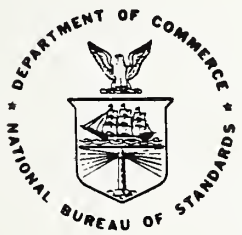
NBS

A Plan for the Development of the Generic Framework and Associated Computer Software for A Consolidated Compartment Fire Model Computer Code

Glenn P. Forney and Leonard Y. Cooper

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Fire Research
Gaithersburg, MD 20899

January 1987



U.S. DEPARTMENT OF COMMERCE
NATIONAL BUREAU OF STANDARDS

NBSIR 86-3500

A PLAN FOR THE DEVELOPMENT OF THE GENERIC FRAMEWORK AND ASSOCIATED COMPUTER SOFTWARE FOR A CONSOLIDATED COMPARTMENT FIRE MODEL COMPUTER CODE

Glenn P. Forney and Leonard Y. Cooper

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Fire Research
Gaithersburg, MD 20899

January 1987

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

TABLE OF CONTENTS

LIST OF FIGURES

A PLAN FOR THE DEVELOPMENT OF THE GENERIC FRAMEWORK AND ASSOCIATED COMPUTER SOFTWARE FOR A CONSOLIDATED COMPARTMENT FIRE MODEL COMPUTER CODE

by

Glenn P. Forney and Leonard Y. Cooper

ABSTRACT

A plan is presented for the development of computer software to support the generic framework of a Consolidated Compartment Fire Model (CCFM) computer code. The several software modules which will make up particular application versions of the CCFM are identified. These modules are then classified as either generic or application-specific. The characteristics and a detailed plan for the development of the generic modules are outlined.

Descriptions of the CCFM application products which will use the generic framework were presented. All of these will be designed in response to clear needs of the fire science and technology community. In terms of modeling sophistication of CCFM products, the prototype application will be at the simple end of the spectrum. It will provide simple guidance for engineering fire safety design. At the other extreme will be a benchmark-like compartment fire model computer code. At an intermediate level of sophistication will be an Application product which will be useful as an updated tool for Fire Hazard Assessment.

Keywords: compartment fires; computer programs; fire models; fire research; hazard assessment; room fires; smoke movement

A PLAN FOR THE DEVELOPMENT OF THE GENERIC FRAMEWORK AND ASSOCIATED COMPUTER SOFTWARE FOR A CONSOLIDATED COMPARTMENT FIRE MODEL COMPUTER CODE

by

Glenn P. Forney and Leonard Y. Cooper

1. INTRODUCTION

The purpose of this report is to outline a plan to develop the generic framework and associated computer software for a well-documented Consolidated Compartment Fire Model (CCFM) computer code. The general characteristics of the generic code will follow guidelines similar to those originally outlined for the Benchmark Compartment Fire Model (BCFM) code development [1]. However, these guidelines will be somewhat extended in the sense that the generic code framework will be usable over a wider range of application. As required, it will be possible to implement the generic code framework at different levels of modeling detail, thereby satisfying compartment fire modeling needs ranging from simple engineering fire safety guidance, at one extreme, to sophisticated BCFM computer code calculations at the other.

A key requirement of the generic code framework is that it be flexible in the sense that the structure of the code readily allows enrichment or simplification of its physics and numerics and of its input/output specifications. Such flexibility will allow versions of the code to be usable as a vehicle for testing new physical/chemical process algorithms and algorithm combinations. It will also guarantee that official published updates of the code can be carried out in a orderly manner and allow for revisions of the code to

be maintained at a technical level which is current in relation to advances in fire science and technology.

When one examines simulation software in general and compartment fire model simulation software in particular one observes a pattern in their overall function and structure. For example, most compartmental fire simulators can be characterized in the following way: The user inputs data; the program simulates the compartment fire-generated environment as it develops over a period of time; the program outputs data so that the user can analyze results. A major element of the new CCFM will be a framework and associated computer software which takes full advantage of this general pattern. This framework will consist of several independent modules and a well-defined protocol that interfaces them together and that allows required physics and numerics subroutines to be interfaced in turn.

The motivation for developing a consolidated code in this way is twofold. First, it allows experts to concentrate on the subject of their expertise. For example the physicist/fire modeler can work on the "physical algorithms" module, the mathematician/numerical analyst can work on the "numerics" module and so on. Second, modules designed for use by one application can be adapted for use by another application. So when improvements are made in the "numerics" or "physical algorithms" modules they will be available to all applications that use them, and greater productivity can result by consolidation of effort.

The specific objective of the CCFM project is to develop generic and application-specific software for a well-documented, user-friendly, modular and easily updated CFR CCFM computer code, and to use this software to implement different applications of the CCFM which are capable of simulating a broad range of compartment fire phenomena. The CCFM applications will address a variety of particular needs of CFR and the fire science and technology communities.

The objective will be met by carrying out the following tasks: Develop the generic framework and associated computer software for the CCFM; identify a specific set of phenomena to include in a prototype application of the CCFM software; use components from existing compartment fire models and results from recent submodel research activities to establish the set of equations to describe these phenomena; expand the number of entries in the catalog of modular submodel algorithms/subroutines [2] to include all of these equations; construct a numerics module for solving this set and more general application-specific sets of model equations; develop other application-specific software for the prototype CCFM application (e.g., for driving its input/output), and complete a working prototype computer model; use the prototype CCFM application to illustrate the feasibiliy and merits of the CCFM concept; generate more sophisticated applications of the CCFM as required.

This report is related to several of the above tasks, but focuses mainly on the first of these. Its purpose is to outline a plan for the development of the generic framework and associated computer software for the CCFM computer code.

2. PLAN FOR THE GENERIC FRAMEWORK OF THE CCFM COMPUTER CODE

2.1 Design Goals

It is anticipated that persons other than the original developers will be using applications derived from the generic framework discussed in this report. Therefore, in the design stage it is important to consider design issues such as program modularity, portability and user friendliness.

The CCFM code will achieve modularity by having a flexible structure. There are two aspects to the goal of a structured program. First, the developer should be able to make program changes with relative ease. Second, the user should have a choice of numerical and/or physical algorithms when doing simulation. To a limited degree then, the user can also determine the sophistication of the physics and the numerics used in a simulation run of a particular CCFM application.

The CCFM should be portable, i.e. it should be targeted to run on computers readily available to fire science practitioners. This should include: supercomputers such as the Cyber 205, minicomputers such as the Perkin-Elmer 3252 and microcomputers like the IBM PC/XT,AT with a math co-processor chip. This chip allows floating-point computations to be performed in the hardware. This is significantly faster than the alternative of performing these calculations in software.

In order to accomplish this goal of portability the CCFM should be coded in a language that is standardized and readily available on computers in the above mentioned computing environments. At the present time FORTRAN is the only language available that fits this criteria. In addition, coding should be done in a "vanilla" version of FORTRAN 77 and not rely on machine dependent features. Further, adherence to the FORTRAN 77 standard should not be so strict as to unnecessarily restrict the computers that the CCFM can run on.

Another design goal that should be incorporated into the CCFM is user-friendliness. First, the non-developer should be able to input, retrieve and interpret program results with relative ease. Next, the user should be given a chance to recover from any mistake he or she might make while entering data. Finally, the user should have the option of saving input data to a file so that subsequent model runs can be made more conveniently. Also, the output should be formatted in a way so that one experienced with fire science technology can understand it.

The framework for the CCFM will be developed with these design goals in mind.

2.2 A Framework for a Compartment Fire Simulator

As mentioned in the introduction, a program that simulates fire phenomena can be split into logical pieces or modules each of which performs a certain specified function. This section will identify the modules that make up

the generic framework and the areas of these modules that are common to any simulation.

The program modules in the generic framework as illustrated in Figure 1 are: Input, Model, Output and Data structures. Each module communicates with other modules using well-defined interfaces. It is the existence of these interfaces that allow the work on various pieces to proceed independently. Briefly the various program modules are defined as follows.

Input Module. The input module is responsible for soliciting input from the user. This should be done in a user-friendly manner, which, in the input module, means that the user should be given the opportunity to recover from mistakes.

A generic menu format has been designed that allows the developer to construct a tableau or menu of user requests. A menu consists of a collection of keywords or phrases arranged in rows and columns. These key's are the options that are available to the user. The user then types in a choice given by the menu. The menu routines handle the work of printing out the menu, soliciting input from the user, verifying that inputs are legal and if so what row and column of the menu the request came from. The routines that do these tasks are generic. That is, they will be the same from one application of the generic framework to the next.

An application-specific routine called the input driver must be written that implements the various commands given in the menu. Figure 2 shows how

the input module can be broken into generic and application-specific sections of code. The outline or skeleton of the input driver will be the same in each application of the generic framework. The input driver calls various generic menu routines that are written only once. The utilities in Figure 2 are generic routines that aid the input driver in handling input in a user-friendly way. For example some of the tasks it performs are: data type conversion, i.e. character to integer, file allocation and character string manipulation.

The routines in the input module that are generic or application-independent are the menu handling routines and the utilities. The input driver is application-specific and needs to be written for each application of the generic framework.

Model Module. The model module can be split into two parts as illustrated in Figure 3. The first part is the physics section which consists of the catalog of compartment fire subroutines [2] and a physics driver that calls the catalog routine entries required in a particular application. The compartment fire catalog will be the same for each application of the generic framework. The physics driver, on the other hand, will depend on the particular application version of the code.

The compartment fire catalog consists of a collection of FORTRAN subroutines each of which calculates a characteristic of the compartment fire-generated environment. These routines have one or more inputs and one output. An output from one routine may be an input to another one. So when a catalog routine is called, all inputs to this routine must be available to it. In

other words, the physics driver needs to call the appropriate catalog routines.

The second piece of the Model module is the numerics section which consist of those routines that are required to solve the necessary equations. The generic fire modeling equations are [3]:

$$0 = F(U,V,W,t) \tag{1}$$

$$dV/dt = G(U,V,W,t) \tag{2}$$

$$W = H(U,V,W,t) \tag{3}$$

$$Y = I(U,V,W,t), \tag{4}$$

where equation (1) denotes the zero finding equations, (2) gives the differential equations, (3) gives the fixed point equations, and (4) gives the eliminated equations. The vectors U, V, W and Y are the quantities that are modeled in the compartment fire simulator and equations (1) through (4) are the relations that these quantities must satisfy. The physics driver then calculates the functions F, G and H at time t. The numerics package calculates estimates for U, V, and W satisfying equations (1) through (3) at a new time $t + \Delta t$. A flowchart of this process is illustrated in Figure 4.

An application-specific interface routine called the physics/numerics interface transfers data between the physics and numerics section. Typically the data structures for the Physics module consists of a collection of arrays that are dimensioned for the number of objects or number of rooms, etc. If the number of objects in the simulation is less than the maximum possible then the Physics data structures will have gaps. The numerics section, however, expects the data it works with to be contiguous. Therefore we need a protocol

that compresses or packs the Physics data structures to form the numeric data structures and also unpacks the numeric data structures to form the Physics data structures. This protocol will also indicate to the numerics section the number of equations of each type that are to be solved. The choice of equations types anticipated are zero-finding, differential, fixed-point and eliminated equations. This process of packing and unpacking data structures is done in both the Harvard VI and FIRST fire simulation codes.

The physics/numerics interface is application-specific. It will depend upon the data structures that are set up in the physics driver routine. Work remains to be done that will give guidelines as to how to set up the Physics data structures so that the job of writing the physics/numerics interface routines will be made easier.

The numeric routines will be the same for each application of the generic framework. At this time it is anticipated that the numerics package will consist of subroutines that implement the Newton-Raphson method and a non-linear Gauss-Seidel fixed point iteration algorithm. Other types of numerical procedures will be investigated over the course of this project to determine their suitability for solving equations arising in the compartment fire model setting.

The generic or application-independent routines in the Model module are the compartment fire catalog and the numerics section. The sections of code that need to be written for each application of the generic framework are the physics driver and the physics/numerics interface.

Output Module. The output module will consist of routines that will display simulation results in a printed or graphical form. The output format will be designed so that someone familiar with concepts in fire science and technology, but not necessarily the inner workings of the simulator, will be able to readily interpret the simulation results. Further, the output from a simulation run will be available in a format easily plotted on graphical devices. It is anticipated that the high-level graphics routines will be device independent. Low-level routines on the other hand will exist for each graphical device to be supported. Work at CFR has been done on incorporating graphics into applications in a device independent way [4]. It is anticipated that this work can be incorporated into an implementation of the generic framework.

Data Structure Module. The data structure module will organize and contain those pieces of data to be saved at the end of a time step. The data will be stored in two places, internal or central memory and external memory, i.e. disk or tape. The format for internal data structures will be as follows: If the value of a variable needs to be saved then it should be placed in a labeled common. The variables in this common should be stored by type, double precision first, followed by single precision, integer and logical. A variable that only needs to be saved once at the beginning of a simulation run is a time independent variable otherwise if it needs to be saved at the end of each time step it is a time dependent variable. Time dependent and independent variables need to be allocated in different labeled common blocks. The file interface routines, to be discussed next, are responsible for transfer-

ring values of variables between internal and external memory. An expanded form of the data structure module is given in Figure 5.

File Interface. The file interface is responsible for transferring data between internal and external memory. External memory will consist of sequential disk files, random access disk files and sequential tape files. Internal memory will consist of a collection of labeled common blocks. These common blocks contain data that the modeler/developer wishes to save at the end of a time step. Data in these common blocks will be grouped by type, i.e. time dependent data should be stored in a separate labeled common from time independent data. Further, data in any given labeled common should be in the following order: double precision first, followed by single precision, integer and logical. The purpose of this criteria is to allow generic routines to be written that can transfer data between random access and sequential files and between internal and external memory.

A random access file will be used to save results from a simulation run when subsequent runs will be performed on the same type of computer. A sequential file is used when simulations runs are started on one computer and finished on another. The purpose for this is the following. A computer typically can read another computer's sequential file's but not their random access files. Random access files on the other hand can be accessed more quickly and stored more compactly than sequential files.

The format of the internal data structures will be specified as data to

the application. Therefore, the file interface routines are generic and will be the same for each application of the generic framework.

3. THE GENERIC FRAMEWORK AND ITS APPLICATION

3.1 Proposed Products

The first application of the generic framework will serve the dual purpose of testing the generic framework concept and providing a well-documented, user-friendly, compartment fire model simulator. It is anticipated that the first application of the generic framework will be suitable for Fire Safety Engineering and Design. The goal is to provide a prototype application of the generic framework which is relatively simple in modeling detail, a product that is simple in scope while being useful as a tool of analysis for fire science practitioners.

Subsequent to this activity, at least two other applications of the generic framework are envisaged. One application will be useful as an updated tool of Fire Hazard Assessment. The other, the most sophisticated of the applications, will be the richest in terms of accuracy of simulation of compartment fire-generated phenomena. The latter application has been referred to previously as the benchmark compartment fire model [1].

As noted earlier, the architecture of these three products will be basically the same. Certain routines are application-specific, namely, the

input driver, the physics driver and the physics/numerics interface routine. Most of the rest of the code to be developed for the prototype application will find common use in these and subsequent CCFM application packages.

The generic framework is possible because the equations sets describing the compartment fire phenomena in CCFM applications are basically of the same class. These sets will differ in the degree of sophistication and number of the fire catalog routines used, but they all can be reduced to some combination of zero-finding, differential and fixed point equations.

3.2 Features of the Generic Framework

Many portions of a simulation package such as the numerics or input module serve the same purpose in any given application. By identifying these parts and writing them in an independent way they may be used in more than one software product. For example the same Gauss-Seidel algorithm will be useful in all or most CCFM applications that have non-linear fixed point equations. Therefore, the Gauss-Seidel solver will be an important element of the CCFM numerics module.

Once the interfaces between modules are established, work will proceed along independent paths, making a team effort possible. For example, mathematician/numerical analyst(s) will work on the numerics of solving the non-linear equations; engineer/physicist(s) will work on the catalog module, and write drivers that call the appropriate catalog routines to be solved by the

numerics; and people familiar with the needs of potential users of an application will design a menu of appropriate user inputs and outputs.

4. SUMMARY

This report presented a plan for the development of computer software to support the generic framework of a Consolidated Compartment Fire Model (CCFM) computer code. The several software modules which will make up particular application versions of the CCFM were identified. These modules were then classified as either generic or application-specific. The characteristics and a detailed plan for the development of the generic modules were outlined.

Descriptions of CCFM application products which will use the generic framework were presented. All of these will be designed in response to clear needs of the fire science and technology community. In terms of modeling sophistication of CCFM products, the prototype application will be at the simple end of the spectrum. It will provide simple guidance for engineering fire safety design. At the other extreme will be a benchmark-like compartment fire model computer code. At an intermediate level of sophistication will be an application product which will be useful as an updated tool for Fire Hazard Assessment.

5. REFERENCES

[1] Cooper, Leonard Y. , Rockett, John A., Mitler, Henri E. and Stroup, David W., A Program for the Development of A Benchmark Compartment Fire Model Computer Code, U.S. National Bureau of Standards Report, NBSIR 85-3252.

[2] Stroup, David, W., A Catalog of Compartment Fire Model Algorithms and Associated Computer Subroutines, to appear as an NBSIR.

[3] Gahm, J.B., Computer Fire Code VI - Volume 1, Harvard University Report to U.S. National Bureau of Standards, NBS-GCR-83-451, 1983.

[4] Jones, Walter W. and Fadell, Alicia B., A Device Independent Graphics Kernel, U.S. National Bureau of Standards Report, NBSIR 85-3235.

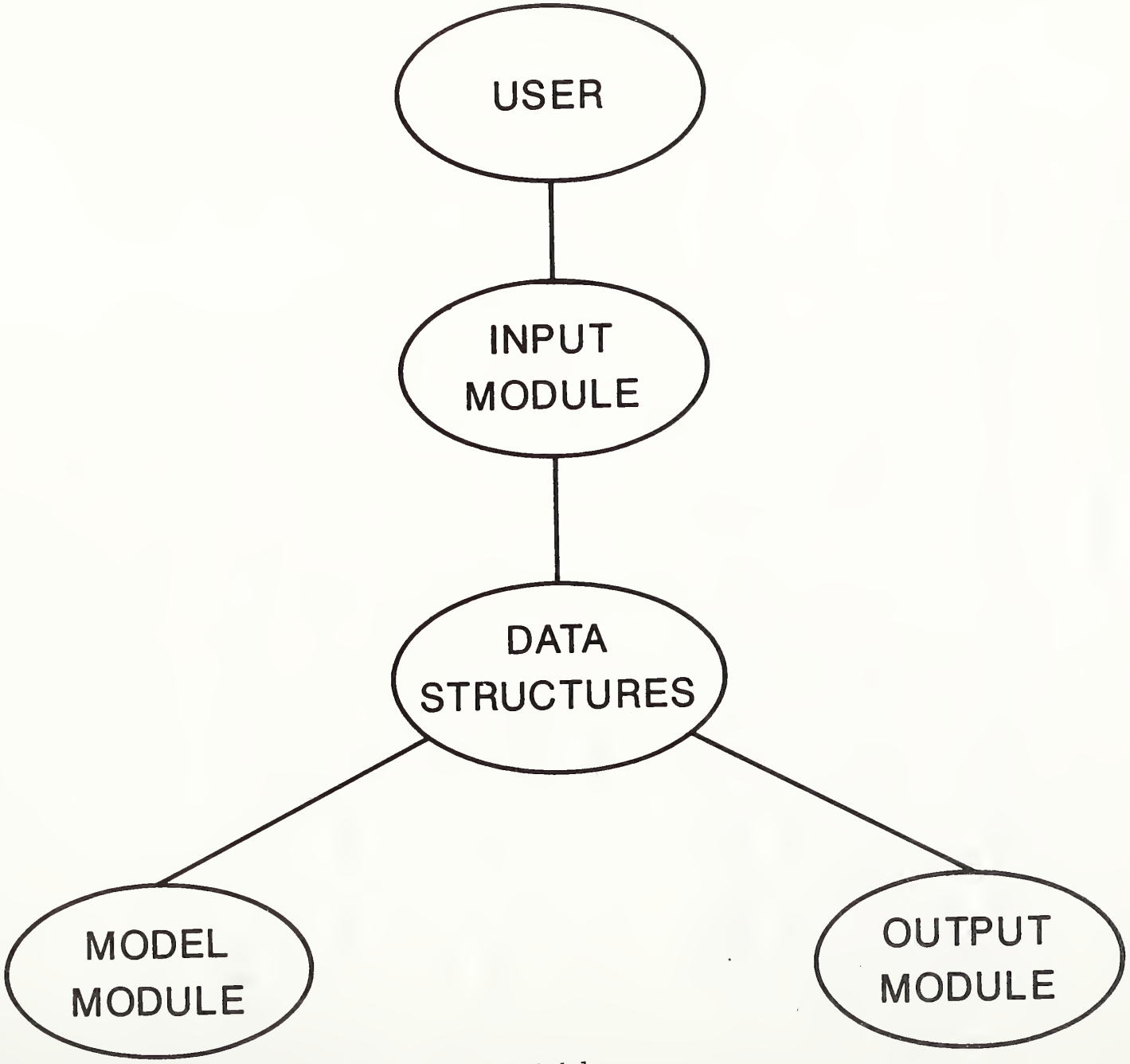


Figure 1. Diagram of a Generic Compartment Fire Model

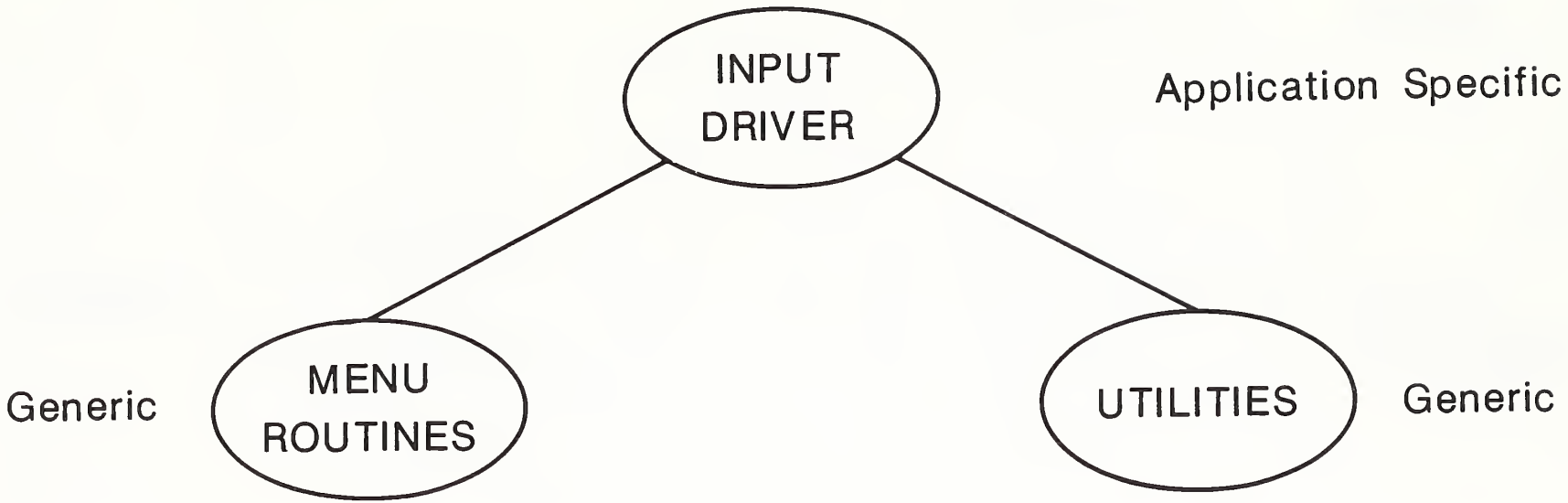


Figure 2. Diagram of the Input Module

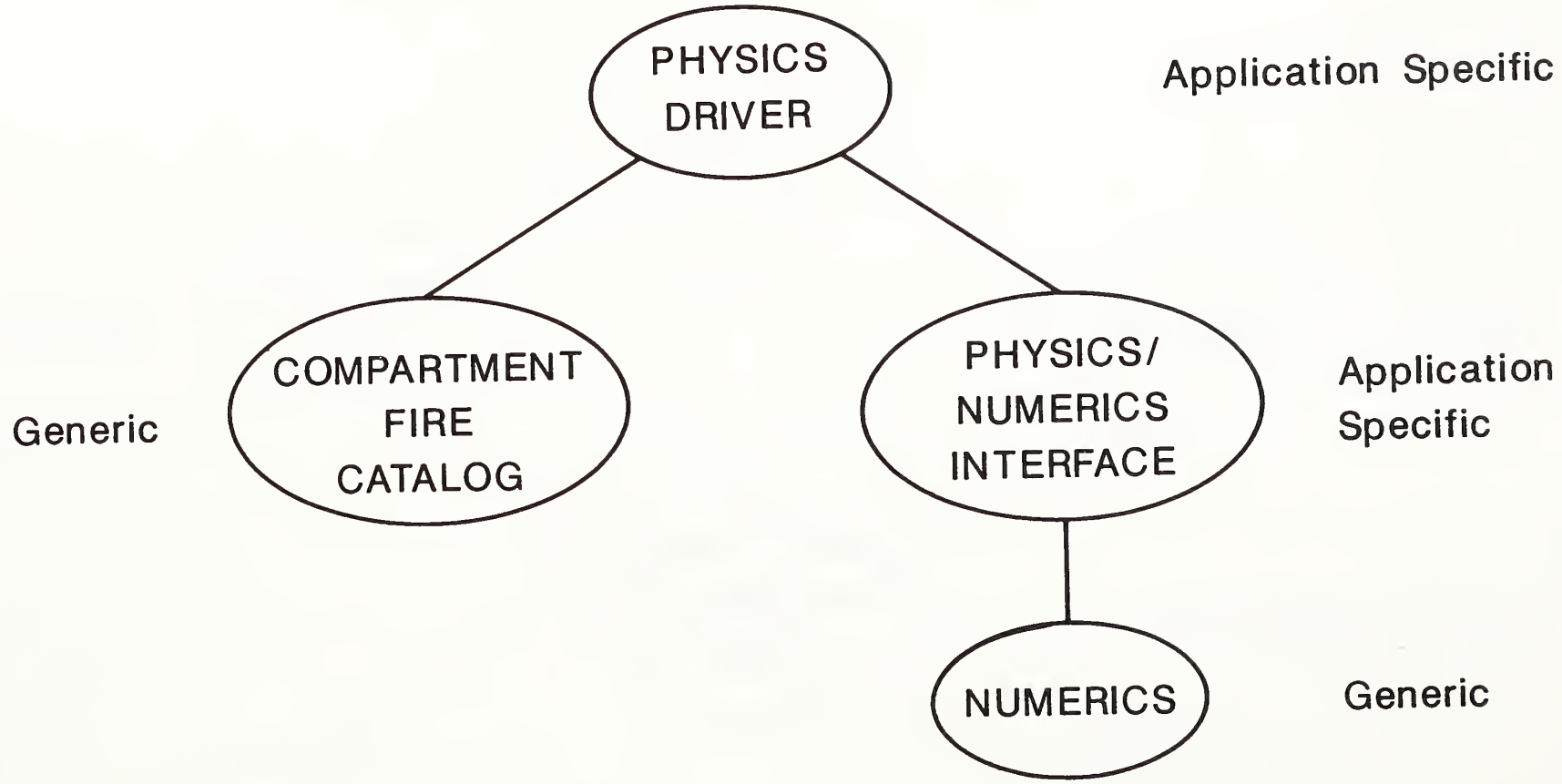


Figure 3. Diagram of the Model Module

Solve: $0 = F(u,v,w,t)$

$dv/dt = G(u,v,w,t)$

$w = H(u,v,w,t)$

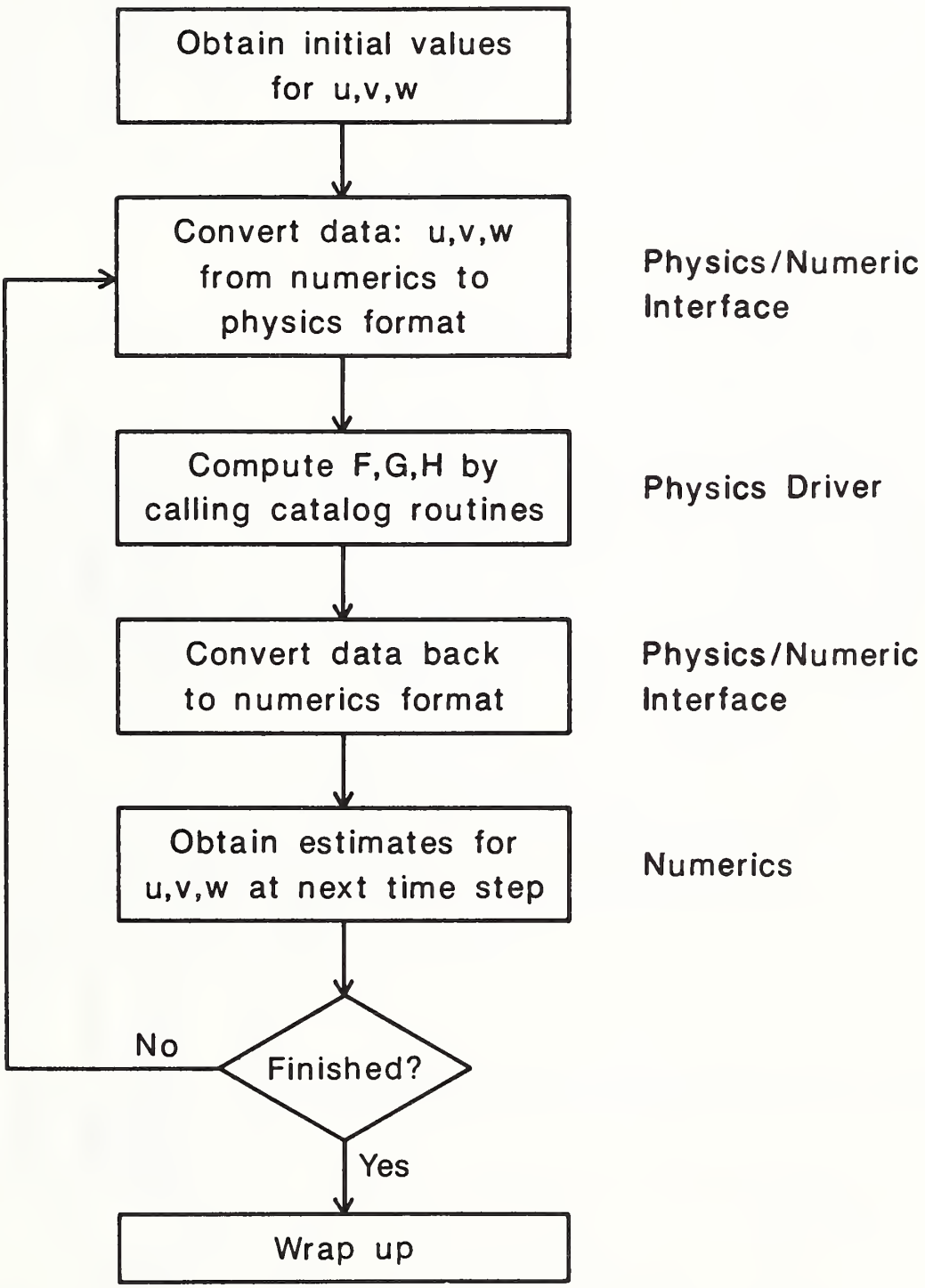


Figure 4. Outline of the Solution Process

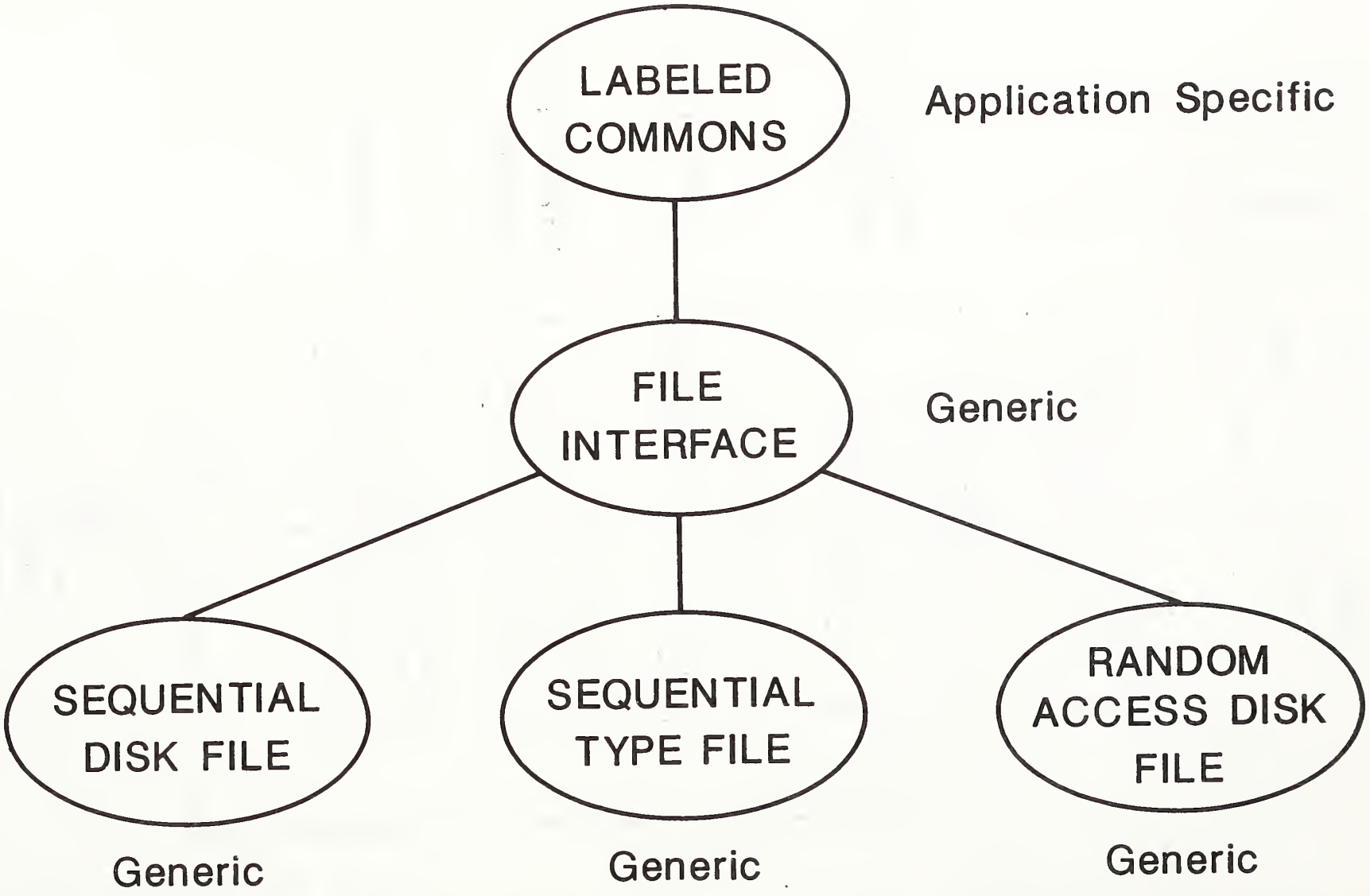


Figure 5. Diagram of the Data Structures Module

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBSIR-86/3500	2. Performing Organ. Report No.	3. Publication Date January 1987

4. TITLE AND SUBTITLE

A Plan for the Development of the Generic Framework and Associated Computer Software for a Consolidated Compartment Fire Model Computer Code

5. AUTHOR(S)

Glenn P. Forney and Leonard Y. Cooper

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234 Gaithersburg, Maryland 20899

7. Contract/Grant No.

8. Type of Report & Period Covered

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

A plan is presented for the development of computer software to support the generic framework of a Consolidated Compartment Fire Model (CCFM) computer code. The several software modules which will make up particular application versions of the CCFM are identified. These modules are then classified as either generic or application-specific. The characteristics and a detailed plan for the development of the generic modules are outlined.

Descriptions of the CCFM application products which will use the generic framework were presented. All of these will be designed in response to clear needs of the fire science and technology community. In terms of modeling sophistication of CCFM products, the prototype application will be at the simple end of the spectrum. It will provide simple guidance for engineering fire safety design. At the other extreme will be a benchmark-like compartment fire model computer code. At an intermediate level of sophistication will be an Application product which will be useful as an updated tool for Fire Hazard Assessment.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

compartment fires; computer programs; fire models; fire research; hazard assessment; room fires; smoke movement

13. AVAILABILITY

☒ Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

☒ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES

15. Price