NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:
The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

Developing Reusable Model Libraries in the ASCEND Environment

Joseph J. Zaher

EDRC 06-108-91

CARNEGIE MELLON UNIVERSITY

DEVELOPING REUSABLE MODEL LIBRARIES IN THE ASCEND ENVIRONMENT

Engineering Design Research Center
Research Report Series

submitted by

JOSEPH J.ZAHER

Pittsburgh, PA 15213
My, 1991

This work has **been supported** by the Engineering Design Research Center, a NSF Engineering Research Center.

University Libraries
Carnegie Mellon University
Pittsburgh PA 15213-3890

Table of Contents

List of Figures

1. Introduction

In this report, the model-building capabilities of the ASCEND (Advanced System for Computations in Engineering Design) environment are demonstrated. In chemical engineering, many simulation systems, including conventional flowsheeting programs, provide the user with a predefined set of abstractions, such as unit operations (Piela, *et al.*, 1991). These abstractions remain hidden from the user and cannot be readily tailored for specific modeling needs. The model-building language within ASCEND, however, is equation-based and offers more flexibility. It provides many of the object-oriented abstractions that are conducive to a more exact approach to modeling by allowing models to be structured more like die physical systems they are meant to represent (Westerberg, *et aL*, 1991). This is done through the use of two objects (ATOMs and MODELs) and five operators (REFINES, IS_A, IS_REHNED_TO, ARE_THE_SAME, and ARE_ALKE). Analogous to the use of unit operations to aid engineers in constructing flowsheets, libraries can be developed in ASCEND to aid in the proper formulation of equational models. It is the purpose of this work to illustrate the above language primitives in the development of standardized ASCEND structures for mathematics and thermodynamics modeling. The use of these libraries is further demonstrated by formulating and solving an adiabatic catalytic reactor. The rationales for taking such a highly structured approach toward model construction will be explained.

2. The Modeling Language

(Piela, *et al.*, 1991) gives a detailed syntactical description of the language. The following is a summary of die language features required to understand this report

The ASCEND language is strongly typed, requiring that the type of every part in every model be specified. Data in ASCEND are limited to be of the five pre-defined elementary types *(integer, string, boolean, unit,* and *real). Integers* and *strings* are used mainly for indexing arrays of objects and arc required to be given at compilation time. *Boolean, unit,* and *real* valued **ATOMs,** on **the** other hand, may have their values changed. *Reals* are unique in that they have an added notion of dimensionality and are used often as meaningfully named constants in the modeling equations. The implicitly valued ATOM is a structured object which may only contain elementary type attributes. Algebraic variables are most generally declared to be of the pre-defined type *generic_real* which is a REFINEment of the elementary type *real.* Because the **ATOM** *generic_real* REFINES *real,* it inherits the numeric and dimensional features and provides some additional elementary type attributes as parts to distinguish it as a variable. These attributes include a *real* lower and upper bound, a *boolean* fixed flag, a *real* nominal value, etc., to define its interaction with the equation solver. The *genericjreal* can be further REFINEd to any commonly encountered engineering variable type where the dimensionality and perhaps other attributes can be specified. Figure A-1 in the Appendix contains a listing of an ATOMs library file defining some useful thermodynamic variable types.

The REFINES operator may also be used to inherit the attributes of a previously written MODEL definition. Building MODELS in ASCEND involves describing MODEL types which are composed of instances and configurational relationships between them. Instances are declared locally within a MODEL using the IS_A operator preceded by a list of instance names and followed by a type. If the type is *genericjrecd* (or a refinement thereof), then the instances become system variables and may also exist in equational relationships with other ATOMS. Equations in ASCEND are written in a declarative style, and are treated as objects which can be optionally named by the user. Dimensional consistency is automatically checked by the system. Each equation is given a *boolean* included flag attribute which can be set by the user to specify whether it is to be satisfied or ignored by the solver. Equations from separate MODELS can be accumulated in one simulation by instancing MODEL types within MODEL definitions. This illustrates how complex models can be decomposed into smaller simpler building-Mock models for a more clear understanding of the necessary equations. In addition to a declarative description, all MODELs have a procedural section with which initializations and specifications can be made to prepare the MODEL for solving. Through the evironment, MODELs instanced within a simulation can be isolated in the ASCEND solver for local investigation. This provides a means in which the initial values for some variables can be propagated throughout the MODEL by making temporary local specifications, i.e. assigning a TRUE value to some of the variables' fixed flag.

The REFINES operator, as it applies to MODELs, allows the defining of a MODEL as a modification of a more general definition. All of the equations and variables defined in the general MODEL become applicable in the REFINEd MODEL. Such a general MODEL definition is referred to as a base and serves as a root node of an inheritance hierarchy (Piela, *et ai,* 1991). An inheritance hierarchy is formed to support varying degrees of specialization or rigorousness and to also maximize code re-use. If an instance of a base type is made within a MODEL definition, then flexibility is provided to specialize the instance via any of the ancestral paths made available through type inheritance. This deferred binding is accomplished with the ISJREFINEDJTO operator. With regard to thermodynamics.

the IS JREFINEDJIO operator can be used, for example, to upgrade a vapor property MODEL assuming ideal gas behavior to a more rigorous one which utilizes Pitzer's corresponding state correlations. Provision, however, is necessary to have previously declared a pitzer MODEL as a REFTNEment of the ideal gas MODEL.

The final operators to be introduced offer the configurational abstractions of object grouping and merging. The first form of object grouping is done using arrays which allows the creation of a stack of ATOMs or MODELS (currently indexed over *integer* or *string),* all of the same base type. Each element may undergo deferred binding individually. The ARE.ALKE command is used to form object cliques by forcing two or more objects (ATOMS or MODELS) to be of the same type, regardless of the degree to which any one is REFINEd. This is useful in propagating the effect of an IS_REFINED_TO command on a large group of objects and can prevent configurational modeling errors by forcing structural constraints. Finally, the merge operator ARE_THE_SAME goes beyond the ARE_ALIKE command ito that it condenses the clique of objects into one unified object It is the more important configurational operator by providing the means of connectivity necessary in the construction of most engineering models. In addition, size and memory requirements of a simulation can be reduced. For example, by ARE_THE_SAMEing variables themselves rather than equating their values, equations can be eliminated.

3. Building a Mathematics and Thermodynamics Library

The basic framework for solving two-point boundary value problems has been previously given (Piela, *et al.,* 1991). This has been found to provide a suitable framework for most numerical methods involving ODE and quadrature integration, and interpolation. To further illustrate the structure, some visual aids are used in Figure 3-1 (Zaher, 1990). MODELS are depicted as 2-dimensional boxes. Arrays are represented by a stack of oveiiapping MODELS. The ARE_THE_SAME (ATS) and ARE_ALIKE (AA) commands can be accomplished by connecting the corners of objects with labeling to help clarify which of the two is intended. Instancing is pictorialized by simply overlaying small MODELs (instances) in large ones with side to side connections from the instances to their base types. Specialization is visually formalised as a vertical hierarchy of MODELs connected top to bottom, where the most general base type lies at the top of the tree. An IS_REFINED_TO link can be drawn similar to that of an IS_A with the correct labeling.

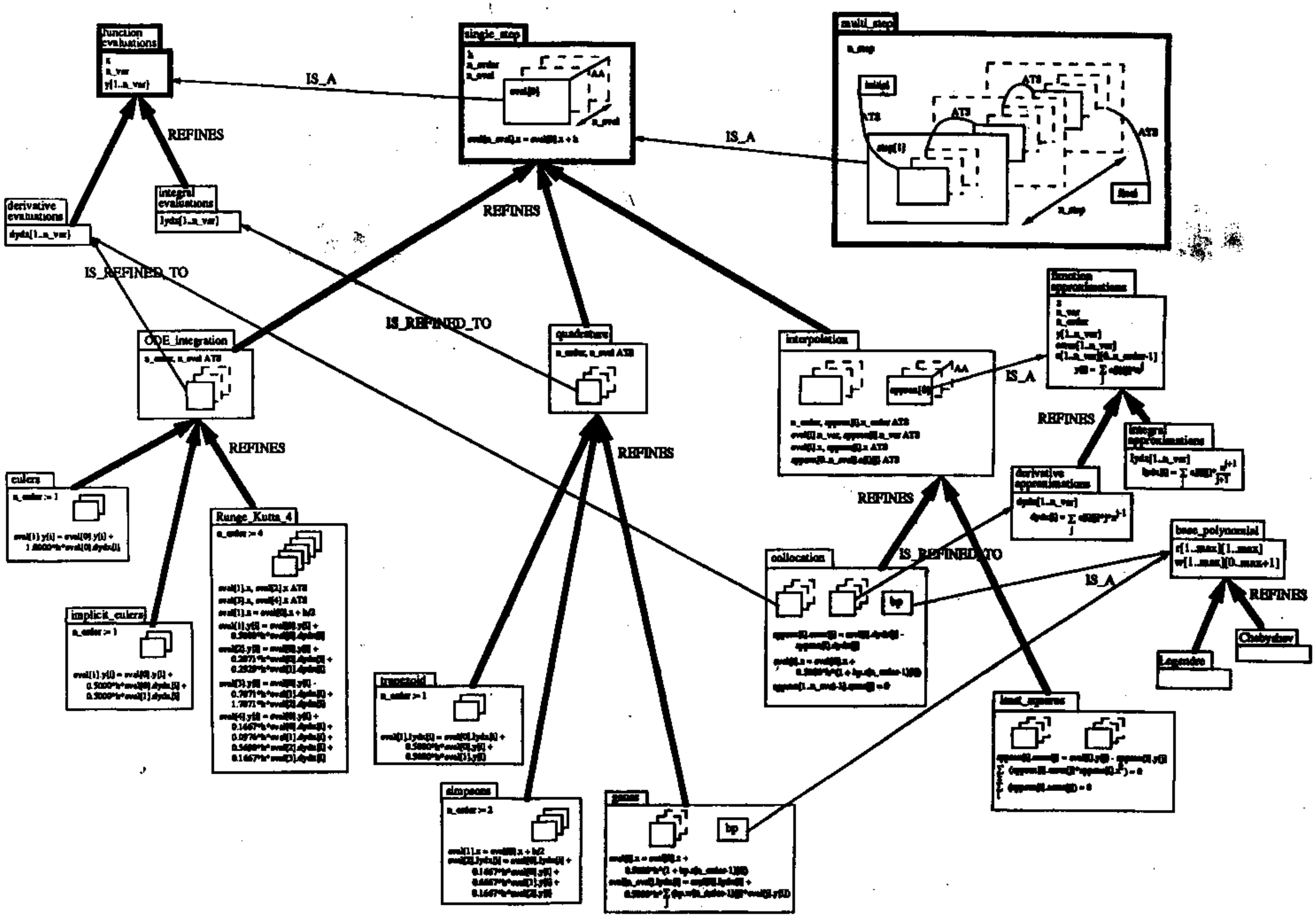


Figure 3-1: An ASCEND structure for mathematics

A listing of the ASCEND code corresponding to the building of the mathematical structure is given in Figure A-2 of (he Appendix. It is shown that a mathematical support structure can be fabricated by instancing an array of MODELS which themselves contain an array of MODELS. The purpose of making an array of *single_step* MODELS is to break the domain up into one-dimensional finite elements. Each element or *step[]* consists of an array of *function_evaluations* MODELS, the length of which being determined by the accuracy of the element. Connectivity is provided by merging the last *eval[]* instance of each element with the first *eval[]* instance of the element immediately following. For integration (ODE or quadrature), specifications include the user-supplied MODEL containing the equations to be integrated, the number of steps per the integration *(nstep)*, the step-wise integration method to be used, and any additional specifications required by the integration method. To solve a system of DAEs (Differential-Algebraic Equations), for example, a user must REFINE the MODEL *derivative_evaluations* to be a more descriptive MODEL where the problem-specific differential and algebraic equations can be entered. Then, within the instanced *multijstep*, an IS_REFINED_TO link can be used to upgrade the *eval[]* instances to be of the problem-specific type. This is simplified by ARE_ALIKEing all *eval[]* instances in all *step[]* instances of *multi_step*. Then, the upgrade can be accomplished by simply binding only one (namely the *initial)* instance. In addition, the number of elements to be used is specified and each *step[]* instance can be bound to any of the available methods. For interpolation, the order of the approximating function *(n_prder)* to be used for fitting along with the number of elements and the type of interpolating method to be used must be given. If the collocation method (Finlayson, 1980) is selected, DAEs must be provided as needed by an integration method. Otherwise, data interpolation is done by entering the data points directly at the *multijstep* instance level.

Arrays also play a major part in the development of a thermodynamics framework to provide a means for implementing non-ideal mixture thermodynamics. Figure 3-2 provides a layout of the structure. First, a library of component data is put together where the physical property constants for each individual species can be made UNIVERSAL, as listed in Figure A-3 for some selected components (Reid, Prausnitz, and Sherwood, 1977). The ASCEND operator, UNIVERSAL, is used for automatically invoking an ARE_THE_SAME command on all instances of the applied type. In Figure A-4, the ASCEND listing corresponding to pure and mult-component thermodynamic property calculations is given (Smith and Van Ness, 1987). For pure component thermodynamic property calculations, some non-ideal methods are given for both the vapor and liquid phases as REFINEments of the general ideal and incompressible MODELS.

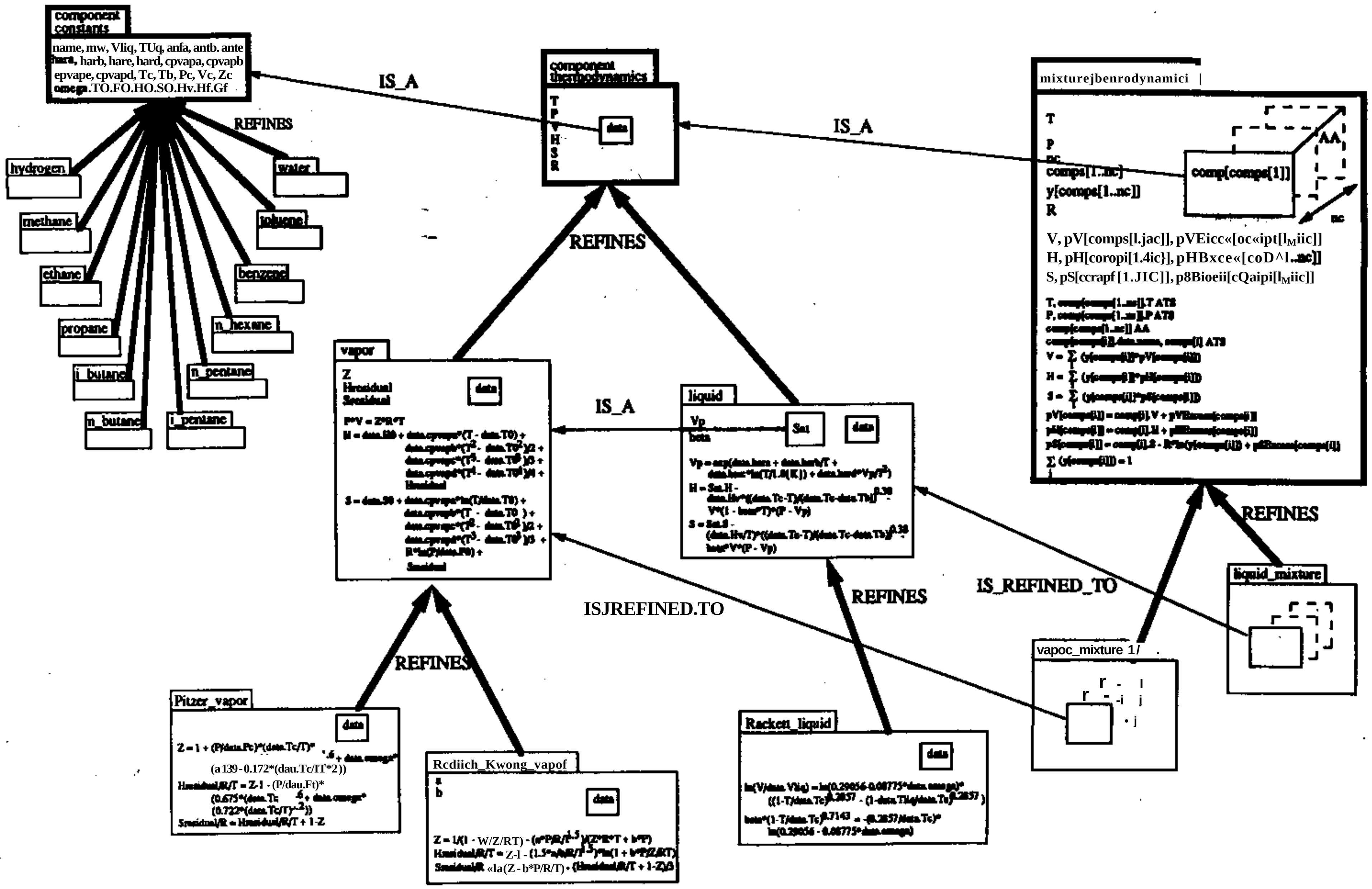

component constants
name, mw, Vliq, TUq, anfa, antb. ante
hara, harb, hare, hard, cpvapa, cpvapb
epvape, cpvapd, Tc, Tb, Pc, Vc, Zc
omega.TO.FO.HO.SO.Hv.Hf.Gf
IS_A
REFINES
hydrogen
methane
ethane
propane
i_butane
n_butane
i_pentane
n_pentane
n_hexane
benzene
toluene
water
component thermodynamics
T
P
V
H
S
R
data
IS_A
mixturejbenrodynamici
T
P
nc
comps[1..nc]
y[comps[1..nc]]
R
comp[comps[1]]
AA
nc
V, pV[comps[l.jac]], pVEicc«[oc«ipt[lMiic]]
H, pH[coropi[1.4ic}], pHBxce«[coD^1..nc]]
S, pS[ccrapf[1.J1C]], p8Bioeii[cQaipi[lMiic]]
REFINES
vapor
Z
Hresidual
Sresidual
data
P*V = Z*R*T
IS_A
liquid
Vp
beta
Sat
data
ISJREFINED.TO
REFINES
IS_REFINED_TO
REFINES
liquid_mixture
vapoc_mixture 1/
REFINES
Pitzer_vapor
data
(a139-0.172*(dau.Tc/IT*2))
Hresidual/R/T = Z-1 - (P/dau.Ft)*
Sresidual/R = Hresidual/R/T + 1-Z
Rcdiich_Kwong_vapof
a
b
data
Z = 1/(1 - W/Z/RT) - (a*P/R/T^1.5)/(Z*R*T + b*P)
Hresidual/R/T = Z-1 - (1.5*a/b/R/T^1.5)*ln(1 + b*P/Z/R/T)
Sresidual/R «la(Z - b*P/R/T) - (Hresidual/R/T + 1-Z)/3
Rackett_liquid
data

Figure 3-2: An ASCEND structure for thermodynamics modeling

Non-ideal vqgor MODELs provide a means with which to estimate compressibility factors other than 1.0 while non-incompressible liquid MODELs provide a means with which to estimate the effect of temperature and pressure on the liquid density. For phases of more than one component only ideal mixture MODELs are available in the current library. It is desired to implement a UNIQUAC REFINEment of the *liquidjnixture* MODEL as well as a mixing rule REFINEment of the *vaporjnixture* MODEL in the near future. Although not shown in the pictorial structure, the library also provides stream MODELs for basic flow process simulation. *Singlejthasejstream* MODELs will contain a mixture model to calculate the total (V, //, and S) and partial molar properties *(pV[J$_t$ pH[],* and *pS[])* from the temperature, pressure, and composition using an array of *component^thermodynamics* MODELs. An additional specification for streams is the flow rate which becomes the key attribute for converting all intensive properties to extensive properties. *Multij>hase_stream* MODELs will require a means to estimate the distribution of components among the phases. This is done using a *phasejniscibility* MODEL (or its more REFINEd *phase^equilibrium* MODEL) which contains an array of *mixture^thermodynamics* instances, one for each phase present The miscibility calculation becomes an important part of most equilibrium separation simulations.

4. Example

To demonstrate the use of the above libraries in an engineering application, a chemical reactor will be simulated. The reaction kinetics to be studied is that of vapor phase isomerization of normal pentane in the presence of hydrogen. The isomerization process is widely used among petroleum refineries as a non-additive method for the octane upgrading of hydrocarbon streams. The process explored in this example is shown in Figure 4-1. It was designed to treat approximately 3000 barrels per day of a *75»»i**nC$_5$H$_l$2/25^{mai}*$^{\%}$iC$_5$H$_l$2* feedstock. A vent was to be controlled to maintain a recycle with no less than 9O^{m0M} hydrogen. A zeolite-based platinum catalyst is used to site the isomerization reaction. The reaction is carried out at 525 °*F* in the presence of 1.25 moles of hydrogen recycle per mole of hydrocarbon feed in order to suppress excessive decomposition of the pentanes to the undesired cracked gases, ethane and propane. The competing reactions are shown in Figure 4-2.

Kinetic models have been proposed to quantify their selectivity (Voorhies and Bryant, 1968). Experimental data have suggested that all species present are assumed to be in adsorption equilibrium and that the adsorption equilibrium constants for all components are equivalent. A unified correlation K_o has been found to be a function of temperature, T, in the range 900-1100 Rankine.

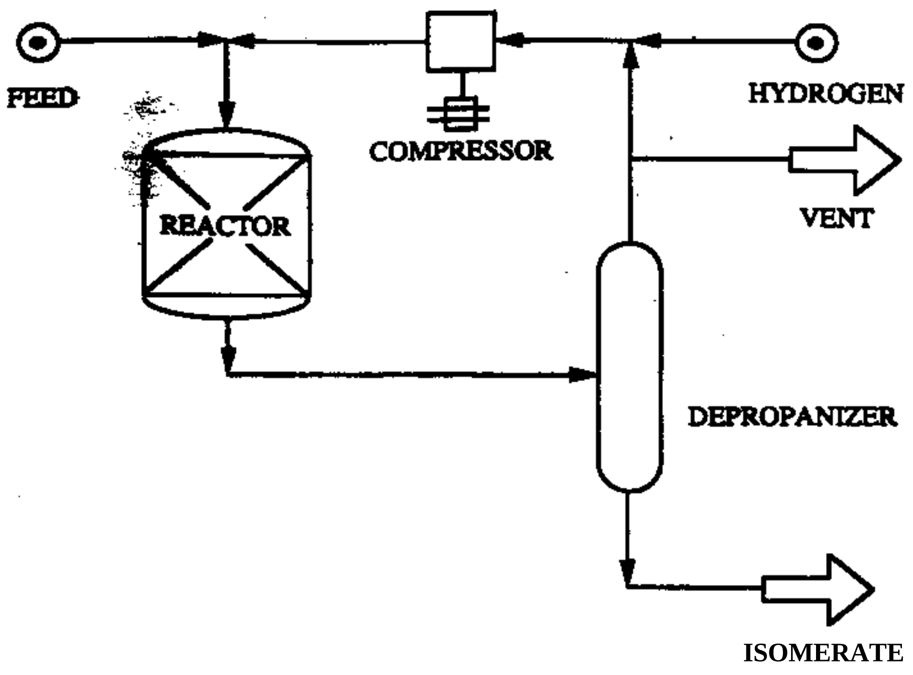


Figure 4-1: Isomerization Process Flow Diagram

$$H_{2(g)} + *_{(a)} \underset{}{\overset{K_0}{\rightleftharpoons}} H_{2(a)}$$

$$C_2H_{6(g)} \quad *_{(a)} \overset{K_0}{\rightleftharpoons} C_2H_{6(a)}$$

$$C_3H_{8(g)} \quad *_{(a)} \overset{K_0}{\rightleftharpoons} C_3H_{6(a)}$$

$$iC_5H_{12} + * \overset{K_0}{\rightleftharpoons} iC_5H_{12(a)}$$

$$nC_5H_{12(g)} + *_{(a)} \overset{K_0}{\rightleftharpoons} nC_5H_{12(a)}$$

$$nC_5H_{12(a)} \underset{k_r}{\overset{k_f}{\rightleftharpoons}} iC_5H_{12(a)}$$

$$H_{2(g)} + nC_5H_{12(a)} \xrightarrow{k_n} C_2H_{6(g)} + C_3H_{8(g)}$$

$$H_{2(g)} + iC_5H_{12(a)} \xrightarrow{k_i} C_2H_{6(g)} \quad C_3H_{8(g)}$$

Figure 4-2: Adsorption, Isomerization, and Cracking Reactions

K_o *m* 0.63726-0.0010452r-4.2182«-7r^2

The rate constants have been expressed in the Annenius fonn as functions of temperature as **well.**

/ n o 8500

_,A_ 10100

k_r = e<^{10}-^{7} " T ")

, _ , 10000)

k_t = «<^{7}-^{3} " 11000

k_n *m* e(7·, _ T)

Finally, the rate equations for components C_2H_6, iC_5H_{12}, **and** nC_5H_{12} **are** given by

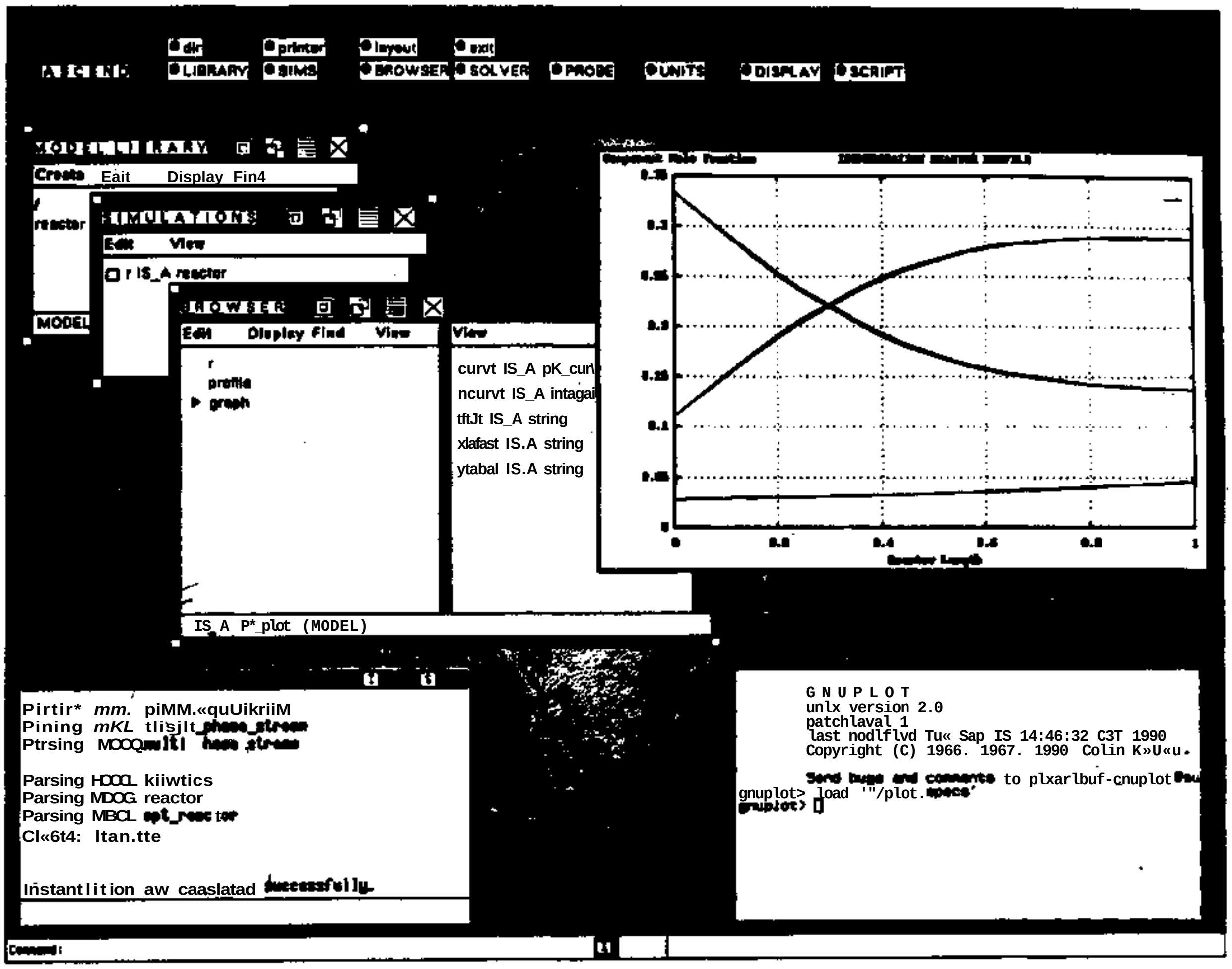
foc$_2$//$_6$ = w i (k. +^)

$$\frac{dy_{iC_5H_{12}}}{dx} = \frac{W}{VF}\frac{1}{(1+K_0P)^2}(k_f y_{nC_5H_{12}} - (k_r + k_i) y_{iC_5H_{12}})$$

$$\frac{dy_{nC_5H_{12}}}{dx} = \frac{W}{VF}\frac{1}{(1+K_0P)^2}(k_r y_{iC_5H_{12}} - (k_f + k_n) y_{nC_5H_{12}})$$

where P is **the** pressure, *W* is the mass of catalyst, and V, F, and y_i are the molar specific volume, molar flow **rate, and** compontential mole fraction, respectively, of the **gas** phase in the reactor.

The code used to formulate this **reactor** is **given in Figure** A-5. The thermodynamics, library was used **to calculate all stream related properties while the** mathematics library was used to integrate the above system of differential equations. The pressure drop across the reactor and heat transfer to and from **the reactor is neglected to** simulate an adiabatic or isenthalpic process for generation of a temperature profile. **The** *derivative^evaluations* MODEL is first REFINEd to the MODEL *kinetics* where an instance of a vapor phase *singlejphasejttearn* is created. This provides a molar specific volume and molar specific enthalpy calculation at the reactor conditions for use in the differential equations. Then, a MODEL *reactor* is written which includes an instance of a *multijtep* called *profile.* By inspection of the reaction stoichiometry, the molar flow rate is considered constant throughout the reactor. Here, the conditions for isenthalpic and isobaric constant flow is declared. For this example, finite-element orthogonal collocation is used to integrate the rate equations with five elements and two collocation points per element.



```
Pirtir* mm. piMM.«quUikriiM
Pining mKL tlisjlt_phase_stream
Ptrsing MOOQmulti hase_stream

Parsing HOOOL kiiwtics
Parsing MDOG reactor
Parsing MBCL opt_reactor
Cl«6t4: ltan.tte

Instantlition aw caaslatad successfully.
```

```
        G N U P L O T
        unlx version 2.0
        patchlaval 1
        last nodlflvd Tu« Sap IS 14:46:32 C3T 1990
        Copyright (C) 1966. 1967. 1990 Colin K»U«u.

        Send bugs and comments to plxarlbuf-cnuplot
gnuplot> load '"/plot.specs'
gnuplot> []
```

Figure 4-3: The ASCEND environment

A typical illustration of the environment following execution is shown in Figure 4-3. The simulation was performed with both ideal gas and pitzer correlated stream property calculations for comparison. First the ideal case was solved to generate values for all of the system variables. Then the non-ideal case was constructed easily by using the IS_REFINED_TO tool of the environment to upgrade each *comp[ ]* instance of each stream's mixture MODEL to the type *Pitzerjtapor.* By solving the ideal case first, the ideal solution provided excellent starting values for the non-ideal case. As can be seen in the figure, the ASCEND environment offers plotting programs which can be accessed through the language. The MODELS *singlejstep* and *multi_step* have been designed to interface with the plotters automatically through use of the merge operator. In the graph, there are two plots (ideal and non-ideal) for the mole fractions of each of the components $C_2//_6$, iC_5H_{l2}, and nC_5H_{l2} as they vary throughout the length of the reactor. It can be seen that non-idealities were neglibible at the selected temperature and pressure.

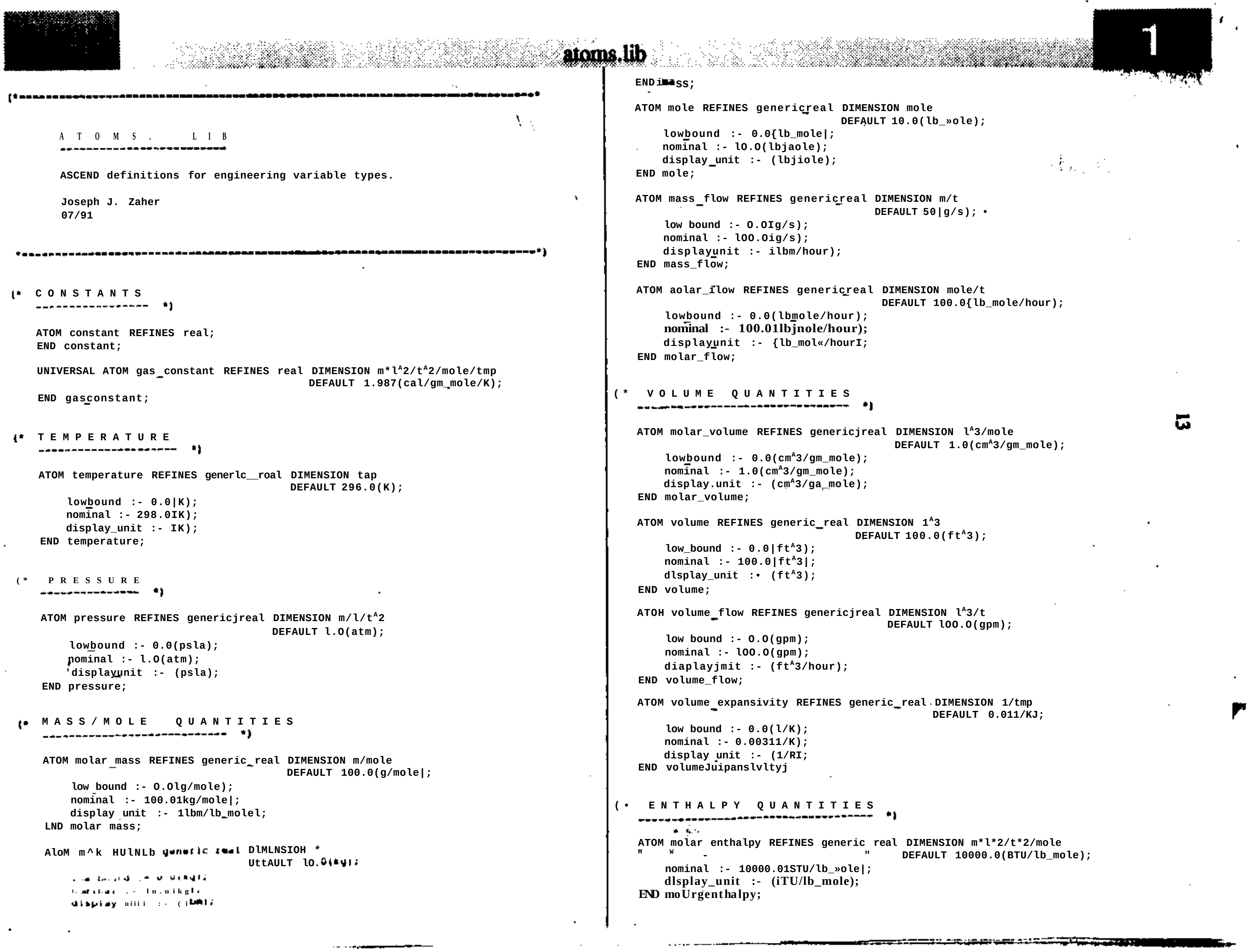
5. Conclusions

In conclusion, it was found that the ASCEND environment promoted a very convenient handling of the modeling problem. The language offered decomposition to allow a suitable breakdown of the physics of the problem for easier formulating. Through type inheritance, flexibility was provided to choose a method of integration and a method for estimating thermodynamic properties easily. Data handling was facilitated with the use of nested array structures where the qualifying names for all of the system variables became more literal and easy to associate. Finally, through merging, it was made possible to communicate information across levels of decomposition, such as by merging the temperature of the reactor with that at which the individual component thermodynamic properties within the stream were to be calculated.

It is felt by this author that a more clear understanding of the process is inevitable when the above structure is applied. Should a modification to the reactor process ever need to be implemented, the abstractions offered by the language will minimize the amount of code re-writing by isolating the MODELS that will be affected.

APPENDIX

Figure A-l



```
(*---------------------------------------------------------------------*

      A T O M S .   L I B
      -------------------

      ASCEND definitions for engineering variable types.

      Joseph J. Zaher
      07/91

 *---------------------------------------------------------------------*)

(* C O N S T A N T S
   ----------------- *)

   ATOM constant REFINES real;
   END constant;

   UNIVERSAL ATOM gas_constant REFINES real DIMENSION m*l^2/t^2/mole/tmp
                                       DEFAULT 1.987(cal/gm_mole/K);
   END gasconstant;

(* T E M P E R A T U R E
   --------------------- *)

   ATOM temperature REFINES generlc__roal DIMENSION tap
                                       DEFAULT 296.0(K);
       lowbound :- 0.0|K);
       nominal :- 298.0IK);
       display_unit :- IK);
   END temperature;

(* P R E S S U R E
   --------------- *)

   ATOM pressure REFINES genericjreal DIMENSION m/l/t^2
                                   DEFAULT l.O(atm);
       lowbound :- 0.0(psla);
       nominal :- l.O(atm);
       'displayunit :- (psla);
   END pressure;

(* M A S S / M O L E   Q U A N T I T I E S
   --------------------------------------- *)

   ATOM molar_mass REFINES generic_real DIMENSION m/mole
                                     DEFAULT 100.0(g/mole|;
       low bound :- O.Olg/mole);
       nominal :- 100.01kg/mole|;
       display unit :- 1lbm/lb_molel;
   LND molar mass;

   AloM m^k HUlNLb generlc real DlMLNSIOH *
                            UttAULT lO.0(kg);
       [illegible]
       [illegible] :- lu.uikgl.
       display uiii :- (ibm);

   ENDimass;

   ATOM mole REFINES genericreal DIMENSION mole
                                   DEFAULT 10.0(lb_»ole);
       lowbound :- 0.0{lb_mole|;
       nominal :- lO.O(lbjaole);
       display_unit :- (lbjiole);
   END mole;

   ATOM mass_flow REFINES genericreal DIMENSION m/t
                                   DEFAULT 50|g/s); •
       low bound :- O.OIg/s);
       nominal :- lOO.Oig/s);
       displayunit :- ilbm/hour);
   END mass_flow;

   ATOM aolar_flow REFINES genericreal DIMENSION mole/t
                                   DEFAULT 100.0{lb_mole/hour);
       lowbound :- 0.0(lbmole/hour);
       nominal :- 100.01lbjnole/hour);
       displayunit :- {lb_mol«/hourI;
   END molar_flow;

(* V O L U M E   Q U A N T I T I E S
   --------------------------------- *)

   ATOM molar_volume REFINES genericjreal DIMENSION l^3/mole
                                   DEFAULT 1.0(cm^3/gm_mole);
       lowbound :- 0.0(cm^3/gm_mole);
       nominal :- 1.0(cm^3/gm_mole);
       display.unit :- (cm^3/ga_mole);
   END molar_volume;

   ATOM volume REFINES generic_real DIMENSION 1^3
                                   DEFAULT 100.0(ft^3);
       low_bound :- 0.0|ft^3);
       nominal :- 100.0|ft^3|;
       dlsplay_unit :• (ft^3);
   END volume;

   ATOH volume_flow REFINES genericjreal DIMENSION l^3/t
                                   DEFAULT lOO.O(gpm);
       low bound :- O.O(gpm);
       nominal :- lOO.O(gpm);
       diaplayjmit :- (ft^3/hour);
   END volume_flow;

   ATOM volume_expansivity REFINES generic_real DIMENSION 1/tmp
                                   DEFAULT 0.011/KJ;
       low bound :- 0.0(l/K);
       nominal :- 0.00311/K);
       display unit :- (1/RI;
   END volumeJuipanslvltyj

(* E N T H A L P Y   Q U A N T I T I E S
   ------------------------------------- *)

   ATOM molar enthalpy REFINES generic real DIMENSION m*l*2/t*2/mole
                                   DEFAULT 10000.0(BTU/lb_mole);
       nominal :- 10000.01STU/lb_»ole|;
       dlsplay_unit :- (iTU/lb_mole);
   END moUrgenthalpy;
   ATOM enthalpy REFINES generic_real DIMENSION a*l^2/t^2
                                      DEFAULT 100000.01BTU);
       nominal :- 100000.O(BTU);
       dlsplay_unit  :-  (BTU|;
   END enthalpy;

   ATOM enthalpy_flow REFINES generic_real DIMENSION B*l^2/t^3
                                           DEFAULT  100000.0{BTU/hourJ;
       nominal :- 100000.0(BTU/hour);
       display_unit  :- (BTU/hour);
   END enthalpy_flow;


(*  E N T R O P Y   Q U A N T I T I E S
   ----------------------------------   *)

   ATOM moUr_entropy REFINES generlc_real DIMENSION »*l^2/t^2/»ole/tBp
                                            DEFAULT  100.0{BTU/lb_»ol«/RJ;
       nominal :- 100.0(BTU/lb_»ole/R);
       display_unit  :-  |BTU/lb_«ol«/R|;
   END molar_entropy;

   ATOM entropy  REFINES generlc_real DIMENSION ••l^2/t^2/tmp
                                     DEFAULT 1000.0{BTU/R);
       nominal :- 1000.0(BTU/R);
       display_unit  :-  {BTU/R);
   END entropy;

   ATOM entropy_flow REFINES generlc_real DIMENSION n*l^2/t^3/tmp
                                          DEFAULT 1000.01BTU/hour/R);
       nominal :- 1000.0(BTU/hour/R);
       display_unit :• (BTU/hour/R);
   END entropy_flow;


(»  D I M E N S I O N L E S S   Q U A N T I T I E S
   ---------------------------------------------   *)

   ATOM factor REFINES genoric_real DIHENSIONLESS
                                    DEFAULT 1.0;
       nominal :- 1.0;
   END factor;

       ATOM fraction REFINES factor DEFAULT 1.0;
           low_bound :- 0.0;
           nominal  :~ 1.0;
           upper_bound :- 1.0;
       END fraction;
```

Figure A-2

```
(*-------------------------------------------------------------------------

        M A T H E M A T I C S  . L I B
        ------------------------------------------------

        ASCEND structure of a mathematical modal library for
        numerically integrating function* or systems of DAE'a
        and performing polynomial interpolation.

        Joseph J. Zaher
        07/91

 -------------------------------------------------------------------------*)

% Include "/ascend/library/plot.asc"

(*  D A T A - B A S E  M O D E L S
    ------------------------------  *)

    MODEL baft©polynomial;

        rIInteger1[integer),
            w[integer)[Integer)                   IS_A real;

            FOR 1:1..6 CREATE
                r[ll(l..l) :- 0.0;
                »[l)[0..1U| :- 0.0;
            END;
            w(0)[0..1) :- 1.0;

    END basejpolynomial;

    UNIVERSAL MODEL Legendre REFINES basejpolynomial;

        INITIALIZE
            PROCEDURE values;
                (* M-0 *)
                -10M0) :-   1.0000000000;  —[01[1] :-  1.0000000000;
                (* M-l •)
                rlDll) :-   0.0000000000;  —[11[0) :-  0.3333333333;
                w[1][1] :-   1.3333333333;  w[l)(2) :-  0.3333333333;
                (• M-2 *)
                r[2)U!  :- -0.5773502692;  -(2)(1) :-  l.0000000000;
                r[2][21 :-  0.5773502692;  -121(21 :-  1.0000000000;
                (• M-3 •)
                r{3][1| :- -0.7745966692;  -13)(1) :-  0.5555555556;
                r(3)(21 :-  0.0000000000;  -(31(2) :-  0.8888888889;
                r{3|[3) :-  0.7745966692;  -13)131 :-  0.5555555554;
                (• M-4 -)
                r{4||1| :- -0.8611363116;  -14)[1) :-  0.3478548451;
                r(4)|2) :- -0.3399810436;  -(41(2) :-  0.6521451549;
                r(41(31 :-  0.3399810436;  -14113) :-  0.6521451549;
                r|4||4| :-  0.8611363116;  -14)141 ;-  0.34785484*1;
                (• M-* •)
                ON11    :-  0 9061/*84*»;  -I*Ini   :-  0.23692688*1;
                1*1Ii\  :-  0.*JS«|>«)10); -mm      :•  0.4/8fc28t?0*;
                i1">It*1 :- 0.UuOOUOOUOO;  «I*I in  :-  O.**8MM889j
                t%114|  :-  0 *lS4tf>101;  •l*iI4|  :-  0.4J8628470*;
                r1*11*1 ;-  0.90*17984*9;  w[5][5]  :-  0.2369268851;
                (• M-6 *)
                r[6][1] :- -0.9324695142;  w[6][1]  :-  0.1713244924;
                r(6)[2) :- -0.6612093865;  w[6)[2)  :-  0.3607615730;
                r[6)[3) :- -0.2386191861;  w[6][3]  :-  0.4679139346;
                r(61(4) :-  0.2386191861;  w[6][4]  :-  0.4679139346;
                r(6)[5) :-  0.6612093865;  w(6][51  :-  0.3607615730;
                r(6)(6) :-  0.9324695142;  w[6][6]  ;-  0.1713*449*41
            END values;

    END Legendre;

    UNIVERSAL MODEL Chebyshev REFINES base_polynomial;

        INITIALIZE
            PROCEDURE values;
                (• M-0 •)
                w(0)l0) :-  1.0000000000;  w[0][1]  :-  1.0000000000;
                (* M-l *)
                r[l)[l) :-  0.0000000000;  w[1][1]  :-  3.1415926540;
                (• M-2 •)
                r(2)[ll :- -0.7071067810;  w[2][1]  :-  1.1107207350;
                r[2][21 :-  0.7071067810;  -12)12)  :-  1.1107207350;
                (* M-3 •)
                r[3)[l) :- -0.8660254030;  w[3][1]  :-  0.5235987750;
                rl3)[2) :-  0.0000000000;  M(31(2)  :-  1.0471975510;
                r[3)[3) :-  0.8660254030;  -13)(3)  :-  0.5235987750;
                (• M-4 •)
                r[4](1] :- -0.9238795320;  w(4)[l)  :-  0.3005588660;
                r[4)(2) :- -0.3826834320;  M(4)[2)  :-  0.7256132880;
                r[4)[3) :-  0.3826834320;  -14)[3)  2-  0.7256132680;
                r(4)(4) :-  0.9238795320;  w[4][4]  :-  0.3005588660;
                (* M-5 *)
                r[5][1] :- -0.9510565160;  w[51(l)  :-  0.1941611040;
                rl5|(2) :- -0.5877852520;  w(5)(2)  :-  0.5083203690;
                r[5)(3) :-  0.0000000000;  -(5)(3)  :-  0.6283185300;
                r(5)[4) :-  0.5877852520;  tf(5)[4) :-  0.5083203690;
                r[5][5] i«  0.9510565160;  w[5][5]  :-  0.1941611040;
                (• M-6 *)
                r[6][1] i- -0.9659258260;  w[6][1]  :-  0.1355173350;
                r[*)[2) :- *0.7071067810;  -16)(2)  :-  0.3702402450;
                r(6)(3) :- -0.2588190450;  -16)(3)  :-  0.5057575800;
                r(61(4) :-  0.2588190450;  w[6][4]  :-  0.5057575800;
                r[€U51  :-  0.7071067810;  w[6][5]  :-  0.3702402450;
                r[6][6] :-  0.9659258260;  -16)(6)  :-  0.1355173350;
            END values;

    END Chebyshev|

(*  E V A L U A T I O N  M O D E L S
    --------------------------------  *)

    MODEL funotlon^evaluatlons;

        n_var#  n^eq                               IS_A integer;
        »,  y(integer1                             IS_A generic_real;

            n_«q,  n^var  ARE^THE_SAME;

    EMD functlon^evaluations;

    MODEL derlvatlve_evaluatlons REFINES functlon_evaluations;

        dy<Ulinteger)                              IS_A generic_real;
    END derlvatlveevaluation*;

    MODEL integralevaluation* REFINES function_evaluationa;

        Iydx(Integer)                          IS_A genericreal;

    END integral_•valuation*|

(•  A P P R O X I M A T I O N   M O D E L S
    ------------------------------------------  *)

  MODEL lunctionapproxlaatlona;

    n_var, n_eq, n_order                       IS_A integer;
    x, yllnteger)                              ISjA genericreal;
    error(integer)                             IS_iA genarlcreal;
    c(Integer)(Integer)                        ISjA ganarlcraal;
    y_ter»[Integer)(Integer)                   If_A ganerlcreal;

        neq, nvar ARE_THE_SAME;
        FOR l:l..n_var
        CREATE
            y term[i][0], c[i][0] AR£_THE_SAJtf,

            FOR j: l..n_order-l
            CREATE y_term[i][j] = c[i][j] * x^j;
            END;
            y[i] - SUM(y_tar«li)(0..n_order-l));
        END;

    INITIALIZE
        PROCEDURE fxn_«paca;
            x.fixed :- true;
            y(l..n_var).flxad :- true;
            yternll..nvarjt0..n_order-l).fix«d :- falt«;
            error[1..nvar) :- 0.0;
            error(1..nvar).fixed :- true;
            ctl..n_varl(l..n_ord«r-ll :- 0.0;
            c(l..n_varl(l..n_ord«r-ll.fixed :- trua;
            c[l..nvar)(0).fixed :- falaa;
        END fxn_«peca;

  END* function_approximation*;

    MODEL derivatlve_approxl»ationa REFINES functlon_approxlautiona|

        dydx[Integer)                          ISA generlc_real;
        dydx_ter»(Integer)(integer)            IS_A ganarlc_raal;

            FOR l:l..n_var
            CREATE
                dydx_ter»(l)(U# cl1](1] ARE_THE_SAME;
                FOR j: 2..n_order-l
                CREATE
                    dydx_term[i][j] = c[i][j] * j*x^(j-1);
                LNU;
                dydx[i] = SUM(dydx term[i][1..n order>l|);


        .t».I,A.,iI

            HUN  f*n  %imc»;

                dydx(l..n_var).fixad :- falaa;
                dydx_ter«(1..n_var)[1..n_ordar-lJ.flxad :- faXaa;
            END darapaca;

    END dar1vatlve_approxi«atIons;

    MODEL intagral_approxl«atlon* REFINES functlonjapproximations;

        Iydx(Integer), 10                      IS_A generic_real;
        Iydxterm(integer)(integer)             IS_A generic_real;

            FOR i:l..n_var
            CREATE
                FOR j: 0..n_ordar-l
                CREATE
                    Iydx_tar»U)lJI - c[i][j] * x^(j+1)/(j+1);
                END;
                y(l) - SUM(y_temll) 10..n_order-l));
                dydx(i) - 8UM(dydx_tara(l)[l..n_ordar-in;
                Iydx(l) - 10 • SUM(Iydx_ter«li)(0..n_ordar-l));
             END;

        INITIALIZE
            PROCEDURE lnt_ap«ca;
                RUN fxn apaca;
                Iydxd.Tn_var).fixed :- falae;
                Iyd*_tanid..n_varno..n_ord*rl.fUad :- falaa;
            END lnttpeca;

    END lntegral_approxUationa;

(•  P R O P A G A T I O N   M O D E L S
    ------------------------------------------  *)

  MODEL «ingla_atep;

      aval(integer)                            IS_A function_evaluatlona;
      h                                        IS~A ganaricraal;
      n_eval, n_order, n_var                   IS_A integer;

          eval(0..n aval) ARE_ALIKE;
          eval(0..n~aval).n_vtr ARE_THE_SAME;
          n_var, «val(0..n_•vaij.n.var ARE_TUE_SAME;
          aval(n_evalj.x - •v!l[0).x • h;

  END aingle_atep;

 (•  N U M E R I C A L   I N T E G R A T I O N
     ------------------------------------------  *)

    MODEL integration REFINES ilngl«_«tep;

          n^ordtr, n^aval ARf_THE^SAME;

    END Integration;


    I*  O D E   S V I T E M S
        .................—-  *)

        MODEL QOE integration REFINES integration;
    eval[0..naval)                      IS_REFINED_TO  -
                                            darlvatlve_•valuations;

END ODEintegratlon;

    MODEL «ulir B&nMM QOI^latagration;

        norder                          -  :- 1;

            FOR 1:1..n_var
            CREATE
                ival{l|.y[l] - eval{0|.y{l] • h*«val(01.dydx(l);
            END;

        INITIALIZE
            PROCEDURE apaca;
                RUN aval(0..naval).apaca;
                avald-.n_aval).x.flxad ;- falaa;
                aval(l..n_aval).y(l..n_var).flxad :- falaa;
                aval[naval).x.flxad :- trua;
                h.flxad :- falaa;
            END specs;

    END eular;

    MODEL lnpUclt_aular REFINES ODEintegratlon;

        norder                          :* 1;

            FOR i:l..n_var
            CREATE
                eval(l).y[ll - aval[0).yd) • <h/2.0)*
                    (aval(O).dydxdl • aval(l) .dydx[l)) ;
            END;

        INITIALIZE
            PROCEDURE spaca;
                RUN aval(O..n_aval}.apaca;
                aval[l..n_avall.x.flxad :- falaa;
                •val(l..n_avall.yll..n_var).fl*ad :- falaa;
                aval(n_aval).x.fl»ad :- trua;
                h.flxad :- falaa;
            END apaca;

    END impllclt_aular;

    MODEL Runga_Kutta_4 REFINES ODE_lntagratlon;

        n_ordar                         :" *»

            avaldl.x, aval(2].x ARE_THE_SAME;
            aval(3|.x( aval(4|.x ARE_THE_SAHE;
            avalfll.x - aval[0).x • h/2.0;
            FOR l:l..n_var
            CREATE
                evall1|.ydl - •val|0|.y[l) •
                    aval|0|.dydx|l|*h/2.0;
                •v«l|2| ,ylI I - •v*U0|.yll| *
                    (•vjl |0| .<Jyd*| I 1*0. 2OMO4781 «
                    «v«l | 1 | .dyd*| I | *U. ;«*;•*121i| *h;
                .v4i| M ylI I - «v«l |0i ylI I •
                    j-«v*l | I | .uya»i II#vi. lOMtiaill •
                    aval|2|.dydx|l|*1.[illegible]*h;
                aval|4|.y|l| - av*>I01-VI*•*

                    (aval(O).dydx(l) +
                    aval 11 l.dydxd 1*0.585786436 •
                    aval(2).dydxd]*3.4142135*2 •
                    aval[3).dydxtl|)*h/«.0;
            END;

        INITIALIZE                                *    -i.wr.
            PROCEDURE apaca;                                5
                RUN aval(0..neva1).apaca;
                aval[l..n_evall.x.flxad :- falaa;
                evald-.nevall.y[l..n_var).flxad j- falaa;
                aval[neval).x.fixed :- trua;
                h.flxad t- falaa;
            END apaca;

    END Runge_Kutta_4;


(» QUADRATURES
   ---------------------  •)

MODEL quadrature REFINES Integration;

    aval(0..n_aval)                     ISREFINEDTO
                                            lntagral_avaluatlona;

END quadrature;

    MODEL trapasold REFINES quadrature;

        norder                          i- 1;

            FOR 1:1..n_var
            CREATE
                avaldl).lydx(l] - aval(0).Iydxd) + (h/2.0)*
                    (avaldl.ydl • •valCOl.ydl);
            END;

            INITIALIZE
                PROCEDURE apaca;
                    RUM aval(0..n_eva1).apaca;
                    aval(l..n aval).x.flxad :• falaa;
                    eval(l..n"eval).lyd*d..n_var).fixed :- falaa;
                    avaljn_aval).x.flxad :- trua;
                    h.flxad i* falaa;
                END apaca;

    END trapazold;

    MODKL alapaona REFINES quadratura;

        n^ordar                         :- 2;

            avaldl.x * aval(0).a + h/2.0;
            FOR l:l..n_var
            CREATE
                aval[2).lydxdl - aval[0).Iydxd) + (h/6.0)*
                    (aval(0|.y(ll • 4.0»avald).yd) + aval(2).y(l))|
            END;
        INITIALIZE
            PROCEDURE specs;
                RUN eval(0..neval).specs;
                avald..n_aval).x.fixad :- falaa;
                aval(l..n_aval-l).Iydx(l..n_var).raqul{ad  :-  falaa;
                aval(l..n_aval).Iydx(l..n_var).flxsd  :-  falsa;
                aval[n_aval).x.flxsd  :-  trua;
                h.flxad  I-  falsa;                          X
            END apaca;

    END siapsona;

    MODEL Gauaa_quadratura REFINES quadratura;

        point*                          I*_A  basa_polynoaial;
        walght (lntagar) (lntagor)      If Ji ganaxlcjraal;

            FOR l:l..n_sval-l
            CREATE
                aval[l).x - aval(01.x +  (h/2.0)*
                    (I • points.r(n_ordar-11(1));
            END;
            FOR l:l..n_var
            CREATE
                FOR j:O..n_aval
                CREATE
                    weightUHJl  -  poti>ts.Wn_ordar-lUj)»
                                   aval[j].y(ll;
                END;
                •val(n_ml).Iy<U(l)  - •val(O) .Iydx[l|  t  (h/2.0)*
                    SUMO»lght(l)(0..n_aval));
            END;

        INITIALIZE
            PROCEDURE apacs;
                RUM aval[O..n_aval).apaca;
                •val(l..njaval).x.flxad  :-  falaa;
                aval(l..n_aval-l).Iydx[l..n_var).raqulra4  :-  falaa;
                aval[n_aval).Iydx(l..n_var).ClxMl  :-  falsa;
                aval(n~aval).x.flxa<l  :-  trua;
                walght[l..n_var|(O..njaval).flxad  :•  falsa;
                h.flxad  :-  falaa;
            END apaca;

    END Gauaa_quadratura;


(*   P O L Y N O M I A L   I N T E R P O L A T I O N


  MODEL lntarpolatlon REFINES slngla.stap;

      approx(lntagar)                    ISA  function__*p0ro«ls)atlona;
      graph                              IS_A pltj>lot7

          *pprox(0..n_aval)  ARE_ALIKE;
          n_ordar,  appro*(0..n_aval1.n^ordar ARE^THE^SAME;
          FOR l:0..n_aval
          CREATE
              •v*l|lj.n _v»r,  •ppro«(l|.n_w«r  A*E_THE_SA*E;
              •w*l|l|.«,  «ppro«(l|.» AAE _THE _SAME;
          k NL);
          lu«  I . i   n  w«f
          , MIA! t
              torn  !  v   »•  . i J t i  4
              v MI A I I

              IHU,
```

```
        END;

        (*——————————PLOTTING——————————*)
        graph.ncurva,  nvar ARE_THE_SAME;
        graph.curva[l..graph.ncurva).npnt,  n_aval  AaiJTHE^SAME;
        FOR 1:1..graph.ncurva
        CREATE
            FOR j:O..graph.curva[l).npnt
            CREATE
                graph.curva(i).pnt[j).xf  aval[j).x ARE THE_SAME;
                graph.curvs(U.pnt[j).y,  avaUJl:y(i)  IIJTHE_SAME;
            END;
        END;

END interpolation;

    MODEL laaat_squaraa REFINES Interpolation;

        moment[lntag«r) [intagarUlntagar]  ISA  ganarlc_raal;

            FOR  i:0..n_aval
            CREATE
                FOR  j:l..n__var
                CREATE
                    approxdl.arrorOl  -  aval[l).y(j)  -
                                         approxUl.yUl;
                END;
            END;

            FOR 1:1..n_ordar-l
            CREATE
                FOR j:l..n_var
                CREATE
                    FOR k:0..n_aval
                    CREATE
                        moment[i][j][k]  -  <approx(kl.arror(j))*
                                            Upprox[lc).x)^l;
                    END;
                    SUM(moment(lHj)[0..n_aval])  - 0;
                END;
            END;

            FOR ltl..n_var
            CREATE
                lUMUpprox[0..n_aval).srror[l))  -  0;
            END;

        INITIALIZE
            PROCEDURE spacs;
                RUN avsl(0.,n_avalj.apaca;
                RUM  approx(0..n_ava1).fxn_apaca;
                approxl0..n_aval|.arrorCl..n_var|.fixad  :-  falsa;
                approx(0..n_aval).y(l..n_var|.fixad  :-  falsa;
                •omant(l..n_ordar-lHl..n_varl[0..n_aval|.fi«ad  :-  falaa;
                approx{0].cll..n_varMO..n_ordar-ll.fUad  :-  falaa;
                h.flxad  :-  falaaj
            END spscs;

    END  l««*t_squaraa!

    MU>iL collocation RLK1NLS  lntsrpolatIon;

        points                              ISA  basejpolynoalal;
        •vallO..n_«val|                     IS_REFINED_TO
    appro*[O..n_eval)                        derlvativejevaluations;
                                        IS_REFINED_TO ~
                                             derlvative_approxlsations;

        n_order, n eval ARE_TH*_SAME;
        FOR i:0..n~eval    *"
        CREATE
            FOR j:l.,n_var
            CREATE
                eval[i).y[j), appro»Ul.y(JI ARE_THE_SAME;
                appro*(l).error[j) - eval[l).dydx[jl -
                                         appro*U).dydx[j);
            END;
        END;                                  *"'
        FOR i:l..n_eval-l
        CREATE
            eval[i).x - eval(O).x • (h/2.0)*
                (1 + polnts.r(n_or4er-l)UUf
        END;

    INITIALIZE
        PROCEDURE Specs;
            RUN eval(O..n_eval).specs;
            RUN appro*[O..n eval).der_specs;
            eval[l..n_eval)7y[l..n_varl.fixed 2- false;
            eval[O|.yll..n_var).fixed  f  true;
            ev»l(l..n_eva1-1).*.fixed 1- false;
            h.fi*ed  :- false;
            appro*l0..n_evall.errorll..n_varl  2- 0.0;
            approx(l..n_eval-l].errorll..n_varl.fixed  :- true;
            approxl0#n_eval|.error(l..n_varl.fixed  :- false;
            appro*[0).ell..n_var)[O.-n^order-U.fixed  2- false;
        END specs;

END collocation;


(*  P R O F I L E   M O D E L S
    ---------------------«--      *)

    MODEL multlstep;

        n_step, nvar                          ISA integer;
        •tep(Integer)                         ISA single_step;
       •^initial, final                       IS-A function_evaluations;
        graph                                 IS_A pltjplotj

            initial, steplll.evallO) ARE_THE_SAME;
            n var, «tep(l..n_stepl.n_var ARE_THE_SAME;
            final, step[n_step|.«val[etep[n_stepl.n_evall ARE_THE_SAME;
            FOR l:l..n_»tep-l
            CREATE
                stepli|.evaUstepli).n_eval|, stepli+1) .eval[0] ARE_THE_SAME;
            END;

            (*-------------- PLOTTING. . . . . . . . . . ---*)
            graph.ncurvi, n v*r ARE THE SAMt;
            graph.curv«1 i . gi 4pr».ncuiv«| . n^ni.
              [illegible] :MI - >knt.
            [illegible]
             [illegible]
                [illegible]
                [illegible]
                    graph.curve[i] pnt[j] [illegible]
                step(j].eval(0).x ARE_THE_SAN«;
            graph.curve[11.pnt(j-1).y7
                step[j].eval(0).y(l) ARE_THE_SAME;
          END;
          graph.curvelD.pnt[graph.curveli).npnt).x, final.x ARX_TN1_|4I*|
          graph.curve(l) .pnt [graph.curve(i) .npnt).
          y, final.y[l) ARE_THE_SAME;
        END;

    INITIALIZE
        PROCEDURE specs;
            RUN step[1..nstep).specs;
            step(l..n_step).eval[0).x.fixed :- false;
            final.x.fixed :- false;
            stepI1..n_step).h.fixed :- true;
            initial.x.fixed :• true;
            step[l..n_step).eval[01.y(l..n_var).fixed :- false;
            Initial.yll..n_var).fixed :- true;
        END specs;

END Multi_step;


( •M U L T I - S T E P    M E T H O D S
    -------------------------------    *)

    MODEL Ada*s_Bashforth_4 REFINES snilti_step;

        step[1..3]                            IS REFINED_TO
                                                ~ Rungejtutta_4;
        step[4..n_step]                       IS REFINED_TO
                                                ~ OOC_Integration;
        step(4..natep).n_order                :- 1|
        etep(4..n~step).n_eval                :- If

            FOR 1:4..n_step
            CREATE
                FOR j:l..step[l).eval(O).n_var
                CREATE
                    stepm.evalUl.ylj) - step[ij.evai 101 .y[jl •
                        (step(l).h/24)*
                        (SS*step(ll.eval(01.dydxljl -
                         59*atep[i-l).eval[0).dyd*[j) •
                         37*step(l-21.eval(0).dydx(j) -
                         t*step(l-3).eval(0).dydx(j));
                END;
            END;

        INITIALIZE
            PROCEDURE AB4_spacs;
                RUN step(1..3).specs;
                step(l..n_stepl.eval[01.x.fixed :- false;
                final.x.fixed »- false;
                step[l..n_step).h.fixed :- true;
                initial.x.fixed :- true;
                step[l..n_step|.eval[01.y[l..n_var).fixed :- false;
                initial.y[l..n_var).fixed :- true;
            END AB4_apecs;
    END ka+ms luahforth_4;

    NOOLL 1SO4« RU1NES Aulll «L«p;

        d                                     IS_A derivative_evaluation*;
    nstep                                    ISA  Integer;
    y(integer)(integer)                      ISA  generlc_real;
    x(Integer)                               IS A generlc~real;
    rtol(Integer)                            IS~A real;
    atol[Integer)                            IS~A real;
    step(l..n_step)                          IS RCFINED_TO integration;
    »tepj1..n~stepj.o__eval                  :-~1;

        natep, n_»tep ARE_THI_SAME;
        d.nvar, n_var ABE_TW_«AM«;
        x(O..natep)# d.x AtE_AUEE;
        FOR l:l..d.n_var
        CREATE
            y(O..nstepUl|, d.ytl) AiE.AUKE;
        END;
        FOR l:l..n_atep
        CREATE
            FOR j:l..tf.n_var
            CREATE
                etep(l).eval[O).y(jl, y(l-l|(j) AAS.THE.SAME;
            END;
            atep(i|:eval(O).x, x(l-l) ARE_THE_tAM«;
        END;
        FOR k:l..d.n_var
        CREATE
            atep[n_atep).eval[l).y[k)f y(natep)(k) ARE_THE_SAME;
        END;
        8tep(n_atep).eval[l).x#.ji[natep) ARE_THE_SAME;

    INITIALIZE
        PROCEDURE specs;
            d.x.fixed :- true;
            d.y(l..d.n_eql.fixed :- true;
        END specs;    ~

END lsode;
```

Figure A-3

```
(*------------------------------------------------------------------------*

      C O M P O N E N T S .   L I B
      -----------------------------

      ASCEND structure of compon+nt physical property
      constants.

      Joseph J. Zaher
      07/91

 *------------------------------------------------------------------------*)


llnclude "-/ascend/library/atoms.lib"

(*  G E N E R A L   C O M P O N E N T
    ---------------------------------*)

    MODEL component_constants;

        name                                    IS_A string;
        mw,
        Vllq, Tliq,
        anta, antb, ante,
        hara, harb, hare, hard,
        cpvapa, cpvapb, epvape, cpvapd,
        Tc, Tb, Pc, Vc, Zc, omega,
        TO, PO, HO, SO, Hv, Hf, Gf              IS_A constant;

        INITIALIZE
            PROCEDURE reference;
                TO :- 298.15IK);
                PO :- l.O(atm);
                HO :- Hf;
                SO :- (Hf - Gf)/TO;
            END reference;
    END componentconstants;


    ( * H Y D R O G E N
      -----------------*)

    UNIVERSAL MODEL Hydrogen REFINES component_constants;

        name                                    :- 'H2';

        INITIALIZE
            PROCEDURE values;
                mw :- 2.016|g/gm_mole»;
                anta :- 13.6333;
                antb :-  164.90|K|;
                ante :•   3.19{K»;
                hara :- 12.050;
                [illegible] :- -m .v*iKi;
                [illegible]
                [illegible]
                [illegible]
                [illegible]
                cpvapd :- -1.290e-6(cal/gm mole/K^3);
                cpvapd :- 1.826e-9{cal/gm_mole/K^4};
                Tc :- 33.3|K);
                Tb :- 20.37(K|;
                PC :- 12.8(atm);
                Vc :- 65.0(cm^3/gm_mole);
                Zc :- 0.305;
                ome<)a :- -0.22;
                Hv :- 904(J/gm_mole};
                Hf :- 0.00{J/gm_mole);
                Gf :- 0.00(J/gm_mole);
                Tliq :- 20.0(K);
                Vllq :- 28.3944{cm^3/9D_sole};
                RUN reference;
            END values;
    END Hydrogen;


    (• M E T H A N E
       ------------- *)

    UNIVERSAL MODEL Methane REFINES component_constants;

        name                                    :• 'CH4';

        INITIALIZE
            PROCEDURE values;
                mw :- 16.042<g/g»jsole);
                anta :- 15.2243;
                antb :-  897.841KI;
                ante :-  -7.16JKW
                hara :-  30.715/
                harb :- -1300.tl(K);
                hare :- -2.641;
                hard :- 0.442(lT2/mmHg};
                cpvapa :- 4.59t{cal/gm_mole/K);
                cpvapb> :- 1.24S«-2(cal/gm_mole/K^2);
                epvape :- 2.i60e-6{cal/^i_mole/K^3);
                cpvapd :- -2.703e-9|cal/gm_mole/K^4);
                Tc :- 190.7{K»;
                Tb :- 111.67iK»;
                Pc :- 45.4{atm|;
                Vc :- 99.0tcm^3/gm_mole);
                Ic :- 0.288;
                omega :- 0.008;
                Hv :- 8180|J/gm_moleW
                Hf :- -74840U/ga_moU);
                Gf :- -5083S.6(J/gmjROle);
                Tllq :- 111.7IK);
                Vllq :- 37.7459{cm^3/gB_mole);
                RUN reference;
            END values;
    END Methane;


    (* E T H A N E
       -------

    UNIVERSAL MODEL Ethane REFINES componentj;onstants;

        name                                    :- 'C2H6';

        INITIALIZE
            PHOCEOORL values;
                mw :- 30.068|g/gm_mole|;
            anta  :- 15.6637;
            antb  :-' 1511.42UI;
            ante  :- -17.16|K|;
            hara  :- 30.759;
            harb  :- -2464.42(K);
            hare  :- -3.601;
            hard  :- 1.073{K^*/mmHg};
            cpvapa :- 1.292<oal/g»_»ol«/K|; -
            cpvapb :- 4.254ct-2|eal/gB MBI«/K^2};
            epvape :- -1.6S?«-S|eai/g»_»«U/K^3);
            cpvapd :- 2.001«-t|e«l/fBj»U/K^4);
            Tc :- 305.4(K);
            Tb :- 104.531IK);
            Pc :- 40.2(ata);
            Vc :- 140.0(CB^3/ga_aoU)f
            Zc :- 0.205;
            OMga :- 0.OM;
            Hv :- 147201J/g« aol«|;
            Hf :- -04667|J/gi_ioU|;
            Gf :- -329201J/ga «oU|;
            Tilq :- lU.OIKI;"
            Vllq :- 54.0606|cs^3/gB_»ol*)f
            RUM reference;
        END valuta;
    END Ethan*;

    (*  PROPANE
        ------------  *)

    UNIVERSAL MODEL Propar* REFINES component_conatants;

        naa«                                          :« 'C3H0';

        INITIALIZE
            PROCEDURE  valu«s;
                mw :- 44.097(g/gmj mole);
                ant*  :- 15.7260;
                antb  :- 1062.46<K);
                ante  :- -25.1t(K);
                hara  :- 43.492;
                harb  :- -3266.92(10;
                hare  :- -4.179;
                hard  :- 1.01{K*2/amHg|;
                cpvapa  :- -1.009(cal/g»jiol«/K);
                cpvapb  :- 7.315«*2(cal/gM_«ol*/K^2);
                cpvapo  i- -3.709«-5(cal/ga_«oU/K^3);
                cpvapd  :- 7.670«~9{cal/gBjioU/ir4);
                Tc  :- 369.9{K|;
                Tb  :- 229.99KK);
                Pc  :- 41.9(ata);
                Vc  :- 203.0(c»^3/9i_«ole);
                Zc  :- 0.201;
                oiega  :- 0.152;
                Hv  :- 1077O(J/gn_mole);
                Hf  :- -103050|J/gn_»oUI;
                Cf  :- -23472U/OB_aol«|;
                Tllq  :- 231.O1K);
                Vllq  :- ?5.7680(cm^3/gm_mole);
                MUM  (·(·(ence;
            END valuea;
    i HO  f i ^^Aum.
```

```
    (*  PROPYLENE
        ------------  *)

    UNIVERSAL NODEL Propyl«n« REFINES coa^»n«nt_constants;

        name                                 :- 'C3Hf;

        INITIALIZE
            PROCEDURE valuta;
                •w  :- 42.001|g/g*jmole);
                anta  :- 15.7027;
                antb  :- 10O7.531KI;
                ante  :- -26.15IK);
                hara  :- 44.704;
                harb  :- -3260.31(K);
                hare  :- -4.379;
                hard  :- 1.63{K"2/MHg);
                cpvapa  :• O.006tcal/gBjK>l«/K|;
                cpvapb  :- 5.602«-2(cal/gnjM>l«/K^2);
                epvape  :- -2.771«-S|cal/gm_ eU / n i;
                cpvapd  :- 5.26to-9(cal/g»_wole/K^4);
                Tc  :- 365.11{K|;
                Tb  :- 225.423{K);
                Pc  :- 4S.6Uta);
                vc  :- 101.O(oa^3/g«_«ole);
                Zc  :- 0.275,
                oaaga  :- 0.140;
                Hv  :« 19653.92|j/gB_»ol«|;
                Hf  :• 20414.0(J/ga wolm\t
                Gf  :- 62710.O(J/e»~Bol«);
                Tllq  :- 223.0IKII
                Vllq  := 6075M(cm^3/gm_mole);
                RUN r«f«r««c«f
            END valuaa;
    END Propyl«A«;

    (*  i_BUTANB
        ------------  *)

    UNIVEMAL MODEL l.ButM* MUIXNES oc«pon#nt_con«taftt»;

        name                                 :- '1C4H10';

        INITIALIZE
            PROCEDURE valU«a;
                •w  :- 50.124(g/gft_«ol«);
                aota  :• 15.5301;
                antb  :- 2032.73IK);
                ante  :- -33.15IK);
                hara  :- 46.141;
                harb  ;• -3771.21|K|;
                hare  t- -4.509;
                hard  :- 2.571IT2/»Hg};
                cpvapa  i» -0.332(cal/ga_aol«/K|;
                cpvapb  i- 9.l09*-2(oal/g«_»oU/K^2);
                cpvapo  :• -4.409«-5Ical/g«_aol«/K^3\;
                cpvapd  :• 6.915«-9|cai/g»_»oU/K*4i;
                Tc  :• 400.UK);
                Tb  :• 241.424{K);
                Pc  :• 3t.o<«t«l;
                Vc  :• 263.0(cm^3/gm_mole);
                Ic  :• 0.213;
                omega  I- 0.176;
            Hv :- 21190{J/gm_mole);
            Hf :- -134500tJ/gm_aole);
            Gf :- -20878.0(J/g«_mole);
            Tllq :- 293.0(K);
            Vllq :- 104.3519(c«^3/g»_mole);
            RUN reference;
        END values;
END i_Butane;


(* n _ B U T A N E
   ---------------- *)

UNIVERSAL MODEL n_Butane REFINES compoiNrot_constants; -

    name                                    :- 'nC4H10';

    INITIALIZE
        PROCEDURE values;
            mw :- 58.124(g/gm_mole);
            anta :- 15.6782;
            antb :- 2154.90(K);
            ante :- -34.42(K1;
            hara :- 48.334;
            harb :- -4065.57iK»;
            hare :- -4.781;
            hard :- 2.68{K^2/m*Hg);
            cpvapa :- 2.266(cal/g«joole/K»;
            cpvapb :- 7.913e-2|cal/gm_mole/K^2);
            epvape :- -2.647e-5(cal/gm «ole/K"3);
            cpvapd :- -0.674e-9lcal/g*~_"*ole/ir4);
            Tc :- 425.17(K|;
            Tb :- 272.665(K);
            Pc :- 37.5(atm);
            Vc :- 255.0(cm^3/gm_mole);
            Zc :- 0.274;
            ome<;)a :- 0.193;
            Hv :- 22310(J/g»_mole);
            Hf :- -124730|J/gm_mole);
            Gf :- -17154{J/gm_mole);
            Tllq :- 293.0(K|;
            Vllq :- 1000.3869(cn^3/gmjnole);
            RUN reference;
        END values;
END nButane;


(* 1 _ P E N T A N E
   ----------------- *)

UNIVERSAL MODEL i_Pentane REFINES component^constants;

    name

    INITIALIZE
        PROCEDURE values;
            DM :- 72.1blig/gm_mole);
            anta :- lb.6338;
            anib :- 2 348.67JKI;
            antc :- - 40.0^>| K|;
            hara :- 50.428;
            harb :- -4565.64(K);
            harc :- 5.0ii;
            hard :- 3.55{K^2/mmHg);
            cpvapa :- -2.275(cal/gm_mole/K);
            cpvapb :- 1.210e-l|cal/gm_mole/K^2);
            epvape :- -6.519e-5(cal/gm_mole/K^3);
            cpvapd :- 1.367e-8(cal/gm_mole/K^4);
            Tc :- 461.0(K);
            Tb :- 300.999(K);
            Pc :- 33.4(atm|;
            Vc :- 306.0(cm^3/gm_mole);
            Zc :- 0.271;
            omega :- 0.227;
            Hv :- 24452|J/gm_molel;
            Hf :- -155185»J/gm_mole);
            Gf :- -1481UJ/gm_mole);
            Tllq :- 293.01K);
            Vllq :- 116.3726|g/cm^3);
            RUN reference;
        END values;
END IPentane;


(* n _ P E N T A N E
   ---------------- *)

UNIVERSAL MODEL n_Pentane REFINES componentconstants;

    name                                    :- 'nCSH12';

    INITIALIZE
        PROCEDURE values;
            mw :- 72.151(g/gm_mole);
            anta :- 15.8333;
            antb :- 2477.07(K);
            ante :- -39.94(k|;
            hara :- 52.682;
            harb :- -4827.081*);
            hare :- -5.313;
            hard :- 3.68|K»2/mmHQ);
            cpvapa :- -0.8*8ioal/gm_mole/K);
            cpvapb :- l.l«4e-l{cal/gm_mole/K^2);
            epvape :- -6.163e-5|cal/gm_mole/K^3);
            cpvapd :- 1.2i7e-81cal/gm_mole/K^4);
            Tc :- 489.8(K);
            Tb :- 309.187IK);
            Pc :- 33.3{atm);
            Vc :- 304.0(cm^3/gm_mole);
            Zc :- 0.262;
            omega ;- 0.251;
            Hv :- 25770(J/gm_mole);
            Hf :- -146400{J/g« mole);
            Gf :- ~8368.0|J/ga_mole);
            Tliq i* 293.0(K);
            Vllq :- 115.2572{ca^3/gm_molel;
            RUN reference;
        END values;
END n_Pentane;


(* o^H E X A N t
   ---------------- *)

UNIVERSAL MOOL n^Me*ine REFINES componentconstantc;

    name                                    :- 'nC6H14';
```

```
      INITIALIZE
          PROCEDURE values;
              •» :- 86.178{g/ga »ole);
              anta :- 15.83**; "
              antb :- 2*97.SS(K);
              ante :- -49.78(K)#
              hara :- 57.27*/
              harb :- -5587.42|K)|
              hare :- -5.885;
              hard :- 4.778|K*2/atfQ);
              cpvapa :• -1.054(cal/ga_Mole/K);
              cpvapb :- 1.390*-l{cal/gii M»1*7K^2);
              epvape :- -7.449e-5|cal/gi_ole/K^3);
              cpvapd :- 1.551e-8|cal/g»j»l«/K^4)i
              Tc :- 507.9{K);
              Tb :- 341.887(K);
              Pc :- 29.3{at»);
              Vc :- 370.0(ca^3/g»_«ole);
              2c :- 0.2*0;
              onega :- 0.29*;
              Hv :- 28850fJ/g«_»ole);
              Hf :- -1*7200|J/gB_»ole);
              Gf :- -251.01J/gp_SK>le|;
              Tllq :- 293.0|K);
              Vllq :- 130.7709(g/cM^3);
              RUN reference;
          END values;
END i\ Hexane;


(*  B E N Z E N E
    -------------  *)

UNIVERSAL MODEL Benzene REFINES coapon«nt_constants;

    nam«                                          :- 'C*H*';

    INITIALIZE
        PROCEDURE values;
            nw :- 78.114{g/gm_nole);
            anta :- 15.9008;
            ancb :- 2788.51(K);
            ante :- -52.36(K|;
            hara :- 52.1;
            harb :- -5557.*1{K);
            hare :- -5.072;
            hard :- 3.*l|K^2/m«Hg);
            cpvapa :- -8.101lcal/g«jnol«/K);
            cpvapb :- 1.133e-Hcal/g»_»ol«/K^2| ;
            epvape :- -7.20*«~5{cal/gajK>l«/K^3);
            cpvapd :- 1.703«-8{cal/g»_»ol«/K*4};
            Tc :- 562.2|K);
            Tb :- 353.252IK);
            Pc :- 48.3(atn|;
            Vc :- 259.0(cm^3/gm_mole);
            Ic :- 0.271;
            omega :- 0.212;
            Hv :- [illegible]57.0(cal/gm mole);
            H( :- [illegible] mole);
            i.f := W*«fcJ.UtJ/ga mole);
            T[illegible] - 298.0(K),
            V[illegible] - 88.[illegible](cm^3/gm mole),
            RUN reference;
        END values;
END Benzene;

(*  T O L U E N E
    -------------  *)

UNIVERSAL MODEL Toluene REFINES component_constant•;

    name                                          :- 'C7H8';

    INITIALIZE
      PROCEDURE value*;
            •w :- 92.13{g/g«_»ole|;
            anta :- U.0137;
            antb :- 3O9*.S2(K|;
            ante :- ~53.*7(K);
            hara :- 5*.785;
            harb t• -*283.50(K|;
            hare :- -5.*81;
            hard :- 4.84tK^2/MHg};
            cpvapa :« -5.tl7|cal/g»_nole/K);
            cpvapb :- l.224e-ltcal/gn_»ole/K^2|;
            epvape I- -*.*05e-5{cal/ga_»ole/K^3);
            cpvapd :- 1.173e-8(cal/gB_Mole/K^4»;
            Tc :- S91.I{K||
            Tb :- 393.77*(K);
            Pc :- 40.6lat»);
            Vc :- 3l6.0(C*^3/g"_«ole};
            Zc :- 0.2*4;
            ooega !• 0.257;
            Hv :- 7930.0ical/g»_«ole|;
            Hf i- S.003«4(J/ga_sole);
            Gf :- 122005.0(J/g»jsole);
            Tllq :- 293.0(K);
            Vllq :- 10*.2*30(CB^3/gBjK>le);
            RUN reference;
        END values;
END Toluene;


(•  H A T E R
    ---------  *)

UNIVERSAL MODEL Hater REPINE! component_constants;

    na«e                                          :- 'H2O';

    INITIALIZE
        PROCfiDURS values;
            mmi* 18.00(g/g»_Mill;
            anta t• 18.303*i
            aotb !• 381*.44|M/
            anco :- -46.13IK),
            hara :- 55.33*;
            harb i- -68*9.50(KI/
            hare :- -5.115;
            hard :- 1.05(K*2/«»Mg);
            cpvapa ;- 7.701(cal/g«jnole/K);
            cpvapb := 4.595e-4(cal/gm mole/K^2);
            cpvapc :- 2.S21e-4|cal/g«>ole/K^3t;
            cpvapd := -0.859e-9(cal/gm_mole/K^4);
            Tc i* 447.4JK»;
            Tb := 373.15(K|
            fc :- 2U.*(at«i;
```

24

```
                Vc  :- S6.0(ca^3/gB_«ole);
                2c  :- 0.229;
                omega :• 0.344;
                Hv  :- 40650{J/grn_sole);
                Hf  :- -241826|J/gm_mo U|;
                Gl  :- -228614.0{J/gm_mole};
                Tliq :- 293.0(K|f
                VIlq :- 18.0361(oi^3/g»jK>X«);
                RUN reference;
            END values;
    END Water;
```

Figure A-4

```
(*------------------------------------------------------------------------*

       T H E R M O D Y N A M I C S .   L I B
       -----------------------------------------

       ASCEND structure of tbexmodyaamle properties for single
       and multi-phase streams of pur* and mixed components.

       Joseph J. Zaher
       07/91

*------------------------------------------------------------------------*)

%Include *Wascend/library/components,lib*

ATOM specUs REFINES string;
END specie*;

ATOM states REFINES string;
END states;

(*  P U R E - C O M P O N E N T   M O D E L S

   MODEL component_thermodynamlcs;  (* Intensive *)

       R                                   IS_A gas_constant;
       data                                IS_A component_oonstants;
       T                                   IS_A temperature);
       P                                   U_A  pressure;
       V                                   It_A molar_volume;
       H                                   is'A molar_enthalpy;
       S                                   IS~A molar_entropy;

   END component_thermodynamics;

   (* V A P O R   P H A S E
      -------------------- *)

       MODEL vapor REFINES component_thermodynamics;

           Z                               IS_A factor;
           HResldual                       ISA  molar_enthalpy;
           SResidual                       IS~A molar jentropy;

               P*V - Z*R*T;
               H - data.HO +
                   data.cpvapa*(T - data.TO) •
                   data.cpvapbMT^2 - data.TO*2)/2 •
                   data.cpvepc*(T*3 - data.T0*3)/3 •
                   dat.i.cpvapd*(T^4 - data.T0^4)/4 •
                   HR«sidu4l;
               S - data.SO •
                   data.cpvapa•I»i(T/d4t*.TO) •
                   data.cpval*j* lT - 4«l«.TO) •
                   data.cpvapc• \1'J - Osis.lQ'JlII •
                   data.cpvapd•(T*J - Oaia.TO'D/1 •
                   R*ln(P/data.PO) +
                   SResidual;

           INITIALIZE
               PROCEDURE specs;
                   T.fUed :•• true;
                   P.fixed :•• true;
                   V.fixed :•- false;
                   H.fixed :•- false;
                   S.fixed :•• false;
                   I :- 1.0;
                   Z.fixed :- true;
                   HResldual :- 0.0|cal/gm_mole);
                   HResldual.fixed :- true;
                   SResldual :- 0.0(cal/gm_mole/K);
                   SResldual.fixed :- true;
               END specs;

       END vapor;

       MODEL Redllch_Kwong_vapor REFINES vapor;

           a( b                                IS_A real;

               Z - Z*R*T/(Z*R*T - b*P) - <a*P/R/T^1.5)/U*R*T • b*f);
               HResldual/R/T - Z - 1.0 - <1.5*a/b/R/T*1.5)*lnU • o*P/*/a/T)|
               SResldual/R - ln(Z - b*P/R/T) • (HResldual/RVT f 1.0 - l)/)f

           INITIALIZE
               PROCEDURE RedlichKwong_specs;
                   a :- 0.4274i*R*2*data.Tc^2.5/data.Pc;
                   b :- 0.0«€«4*R*data.Tc/data.Pc;
                   t.low_bound :• data.Zc;
                   Z.fixed :- false;
                   HResldual.fixed :- false;
                   SResldual.fixed :- false;
               END Redllcn_K*ong_epecs;

       END Redllchjlwong_vapor;

       MODEL Pltier_vapor REFINES vapor;

               Z - 1.0 • <P*data.Tc/T/data.Pc)*
                   ((0.0t) - 0.422*(data.Tc/T)^1.6) + data.omega*
                   (0.139 - 0.172*(data.Tc/T)^4.2));
               UResldual/R/T - Z - 1.0 - (P/data.Pc)*
                   <(0.§7S*(data.Tc/T)^2.f) • data.omega*
                   (0.722*(data.To/T)^S.2>);
               SAesldual/R - HResidual/R/T * (1.0-Z);

           INITIALIZE
               PROCEDURE Pitter_specs;
                   Z.low_bound :• data.Zc;
                   Z.fixed i* false;
                   HResldual.fixed :- false;
                   SResldual.fixed :- false;
               END Pltier.spscs;

       END Piti«r_vapor;

   (* L I Q U I D   P H A S C
      -------------------- *)

       MODEL liquid REFINES aomponent_th*r*odyna*lcs;
```

```
beta                                IS_A volume_expansivity;
VP                                  IS_A pressure;
Sat                                 IS_A vapor;

    T, Sat.T ARE_THE_M9ttj
    VP, Sat.P ARK_THE_lAMt;
    data, Sat.data" AUjrMSjMME;
    VP - 1.0(mmHg)*exp(data.hara •
                   data.harb/T +
                   data.harc*ln(T/l{KJ) •
                   data.bard*VP/T^2)j
    H - Sat.H -
        data.HV((data.Tc-T)/ (data.Tc-data.Tb)**0.38 +
        VM1.0 - beta*T)*(* - VP);
    S - Sat.S -
        (data.Hv/T)*((data.Tc-T)/(data.Tc-data.Tb))^0.38 -
        bata*V»(p - VP);

INITIALIZE
    PROCEDURE specs;
        RUN Sat.specs;
        T.fixed :- true;
        P.fixed :- true;
        V :- data.Vllq;
        v.fixed :- true;
        H.fixed :- false;
        S.fixed :- false;
        beta :- 0.0U/K);
        beta.fixed :- true;
        VP :- l.O(mmHg)*
            exp(data.anta - data.antb/(T • data.antc));
        VP.fixed :- false;
    END specs;

END liquid;

MODEL Rackett_llquld REFINES liquid;

        ln(V/data.Vllq) -
            ln(0.29056-0.08775*data.omega)*
            ((1 - T/data.Tc)*0.2857 -
            (1 - data.Tliq/data.Tc)^O.2857);
        betaMl - T7data.TcrO.7143 - - (0.2857/data.Tc) •
            In(0.29056-0.08775*data.omega);

    INITIALIZE
        PROCEDURE Rackett_specs;
            V.fixed :- false;
            beta.fixed :- false;
        END Rackett_specs;

END Rackettliquid;


f   MiXTUBt  MODELS
    ------------------------  *)

    M[illegible] thermodynamics;

        [illegible]                         IS A temperature;
        [illegible]                         IS   [illegible];
        [illegible]                         IS A Inlm[illegible];
        comps[illegible]                    IS A »|>«c[illegible];
```

```
y(species)                                   IS_A fraction;
data(species)                                IS_A coaponent_constants;
comp(species)                                IS_A component_thermodynamics;
V, pV(species), pVExcess[species)            IS_A molar_volume;
H, pH(species), pHExcess(species)            IS_A molar_enthalpy;
S, pS(species), pSExcess(species)            IS_A molar_entropy;
R                                            IS_A gas_constant;
tV(species)                                  IS_A molar_volume;
tH(specles)                                  IS_A molar_enthalpy;
tS(species)                                  ISA mol«r_eiitropy;

    T, comp(comps(1..nc)).T ARE_THE_SAME;
    P, comp(comps(1..nc)).P ARE_THE_SAME;
    comp(comps(l..nc)| ARE_ALIKE;
    FOR 1:1..nc
    CREATE
        comp(comps(i|j.data.name, comps(l) ARE_THE_SAME;
    END;
    V - SUM(tV[comps[l..nc]|);
    H - SUM(tH(comps(l..nc]|);
    S - SUM(tS(comps[l..nc)));
    FOR l:comps(l..no)
    CREATE
        data(l), comp(i).data ARE_THE_SAME;
        tV[l) - y(l)*pv(l);
        tH(l| - y(l)*pH(l);
        tS(i) - y(l)*pS(l);
        pV(i) - comp(i).V • pVExcess(i);
        pH[l) - compdJ.H • pHExcess(l);
        pS(l) - comp(i).S - R*ln(y(U) • pSExcess(i);
    END;
    SUM(y(comps(l..nc)J) - 1.0;

END mixture_thermodynamics;

MODEL vapor_mixture REFINES mlxture_thermodynamlcs;

    comp(comps(l..nc)|                      IS_REFIIIEO_TO vapor;

    INITIALIZE
        PROCEDURE specs;
            RUM comp(comps(l..nc|).specs;
            T.fixed :- true;
            p.fixed :- true;
            y(oompa(l..nclj.fixed :- true;
            y(comps(nc)).fixed :- false;
            V.fixed :- falss;
            H.fixed :- fa1st;
            S.fixed :- false;
            pV(comps(l..nc)).fixed :- false;
            pH(comps(l..nc)).fixed :- false;
            pf(comps[l..nc|j.fixed :- false;
            pVExcess[comps(1..nc J) :- 0.0(ft^3/lb_moU};
            pVExcess(comps(1..nc)j.fixed :- true;
            pHExcess(comps11..no11 :- 0.0|BTU/lb_mole);
            pHExcess[comps(1..nc|J,.fixed :- true;
            pSExcess(comps(1..nc)) :- 0.0(BTU/lb_mole/R);
            pSExcess(comps(1..nc)|.fixed :- true;
        END specs;
ENO v*por_mtxiure;

MODEL liquid_aliiuit RLtlNLS mlxiura_ch«rmodynamlcs;

    co«plcompsll..nc)|                      I TINED_TO liquid;
    INITIALIZE
        PROCEDURE specs;
            RUN comp(comps[l..nc)l.specs;
            T.fixed :- true;
            P.fixed :- tn»f
            y[coaps[l..no|).fixed :- true;
            y(comps(nc)).fixed x- falsef-
            v.fixed :- false;
            H.fixed :• false;
            s.fixed :- false;
            pv(comps(l..nc|).fixed :• false;
            pH[comps(l..nc)).fixed :- false;
            pS(comps(l.-nc)).fixed :- false;
            pVExcess(comps(l..ncII I- 0.0(ft^3/lb_mol«);
            pVExcess(comps[1..ncj).fixed :- true;*"
            pHExcess(comps(l..nc)) :- 0.0(BTU/lb_mole);
            pHExcess(comps(l..nc)j.fixed :• true?
            pSExcess(comps(l..nc)) :- 0.0(BTU/lbjsole/R);
            pSExcess(compsI1..ncj).fixed :- truef
        END specs;
END liquldmixture;


(* H I S C I B I L I T Y   M O D E L


MODEL phasejilsclblllty;

    T                                   IS_I1 temperature;
    P                                   IS_J1 pressure;
    Mlx(states)                         IS_J1 mixture_thermodynaxdcs;
    alpha(species)(states)[states)      IS_Ji factor;"
    ave_alpha[states)[states)           IS_i1 factor;
    nc, np                              IS~J1 integer;
    camps[Integer)                      IS_ii species;
    phases(Integer)                     IS_ii states;
    data(species)                       IS_1I componentconstants;

        T, Mlxlphases(l..np)).T ARE_THE_SAME;
        P, Mix[phases[l..np)).P ARE_THE_SAME;
        nc, Mix(phases[l..np)).nc ARE_THE_SAME;
        FOR 1; l..nc
        CREATE
            comps(i), Mix[phasesll..np)l.comps[i) AR1_THE_SAME;
        END;
        FOR 1: comps(l..nc)
        CREATE
            data(l), Mix(phases[l..np)|.data(1) ARC^THE.SAHE;
            FOR j: phases(2..np]
            CREATE
                ave_alpha[phases(l)](j)*Mlx1pha»as[l)).y(l) -
                    alpha (l)(phases(l))(j)*Mlx()).yd);
            END;
        END;

    IN1TIALI/L
        PROCEDURE specs;
            RUN Mix[phases[1..np]].specs [illegible]
                ave_alpha[phases[1]][phases[2..np]].fixed := false;
            Mix[phases[1]].y[comps[1..nc]].fixed := true;
            T.fixed := true;
            P.fixed :- iiu«,
            alpha[comps»p»11..nc|IIphases[1]][phases[2..np]].fixed := true;
```

```
                ave_alpha[phases(l))(phases[2..npJI.fixed :- false;
        END specs;

    END phase jilsclblllty;

    (• P H A S E   E Q U I L I B R I U M
       ------------------------------- •)

        MODEL phase_equlllbrlum REFINES phase_mlsclblllty;

            FOR 1: comps(l..nc)
            CREATE
                FOR j: phases(2..np)
                CREATE
                    Mix(phases[l)).pH(i) - T*Mix[phases(1J).pi(1) -
                    Mlx(j).pH(l) - T*Mlx(j).pS(l);
                END;
            END;

        INITIALIZE
            PROCEDURE equll_specs;
                T.fixed :- false;
                alpha (comps (l..nc)) (phases(l)) [phases(2..np)l .fU«d :- false;
                ave_alpha(phases(l)l[phases(2..np)) :- 1.0;
                ave_alpha[phases(l)l[phases(2..np)).fixed :- true;
            END equ11_specs;

        END phase_equlllbrlum;


(* S T R E A M   M O D E L S
   ------------

    MODEL slnglejphase_strea«;

        T                       IS_J1 temperature;
        P                       IS~J\ pressure;
        F                       IS_Jt molar_flow;
        nc                      IS~J1 Integer;
        comps (Integer)         IS~J1 species;
        y(species)              is""ifcfraction;
        data(species)           IS_Ji componentconstants;
        Mix                     IS_J1 mlxture_thermodynamlcs;
        V                       IS_J1 molar_volume;
        H                       IS_Ji molar_enthalpy;
        S                       IS_Ji molar_entropy;

            T, Mlx.T ARE THC_SAME;
            P, Mlx.P AAE~fHE_SAME;
            nc, Mlx.no ARE_THE_SAME;
            FOR 1:1..no
            CREATB
                comps(l), Mlx.comps(l) ARE_THE_SAME;
                data(oompa(l)), Mlx.data(oomps(i)) ARE_THESAME;
                ylcompadl), Mix.y(compa(l J) ARE_THE_SAME;
            END;
            V, Mlx.V ARC_THE_SAME;
            H, Mlx.N ARC_THESAME;
            J, Mlx.S A*C_THE_SAME;

        IN1T1AL1XE
            PKOCCOUM «p«cs;
                HUMMix.specs;
            T.fixed  :- true;
            P.fixed  :- true;
            F.fixed  :- true;
            y(comp«[l.-nc)).fixed  :• true;
            ylcomps(ncl).fixed  :-  false;
            V.fixed  :-  false;
            H.fixed  :-  false;
            S.fixed  :-  false;
        END specs;
END slngle_phase_stream;

MODEL multlphasestream;

    T                                     IS_1i temperature;
    P                                     IS_1i pressure;
    F                                     IS~Ji aolar__flow;
    nc, np                                IS~Ji integer;
    compslInteger|                        IS~Ji  species;
    phases(lnteger)                       IS~Jl  states;
    y[specie*|,  tylspecles)[states)      It*!I  fraction;
    data(species)                         IS~J1 component-constants;
    Mlx(states)                         .18*"/i »lxture_ther»odyna»lcs;
    Hlsc                                  IS_Ji phase_«isclbllity;
    phi[states)                           IS~Ji  fraction;
    V,  tV(states)                        IS~*I1 «olar_volujse;
    H,  tH(states)                        IS~JVvk>lar_enthalpy;
    S,  tS(atates)                        IS~J1 solar-entropy;

        T,  Mlsc.T ARE_THE_SAME;
        ?,  Mlsc.P ARE_THE_SAME;
        nc,  Hlsc.nc ARE_THESAME;
        np,  Mlsc.np ARE_THE_SAME;
        FOR l:l..nc
        CREATE
            compsli), Misc.compsl1) ARE_THE_SAM£;
        END;
        FOR  l:comps[l..nc)
        CREATE
            data[l), Mlsc.datall) ARE_THE_SAME;
            y(i] - SUM(ty(i)[phases(l..np)));
            FOR  ):phases(l..npl
            CREATE
                tyllllljl - phl(jl*Mix[j).y(l);
            END;'
        END;
        FOR j:l..np
        CREATE
            phases(j),  Misc.phases!j)  ARE_THE_SAME;
        END;
        FOR  j:phases[1..np)
        CREATE
            Mlx[j},  Misc.Mlx(j)  ARE_THE_SAME;
            tV|j}  -  phl[j)*Mlx[j|.V;
            tH[j]  -  phl|j|*Mlxni.H;
            tS|  j]  -  phiIJI'Mlxlj|.S;
        ENU;
        V  -  ^UM (t V| i [illegible] | l . . np | | J i
        M  -  [illegible] H u [illegible]
        [illegible]
```

```
            T.fixed  :-  true;
            P.fixed  :-  true;
            F.fixed  :-  true;
            y(coaps[l..ncl).fixed  :-  true;
            y[co«ps[nc)).fixed  :-  false;
            FOR  l:co«ps[l..nc)  00
                Mlx[phases[l..np]).y[i)  :• ydl;
                Mix(phases[l..np)l.y(l).fixed  :-  false;
            END;
            phi(phases[l..np)].fixed  :-  false;
            V.fixed  :-  false;
            H.fixed  :-  false;
            S.fixed  :-  false;
        END  specs;
END  multl_phase_stream;
```

Figure A-5

```
(*-----------------------------------------------------------------------*

        I S 0 M . A S C
        ---------------

        ASCEND structure Cor the modeling
        of a pentane lsoawrliatloo prooess.

        Joseph J. 2aher
        07/91

 *-----------------------------------------------------------------------*)

linclude  "-/ascend/library/mathematics.lib*
%Include  "-/ascend/library/thermodynamics.lib*

(*  R A T E   E Q U A T I O N S
    ---------------------------  *)

    MODEL kinetics REFINES derlvative_evaluations;

        catalyst                                    IS_A mass;
        s                                           is"_A slnglejphase_stream;
        t                                           IS"_A temperature;
        p                                           IS"_A pressure;
        n_var                                       := 3;
        s.nc                                        :- 5;
        s.,conps[1)                                 :-*H2';
        s..comps(2]                                 :- 'C2H6';
        s,.coaps[3]                                 :- 'C3H8';
        s.,conps[4]                                 :- '1C5H12*;
        s..compslS)                                 :- 'nC5H12";
        s..data('H2'l                               IS REFINED_TO hydrogen;
        s.,data['C2H6')                             IS~REFINED_TO ethane;
        s.,datal'C3H8'l                             IS~REFINED~_TO propane;
        s..data('iC5H12')                           IS REFINED"_TO ijpentane/
        s..data['nC5H12')                           UNREFINED_TO n_pentan«;
        s..Mix                                      ITRIFINED_TO vapor_mixture;
        x                                           IS_REFINED_TO fraction;

            t, s.t ARE_THE_SAME;
            p, s.p ARE_THE_SAME;
            y(D« «.y('C2H6') ARE_THE _SAME;
            yl21, s.y['lC5H12'l ARB^THE _SAME;
            y(3)( ..y('nC5Hl2'l ARE _THE _SAME;
            s.yl'C2H6M, a.yl'C3Hr| ARI_THE_SAME;

            dydxlll - 1.0(cn^3/g/s|
                      * (catalyst/s.V/s.F)
                      M«xp(7.3 - 10000.0|R)/T)*s.y('lCSH12') •
                        e«p(7.1 - 11000.01R|/T)*s.y('nC5H12'|)
                      /(1 .0 • 1.0|l/p«la»*p
                        *(0.63726-0.0010452(1/R)*t+4.2182e-7(1/R^2|*t^2))^2;
            dydx[2] - 1.0(cm^3/g/s)
                      *(catalyst/s.V/s.F)
                      *(exp(9.9 - 8500.0(R)/T)*s.y('nC5H12') -
                        exp(10.7 - 10100.0(R)/T)*s.y('iC5H12') -
                        exp(7.1 - 10000.0(R)/T)*s.yl'iC5H12'))
                      /(1.0 + l.0(1/pal*|*p
                        •(0.63726-0.001045211/R) •t+4.2182e-7(l/R^2)*f2) )^2;
            dydx[3]  -  1.0(coi^3/g/a)
                        *(catalyst/s.V/s.F)
                        M«»p(10.7 - 10100.0(R)/T)*s.y['lC5H12') -
                          exp(9.9 - 8500.0|R)/T)*s.y('nC5H12'l -
                          exp(7.1 - 11000.0{R)/T)*s.y('nCSH12'U
                        /(1.0 + 1.0{l/p»la)*p
                          *(0.63726-0.0010452(1/R)*t+4.2182e-7(1/R^2)*t^2))^2;

        INITIALIZE
            PROCEDURE values;
                RUN s.data(#H2').values;
                RUN s.data('C2H6').values;
                RUN s.data['C3H8#|.values;
                RUN s.dataf'1C5H12'|.values;
                RUN s.data(•nC5H12*).values;
                catalyst :- 10000.0|lbB);
                s.t :- 985.0(R);
                s.p :- 280.0(psla|;
                s.F :- U25.0|lb_BK>le/hour);
                s.y('H2'l :- 0.50;
                s.y('C2H6'| :- 0.03;
                s.y('C3H8') :- 0.03;
                s.y('lC5H12') :- 0.11;
                s.y('nC5H12') :- 0.33;
                dyds(l).nominal :- 0.001;
                dydJt[2) .nominal :- 0.01;
                dydx(3) .nominal '-' °-01;
            END values;

            PROCEDURE specs;
                RUN s.specs;
                catalyst.fixed :• true;
                s.y(#H2'l.CiJied :- false;
                s.y('C2H6')*Cl]i«d :- true;
                s.y['C3H8').fU«d :- true;
                s.y('iC5H12'l.flx«d :- true;
                s.y('nCSH12*) .fU«d :- true;
                x.fixed :• true;
                dydx[l..n_var).fixed :- false;
            END specs;
        END kinetics;

(*  R E A C T O R   O P E R A T I O N
    ---------------------------------  *)

        MODEL reactor;

            in, out                                     IS_A singlej>hase_strea«;
            profile                                     ISA multl_step;
            ratio                                       ISA factor;
            profile.n_step                              :- 5;
            profHe.step!1..profile.n_step).reorder      :- 3;
            profile.stepd. .profile.n'atepl             ISREFINED_TO collocation;
            profile.itipil..profile.n_step|.points      ISREFINEDTO Legendre;
            profile.initial                             IS_REFINED_TO kinetics;

                in, proflie.initial.s ARE _THE _SANE,•
                but, proflie.final.s ARE _THE _SAME;

                »O* is1..prof 1le.n_ step
                CREATE
                    FOR j:0..proflie.step(l).n_ord«r-l
            CREATE
                profile.steplU.evalljl.s.H,  profile.stepUl .evallj+l| .s.K
                    ARETHESAME;
                proflit.step (U.eval(j).s.F,  profile.step[ll.eval[j+H.s.F
                    ARE_THE_SAME;
                proflleTstepli).avalIjl.s.p,  profile.steplU .eval[ j+U .s.p
                    ARE_THE_IAMB;
            END;
        END;
        ln.y['C2H6'l*(ratio • 1) - 0.05*ratlo;
        in.yC 1C5H12')* (ratio + 1) - 0.25;
        in.yl'nC5H12'l*(ratio • 1) - 0.75j
        ln.F - 500.01lb_M>l«/hour)*(ratio • 1);

    INITIALIZE
        PROCEDURE values;
           FOR 1:1..proflie.nstap DO
                RUN proflie.•tepli).point•.value*;
                FOR }:0..proflie.«tep(l).n_ord«r DO
                    RUN proflie.«tepl11.availjl.value«;
                END;
            END;
            ln.T :- 985.0(R);
            ratio :- 1.25;
            profile.Initial.x :- 0.00;
            proflie.8tep[2).eval[0].K :- 0.20;
            profile.step(31.eval[0].x :- 0.40;
            profile.8tep[4).eval[0J.x :- 0.60;
            proflle.«tep[5).eval(O).a :- 0.80;
            profile.final.x :• 1.00;
        END values;

        PROCEDURE specs;
            RUN In.specs;
            RUN out.specs;
            RUN proflie.specs;
            FOR 1:1..proflie.n-«tep DO
                FOR j:0..proflie.stepd ] .n_order DO
                    profile.stepU).eval[j].s.t.fixed :- false;
                    profile.step(U.eval[j).s.p.fixed :- false;
                END;
            END;
            ln.T.fixed :- true;
            In.P.fixed :- true;
            ln.F.fixed :- false;
            ln.y(ln.cosips[l..ln.nc]).fixed :- false;
            out.t.fixed :- false;
            out.y(out.co*ps(l..out.nc)].fixed :- false;
            ratlo.fixed :- true;
            profile.initial.x.fixed :- true;
            profile.step[2).eval[01.x.fixed :- true;
            profile.step(3).eval[0).x.fixed :- true;
            profile.8tep[4).eval(0).x.fixed :- true;
            profile.stepi5).eval(0).x.fixed :- true;
            profile.flnal.x.fixed :- true;
            prof lie.stepll..5).h.fixed :- false;
        tEND specs;
  LNU [illegible]
```

References

B. A^Finlayson. (1980). *Nonlinear Analysis in Chemical Engineering.* McGraw-Hill Book Company.

P. C Piela, T. G. Eppeiiy, K. M. Westeiberg, and A. W. Wcsterberg. (1991). ASCEND: An Object-Oriented Computer Environment for Modeling and Analysis: The Modeling Language. *Comput. Chem. Engng.*₉75(1), 53-72.

R. C. Reid, J. M. Prausnitz, and T. K. Sherwood. (1977). *The Properties of Gases and liquids.* McGraw-Hill **Book** Company.

J. M. Smith and H. C. Van Ness. (1987). ***Introduction to Chemical Engineering Thermodynamics, 4th*** *Edition.* McGraw-Hill Book Company.

A. Vooihies and P. Bryant (1968). Hydroisomerizatkm of Normal Pentane over a Zeolite Catalyst ***AIChE Journal, 14(6),*** **852-856.**

A. W. Westeiberg, P. C. Piela, R. D. McKelvey, and T. G. Epperly. (July 1991). The ASCEND Modeling Environment and Its Implications . Paper presented at the 4th International Symposium meeting of the Process Systems Engineering.

J. J. Zaher. (October 1990). Graphical Representations of the ASCEND Modeling Language . **Paflbr** presented at the 12th Annual Chemical Engineering Symposium meeting of the Camegie **Mellon** University ChEGSA.