DR-2086-9

SANDIA REPORT SAND83-8253 • Unlimited Release • UC-32
Printed January 1984

# TOPAZ—A Computer Code for Modeling Heat Transfer and Fluid Flow in Arbitrary Networks of Pipes, Flow Branches, and Vessels

SAND--83-8253

DE84 006275

W. S. Winters

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550
for the United States Department of Energy
under Contract DE-AC04-76DP00789

UC-32

SAND83-8253
Unlimited Release
Printed January, 1984

# TOPAZ - A Computer Code for Modeling Heat Transfer and Fluid Flow in Arbitrary Networks of Pipes, Flow Branches, and Vessels

W. S. Winters
Analytical Thermal/Fluid Mechanics Division
Sandia National Laboratories, Livermore
Manuscript Completed March 17, 1983

## ABSTRACT

An overview of the computer code TOPAZ (Transient-One- Dimensional Pipe Flow Analyzer) is presented. TOPAZ models the flow of compressible and incompressible fluids through complex and arbitrary arrangements of pipes, valves, flow branches and vessels. Heat transfer to and from the fluid containment structures (i.e. vessel and pipe walls) can also be modeled. This document includes discussions of the fluid flow equations and containment heat conduction equations. The modeling philosophy, numerical integration technique, code architecture, and methods for generating the computational mesh are also discussed.

## Contents

## List of Figures

## Nomenclature

| Symbol | Description |
|---|---|
| $A$ | Flow cross-sectional area |
| $A_s$ | Control volume heat transfer area |
| $(\frac{A}{L})$ | Heat conduction characteristic length |
| $c$ | Isentropic sound speed |
| $C$ | Solid specific heat |
| $C_p$ | Fluid specific heat at constant pressure |
| $C_v$ | Fluid specific heat at constant volume |
| $D$ | Flow hydraulic diameter |
| $F$ | Vector of residuals |
| $f$ | Friction factor |
| $g$ | Gravitational acceleration |
| $G$ | Net rate of incident radiation per unit area |
| $h$ | Enthalpy per unit mass, convective heat transfer coefficient |
| $I$ | Integer denoting a fluid control volume (fluid control volume node), also fluid inertia constant defined by equation (III-11) |
| $J$ | Integer denoting a fluid stream tube (fluid momentum node) |
| $K$ | Integer denoting a solid heat conduction control volume (solid heat conduction node) |
| $k$ | Thermal conductivity |
| $\bar{k}$ | Average thermal conductivity |
| $L$ | Fluid control volume length |
| $M$ | Fluid control volume mass |
| $N$ | Integer denoting a computational node, variable, or equation |
| $NMAX$ | The maximum number of computational nodes, variables, or equations |
| $P$ | Fluid pressure |
| $Pr$ | Fluid Prandl number |
| $\dot{Q}_s$ | Rate of heat transfer per unit area from the containment to the fluid |
| $\dot{q}_v$ | Rate of heat generation per unit volume |

| | |
|---|---|
| $s$ | Fluid entropy per unit mass |
| $Ra$ | Raleigh number |
| $Re$ | Reynolds number |
| $t$ | Time |
| $T$ | Temperature |
| $u$ | Internal energy per unit mass |
| $v$ | Fluid velocity |
| $v_c$ | Fluid choking velocity |
| $V$ | Volume |
| $W$ | Fluid mass flow rate |
| $y$ | Vector of dependent variables |
| $y'$ | Vector of dependent variable time derivatives |
| $Z$ | Elevation |
| $\alpha$ | Absorptivity |
| $\epsilon$ | Emissivity |
| $\epsilon/D$ | Tube wall relative roughness |
| $\mu$ | Fluid dynamic viscosity |
| $\sigma$ | Stefan-Boltzmann constant |
| $\rho$ | Density |

## I. Introduction

The purpose of this document is to present an overview of the computer code TOPAZ (TRANSIENT-ONE-DIMENSIONAL-PIPE FLOW ANALYZER). TOPAZ was developed to help fulfill a need for a highly general code capable of modeling flow through complex and arbitrary arrangements of pipes, flow branches, and vessels. Although TOPAZ was originally designed to model thermohydraulic phenomena in solar central receivers [1, 2], its application has since been expanded to include a broad range of non-solar heat transfer/ fluid flow problems of interest to Sandia. Features of TOPAZ include the following:

- One-dimensional, transient modeling of internal fluid flows
- Two-dimensional, transient modeling of heat conduction in fluid containment structures (vessel and pipewalls)
- Coupling between fluid flow and containment heat conduction calculations
- Capability of modeling incompressible flows, including thermally compressible flows for which fluid properties vary with temperature.
- Capability of modeling compressible flows and associated phenomena including flow choking and the propagation of shock and rarefaction waves
- Capability of linking flow domains together to form arbitrary arrangements of pipes, valves, flow branches and vessels.
- Capability of modeling a wide variety of flow and heat conduction boundary conditions; including arbitrary user-specified boundary conditions
- Capability of including user-specified control equations (i.e., any equations which describe a functional relationship between time, space, and the dependent variables in the system model)
- Capability of applying a number of constitutive models (e.g. heat transfer coefficients, form loss factors, friction factors, etc.) including user-specified models
- A high degree of code modularity which permits the addition of new fluid and solid properties, constitutive models, and boundary conditions as the need arises
- Fully implicit integration of all model equations without operator splitting, i.e., the same numerical technique is uniformly applied to all equations.

This report will provide the reader with an overall view of TOPAZ capabilities. Section II of the report will focus on the coding and modeling philosophy. A discussion of the assumptions used in generating heat transfer and fluid flow models is also presented. Discussions of the fluid conservation equations and containment heat conduction equations are given in Sections III and IV respectively. The concept of a TOPAZ computational mesh will be introduced in Section V. Methods of generating such meshes will also be discussed. Section VI contains a discussion of the fully implicit numerical technique used to solve the TOPAZ model equations. The overall TOPAZ code architecture will be described in Section VII. Future activities associated with the further development of TOPAZ and its applications will be discussed in Section VIII.

In preparing this document an effort was made to present a TOPAZ overview. Rather than dwelling on any one particular facet of the code, all aspects of the code are discussed in a general way. Readers interested in specific details are urged to consult the references listed in Section X. Moreover, this report is not a user's manual for TOPAZ. Specific calculational results using TOPAZ will not be presented. Such information will be documented separately in future reports.

## II. TOPAZ Coding and Modeling Philosophy

### 1. Similarity with LOCA Codes

The nuclear power industry has developed a number of highly general computer codes for modeling internal flows through complex piping networks. The motivation behind such code development has been the need to model events such as the postulated 'loss of coolant accident' (LOCA) in nuclear reactors. A LOCA is an off-design emergency situation in which a primary or secondary coolant line or component ruptures. For most reactors the coolant is high temperature, high pressure water which experiences blowdown (nearly instantaneous boiling via reduced pressure) when the rupture occurs. A major concern in a LOCA is that insufficient coolant will remain in the system resulting in a partial core meltdown. The LOCA codes are designed to predict the duration of blowdown events and the functioning of emergency systems.

Most notable among the family of LOCA codes are RETRAN [3], a code developed under private EPRI funding, and the RELAP [4,5] family of codes developed primarily with NRC funding. With the exception of RELAP5, these codes deal exclusively with equilibrium water-steam flows. RELAP5 has been extended to treat nonequilibrium water-steam mixtures.

The LOCA codes enable the user to model fluid flow using one-dimensional, transient conservation equations (continuity, momentum, and energy), which are partial differential equations with time and axial distance as the independent variables. The equations are discretized in space using a special displaced mesh upwind differencing scheme. This differencing permits the modeling of branching flows as well as single stream flows. Because of this important feature it was decided to employ the same type of spatial differencing scheme in the TOPAZ fluid flow modeling.

In many ways TOPAZ has been modeled after the existing LOCA codes. This is particularly true with regard to the modeling of fluid flow. Despite these similarities TOPAZ retains a number of features which are not found in the openly available versions of LOCA codes. For example:

- A fully implicit numerical solver DASSL [6] is used to integrate all model equations in time.

- Flows are not restricted to water-steam.
  Any incompressible or compressible fluid for which complete thermodynamic data is available can be modeled. At this writing, properties for the following fluids are "hardwired" into the code: air, the hydrogen and helium isotopes, molten salt, and liquid sodium.
- A variety of special boundary conditions are available.
- Transient heat conduction in the fluid containment is modeled in two dimensions rather than one.
- Code modularity permits addition of new boundary conditions, fluid properties, equations of state, and constitutive relationships (e.g. heat transfer correlations, etc.) as the need arises.
- Specialized reactor component and reactor kinetics models are not hardwired into the code.

An effort has been made to develop TOPAZ for the purpose of modeling general fluid flow/heat transfer situations. Wherever possible, useful LOCA code modeling techniques have also been incorporated.

Typically, hundreds of manyears are expended in developing a single LOCA code. Much of this development centers around making the code "user-friendly" while still retaining generality. LOCA code users set up their models exclusively through a single input data file or deck. User reprogramming of the code is seldom, if ever, done; yet, the code can be used to model arbitrary arrangements of pipes, branches and vessels.

In order to incorporate this kind of generality, TOPAZ was developed to be used in conjunction with a Fortran driver program. The driver program is intended to take the place of an extensive (and expensive) user-friendly interface. A typical driver program severely restricts the modeling to a specific class of problems. The code DRAC [2], for example, is a TOPAZ driver program which models a specific class of fluid flow-heat transfer problems associated with solar receivers. Although DRAC is user-friendly, its application is restricted to a specific set of boundary conditions and flow geometry.

Developing TOPAZ to be used in conjunction with a driver program was considered a practical alternative to writing a code with a completely general but prohibitively expensive user-friendly interface. Discussions relating to how the driver program fits into the TOPAZ code architecture are presented in later sections of this report.

## 2. TOPAZ Verification

In order for TOPAZ and any related user-friendly driver programs to be of value, their development must be closely linked to a program of code verification. This verification can be accomplished by comparing TOPAZ calculations to three sources:

- Predictions with known analytical solutions
- Other code calculations
- Laboratory experiments

For many simple problems (e.g. two-dimensional heat conduction, propagation of shock and rarefaction waves, simple one-dimensional-steady flows, etc.) analytical solutions are available for direct comparison with TOPAZ predictions. For more complicated geometries, it may be necessary to rely on comparisons with independently developed codes. Although such code comparisons result in increased confidence, true code verification depends on direct comparison with well controlled experiments. This is particularly true when the purpose of such comparisons is to evaluate a new constitutive model (form loss model, heat transfer correlation, etc.)

A considerable amount of TOPAZ code verification has already taken place (e.g. reference [2]). Future verification of form loss models, vessel heat transfer models, and the flow branching model will rely heavily on the availability of experimental data. Much of this data has yet to be obtained. Until it is, many applications of TOPAZ (or any similar code) must be regarded as experimental.

## 3. Modeling Assumptions

In order to understand the extent to which the physics of fluid flow and heat transfer are being resolved in TOPAZ, it is useful to list the modeling assumptions:

- Fluid flow is treated as one-dimensional and transient.
- Fluid flow is either incompressible or compressible with all local fluid properties a function of the local thermodynamic state.
- Heat conduction in the flowing fluid is neglected.
- Multi-dimensional, transient effects associated with convective heat transfer from the containment to the flowing fluid are treated using one-dimensional quasi-steady, locally-applied heat transfer correlations.
- Multi-dimensional, transient effects associated with fluid-pipe wall friction are accounted for using one-dimensional

quasi-steady, locally-applied friction-factor correlations.

- Multi-dimensional, transient effects associated with changes in direction (restrictions, branches, tube bends, etc.) are accounted for using one-dimensional quasi-steady, locally-applied, form-loss correlations.
- Flowing fluids are non-reacting and do not mix with flowing fluids of different species.
- Viscous dissipation is neglected in the fluid energy equation.
- Heat conduction in the fluid containment (pipe and vessel walls, etc.) is assumed to be two-dimensional and transient.
- The effect of containment property variations due to temperature are included
- The effect of standard gravitational acceleration (e.g. 1 g fluid head effects) is included in the fluid flow modeling. However, time and directional dependent body forces, such as those resulting from angular rotation or linear acceleration of the piping and vessel configuration are not accounted for at this time.

The above assumptions give rise to the fluid flow and fluid containment heat conduction equations presented in Sections III and IV, respectively.

# III. Fluid Flow Equations

## 1. Introduction

The TOPAZ fluid flow equations consist of three integral conservation equations for continuity, energy, and momentum. The equations presented here are given in the form of spacially discretized differential equations with time as the independent variable. They can be derived directly from the fully compressible partial differential Navier-Stokes Equations using appropriate applications of Leibnitz Rule and Green's Theorm. A detailed derivation of these equations is rigorously presented in references [3] and [4].

Before presenting the conservation equations, it is useful to describe the computational mesh. As with most numerical solutions of the Navier-Stokes equations, TOPAZ modeling requires that the fluid domain be spacially discretized in order to eliminate space as an independent variable. Thus the three partial differential equations (PDE's) which describe the conservation of mass, energy, and momentum within the fluid domain are replaced by a much larger set of ordinary differential equations (ODE's) with time as the independent variable. The number of ODE's is directly proportional to the number of mesh points (nodes, cells, or control volumes) used in discretizing the fluid domain. Naturally the finer the computational mesh, the better the ODE's approximate the behavior of the PDE's.

The actual spatial discretization used in TOPAZ is commonly referred to as the 'displaced mesh' or 'control-volume, stream-tube' method. This discretization or discretizations similar to it, are used in virtually all nuclear reactor LOCA simulation codes [3, 4, 5] as well as many 2-D and 3-D transient codes used to describe flow and heat transfer in fluid domains. Basically the discretization method divides the fluid domain into a finite number of control volumes, where continuity and energy are conserved. Momentum is conserved at control volume interfaces (i.e., the boundaries between control volumes).

The TOPAZ discretizations can best be understood through the aid of Figure 1. The figure shows two control volumes and three control volume interfaces. In this particular case the fluid is flowing in a single stream through a channel having a jump in cross-sectional flow area. As will be illustrated later, this discretization can also be used in modeling flow branches and directional changes in the flow.

For the situation depicted in Figure 1, the control volume I is bounded by lines connecting points 1, 2, 3, 10, 11, 12, 1 and control volume I+1 is bounded by lines connecting points 3, 4, 5, 6, 7, 8, 9, 10, 3. The stream tube J bounded by lines

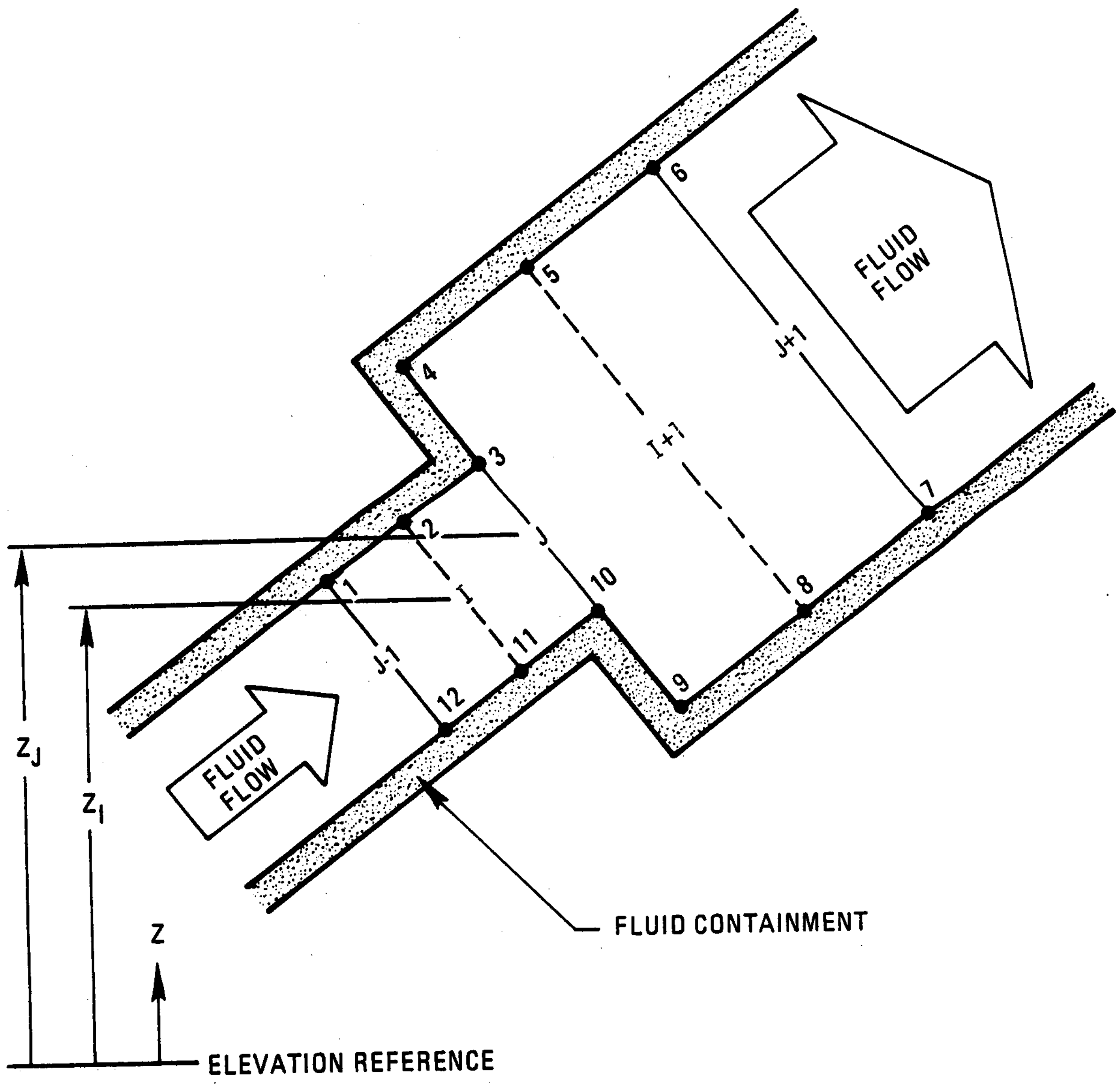


Figure 1. TOPAZ Spatial Discretization for a Single Flow Stream

connecting points 2, 3, 4, 5, 8, 9, 10, 11, 2 connects control volume I to control volume I+1.

In general a control volume may have any number of stream tubes attached to it, but a stream tube can connect only two control volumes. The significance of this statement becomes more apparent in flow branching situations such as the one depicted in Figure 2. Control volume I3 has three stream tubes attached to it, namely J2, J3 and J4. However, all of the stream tubes shown (J1, J2, J3, J4, J5, and J6) connect two control volumes.

In the TOPAZ code and in most reactor LOCA codes (e.g. references 3-5) continuity and energy equations are written for control volumes and momentum equations are written for stream tubes. These equations, together with appropriate boundary conditions, initial conditions, constitutive relationships and fluid state equations completely describe the thermal-hydraulic behavior of fluids flowing in arbitrary networks of pipes, branches, and vessels. Such a model is of course restricted to the limitations of one-dimensional, transient analysis and the assumptions set forth in the previous section. All two- and three-dimensional effects in the flow including fluid-containment heat exchange and pressure losses due to abrupt area changes, tube bends, and branching are accounted for through constitutive (empirical and semi-empirical) relationships. Some of these relationships are well documented and highly reliable such as the Moody friction correlation [7] and numerous forced convection heat transfer correlations for fully developed interior tube flow [8]. For the most part these relationships have been obtained for steady flow. In keeping with standard practice in one-dimensional, transient modeling, the steady relationships are used directly in the unsteady equations. Hence with regard to the application of the constitutive relationships, the flow is regarded as 'quasi-steady'. For most pipe flows of interest to Sandia the 'quasi-steady' assumption produces accurate results. Relaxation of the quasi-steady assumption, when necessary, can only be accomplished through combined analysis and experimentation. With the exception of a few studies such as reference [9] which examines the unsteady effects in tube wall friction, literature on this subject is surprisingly sparse. Codes such as TOPAZ together with careful experimentation could significantly advance the state-of-the-art in this area.

The next three subsections in this chapter present the conservation equations used in TOPAZ fluid flow modeling. As mentioned previously their complete derivation is presented in references [3] and [4]. The spacially discretized conservation equations have been modified somewhat to suit the particular needs of this study. These modifications will be discussed in detail.

It may be useful for the reader to view the fluid flow model as a system of three

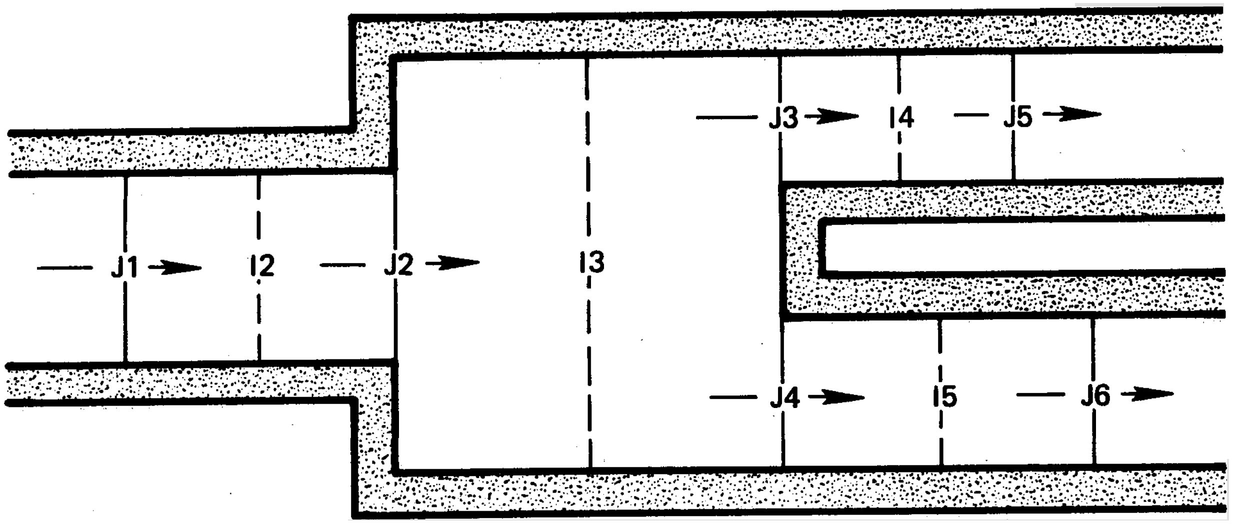


Figure 2. TOPAZ Spatial Discretization for a Flow Branch

equations (continuity, energy and momentum) solved for three dependent variables: pressure, P, internal energy per unit mass, u, and mass flow rate W. The state of the fluid at any point in the flow is uniquely determined by the two thermodynamic variables P and u, i.e. $\rho = \rho\,(P,u)$, T = T(P, u), etc. By formulating the equations in terms of P and u, calculations in two-phase as well as single-phase flow are straightforward. The state of the fluid in all phase regions (subcooled, saturated, and super heated) are uniquely determined by P and u. Such would not be the case if, say, P and T were selected as the fluid model dependent variables.

The final section in this chapter presents a brief discussion of the constitutive and property relations required to complete the fluid flow modeling.

## 2. Fluid Continuity Equation

Continuity of flow for an arbitrary control volume can be expressed as

$$\frac{dM_I}{dt} = \sum_J W_J. \qquad (III-1)$$

Simply stated equation (III-1) says that the time rate of change of mass within a control volume I is equal to the net mass flow rate into the control volume. $W_J$ represents the mass flow rate $(\rho Av)_J$ evaluated at a momentum node (stream tube) J. If a flow stream exits the volume, $W_J$ would take on a negative sign. Hence continuity for volume I3 in Figure 2 can be written as

$$\frac{dM_{I3}}{dt} = \sum_{J=J2}^{J4} W_J = W_{J2} - W_{J3} - W_{J4}. \qquad (III-2)$$

In the TOPAZ code, equation (III-1) is expressed in terms of thermodynamic state variables, hence

$$\frac{dM_I}{dt} = \frac{d(\rho V)_I}{dt} = V_I\left[\left(\frac{\partial \rho}{\partial P}\right)_u \frac{dP}{dt} + \left(\frac{\partial \rho}{\partial u}\right)_P \frac{du}{dt}\right]_I. \qquad (III-3)$$

The final form of the TOPAZ mass balance then becomes

$$V_I\left[\left(\frac{\partial\rho}{\partial P}\right)_u \frac{dP}{dt} + \left(\frac{\partial\rho}{\partial u}\right)_P \frac{du}{dt}\right]_I = \sum_J W_J. \qquad (III-4)$$

The primary advantage of such a formulation is that no modifications need be made in treating compressible or incompressible flows. For an incompressible fluid the property $\left(\frac{\partial\rho}{\partial P}\right)_u$ is zero.

## 3. Fluid Energy Equation

Conservation of energy for an arbitrary control volume can be expressed as

$$\frac{d}{dt}\left[\rho V\left(u+\frac{1}{2}v^2\right)\right]_I = \dot{Q}_S + V\dot{q}_v + \sum_J W_J\left[u+\frac{P}{\rho}+\frac{1}{2}v^2+g(Z-Z_I)\right]_J \qquad (III-5)$$

Stated simply equation (III-5) says that the time rate of change of kinetic and internal energy within a control volume I is equal to the rate of heat transfer into the control volume, $\dot{Q}_s$, from the fluid containment plus the volumetric rate of heat generation, $V\dot{q}_v$, within the control volume plus the net rate of convected energy (sum of internal energy, flow work, kinetic energy and gravity work) into the control volume.

A few words are required regarding how and where the various properties are evaluated in the flow. As previously stated $W_J$ is the mass flow rate evaluated at the center of a stream tube. The mean control volume velocity $v_I = W_I/\rho_I A_I$ where $W_I$ is the mean control volume mass flow rate. A number of methods are available for calculating $W_I$ [3,4,5] in terms of the $W_J$'s. For a single stream flow (Figure 1) all these methods collapse to a form which states that $W_I$ is the sum of the upstream and downstream $W_J$'s divided by 2, i.e., the arithmetic average. For a control volume with three or more adjacent stream tubes TOPAZ employs the method used in reference [3] for computing $W_I$, namely

$$W_I = \frac{1}{2}\left[(\sum_J W_J)_{in} + (\sum_J W_J)_{out}\right] \qquad (III-6)$$

All thermodynamic properties for the fluid are associated with control volume centers, i.e., $P_I$, $u_I$, $\rho_I$, etc. A question immediately arises as to how these quantities are extrapolated to the stream tube center (i.e., the location where $W_J$ is computed). Like most LOCA codes, TOPAZ utilizes the 'upwind extrapolation'. The quantities u, P, and $\rho$ in the brackets on the right hand side of equation (III-5) are assumed to be identically equal to their upwind control volume counterparts. This convention is maintained for all fluid properties at momentum nodes. Hence we can write the following equation to describe the conservation of energy for the control volume I3 in Figure 2:

$$\begin{aligned}\frac{d}{dt}\left[\rho_{I3}V_{I3}(u_{I3} + \frac{1}{2}v_{I3}^2)\right] &= (\dot{Q}_s)_{I3} + (V\dot{q}_v)_{I3} \\ &+ W_{J2}\left[u_{I2} + \frac{P_{I2}}{\rho_{I2}} + \frac{1}{2}v_{I2}^2 + g(Z_{J2} - Z_{I3})\right] \\ &- W_{J3}\left[u_{I3} + \frac{P_{I3}}{\rho_{I3}} + \frac{1}{2}v_{I3}^2 + g(Z_{J3} - Z_{I3})\right] \\ &- W_{J4}\left[u_{I3} + \frac{P_{I3}}{\rho_{I3}} + \frac{1}{2}v_{I3}^2 + g(Z_{J4} - Z_{I3})\right]\end{aligned} \qquad (III-7)$$

where

$$v_{I2} = \frac{W_{I2}}{\rho_{I2}A_{I2}}, v_{I3} = \frac{W_{I3}}{\rho_{I3}A_{I3}} \qquad (III-8)$$

and

$$W_{I2} = \frac{W_{J1} + W_{J2}}{2}, W_{I3} = \frac{W_{J2} + W_{J3} + W_{J4}}{2}. \qquad (III-9)$$

The formulation described here is identical to upwind differencing the energy equation. Such differencing is generally first order accurate. TOPAZ provides the user with the option of selecting pure upwind, pure centered, or a combination of the two, thus increasing the spatial differencing accuracy to as high as second order. Presently this option is only available for sections of the flow network having single flow streams of uniform cross-sectional flow area.

## 4. Fluid Momentum Equation

Conservation of momentum for an arbitrary stream tube (momentum node) J such as the one bounded by lines connecting points 2, 3, 4, 5, 8, 9, 10, 11, 2 in Figure 1 can be expressed as:

$$\begin{aligned} I_J \frac{dW_J}{dt} &= (P_I - P_{I+1}) + \left( \frac{W_I^2}{\rho_I A_I^2} - \frac{W_{I+1}^2}{\rho_{I+1} A_{I+1}^2} \right) \\ &- \frac{f_I L_I W_I \mid W_I \mid}{4 D_I \rho_I A_I^2} - \frac{f_{I+1} L_{I+1} W_{I+1} \mid W_{I+1} \mid}{4 D_{I+1} \rho_{I+1} A_{I+1}^2} \\ &+ g[\rho_I (Z_I - Z_J) - \rho_{I+1}(Z_{I+1} - Z_J)] \\ &- \frac{K_J W_J \mid W_J \mid}{2 \rho_J A_J^2} \end{aligned} \qquad (III-10)$$

where $f_I, L_I, D_I$ are the Moody friction factor, control volume length, and control volume diameter, respectively. The symbol $I_J$ is defined as

$$I_J = \frac{L_I}{2A_I} + \frac{L_{I+1}}{2A_{I+1}}. \qquad (III-11)$$

The term on the left hand side is the inertia associated with the fluid occupying the stream tube J. The first bracketed term on the right hand side represents the net pressure acting on the entrance and exit of the stream tube. The second bracketed term represents the net momentum crossing the stream tube boundaries. The third and fourth terms represent the Moody tube wall friction force acting on the fluid. The fifth term in brackets accounts for gravity forces on the fluid. The last term on the right hand side accounts for multi-dimensional flow effects associated with a change in flow direction (tube bends) and/or sudden area expansions. The symbol $K_J$ is a form loss function based on the flow area $A_J$. $A_J$ is usually taken to be the smaller of areas $A_I$ and $A_{I+1}$. Further discussion of this function will be presented in the next section.

## 5. Constitutive and Property Relationships

It is sometimes useful to associate a particular conservation equation with a particular fluid variable in order to determine the additional constitutive and property relationships required to complete the model. With this in mind one can think of the continuity equation (III-4) as an equation for pressure, P, the energy equation (III-5) as an equation for internal energy, u, and the momentum equation (III-10) as an equation for mass flow rate, W. Having computed two independent thermodynamic properties, namely P and u, the complete local state of the fluid is specified. The remaining properties required by the TOPAZ fluid model may then be calculated directly from known state and property relations. Some state variables such as $\rho, \left(\frac{\partial\rho}{\partial P}\right)_u$, and $\left(\frac{\partial\rho}{\partial u}\right)_P$ appear directly in the conservation equations, while others such as $C_v, C_p, \mu$, k, and c appear indirectly in correlations for constitutive models. The property relationships required in the TOPAZ fluid model may be summarized as follows:

$$Density\ \ \rho = f_1(P,u) \qquad (III-12)$$

$$\left(\frac{\partial\rho}{\partial P}\right)_u = f_2(P,u) \qquad (III-13)$$

$$\left(\frac{\partial\rho}{\partial P}\right)_P = f_3(P,u) \qquad (III-14)$$

$$Temperature\ T = f_4(P,u) \qquad (III-15)$$

$$Enthalpy\ h = f_5(P,u) \qquad (III-16)$$

$$Specific\ heats\ C_v = f_6(P,u) \qquad (III-17)$$

$$C_p = f_7(P,u) \qquad (III-18)$$

$$Viscosity\ \mu = f_8(P,u) \qquad (III-19)$$

$$Thermal\ Conductivity\ k = f_9(P,u) \qquad (III-20)$$

$$Isentropic\ sound\ speed\ c = \left(\frac{\partial P}{\partial\rho}\right)_s^{\frac{1}{2}} = f_{10}(P,u) \qquad (III-21)$$

As previously mentioned the functions (III-12) through (III-21) for air, the hydrogen and helium isotopes, molten salt, and liquid sodium have been "hardwired" into TOPAZ property subroutines. The user may select any of these property sets using an integer flag as part of the input data.

Real gas equation-of-state models for the hydrogen and helium isotopes were obtained from the Abel-Noble viral expansion of the van der Waals equation. These equation-of-state models are discussed in reference [18].

New fluid property sets (including two phase fluids) can be easily added as the need arises. Often property data is only available as a function of $P$ and $T$ rather than $P$ and $u$. Direct incorporation of such data can easily be accomplished providing the relationship $T$=f($P$,$u$) is known.

The additional relationships required to complete the TOPAZ fluid model are $\dot{Q}_s, \dot{q}_v, f$, and $K_J$. $\dot{Q}_s$, the rate at which thermal energy is transferred from the tube or vessel walls to the fluid, is obtained from Newton's Law of Cooling:

$$\dot{Q}_s = A_s h(T_K - T_I) \qquad (III-22)$$

where $A_s, h, T_K$, and $T_I$ are the fluid/wall interface area, heat transfer coefficient, local wall temperature, and local fluid temperature respectively. Numerous models for $h$ are available to the TOPAZ user including the following:

1. Adiabatic flow, $h$=0
2. Isothermal flow, $h \rightarrow \infty$
3. Forced convection tube flow, h = f($Re, Pr$)
4. Vessel heat transfer, $h$ = f ($R$a)
5. User supplied functions for h.

For those flows requiring internal heat generation the user may supply a constant value for $\dot{q}_v$ (default is $\dot{q}_v = 0$) or an arbitrary function for $\dot{q}_v$ in terms of time, space, or system variables.

The friction factor used in fluid modeling may be a constant specified by the user or the familiar Moody friction model [7]:

$$f = f(Re, \epsilon/D) \qquad (III-23)$$

where $\epsilon/D$ is the tube wall relative roughness. Additional friction factor models can be easily added to TOPAZ as the need arises.

The final constitutive model required to model fluid flow is the form loss function $K_J$. The form loss function is intended to account for a multitude of local effects including reversible multidimensional effects as well as irreversible losses associated with friction in expanding or turning flows. The precise form of the form loss term in the momentum equation (last item in Equation III-10), though familiar, is somewhat arbitrary. Other forms of this loss term can be easily incorporated into the code, providing such forms have been experimentally verified.

With the loss term in its present form, the user may specify a value for $K_J$ which is either a constant, an arbitrary function, or one of the TOPAZ default functions. Reference [10] contains a number of $K_J$ correlations for incompressible and low Mach number flows. Two of these correlations, i.e., abrupt expansion and abrupt contraction are built into the TOPAZ code.

If a completely reversible flow is being modeled at an area change, the user is still required to specify a finite value for $K_J$ in order to account for reversible acceleration and deceleration effects. Reference [3] shows that for incompressible and low Mach number flows ($Ma < .3$), the following equation is suitable for describing reversible expansion and contraction:

$$K_J = \begin{cases} -A_J^2\left(\dfrac{1}{A_{I+1}^2} - \dfrac{1}{A_I^2}\right) W_J > 0 \\ \\ +A_J^2\left(\dfrac{1}{A_{I+1}^2} - \dfrac{1}{A_I^2}\right) W_J < 0 \end{cases} \qquad (III-24)$$

The subscripts J, I, and I+1 refer to orientation in Figure 1. Equation (III-24) is built into the TOPAZ code as a user option.

The final constitutive model to be discussed here is the choked flow model, applicable to compressible flows. When the mass flow rate calculated from equation (III-10) exceeds local choked flow velocity, the momentum equation is no longer valid. This represents the onset of choked flow. When this occurs, the momentum equation is replaced by the following equation

$$W_J = \rho_J A_J v_c \qquad (III-25)$$

where $v_c$ is the local choking velocity. When the flow becomes choked, the mass flow rate, $W_J$, continues to be calculated by equation (III-25) until the right hand side of equation (III-10) changes sign thus indicating a deceleration of the flow below the choked value. Such an event signals the onset of unchoked flow and $W_J$ is again calculated from the momentum equation (III-10).

This modeling exhibits all the quantitative as well as qualitative characteristics of choked flow. In a subsequent report it will be demonstrated that when the local (upwind) isentropic sound speed, c, is used in place of $v_c$ in equation (III-25), choking will occur when the local pressure ratio exceeds the critical pressure ratio for the gas. Furthermore a decrease in the downstream pressure will have no influence on the local mass flow rate. This, of course, is classical choked flow behavior, (e.g. reference [11]).

The user may supply arbitrary functions for $v_c$ in order to implement specialized choked flow models. This is frequently done in the simulation of LOCA's. The isentropic model $v_c = c = \left(\frac{\partial P}{\partial \rho}\right)_s^{\frac{1}{2}}$ is the default model for choked flow in TOPAZ.

## IV. Fluid Containment Heat Conduction Equations

TOPAZ permits the user to couple fluid flow calculations with containment heat transfer calculations. The term containment refers to the solid structure which surrounds the fluid, i.e., the tube or vessel walls. Containment heat transfer calculations fall into three groups:

1. Interior two-dimensional, transient heat conduction within the containment
2. Convective heat exchange between the flowing fluid and the containment
3. Convective and radiative heat exchange between the ambient and the containment.

The second and third groups can be thought of as methods for implementing boundary conditions for the containment heat conduction calculation (group 1). TOPAZ could in fact be used solely as a heat conduction code for two-dimensional, transient heat conduction calculations. This can be accomplished by including only solid heat conduction nodes as part of the computational mesh and then supplying time dependent boundary conditions as required.

Two-dimensional, transient heat conduction is usually sufficient to describe most fluid flow-heat transfer situations of interest to TOPAZ users. TOPAZ can be reprogrammed for fully three-dimensional, transient heat conduction analysis but it was felt that the additional generality would seldom be used and computer storage and execution time could be substantially increased. This is due partially to the fact that the equations used to describe heat conduction are solved with exactly the same numerical technique as the fluid flow equations, i.e., each conduction node adds an additional differential equation to the TOPAZ model. Since the solution technique is fully implicit, any attempt to model a complex three-dimensional, transient heat conduction problem could completely consume the available computer storage. Such highly general heat conduction problems are solved more efficiently using specialized heat conduction codes such as TACO [12] or SAHARA [13]. However, these codes are not designed to couple heat conduction calculations to thermal-hydraulic calculations for flowing streams.

Typical TOPAZ heat conduction meshes are shown in Figures 3 thru 5. Figure 3 depicts an interior heat conduction control volume (node K) surrounded by four other control volumes (nodes K1, K2, K3, K4). A simple heat balance on node K, the shaded control volume, may be written as

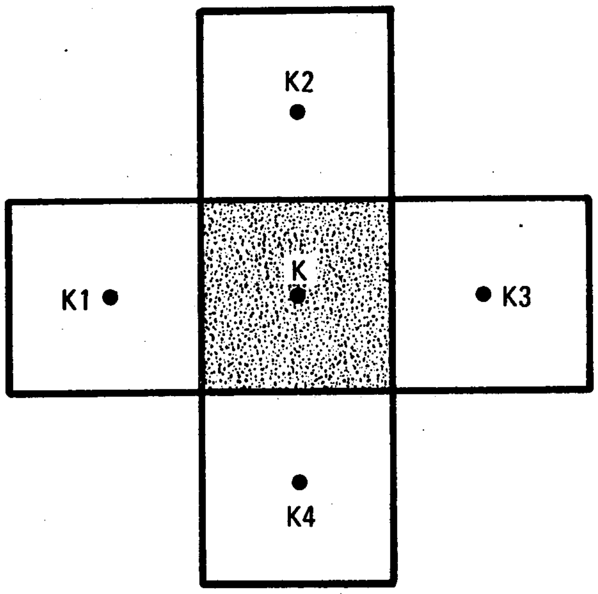


Figure 3. TOPAZ Interior Solid Heat Conduction Mesh

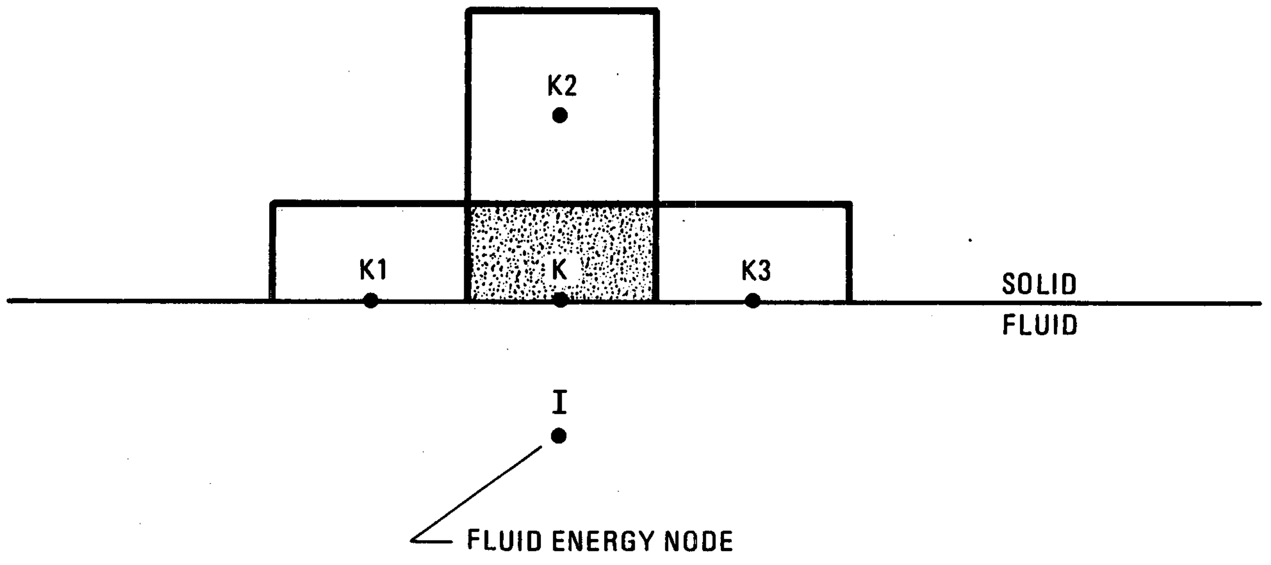


Figure 4. TOPAZ Fluid/Solid Heat Conduction Mesh

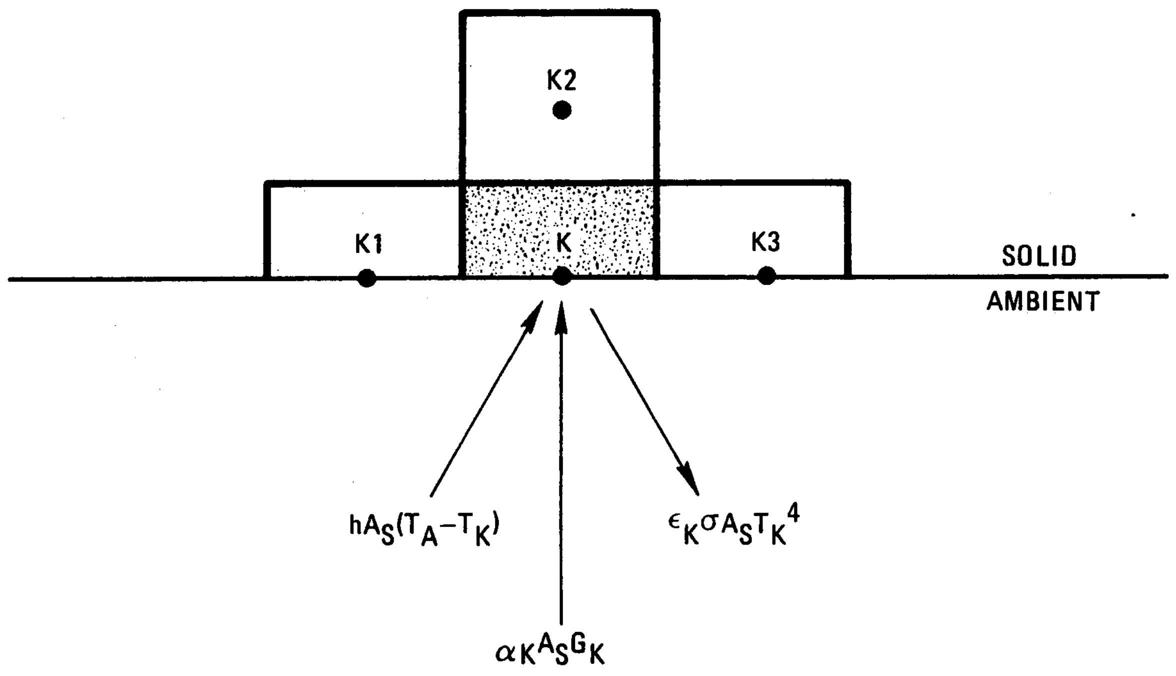


Figure 5. TOPAZ Ambient/Solid Heat Conduction Mesh

$$V_K \rho_K C_K \frac{dT_K}{dt} = \sum_{i=1}^{4} \left(\frac{A}{L}\right)_i \bar{k}(T_{Ki} - T_K) + \dot{q}_v V_K. \qquad (IV-1)$$

where $V_K, \rho_K, C_K, T_K, \dot{q}_v$ are the volume, density, heat capacity, temperature, and heat generation rate per unit volume associated with control volume K. The user may specify either a constant value for $\dot{q}_v$ (default is $\dot{q}_v = 0$) or any arbitrary function, i.e., $\dot{q}_v$ may be represented as a function of time, space, or any dependent variable in the model. The quantity $\bar{k}$ is the average thermal conductivity over the heat flow path from node $K_i$ to node $K$. It is approximated by evaluating the local material conductivity at a temperature equal to $(T_{Ki} + T_K)/2$. The symbol $(\frac{A}{L})_i$ is a characteristic length for conduction heat transfer between node $K_i$ and node $K$. In simple rectangular coordinate systems such as that depicted in Figure 3, $(\frac{A}{L})_i$ is equal to the heat transfer area divided by the length of the heat flow path from node $K_i$ to node $K$. Appropriate values for $(\frac{A}{L})_i$ may be defined for other coordinate systems such as spherical or cylindrical.

As indicated previously, TOPAZ has the capacity of modeling heat conduction at solid/fluid and solid/ambient interfaces. Hence it is possible to couple heat conduction calculations to the fluid flow and ambient boundary conditions. Figure 4 illustrates a typical surface heat conduction control volume (node $K$) in contact with a flowing fluid having a local temperature $T_I$. The energy balance for the shaded control volume (node $K$) may be expressed as

$$V_K \rho_K C_K \frac{dT_K}{dt} = \sum_{i=1}^{3} \left(\frac{A}{L}\right)_i \bar{k}(T_{Ki} - T_K) - hA_s(T_K - T_I) + \dot{q}_v V_K \qquad (IV-2)$$

where $V_K, \rho_K C_K, T_K, \dot{q}_v, \bar{k}$ and $(\frac{A}{L})_i$ are defined as they were for Equation (IV-1). $A_s$ is the heat transfer surface area separating the solid heat conduction control volume $K$ and the fluid energy control volume I. The symbol $h$ is the local convective heat transfer coefficient. Note that $hA_s(T_K - T_I)$ is identical to $\dot{Q}_s$ in the fluid energy equation (III-5), thus indicating that all the heat leaving the exterior surface of the containment is absorbed by the fluid.

Figure 5 illustrates a typical heat conduction control volume (shaded node $K$) exposed to an ambient convective and radiative boundary. This heat conduction model is particularly useful in modeling surfaces exposed to incident thermal radiation such as those encountered in the analysis of solar central receivers [1, 2]. A heat balance for the shaded node $K$ takes the form:

$$V_K \rho_K C_K \frac{dT_K}{dt} = \sum_{i=1}^{3} \left(\frac{A}{L}\right)_i \bar{k}(T_{Ki} - T_K) + hA_s(T_A - T_K) + \dot{q}_v V_K - \epsilon_K \sigma A_s T_K^4 + \alpha_K A_s G_K \qquad (IV-3)$$

where $\epsilon_K$, $\alpha_K$, and $G_K$ are the local emissivity, absorptivity, and net rate of incident radiation per unit area. The user may specify constants or arbitrary functions for $\epsilon_K$ and $G_K$. $\alpha_K$ is a user supplied constant.

This completes the presentation of equations for heat transfer modeling in the fluid containment. The modularity of the TOPAZ code permits the addition of new conduction and convective boundary condition models as the need arises.

## V. Generating a TOPAZ Model

### 1. The TOPAZ Computational Mesh

In the previous two sections, six different model equations were introduced, namely:

1. Fluid continuity equation
2. Fluid energy equation
3. Fluid momentum equation
4. Interior solid heat conduction equation
5. Fluid/solid heat transfer equation
6. Ambient/solid heat transfer equation.

The first three are devoted to modeling the thermal-hydraulics of flowing fluids, while the latter three model thermal energy transport in the fluid containment.

In discussing the TOPAZ computational mesh, it is helpful to introduce the concept of "nodes". Each node represents a position in space where a single dependent variable (P, u, W, or T) is associated with a single model equation. Hence there are continuity nodes associated with P, energy nodes associated with u, momentum nodes associated with W, and heat conduction nodes associated with T.

A typical fluid control volume has two nodes at its center, a continuity node and an energy node. Because of the previously described displaced mesh, momentum nodes are located at the control volume interfaces or stream tubes which are displaced from the connecting continuity/energy node pairs.

In addition to the six node types described in the previous section, TOPAZ also contains additional node types to describe frequently encountered flow boundary conditions (e.g. constant pressure, constant internal energy, constant mass flow rate, etc.) and an arbitrary node type which enables users to add arbitrary algebraic or differential equations to the model. By linking nodes together, it is possible for a user to generate a computational model for any tube-vessel flow geometry, regardless of complexity. Of course the ultimate success of such modeling is subject to the assumptions stated in Section II and the accuracy of any correlations required to reduce multi-dimensional transient flow behavior down to one-dimensional, transient flow behavior. Computer storage may also place practical limitations on the solution of the TOPAZ model equations.

In arranging nodes to form a computational mesh, the following simple rules must be followed in order to insure that the model will make physical sense:

1. Fluid continuity and energy nodes occur in pairs. If node I is a continuity node then I + 1 must be an energy node at the same axial location in the flow.

2. Pairs of fluid continuity and energy nodes may be connected to any number of momentum nodes (hence branching of flows) and any number of fluid/solid heat conduction nodes.

3. Fluid/solid heat conduction nodes which are connected to adjacent fluid continuity-energy node pairs must be physically located at the same axial location as the fluid continuity-energy node pair.

4. All fluid continuity-energy node pairs must be separated by one and only one fluid momentum node.

Application of these rules for generating a mesh can best be illustrated through an example. Suppose one is interested in modeling the flow of a high pressure compressible fluid from one spherical vessel to another. The two vessels are connected by a single tube and the user wishes to predict transient temperature excursions in the heat containment (two vessels and the tube) as well as the fluid property variations as a function of space and time. Figure 6 illustrates how one might connect nodes together to build a TOPAZ model. The figure indicates that one continuity-energy node pair has been designated for each vessel and five equally spaced node pairs have been connected together to form the tube. Such a mesh is equivalent to modeling fluid flow in the tube using an upwind finite differencing scheme with five computational cells. The distribution of heat conduction nodes in the tube wall indicates that radial and axial temperature variations are being resolved (3 nodes in the radial direction and 5 nodes in the axial direction) but circumferential variations are ignored, i.e. heat flow within the tube walls is assumed axisymmetric. The distribution of heat conduction nodes in the vessel walls indicates that only radial temperature gradients are being resolved, i.e. heat flow within the vessel walls is assumed to be spherically symmetric.

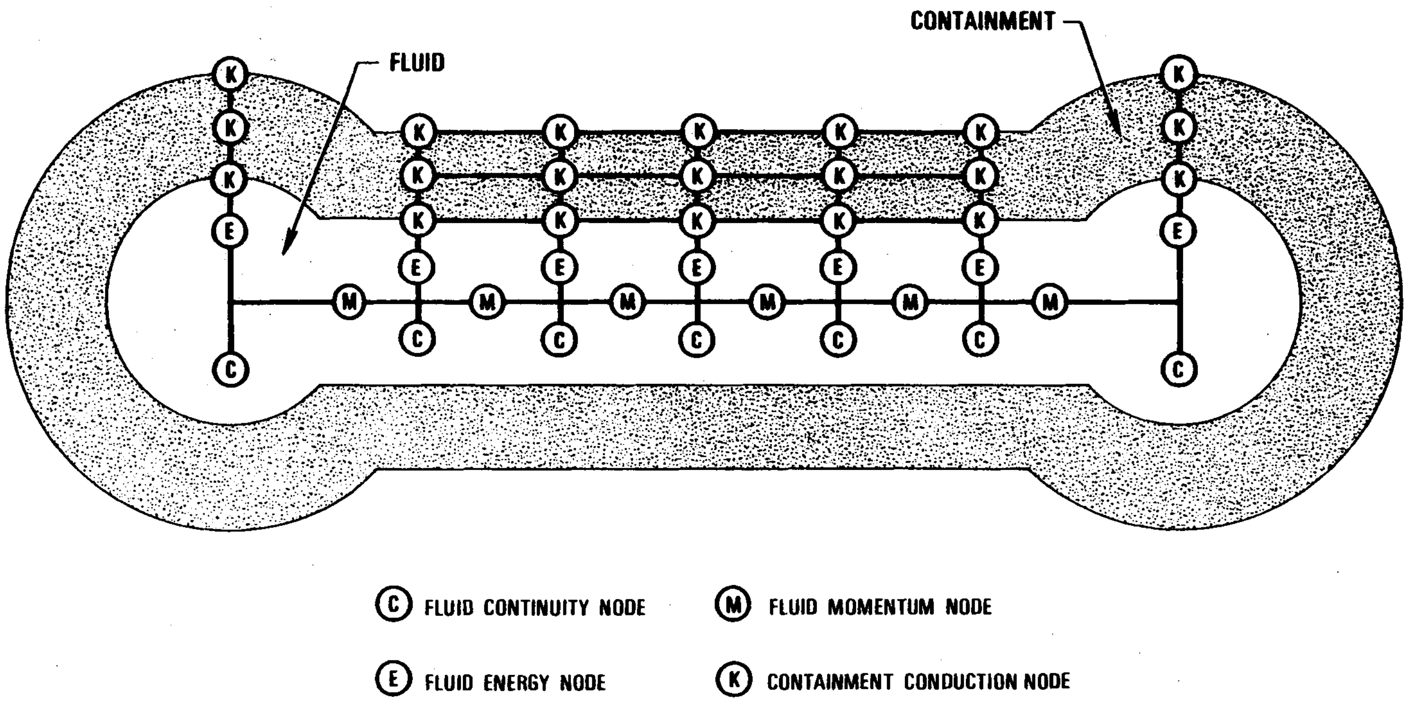


Figure 6. TOPAZ Mesh for Example 1:
Flow Between 2 Vessels

Reference [2] illustrates how radial and circumferential variations in tube wall temperature may be resolved. Such variations are important in the modeling of solar central receivers where thermal boundary conditions are not axisymmetric.

In problems where the heat capacity and thermal diffusivity in the heat containment are large, one might be justified in assigning a constant temperature to tube and vessel walls. Such modeling would completely eliminate the need for heat conduction nodes in the mesh. For the simple example shown in Figure 6, the number of nodes in the TOPAZ mesh would be reduced from 41 to 20. The convective heat transfer between the containment walls and the fluid would then be calculated with a user assigned wall temperature.

Figure 7 illustrates a computational mesh having a flow branch. High pressure gas is transfered from vessel A down tube 1 to a junction where flow divides into tubes 2 and 3. The flow in tubes 2 and 3 terminates in vessels B and C respectively. In this example, three control volumes are used for each tube and the tube walls maintain a user assigned temperature. Note that a single continuity-energy node pair has been designated to represent each vessel and the branch volume.

As with any finite difference technique, the number of nodes employed in the TOPAZ model has a direct bearing on the accuracy of the solution, i.e. the finer the mesh the more accurate the solution. Typically one would obtain TOPAZ solutions with varying numbers of nodes until the addition of further nodes produced negligible changes in the calculated results.

Frequently in compressible flow modeling one must be concerned with supplying enough nodes to adequately resolve steep pressure gradients. As an example, a rapidly opening valve in a piping system carrying a compressible fluid could cause a steep pressure wave to propagate through the flow domain. Since virtually all of the one-dimensional, transient compressible flow physics are embodied in the TOPAZ fluid flow equations, TOPAZ will attempt to resolve and track these waves as they occur in the flow. Success of the numerical method requires that any gradient in the flow be spread over more than one node or control volume. Hence the propagation of a steep front could bring calculations to a halt if the number of axial control volumes is insufficient. Such restrictions are nonexistant in incompressible fluid flow or solid heat conduction modeling.

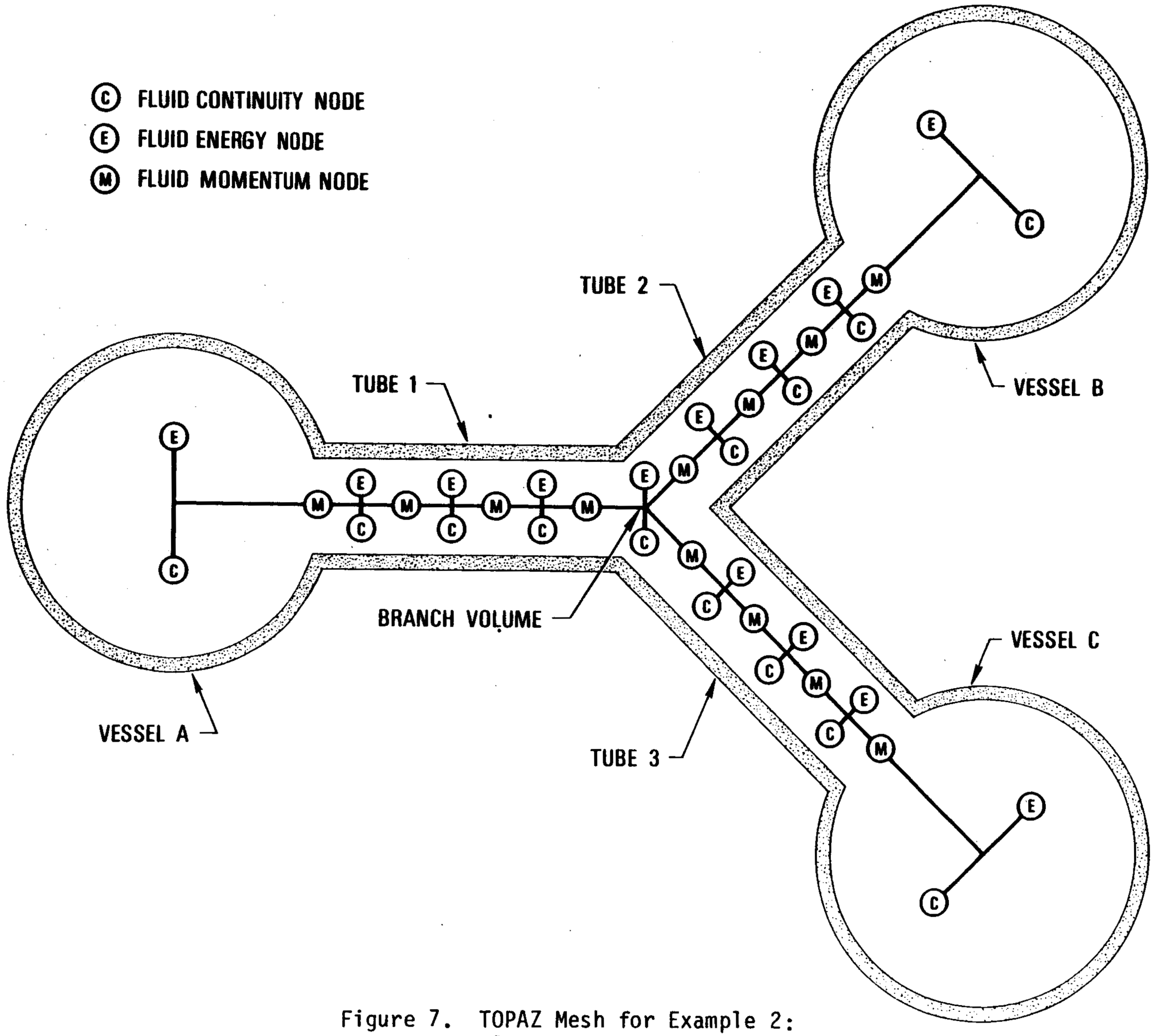


Figure 7. TOPAZ Mesh for Example 2:
Flow Between 3 Vessels

## 2. Generating the Computational Mesh

In the TOPAZ computer code the computational mesh for a model is uniquely defined by elements in the two-dimensional array DATA (N, M) where N is the number assigned to a particular node in the mesh and M is the storage location containing data about the node. The maximum size of the DATA array is DATA(NMAX, 20) where NMAX is equal to the number of nodes in the mesh. The total amount of data associated with a particular node, N, is DATA (N, 1), DATA (N, 2), ... DATA (N, 20). The data stored in these array elements includes such information as:

1. Node type - continuity, energy, momentum, conduction, etc.
2. Material type - stainless steel, $H_2$, air, liquid sodium etc.
3. Mesh connectivity information - number and orientation of each and every node connected to node N.
4. Physical dimensions - diameter, control volume length, elevation from the reference height, etc.
5. Constitutive model information - type of heat transfer, friction factor, and form loss correlations, etc.
6. Printout information - identifies which property to print out at a particular node during major edits (print out of variables at all nodes) or minor edits (print out of variables at 5 user selected nodes).

Generally it is not necessary to fill all elements of the DATA array. Typically the elements DATA (N, 10), DATA (N, 11), ..., DATA (N, 20) remain undefined. One exception to this would be the special case where a fluid continuity-energy node pair would have the maximum 8 momentum nodes connected to it.

Since TOPAZ does not currently possess a totally general user friendly interface, the user is responsible for filling the DATA array for his particular problem. This is usually done in the TOPAZ driver program. Needless to say this could be an extremely tedious process if each element in the DATA array was read in individually as data. A reasonable alternative to this dilemma is for the user to write an "automated" driver program which can act as a mesh generator for filling the DATA array. Through appropriate use of "DO LOOPS", the DATA array can be generated using a relatively small amount of coding.

Although TOPAZ is an extremely general code, the following two types of problems are frequently modelled:

1. Fluid flow with no heat conduction in containment and negligible elevation changes (i.e., the example of Figure 7).

2. Fluid flow with axisymmetric heat conduction in the containment and negligible elevation changes (i.e., the example of Figure 6).

A special set of mesh generating subroutines called PARTS has been written for creating the DATA array for such problems. The user may call specific subroutines for specific component types such as pipes, branch volumes, and vessels. A user may, for example, wish to generate a mesh for a simple constant diameter pipe. The number of individual pieces of data required to fill the DATA array for this ensemble of nodes could be as small as a few hundred, and as large as tens of thousands, depending on the number of fluid control volumes (fluid continuity-energy node pairs) and the number of radial tube wall heat conduction nodes desired by the user. It should be noted, however, that much of the information placed in the DATA array is highly repetitive. Consider the fluid continuity nodes for example. Since the pipe has constant physical dimensions and no elevation changes throughout the flow, all DATA elements of all continuity nodes are virtually identical, except for information relating to nodal connectivity (i.e. the first continuity node in the pipe has different momentum nodes connected to it than say the tenth).

This repetitive characteristic makes it fairly easy to write mesh generation subprograms such as PARTS. With such mesh generation subprograms, the problem of generating a TOPAZ mesh becomes almost trivial. The entire mesh for the example in Figure 6 could be generated through a total of three subroutine calls to the PARTS subprogram, i.e., two calls to a vessel subroutine and one call to a pipe subroutine. The example in Figure 7 could be generated through a total of seven calls, i.e., three calls to the vessel subroutine, three calls to the pipe subroutine, and one call to the branch subroutine.

Complete TOPAZ "user friendly" interface programs can be easily written for specific classes of problems (i.e., fixed flow geometry and boundary conditions). Such interfaces can successfully insulate the user from burden of generating the computational mesh or writting subprograms for boundary conditons. The TOPAZ driver program, DRAC, (Dynamic Receiver Analysis Code), [2] is an example of such an interface. The user need only read in approximately thirty pieces of data through NAMELIST input, and DRAC will generate a computational mesh, specify boundary conditons, and direct the integration of TOPAZ equations for a fairly complicated transient combined heat conduction - fluid flow problem.

In the future, it is expected that other "user friendly" interface codes will be written for the purpose of modeling heat conduction - fluid flow situations of frequent interest.

## VI. Solution of the TOPAZ Equations

The TOPAZ model for a particular problem consists of NMAX mixed algebraic and ordinary differential equations, and NMAX unknowns (one equation and one unknown per computational node). The nodes which make up the model represent locations where boundary conditions, fluid conservation equations (continuity, energy, and momentum), and containment heat conduction equations are applied.

Once the model has been formulated (i.e. formation of the DATA array) the solution of the model equations may then proceed. In TOPAZ these equations are solved using DASSL, a code written by L. R. Petzold [6] of SNLL. DASSL is a family of subroutines (DASTEP, DASSLRT, NJAC, SOLVE, and other library subroutine packages such as LINPACK [14]) designed to perform the fully implicit numerical solution of systems of differential/algebraic equations. No attempt will be made here to outline the details of how DASSL solves the equations. For these details, the reader is referred to reference [6]. However, a few comments will be made to describe the overall solution procedure, and to outline the reasons why DASSL appears to be well suited for solving the TOPAZ model equations.

In order for the DASSL solution to proceed, all the nodal equations, including those presented in Sections III and IV must be recast into the following form:

$$F(t, y, y') = 0 \qquad (VI-1)$$

$$y(t_o) = y_o \qquad (VI-2)$$

$$y'(t_o) = y'_o \qquad (VI-3)$$

where $y' = dy/dt$ and $F, y, y'$ are one-dimensional vectors of length NMAX. The variable $y$ is intended to represent the vector of dependent variables in the TOPAZ model. Physical quantities represented by $y$ include the following:

| | |
|---|---|
| $y = P$ | at fluid continuity nodes |
| $y = u$ | at fluid energy nodes |
| $y = W$ | at fluid momentum nodes |
| $y = T$ | at containment heat conduction nodes |

Equations (VI-2), (VI-3) represent the initial conditions for the TOPAZ model. TOPAZ calculations are usually started from a consistent but trivial set of initial conditions in which all fluids are assumed at rest at $t = t_o$. For many problems such as those in which fluid flow is initiated by opening a valve or starting a pump at $t = t_o$, these initial conditions are realistic. In other problems where the user is interested in describing transient excursions from some steady flow condition, the fluid is first accelerated from rest to the desired steady flow initial conditions. This procedure eliminates the often difficult task of solving the steady state equations to obtain a consistent set of steady flow initial conditions. It has been employed successfully to produce the initial conditions for the DRAC [3] problem.

Equation (VI-1) represents the TOPAZ model equations cast into a special form which will be refered to here as the "residual form". Most popular codes for solving ordinary differential equations require the system equations be cast into the "standard form"

$$y' = f(t,y) \qquad (VI-4)$$

$$y(t_o) = y_o. \qquad (VI-5)$$

Equation (VI-1) simply represents a recasting of equation (VI-4) such that

$$F(t,y,y') = y' - f(t,y) = 0. \qquad (VI-6)$$

For many systems of equations it is difficult or even impossible to solve each equation for a single $y'$. Hence representing the equations in the standard form (VI-4), may not be possible. The DASSL code is not encumbered by this requirement. This is a significant advantage particularly in light of the fact that the TOPAZ fluid continuity Equation (III-4) cannot be cast into the standard form.

In order to solve the system of equations, DASSL replaces the time derivative in (VI-1) by a difference approximation and then solves the resulting equations at the current time, $t_n$, using implicit iterations. If a first order difference in time is employed to approximate $y'$, Equation (VI-1) becomes

$$F\left(t_n, y_n, \frac{y_n - y_{n-1}}{t_n - t_{n-1}}\right) = 0 \qquad (VI-7)$$

where n represents the time step number. The system of ODE's given in (VI-7) have, in effect, been approximated by a system of nonlinear algebraic equations where $y_n$ represents the vector of unknowns at time step n.

DASSL does not restrict itself to approximating $y'$ using only first order approximations. Instead DASSL approximates the derivative $y'$ using the $k^{th}$ order backward differentiation formula where $k$ may range from one to five. The order $k$ and the time stepsize $\Delta t_n = t_n - t_{n-1}$ is automatically chosen by DASSL at every time step based on the behavior of the solution.

The implicit iteration technique used to solve the system of nonlinear algebraic equations is Newton's Method [15]. Newton's Method, like any iteration scheme, requires an initial guess for the vector of unknowns. The method converges most rapidly when the initial guess is an accurate one. DASSL obtains an initial guess for $y_n$ by evaluating the polynomal which interpolates the computed solution at the last $k + 1$ times at the current time $t_n$. An initial guess for $y'_n$ is obtained by taking the time derivative of the interpolating polynomal and evaluating it at $t_n$.

DASSL provides for two options in computing the iteration matrix required in Newton's Method. The user may either provide an analytic Jacobian (matrix of partial derivatives $\partial F_i/\partial y_i$) or request that DASSL compute a numerical Jacobian using finite differences. Because of the complexity of most TOPAZ models the latter option is used exclusively.

DASSL automatically selects a time increment $\Delta t_n$ which satisfies a user specified set of error tolerances. The user specifies absolute and relative error tolerances either uniformly (i.e. one pair of tolerances for all variables) or as a vector of length NMAX (i.e., a different pair for each variable). DASSL automatically recomputes the numerical Jacobian with varying frequency as the solution evolves. The frequency of this calculation depends on the behavior of the solution.

The DASSL user may specify a problem as having either a "banded" or "dense" variable-equation connectivity matrix. (See reference [1] for an explanation of bandedness as it applies to TOPAZ-like flow problems.) Banded problems such as the one shown in the example of Figure 6, can be solved more economically since the number of calculations required to perform the numerical differencing is substantially reduced along with the amount of computer storage. The width of the band (and hence the cost of the calculations) for the problem in Figure 6 will increase as the number of radial heat conduction nodes is increased.

In problems with flow branching, the banded variable-equation connectivity typified by single stream flow problems quickly breaks down. When this happens one must resort to performing DASSL calculations with a "dense" iteration matrix. Such calculations require more storage and computer time. More often than not TOPAZ-DASSL calculations are performed with a dense iteration matrix.

In general the computer storage required by DASSL far exceeds the storage required by the TOPAZ code when a dense iteration matrix is used. Hence it is the storage required by DASSL which ultimately limits the size of any problem modeled by TOPAZ. Such a limitation is typical whenever equations are solved implicitly rather than explicity. Thus far, no TOPAZ simulation to date has required more than 1M words.

In compressible flow problems in which flows choke and unchoke with time, TOPAZ must utilize a special version of DASSL called DASSLRT. DASSLRT contains a root finding capability which permits TOPAZ to identify locations in the flow where choking and unchoking occur. Each time choking or unchoking occurs a "root" is found and DASSLRT returns control of the calculations to TOPAZ. When this occurs TOPAZ replaces the fluid momentum equation with the choked flow equation or vice-versa as required and restarts the calculation. This results in smooth, stable, and accurate transitions between choked and unchoked flows.

The fact that DASSL is an implicit solver has two significant advantages with regard to its application to the solution of the TOPAZ equations. First of all, any TOPAZ model of transient compressible flow is composed of a collection of stiff ODE's since time scales associated with propagating pressure waves in the flow are generally orders of magnitude smaller than time scales associated with the speed of the flow or propagation of thermal diffusion in the containment. An implicit solver such as DASSL is well suited to providing economical solutions for such stiff equation sets.

In addition to facilitating the solution to stiff equations the implicit solution proves to be extremely valuable in its application to TOPAZ because of the inherent difficulty associated with incorporating pressure and velocity boundary conditions in generalized pipe flow modeling. Consider, for example, an explicit upwind finite difference technique applied to a pipe flow where the inlet velocity (or pressure) at the upstream boundary is specified. Properties in the flow field at the new time step can be calculated explicitly by marching in space from the upstream boundary to the downstream boundary. However, if the calculated pressure at the downstream boundary does not match the specified downstream boundary pressure, the inlet pressure (or velocity) must be adjusted and the calculation repeated until the downstream boundary condition is satisfied. Such a flow problem is frequently refered to as a "shooting problem". In such problems, the explicit technique may be highly uneconomical. Not only do time steps have to be small (compared to implicit techniques) to provide stability, but numerous iterations at each time step are required in order to satisfy the shooting problem's boundary conditions. This difficulty is further compounded in compressible flows not only because of increased stiffness in the equations but also due to the existence of choke points in the flow field.

It is by no means impossible to apply explicit techniques to shooting problems. Clever iteration sequences can almost always be devised for a particular flow geometry. However, devising such techniques requires an apriori knowledge of the flow geometry. In a generalized code such as TOPAZ, it is virtually impossible to anticipate all flow geometries of interest to the user. It is highly likely that at some point in time a user will be interested in modeling a complicated collection of vessels, pipes, and multiple flow branches for which any pseudo generalized physically motivated iteration scheme will fail completely.

DASSL and TOPAZ avoid the difficulties associated with the shooting problem by including the boundary conditions as part of the iteration equations. Furthermore, since the iteration procedure is mathematically motivated (Newton's Method) rather than physically motivated, it is usually successful regardless of the number or arrangement of vessels, pipes, or flow branches.

## VII. TOPAZ Code Architecture

In this section a description of TOPAZ code architecture will be presented. A detailed presentation of the code architecture is beyond the scope of this document, hence only a very general overview will be given.

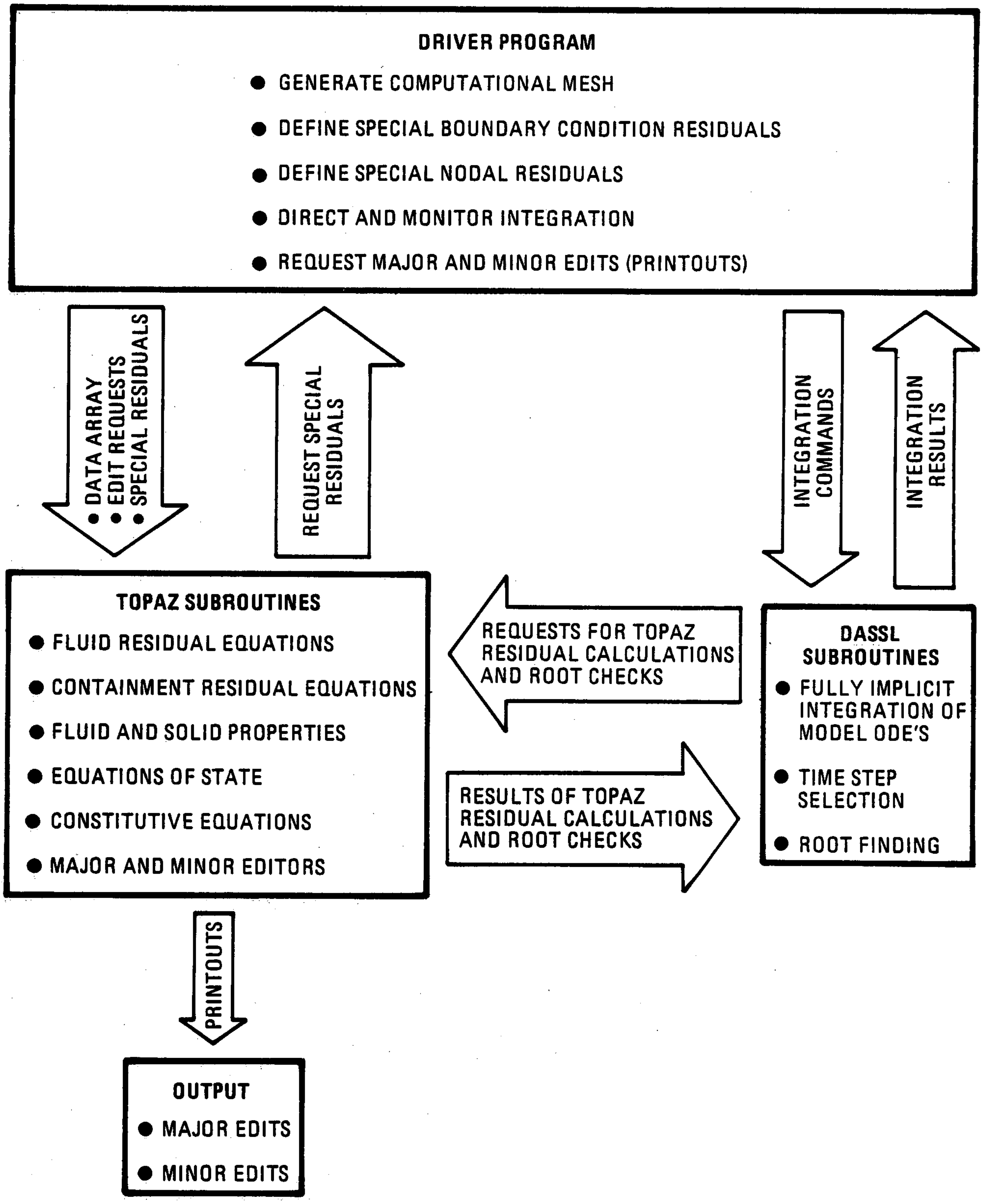
The purpose of this overview is three-fold:

- To show how the user interfaces with TOPAZ
- To illustrate the basic flow of information between major blocks of code
- To illustrate the modularity and generality of TOPAZ

TOPAZ code architecture is illustrated schematically in Figures 8 and 9. Figure 8 illustrates the relationship between the user supplied driver program, the TOPAZ subroutines, and the DASSL subroutines. Figure 9 focuses directly on the TOPAZ block of code, in order to illustrate the flow of information required to accomplish the major goal of calculating NMAX residuals i.e., the F vector defined in Equation (VI-1).

Let us first examine Figure 8. The user's major interaction with TOPAZ and DASSL takes place through the DRIVER program. If a "user friendly" driver program such as DRAC [2] is not available, the user must write his own. The major function of the driver program is to fill the DATA array and monitor the integration. As discussed in Section V the DATA array contains all the information necessary to describe uniquely the user's particular problem. If the user needs to calculate special residuals which are not currently built into TOPAZ, a subroutine (NODE 10) must be included in the DRIVER program for that purpose. These special residuals fall into two groups, namely, special boundary conditions, and special nodal equations (e.g. control equations, etc.). If a user wishes to include special functional relationships for form losses, heat transfer correlations, etc., which are not currently "hard wired" into TOPAZ, he must also include a subroutine (VALUE) as part of the driver program.

The driver program directs the integration by making repeated calls to DASSL. Driver programs are generally written so that the integration is periodically interupted for the purpose of printing out the solution as a function of time. These printouts may be short, i.e. "minor edits" in which dependent variables at five different nodes are printed, or comparatively long, i.e. "major edits" in which the dependent variables at all nodes are printed. The major and minor edit subroutines are part of the TOPAZ block of code.



DRIVER PROGRAM
● GENERATE COMPUTATIONAL MESH
● DEFINE SPECIAL BOUNDARY CONDITION RESIDUALS
● DEFINE SPECIAL NODAL RESIDUALS
● DIRECT AND MONITOR INTEGRATION
● REQUEST MAJOR AND MINOR EDITS (PRINTOUTS)
● DATA ARRAY
● EDIT REQUESTS
● SPECIAL RESIDUALS
REQUEST SPECIAL RESIDUALS
INTEGRATION COMMANDS
INTEGRATION RESULTS
TOPAZ SUBROUTINES
● FLUID RESIDUAL EQUATIONS
● CONTAINMENT RESIDUAL EQUATIONS
● FLUID AND SOLID PROPERTIES
● EQUATIONS OF STATE
● CONSTITUTIVE EQUATIONS
● MAJOR AND MINOR EDITORS
REQUESTS FOR TOPAZ RESIDUAL CALCULATIONS AND ROOT CHECKS
RESULTS OF TOPAZ RESIDUAL CALCULATIONS AND ROOT CHECKS
DASSL SUBROUTINES
● FULLY IMPLICIT INTEGRATION OF MODEL ODE'S
● TIME STEP SELECTION
● ROOT FINDING
PRINTOUTS
OUTPUT
● MAJOR EDITS
● MINOR EDITS

Figure 8. DRIVER/TOPAZ/DASSL Code Architecture

For compressible flow problems the integration may be interupted between print increments. This can happen whenever DASSL "finds a root", i.e. identifies a point in the flow where a transition occurs between choked and unchoked flow. The driver program must contain the necessary logic to restart the integration after substituting the appropriate residual at the choke point (i.e., equation III-10 or III-25).

The next major block of coding illustrated in Figure 8 is DASSL. DASSL receives integration commands from the driver program and returns the integrated results including any error messages. In order to integrate the equations DASSL must periodically evaluate the residuals associated with each and every computational node (i.e. F(N), N=1,2, ..., NMAX). This is accomplished through repeated calls to the RESIDS subroutine located in the TOPAZ block of coding (Figure 9).

The TOPAZ block of coding performs two functions. It prints out major and minor edits as requested by the driver program, and it calculates the residual vector F(N), N=1, 2, ..., NMAX as requested by DASSL. Within the TOPAZ block are subroutines for calculating residuals for each node type. These subroutines are named NODE1, NODE2, NODE3, etc. Other subroutines within the TOPAZ block include subroutines for various material properties, equations of state, and constitutive relationships.

Figure 9 illustrates the flow of information within TOPAZ required to calculate the current values of the residual vector F(N), N=1, 2, ..., NMAX. DASSL calls the RESIDS subroutine which cycles through the entire computational mesh for N=1, 2, ..., NMAX. The first step in calculating the residual at a particlar node, N, is to identify the node type. This information is stored in DATA (N, 1). Knowing the node type enables RESIDS to determine which subroutine (NODE1, NODE2, etc.) to call to determine F(N). Subroutines which make individual residual calculations frequently call on other TOPAZ utility subroutines for the purpose of calculating required fluid properties and constitutive relationships.

Because of the modularity in TOPAZ code architecture new node types, properties, equations of state, and constitutive relationships can be easily "hardwired" into the code as the need arises. This feature increases the codes generality and range of application.

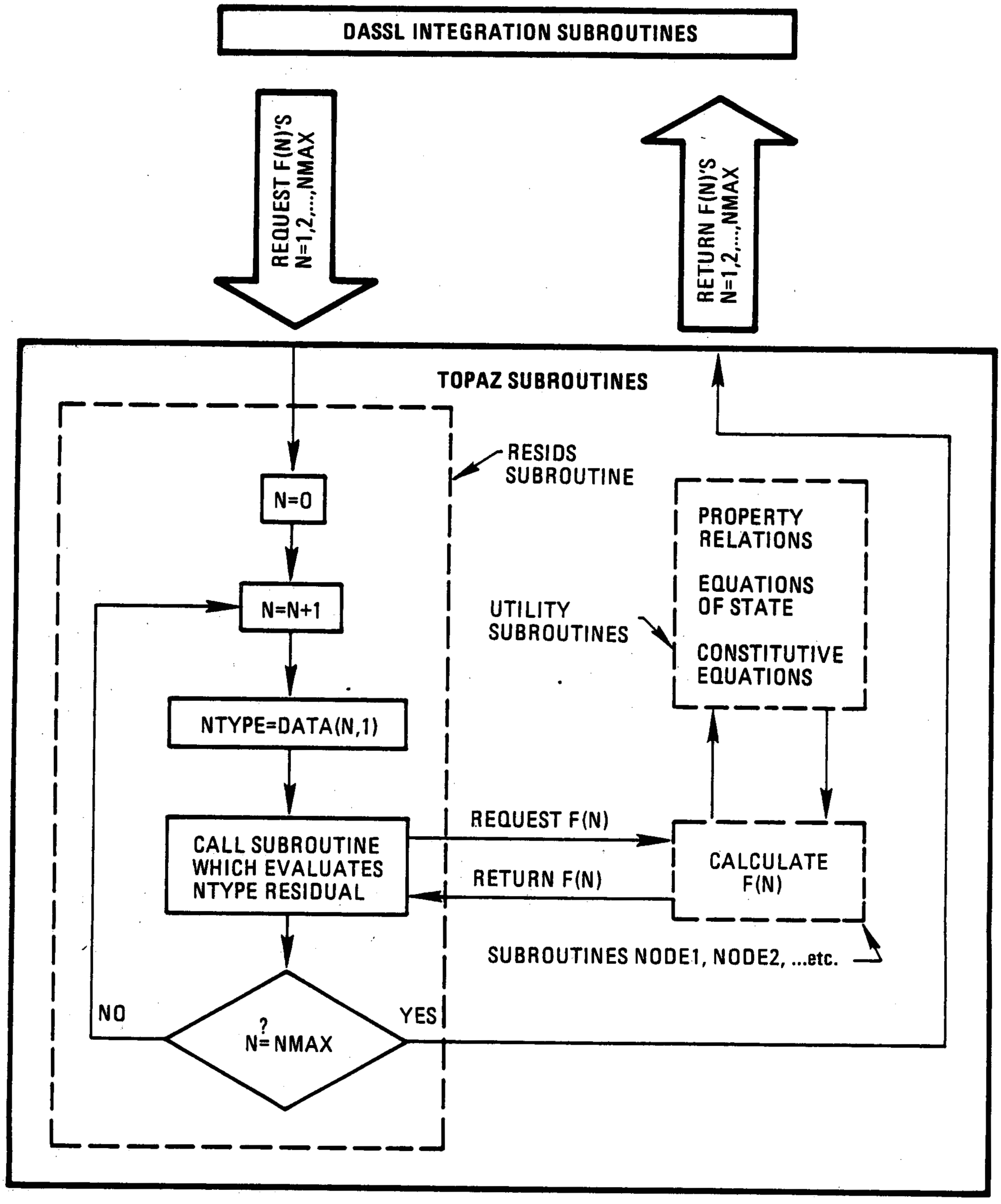


Figure 9. TOPAZ Code Architecture

## VIII. Future Activities

Future activities in the development of TOPAZ will focus on the following areas:

1. Detailed modeling and documentation of fluid-flow heat transfer configurations of specific interest

2. Validation of TOPAZ modeling techniques

3. Writing "user friendly" TOPAZ driver programs

4. Increasing the Generality of TOPAZ

### 1. Detailed Modeling and Documentation

TOPAZ is currently being used to model complex flow problems. These efforts are expected to continue since the code contains a number of features which make it uniquely suited to solving many Sandia problems (e.g. transient flow phenomena, flow branching, etc.). Significant experience in flow modeling is expected to be gained through individual modeling efforts. When appropriate, these efforts will be documented, particularly when experimental test data is available for comparison.

### 2. Validation of TOPAZ Modeling Techniques

In addition to validation through specific modeling efforts, a program of TOPAZ validation will be undertaken in order to evaluate generic multi-dimensional modeling techniques (flow branching, vessel heat transfer, form losses, etc.). In the final analysis, one's ability to successfully model with TOPAZ (or any one-dimensional code) is directly related to the accuracy of constitutive relationships (models for form losses, friction, heat transfer and other multi-dimensional phenomena). A number of currently used constitutive relationships for compressible flow form losses and vessel heat transfer are known to be inaccurate. In the future, TOPAZ validation and model improvement will be tightly coupled to an in-house experimental program aimed at improving form loss and heat transfer models.

A substantial amount of code verification will take place by comparing TOPAZ generated predictions to analytical solutions and other independent computer code predictions. Some verification has already taken place and is documented in reference [2]. Calculational comparisons between TOPAZ and the gas transfer codes TRIC [16] and DUCTFLO [17] will be documented in the near future.

## 3. User Friendly TOPAZ Driver Programs

The manpower required to write a user friendly interface capable of exploiting the complete generality of TOPAZ is prohibitive at this time. However, user friendly driver programs for specific problem types are relatively easy to generate. Such programs would completely insulate the user from the burden of generating a mesh or becoming involved with the details of directing DASSL integration. The user would communicate with the code though a simple concise set of input data without becoming involved in program details. DRAC [2] represents the first of these user friendly driver programs to be written to date. Other user friendly drivers are likely to be written in the near future for the purpose of modeling frequently encountered flow geometries such as those depicted in Figure 6 (compressible flow between two vessels along a single flow path) and in Figure 7 (compressible flow between three vessels via a flow branch).

## 4. Increasing the Generality of TOPAZ

In the early development of TOPAZ, modeling capabilities were directed toward solar central receiver applications [1, 2]. Such modeling was restricted to the flow of single-species, non-reacting, compressible and incompressible fluids. Many non-solar flow problems involve the transport of mixtures of gases in which concentration of species varies throughout the flow domain as a function of space and time. Including such a modeling capability in TOPAZ will require significant code modification. However, plans have been made to add this feature.

## IX. Conclusion

This report is the first in a series of reports documenting the computer code TOPAZ (Transient-One-Dimensional Pipe Flow Analyzer). In this report an overview of the code has been presented. TOPAZ models the one-dimensional transient flow of compressible and incompressible fluids through complex and arbitrary arrangements of pipes, valves, flow branches, and vessels. Heat transfer to and from the fluid containment structures is included in the modeling as well as two-dimensional transient heat conduction within the containment itself.

The purpose of this document was to discuss the TOPAZ modeling assumptions, basic equations, constitutive models, numerical integration techniques, and overall code architecture.

Although TOPAZ is a highly general code, a single "user friendly" interface capable of exploiting the code's complete generality does not presently exist. Application of the code requires that user interface programs be written for specific classes of problems (e.g. reference [2]). These interfaces generate the TOPAZ computational mesh and direct the fully implicit integration of the model equations. The user interface program serves as a driver for TOPAZ.

Future TOPAZ development will include additional code validation, writing of user interface programs and the addition of a gas species tracking capability.

## X. References

[1] W. S. Winters, "Dynamic Modeling of Solar Central Receivers", SAND81-8213, Sandia National Laboratories, Livermore, CA, July 1981.

[2] W. S. Winters, "DRAC - A User Friendly Computer Code for Modeling Transient Thermohydraulic Phenomena in Solar Receiver Tubing", SAND82-8744, Sandia National Laboratories, Livermore, CA, January 1983.

[3] R. V. Moore, et. al., "RETRAN - A Program for One-Dimensional, Transient Thermal-Hydraulic Analysis of Complex Fluid Flow Systems, Volume 1: Equations and Numerics", EPRI NP-407, Energy Incorporated, P.O. Box 736, Idaho Falls, Idaho 83401, January 1977.

[4] RELAP4/MOD5, A Computer Program for Transient Thermal-Hydraulic Analysis of Nuclear Reactors and Related Systems, Users' Manual, ANCR-NUREG-1355, Idaho National Engineering Laboratory, September 1976.

[5] V. H. Ransom, R. J. Wagner, et. a;., "RELAP5/MOD1 Code Manual Volume 1: System Models and Numerical Methods", NUREG/CR-1826, EG&G Idaho, Inc., Idaho Falls, Idaho 83415, March 1981.

[6] L. R. Petzold, "A Description of DASSL: A Differential/Algebraic System Solver", SAND82-8637, Sandia National Laboratories, Livermore, CA, September 1982.

[7] Moody, L. F., "Friction Factors for Pipe Flow", Trans. ASME, vol. 66, No. 8, p. 671, 1944.

[8] F. Kreith, Principles of Heat Transfer, Harper and Row, Publishers, New York, 1973.

[9] King-wah Lam, "Time Dependent Flow Establishment in Long Smooth-Walled Pipes", MS Thesis, University of Toronto, Toronto, 1974.

[10] Flow of Fluids Through Valves, Fittings, and Pipe, Technical Paper No. 410, Crane Co., New York, 1980.

[11] A. H. Shapiro, The Dynamics and Thermodynamics of Compressible Fluid Flow, John Wiley & Sons, New York, 1953.

[12] W. E. Mason, "TACO - A Three-Dimensional Finite Element Heat Transfer Code", SAND83-8212, Sandia National Laboratories, Livermore, CA April 1983.

[13] V. K. Gabrielson, "SAHARA: A Multidimensional Heat-Transfer Computer Code", SAND83-xxxx, (To be Published), Sandia National Laboratories, Livermore, CA, 1983.

[14] J. J. Dongorra, et. al., "LINPACK Users' Guide", SIAM, Philadelphia, 1979.

[15] B. Carnahan, et. al., Applied Numerical Methods, John Wiley and Sons, New York, 1969.

[16] G. L. Clark, "A Zero Dimensional Model of Compressible Gas Flow in Networks of Pressure Vessels - Program TRIC", SAND83-8226 Sandia National Laboratories, Livermore, CA, July 1983.

[17] W. S. Winters, and M. P. Kanouff, "Implementation of Empirical Total Pressure and Enthalpy Loss Data into the DUCTFLO and TOPAZ Gas Transfer Codes", SAND83-xxxx, (To be Published), Sandia National Laboratories, Livermore, CA, 1983

[18] D. R. Chenoweth, "Gas-Transfer Analysis Section H - Real Gas Results via the van der Waals Equation of State and Viral Expansion Extensions of its Limiting Abel-Noble Form", SAND83-8229, Sandia National Laboratories, Livermore, CA, 1983.

UNLIMITED RELEASE

INITIAL DISTRIBUTION

D. J. Cagliostro, LANL, K-555
G. T. Mannell, LLNL, L-125
W. J. Comfort, LLNL, L-140
J. W. Nunziato, 1511
J. C. Cummings, 1512; Attn: F. G. Blottner, 1512
R. S. Claassen, 8000; Attn: A. N. Blackwell, 8200
D. Hartley, 8300
L. Gutierrez, 8400
D. M. Olson, 8100; Attn: R. C. Wayne, 8110
J. D. Gilson, 8130
W. E. Alzheimer, 8150
D. E. Gregson, 8160
S. C. Keeton, 8112
L. D. Bertholf, 8120; Attn: G. A. Benedetti, 8121
M. L. Callabresi, 8122
L. D. Bertholf, 8123
C. S. Hoyle, 8129
R. J. Gallagher, 8124
C. E. Hackett, 8124
M. P. Kanouff, 8124
B. A. Meyer, 8124
M. Abrams, 8125
D. R. Chenoweth, 8125
G. H. Evans, 8125
W. L. Hermina, 8125
M. E. John, 8125
S. Paolucci, 8125
W. S. Winters, 8125 (15)
A. S. Rivenes, 8162
D. J. Bohrer, 8163
G. W. Anderson, 8230
R. J. Kee, 8231
L. R. Petzold, 8231
M. I. Baskes, 8232
G. L. Clark, 8232
P. L. Mattern, 8350
S. C. Johnston, 8351
C. W. Robinson, 8360

B. R. Sanders, 8363
H. Hanser, 8440
A. J. West, 8441
W. R. Hoover, 8442
E. T. Cull, 8445
L. W. Derickson, 8445
A. Lutz, 8445
W. G. Wilson, 8453
Publications Division 8265/Technical Library Processes Division, 3141
Publications Division, for TIC (27)
Technical Library Processes Division, 3141 (3)
M. A. Pound, 8424-2, for Central Technical Files (3)