業規格     Z8762-1969    */
 3:  /*   流量係数　テーブル */
 4:    VAR  x1(8),x2(11),x(11,8)
 5:    TABLE  x  = alph0( x2, x1)     /* x1=Rd , x2=b^4  */
 6:      x1 = (5E3, 1E4, 2E4, 3E4, 5E4, 1E5, 1E6, 1E7)
 7:      x2 = (0.0025, 0.005, 0.01, 0.04, 0.10, 0.15, ..
 8:            0.20,   0.25,  0.30, 0.35, 0.40)
 9:      x  = ..
10:     (0.603,0.600,0.599,0.599,0.598,0.598,0.598,0.597)..
11:     (0.606,0.602,0.602,0.602,0.601,0.601,0.600,0.599)..
12:     (0.611,0.606,0.605,0.604,0.603,0.603,0.602,0.602)..
13:     (0.634,0.626,0.621,0.618,0.617,0.616,0.615,0.614)..
14:     (0.    ,0.658,0.650,0.645,0.642,0.640,0.637,0.636)..
15:     (0.    ,0.684,0.673,0.668,0.663,0.659,0.656,0.655)..
16:     (0.    ,0.710,0.696,0.689,0.688,0.679,0.675,0.674)..
17:     (0.    ,0.737,0.719,0.712,0.705,0.699,0.695,0.693)..
18:     (0.    ,0.763,0.743,0.735,0.727,0.720,0.715,0.713)..
19:     (0.    ,0.792,0.770,0.760,0.750,0.743,0.736,0.733)..
20:     (0.    ,0.0   ,0.797,0.786,0.777,0.767,0.757,0.756)
21:  /*  管内壁あらさ補正用　r0 テーブル          */
22:    VAR  z1(8),z2(7),z(7,8)
23:    TABLE  z = r0( z2, z1)      /*  z1=D/k , z2=b^2  */
24:      z1 = (400, 800, 1200,1600, 2000, 2400, 2800, 3200)
25:      z2 = (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.64)
26:      z  = ..
27:     (1.002,1.000,1.000,1.000,1.000,1.000,1.000,1.000)..
28:     (1.003,1.002,1.001,1.000,1.000,1.000,1.000,1.000)..
29:     (1.006,1.004,1.002,1.001,1.000,1.000,1.000,1.000)..
30:     (1.009,1.006,1.004,1.002,1.001,1.000,1.000,1.000)..
31:     (1.014,1.009,1.006,1.004,1.002,1.001,1.000,1.000)..
32:     (1.020,1.013,1.009,1.006,1.003,1.002,1.000,1.000)..
33:     (1.024,1.016,1.011,1.007,1.004,1.002,1.002,1.000)
34:
35:    /*  Q          体積流量        [m3/h]      */
36:    /*  W          重量流量        [kg/h]      */
37:    /*  P1         流体圧力        [kg/cm2G]   */
38:    /*  delP       差圧            [mmH2O]     */
39:    /*  gam1       流体密度        [kg/m3]     */
40:    /*  mu         流体粘度        [cp]        */
41:    /*  kappa      比熱比          [-]         */
42:    /*  D          上流側管径      [mm]        */
43:    /*  k          管のあらさ      [mm]        */
44:    /*  d          絞り孔径        [mm]        */
45:    /*  b          絞り直径比      [-]         */
46:    /*  alph       流量係数        [-]         */
47:    /*  eps        膨張補正係数    [-]         */
48:    /*  Rd         ﾚｲﾉﾙｽﾞ数       [-]         */
49:    /*  u          管内流速        [m/s]       */
50:
51:  eq:Q    = 0.01252*alph*eps*d^2*SQRT(delP/gam1)
52:     W    = Q * gam1
53:     alph= alph0(b^4, Rd)*rRd
54:     rRd  = ( r0(b^2, D/k)-1)*(LOG10(Rd)/6)^2 + 1   ..
55:                                    WHEN  Rd <  1E6   ..
56:          =   r0(b^2, D/k)          WHEN  Rd >= 1E6
57:     eps = 1 - (0.3707 + 0.3184*b^4)                 ..
58:           *(1-((P2+1.033)/(P1+1.033))^(1/kappa))    ..
59:           ^0.935
60:     Rd   = u*D*gam1/mu
61:     b    = d/D
62:     u    = Q*1E6*4/3.1416/D^2/3600
63:     delP= (P1 - P2)*10000
64:                                          /* 設計計算 */
65:     INPUT   W,P1,delP,gam1,mu,kappa,D,k
66:     OUTPUT  d,b,alph,eps,Rd
67:     RESET   b  #0.5 [0.2, 0.8]  BY eq
68:                                          /* レンジ変更 */
69:  /* INPUT   W,P1,d,gam1,mu,kappa,D,k
70:     OUTPUT  delP,b,alph,eps,Rd
71:     RESET   delP #1000[100,10000] by eq
72:  */
```

4．おわりに（効果と将来の展開）

ＥＱＵＡＴＲＡＮ－Ｍを使用する主な効果（利点）を、プログラミング言語（ＦＯＲＴＲＡＮなど）との比較で言うならば、つぎの５点が挙げられよう。

(1) 結果が早く得られる　……　プログラミングやディバッグに相当する作業がほとんど不要なため、数式モデルを作成してから計算結果が得られるまでの時間が大幅に短縮できる。

(2) 変更が容易　……　モデルの変更や計算条件の変更（入力と出力を取り替えるなど）に簡単に対応できる。

(3) 結果の信頼性が高い　……　ソーステキストに記述した方程式は常に成立しているので、式さえ正しければ結果も正しい。

(4) 本来の仕事に専念できる　……　ユーザーはプログラミングや数値計算などに煩わされることなく、数式モデルの検討や計算結果の検討といった本来のより創造的な仕事に専念できる。

(5) ドキュメント性がよい　……　ソーステキストはそのまま良いドキュメントとなり作成者以外の人にも容易に理解できる。

なお、ＥＱＵＡＴＲＡＮ－Ｍには将来以下のような改良・発展が考えられ、これらが実現すればより広い応用分野での利用が可能になるものと思われる。

(1) より大規模な問題への対応
(2) 常微分方程式への拡張
(3) 複素数変数への対応
(4) 数式処理の応用
(5) 他のシステムとの結合

〔参考文献〕

(1) G.Oguchi, M.Mitsunaga: A powerful language to solve a set of nonlinear equations,International congress "Contribution of computers to the development of chemical engineering and industrial chemistry." at Paris (March 1978)

(2) 小口：創造的技術者の生産性を向上させる方程式解法ソフト―ＥＱＵＡＴＲＡＮ－Ｍ 日経ｺﾝﾋﾟｭｰﾀ　9月30日号 P.207 (1985)

(3) 宮原、林田、須藤：分離における数値計算，　分離技術　No. 5　P.281　(1985)

(4) 宮原他：連載「ＥＱＵＡＴＲＡＮ－Ｍ」技術計算用連立方程式解法言語 ｹﾐｶﾙ･ｴﾝｼﾞﾆﾔﾘﾝｸﾞ　8月号 (1985) ～ 4月号 (1986)

(5) 伊理、恒川、室田：グラフ論的手法による大規模連立方程式の構造的可解性判定とブロック三角化，情報処理学会論文誌，Vol.23,No.1,(1982)