

EPA-
September 1996

HYDROLOGICAL SIMULATION PROGRAM - FORTRAN
USER'S MANUAL FOR RELEASE 11

Brian R. Bicknell
John C. Imhoff
John L. Kittle, Jr.
Anthony S. Donigian, Jr.

AQUA TERRA Consultants
Mountain View, California 94043

Robert C. Johanson

University of the Pacific
Stockton, California 95204

Project Officer

Thomas O. Barnwell
Assessment Branch
Environmental Research Laboratory
Athens, Georgia 30613

IN COOPERATION WITH

OFFICE OF SURFACE WATER
WATER RESOURCES DIVISION
U.S. GEOLOGICAL SURVEY
RESTON, VIRGINIA 20192

ENVIRONMENTAL RESEARCH LABORATORY
OFFICE OF RESEARCH AND DEVELOPMENT
U.S. ENVIRONMENTAL PROTECTION AGENCY
ATHENS, GEORGIA 30613

DISCLAIMER

This report has been reviewed by the Environmental Research Laboratory, U.S. Environmental Protection Agency, Athens, Georgia and approved for publication. Approval does not signify that the contents necessarily reflect the views and policies of the U.S. Environmental Protection Agency, nor does mention of trade names or commercial products constitute endorsement or recommendation for use.

FOREWORD

As environmental controls become more costly to implement and the penalties of judgment errors become more severe, environmental quality management requires more efficient analytical tools based on greater knowledge of the environmental phenomena to be managed. As part of this Laboratory's research on the occurrence, movement, transformation, impact, and control of environmental contaminants, the Assessment Branch develops management or engineering tools to help pollution control officials achieve water quality goals through watershed management.

The development and application of mathematical models to simulate the movement of pollutants through a watershed and thus to anticipate environmental problems has been the subject of intensive EPA research for a number of years. An important tool in this modeling approach is the Hydrological Simulation Program - FORTRAN (HSPF), which uses computers to simulate hydrology and water quality in natural and man-made water systems. HSPF is designed for easy application to most watersheds using existing meteorologic and hydrologic data. Although data requirements are extensive and running costs are significant, HSPF is thought to be the most accurate and appropriate management tool presently available for the continuous simulation of hydrology and water quality in watersheds.

Rosemarie C. Russo, Ph.D.
Director
Environmental Research Laboratory
Athens, Georgia

ABSTRACT

The Hydrological Simulation Program - FORTRAN (HSPF) is a set of computer codes that can simulate the hydrologic, and associated water quality, processes on pervious and impervious land surfaces and in streams and well-mixed impoundments. The manual discusses the structure of the system, and presents a detailed discussion of the algorithms used to simulate various water quantity and quality processes. It also contains all of the information necessary to develop input files for applying the program, including descriptions of program options, parameter definitions, and detailed input formatting data.

The original version of this report was submitted in fulfillment of Grant No. R804971-01 by Hydrocomp, Inc., under the sponsorship of the U.S. Environmental Protection Agency. That work was completed in January 1980.

Extensive revisions, modifications, and corrections to the original report and the HSPF code were performed by Anderson-Nichols and Co. under Contract No. 68-03-2895, also sponsored by the U.S. EPA. That work was completed in January 1981. Versions 7 and 8 of HSPF and the corresponding documents were prepared by Linsley, Kraeger Associates, Ltd. and Anderson-Nichols under Contract No. 68-01-6207, the HSPF maintenance and user support activities directed by the U.S. EPA laboratory in Athens, GA.

The HSPF User's Manual for Versions 10 and 11 were prepared by AQUA TERRA Consultants of Mountain View, CA, incorporating code modifications, corrections, and documentation of algorithm enhancements sponsored by the U.S. Geological Survey, the U.S. EPA Chesapeake Bay Program, the U.S. Army Corps of Engineers, and the U.S. EPA Athens Environmental Research Laboratory. The Version 11 manual and code were prepared under sponsorship of the U.S. Geological Survey under Contract No. 14-08-0001-23472. The manual is available in WordPerfect format.

CONTENTS

| | |
|--------------------|-----|
| Foreword | iii |
| Abstract | iv |

Part

| | |
|---|-----|
| A Introduction. | 1 |
| B General Principles. | 8 |
| C Standards and Conventions (not included). | 24 |
| D Visual Table of Contents (not included) | 24 |
| E Functional Description. | 25 |
| F Format for the Users Control Input. | 284 |

Appendices

| | |
|----------------------------------|-----|
| I Glossary of Terms | 744 |
| II Time Series Concepts. | 752 |

PART A

INTRODUCTION

CONTENTS

| | | |
|-----|---|---|
| 1.0 | Purpose and Scope of the HSPF Software | 2 |
| 2.0 | Requirements for HSPF | 4 |
| 3.0 | Purpose and Organization of this Document | 5 |
| 4.0 | Definition of Terms | 6 |
| 5.0 | Notice of User Responsibility | 6 |
| 6.0 | Acknowledgments | 6 |

1.0 PURPOSE AND SCOPE OF THE HSPF SOFTWARE

The use of models which simulate continuously the quantity/quality processes occurring in the hydrological cycle is increasing rapidly. Recently there has been a proliferation in the variety of models and in the range of processes they simulate. This has been a mixed blessing to a user. To get the benefits of simulation, a user must select a model from a bewildering array and then spend much effort amassing and manipulating the huge quantities of data which the model requires. If the modeler wishes to couple two or more subprocess models to simulate a complete process, he often encounters further difficulties. The underlying assumptions and/or structures of the subprocess models may make them somewhat incompatible. More frequently, the data structures are so different that coupling requires extensive data conversion work.

One reason for these problems is that the boom in modeling work has not included enough work on the development of good model structures. That is, very few software packages for water resource modeling are built on a systematic framework in which a variety of process modules can fit.

With HSPF we have attempted to overcome these problems as far as possible. HSPF consists of a set of modules arranged in a hierarchical structure, which permit the continuous simulation of a comprehensive range of hydrologic and water quality processes. Our experience with sophisticated models indicates that much of the human effort is associated with data management. This fact, often overlooked by model builders, means that a successful comprehensive model must include a sound data management component. Otherwise, the user may become so entangled in data manipulation that his progress on the simulation work itself is drastically retarded. Consequently, the HSPF software is planned around a time series management system operating on direct access principles. The simulation modules draw input from time series storage files and are capable of writing output to them. Because these transfers require very few instructions from the user, the problems referred to above are minimized.

The system is designed so that the various simulation and utility modules can be invoked conveniently, either individually or in tandem. A top down approach emphasizing structured design has been followed. First, the overall framework and the Time Series Management System were designed. Then, work progressed down the structure from the highest, most general level to the lowest, most detailed one. Every level was planned before the code was written. Uniform data structures, logic figures, and programming conventions were used throughout. Modules were separated according to function so that, as much as possible, they contained only those activities which are unique to them. Structured design has made the system relatively easy to extend, so that users can add their own modules with relatively little disruption of the existing code.

Now, a note on the initial contents of the system. Presently, it includes modules which can handle almost all the functions which are available in the following existing models:

- (1) HSP (LIBRARY, UTILITY, LANDS, CHANNEL, QUALITY)
- (2) ARM
- (3) NPS
- (4) SERATRA

The HSPF software is not merely a translation of the above models, but a new system with a framework designed to accommodate a variety of simulation modules; the modules described above are the initial contents. Many extensions have been made to the above models in the course of restructuring them into the HSPF system.

It is hoped that HSPF will become a valuable tool for water resource planners. Because it is more comprehensive than most existing systems, it should permit more effective planning. More specifically, the package can benefit the user in the following ways:

1. The time-series-oriented direct access data system and its associated modules can serve as a convenient means of inputting, organizing, and updating the large files needed for continuous simulation.
2. The unified user-oriented structure of the model makes it relatively simple to operate. The user can select those modules and options that are needed in one run, and the system will ensure that the correct sets of code are invoked and that internal and external transfers of data are handled. This is achieved with a minimum of manual intervention. Input of control information is simplified because a consistent system is used for this data for all the modules.
3. Because the system has been carefully planned using top-down programming techniques, it is relatively easy to modify and extend. The use of uniform programming standards and conventions has assisted in this respect.
4. Since the code is written almost entirely in ANSI standard Fortran, implementation on a wide variety of computers is possible.

2.0 REQUIREMENTS FOR HSPF

In awarding the grant for development of HSPF, the EPA set the following requirements:

1. It must manage and perform deterministic simulation of a variety of aquatic processes which occur on and under land surfaces and in channels and reservoirs.
2. It must readily accommodate alternate or additional simulation modules.
3. It must permit easy operation of several modules in series, and thus be capable of feeding output from any operation to subsequent operations.
4. It must be in ANSI Fortran with minor specified extensions.

With the concurrence of the EPA, we expanded on these requirements:

1. It must have a totally new design. Existing modules should not merely be translated, but should be fitted into a new framework.
2. It must be designed from the top down, using some of the new improved programming techniques, such as Structured Design and Structured Programming.
3. Duplication of blocks of code which perform similar or identical functions should be avoided.
4. The user's control input must have a logically consistent structure throughout the package.
5. Uniform standards and practices must be followed throughout the design, development and documentation of the system.
6. It must have a conveniently operated disk-based time series storage file built on the principle of direct access.
7. The design must be geared to implementation on larger models of the current generation of minicomputers. It must be compatible with Operating Systems which share memory using either the virtual memory approach or a conventional overlay technique.

3.0 PURPOSE AND ORGANIZATION OF THIS DOCUMENT

This report contains all the documentation of the HSPF system. It is designed to:

1. introduce new users to the principles and concepts on which the system is founded
2. describe the technical foundations of the algorithms in the various application (simulation) modules
3. describe the input which the user supplies to run the system

To meet these needs and, at the same time, to produce a document which is reasonably easy to use, we have divided this report into several distinct parts, each with its own organization and table of contents.

Part A (this one) contains introductory material.

Part B outlines the general principles on which the HSPF system is based. This includes a discussion of the "world view" which our simulation modules embody. A firm grasp of this material is necessary before the detailed material can be properly understood.

Part C Standards and Conventions (not included)

Part D Visual Table of Contents (not included)

Part E documents the function of each part of the software. The organization of this part follows the layout of the software itself. The relationship between, and the functions of, the various modules are described, starting at the highest most general level and proceeding down to the lowest most detailed level. The algorithms used to simulate the quantity and quality processes which occur in the real world are described in this part.

Part F describes the User's Control Input; that is, the information which the user must provide in order to run HSPF.

Material which might obscure the structure of this document if it were included in the body of the report appears in Appendices. These include a glossary of terms and descriptions of sample runs.

4.0 DEFINITION OF TERMS

In this document, terms which have a special meaning in HSPF, are enclosed in quotes the first time they occur. Usually an explanation follows immediately. A glossary of terms can be found in Appendix I.

5.0 NOTICE OF USER RESPONSIBILITY

This product has been carefully developed. Although the work included testing of the software, the ultimate responsibility for its use and for ensuring correctness of the results obtained, rests with the user.

The EPA and the developers of this software make no warranty of any kind with regard to this software and associated documentation, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. They shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

While we intend to correct any errors which users report, we are not obliged to do so. We reserve the right to make a reasonable charge for work which is performed for a specific user at his request.

6.0 ACKNOWLEDGMENTS

The original development of HSPF was sponsored by the Environmental Research Laboratory in Athens, Georgia. David Duttweiler was the laboratory director and Robert Swank the head of the Technology Development and Applications Branch, which supervised the project during the code development period. Mr. Jim Falco was the Project Officer initially on the HSPF development work; he was succeeded by Mr. Tom Barnwell who continues to oversee HSPF support activities for EPA.

Recent development of HSPF has been sponsored by the U.S. Geological Survey Water Resources Division in Reston, Virginia. Dr. Alan Lumb is the Contract Officer and directs that effort for the USGS.

The initial HSPF and user manual development work was performed by Hydrocomp, Inc.; members of the entire project team are acknowledged in the original (Release 5.0) version of the user manual (EPA Publication No. EPA-600/9-80-015) published in April 1980. Subsequent revisions and extensions to the HSPF code and user manual were performed by Anderson-Nichols & Co., Inc. and AQUA TERRA Consultants. The primary participants in the work noted above, and their contributions, are discussed below.

Robert Johanson was Project Manager for Hydrocomp on the initial development work, and provided consulting assistance to Anderson-Nichols & Co., Inc. and Linsley, Kraeger Associates on subsequent development and maintenance work.

John Imhoff had primary responsibility for the RCHRES water quality sections, both during the initial development work for Hydrocomp and subsequent modifications and development for Anderson-Nichols and AQUA TERRA.

Introduction

Harley Davis designed, coded and documented much of the PERLND and IMPLND modules for Hydrocomp during the initial development effort.

Delbert Franz participated in the overall design of the system and supervised the work on the time series management system at Hydrocomp.

Jack Kittle performed or directed most HSPF software development activities since 1979. His focus has been on the software structure, time series improvements, and addition of new components. He designed the MUTSIN (Multiple Timeseries Sequential Input) module, water categories in the RCHRES module, conditional Special Actions, major structural improvements to the software, and was responsible for production of Releases 7, 8 and 9. He continues to be a principal advisor in HSPF development and maintenance.

Tony Donigian participated in the design of the PERLND algorithms in the initial project at Hydrocomp. Since 1980, he has been the Principal Investigator/Project Manager, providing overall guidance and supervision, on all projects related to HSPF algorithm development, with particular focus on the improvements to the AGCHEM sections of the program.

Brian Bicknell has been involved in HSPF maintenance and software development since 1981. He has developed or directed all recent algorithm enhancements and software corrections. He added the WDM file interaction, the MASS-LINK and SCHEMATIC blocks, and the FILES block. He directed the development and documentation of Versions 10 and 11, including major enhancements for simulating forest nitrogen, atmospheric deposition, the DSS file interface, and sediment-nutrient interactions and bed temperature interactions in RCHRES.

Tom Jobes has been the primary software engineer with day-to-day responsibility for HSPF development and maintenance since 1992. He implemented most of the enhancements and software corrections in Version 11, with particular responsibility for the DSS file interface, atmospheric deposition, forest nitrogen and plant uptake enhancements in AGCHEM, multiple WDM files, code structure improvements, conditional Special Actions, and water categories.

PART B

GENERAL PRINCIPLES

CONTENTS

| | | |
|-----|--|----|
| 1.0 | View of the Real World | 9 |
| 1.1 | General Concepts | 9 |
| 1.2 | Nodes, Zones, and Elements | 9 |
| 1.3 | Processing Units and Networks | 11 |
| 2.0 | Software Structure | 14 |
| 2.1 | Concept of an Operation | 14 |
| 2.2 | Time Series Storage | 16 |
| 2.3 | Times Series Management for an Operation | 17 |
| 2.4 | HSPF Software Hierarchy | 17 |
| 3.0 | Structure of a Job | 20 |
| 3.1 | Elements of a Job | 20 |
| 3.2 | Groups of Operations | 20 |
| 4.0 | Conventions Used in Functional Description | 23 |
| 5.0 | Method of Documenting Data Structures | 23 |
| 5.1 | Structure of Data in Memory | 23 |
| 5.2 | Structure of Data on Disk Files | 23 |
| 6.0 | Method of Handling Diagnostic Messages | 24 |

FIGURES

| Number | | Page |
|--------|---|------|
| 1-1 | Nodes, zones and elements | 10 |
| 1-2 | Directed and non-directed graphs | 12 |
| 1-3 | Single- and multi-element processing units | 13 |
| 2-1 | Logical structure of the internal scratch pad | 15 |
| 2-2 | Activities involved in an operation | 18 |
| 2-3 | Overview of HSPF software | 19 |
| 3-1 | Schematic of data flow and storage in a single run | 21 |
| 3-2 | Extract from typical User's Control Input, showing how grouping of operations is specified | 22 |

1.0 VIEW OF THE REAL WORLD

1.1 General Concepts

To design a comprehensive simulation system, one must have a consistent means of representing the prototype; in our case, the real world. We view it as a set of constituents which move through a fixed environment and interact with each other. Water is one constituent; others are sediment, chemicals, etc. The motions and interactions are called processes.

1.2 Nodes, Zones, and Elements

The prototype is a continuum of constituents and processes. Simulation of such a system on a digital computer requires representation in a discrete fashion. In general, we do this by subdividing the prototype into "elements" which consist of "nodes" and "zones."

A node corresponds to a point in space. Therefore, a particular value of a spatially variable function can be associated with it, for example, channel flow rate and/or flow cross sectional area. A zone corresponds to a finite portion of the real world. It is usually associated with the integral of a spatially variable quantity, for example, storage in a channel reach. The zone the smallest unit into which we subdivide the world. The relationship between zonal and nodal values is similar to that between the definite integral of a function and its values at the limits of integration.

An element is a collection of nodes and/or zones. Figure 1-1 illustrates these concepts. We simulate the response of the land phase of the hydrological cycle using elements called "segments." A segment is a portion of the land assumed to have areally uniform properties. A segment of land with a pervious surface is called a "Pervious Land-segment" (PLS). Constituents in a PLS are represented as resident in a set of zones (Fig. 1-1a). A PLS has no nodes. As a further example, consider our formulation of channel routing. We model a channel reach as a one dimensional element consisting of a single zone situated between two nodes (Fig. 1-1b). We simulate the flow rate and depth at the nodes; the zone is associated with storage.

The conventions of the finite element technique also fall within the scope of these concepts. Figure 1-1c shows a two dimensional finite element used in the simulation of an estuary. Three nodes define the boundaries of the triangular element. A fourth node, situated inside, may be viewed as subdividing the element into three zones. This last type of element is not presently used in any HSPF module, but is included in this discussion to show the generality provided by HSPF. The system can accommodate a wide variety of simulation modules.

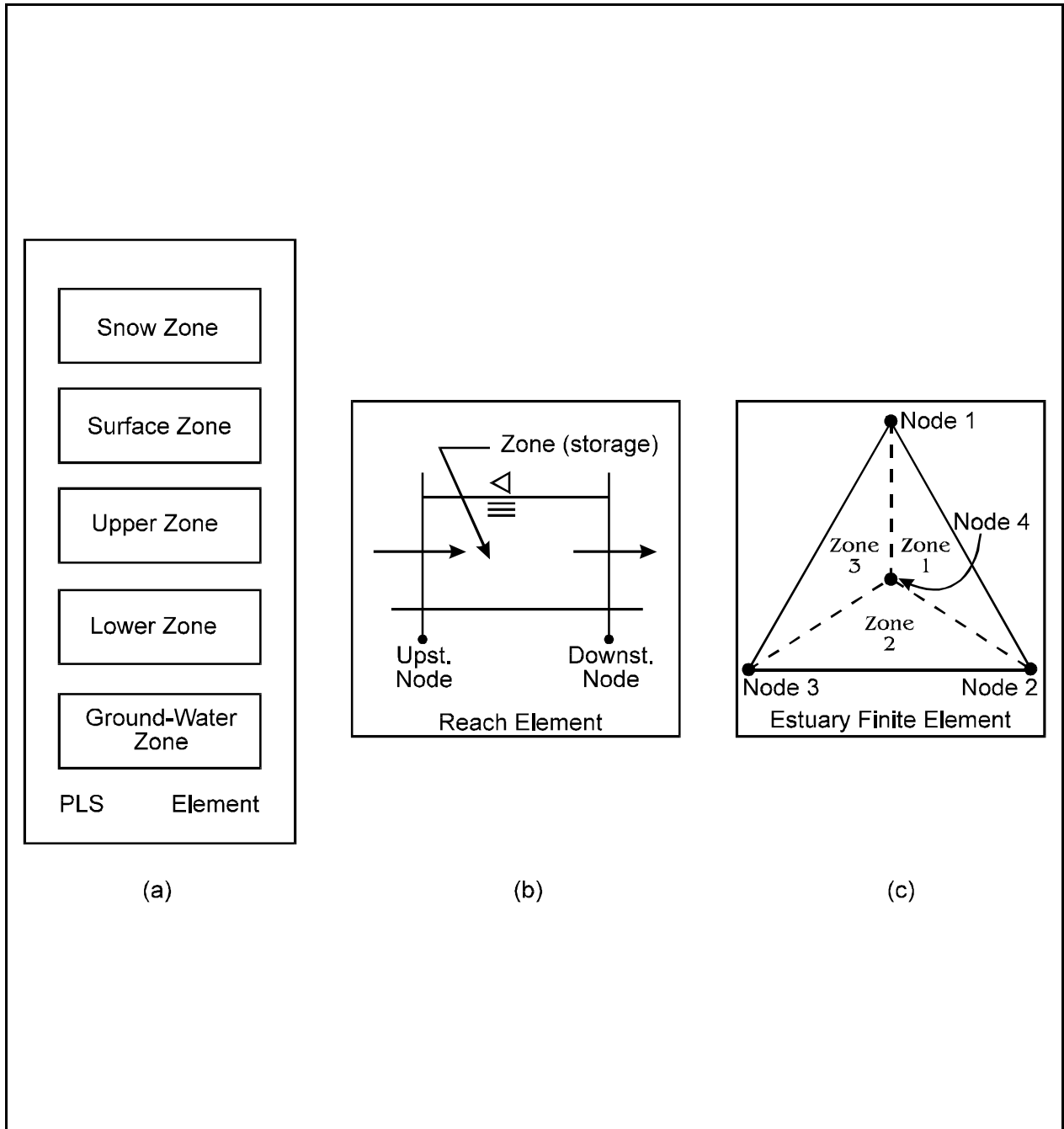


Figure 1-1 Nodes, zones, and elements

There are no fixed rules governing the grouping of zones and nodes to form elements. The model builder must decide what grouping is reasonable and meaningful, based on his view of the real world processes being simulated. In the foregoing material we presented some elements used in HSP and other systems. In general, it is convenient to define elements so that a large portion of the real world can be represented by a collection of conceptually identical elements. In this way, a single parameter structure can be defined which applies to every element in the group. Thus, each element is a variation on the basic theme. It is then meaningful to speak of an "element type." For example, elements of type "PLS" all embody the same arrangement of nodes and are represented by sets of parameters with identical structure. Variations between segments are represented only by variations in the values of parameters. The same applies to any other element, such as a Reach, layered lake or a triangular finite element.

As illustrated in the above discussion, nodes are often used to define the boundaries of zones and elements. A zone, characterized by storage, receives inflows and disperses outflows; these are called "fluxes." Note that if the nodal values of a field variable are known, it is often possible to compute the zonal values (storages). The reverse process does not work.

1.3 Processing Units and Networks

To simulate a prototype we must handle the processes occurring within the elements and the transfer of information and constituents between them. The simulation of large prototypes is made convenient by designing a single "application module" for a given type of element or element group, and applying it repetitively to all similar members in the system. For example, we may use the RCHRES module to simulate all the reaches in a watershed using storage routing. This approach is most efficient computationally if one element or group of elements, called a "processing unit" (PU), is simulated for an extended period of time before switching to the next one. To permit this, we must be able to define a processing sequence such that all information required by any PU comes from sources external to the system or from PU's already simulated. This can only happen if the PU's and their connecting fluxes form one or more networks which are "directed graphs." In a directed graph there are no bi-directional paths and no cycles. Figure 1-2 shows some directed and non-directed graphs.

The requirement that PU's form directed graphs provides the rule for grouping elements into PU's. Any elements interacting with each other via loops or bi-directional fluxes must be grouped into a single PU because none of them can be simulated apart from the others.

Thus, we can have both single element and multi-element PU's. A PLS is an example of the former and a channel network simulated using the full equations of flow exemplifies the latter (Fig. 1-3). A multi-element PU is also known as a "feedback region." The collection of PU's which are simulated in a given run is called a "network."

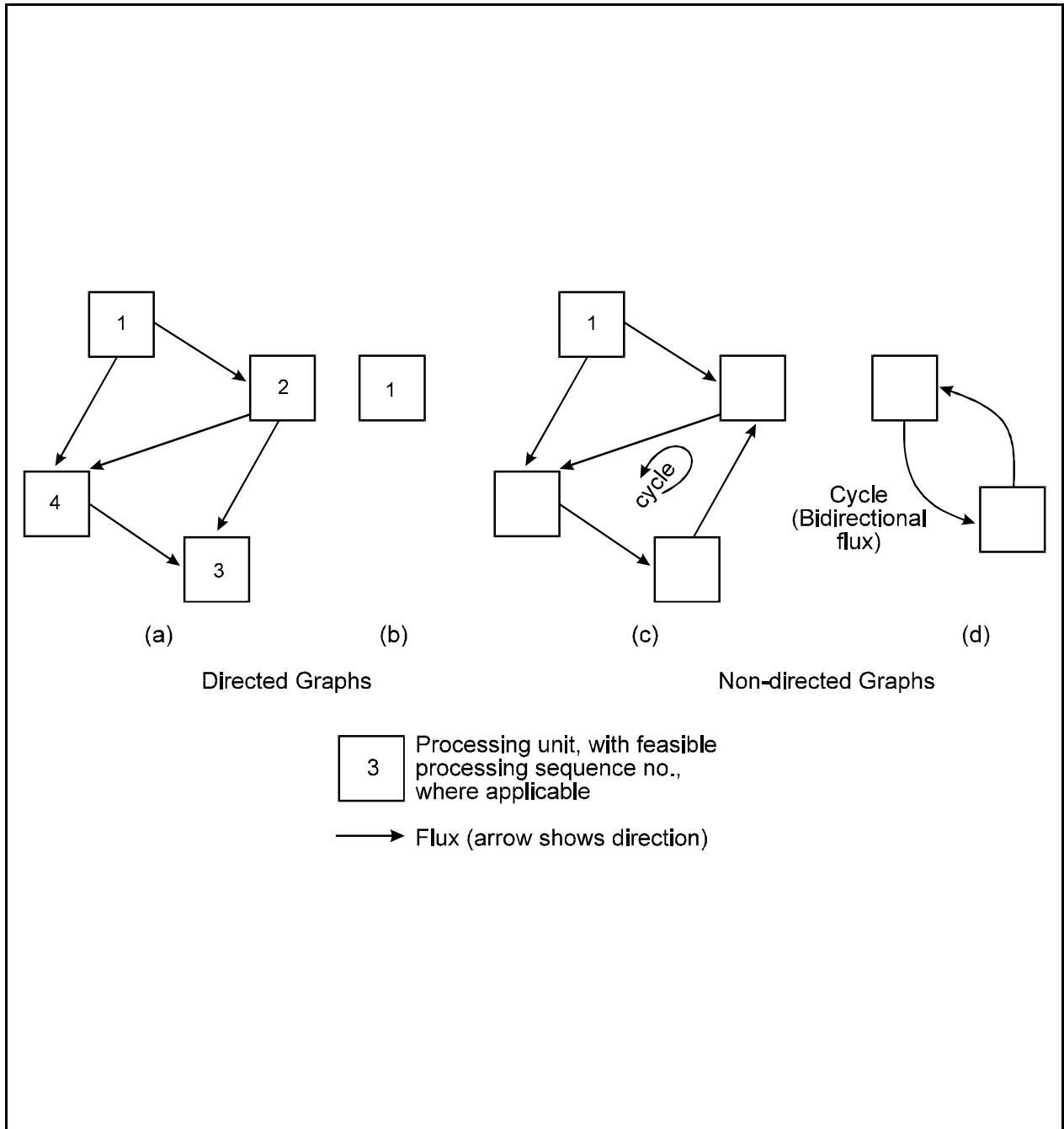


Figure 1-2 Directed and Non-directed graphs

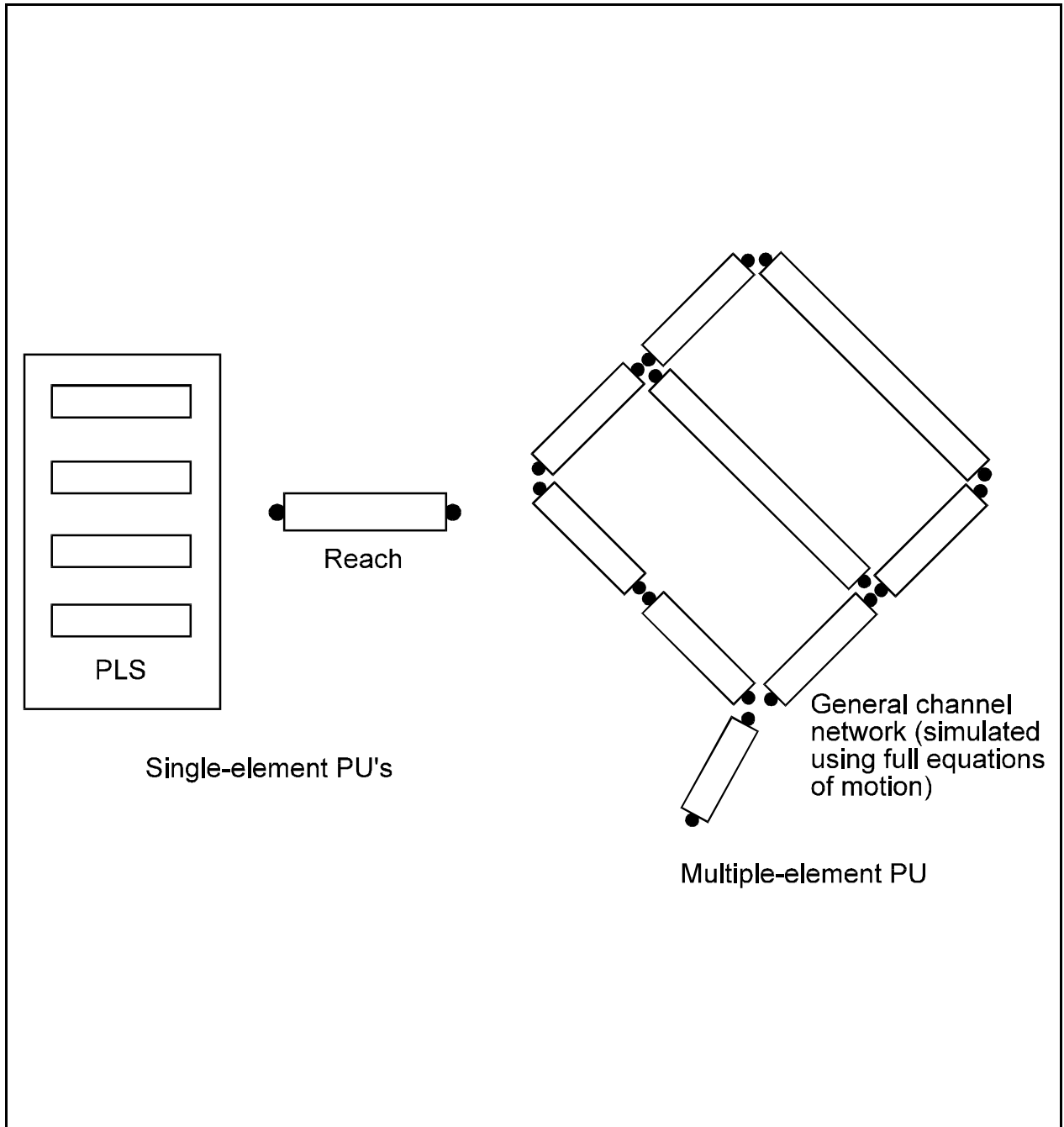


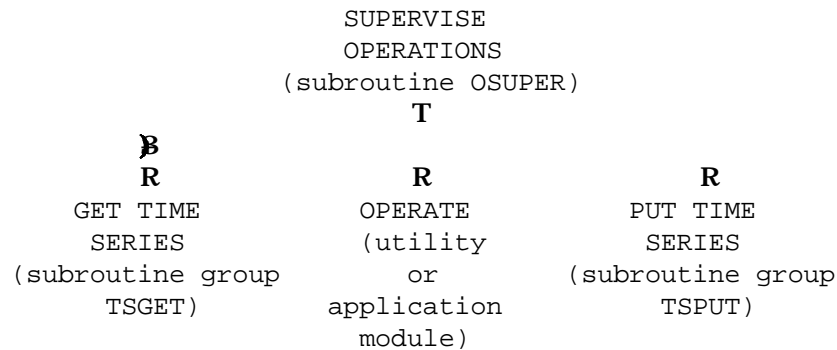
Figure 1-3 Single- and multi-element processing units

The processes which occur within a PU are represented mathematically in an "application model." The corresponding computer code is called an "application module" or "simulation module."

2.0 SOFTWARE STRUCTURE

2.1 Concept of an Operation

A great variety of activities are performed by HSPF; for example, input a time series to the WDM file, find the cross correlation coefficient for two time series, or simulate the processes in a land segment. They all incorporate two or more of the following functions: get a set of time series, operate on the set of input time series to produce other time series, and output the resulting time series. This applies both to application modules (already discussed) and to utility modules, which perform operations ancillary or incidental to simulation. Thus, a simulation run may be viewed as a set of operations performed in sequence. All operations have the following structure:



The OPERATE function is the central activity in the operation. This work is done by an "operating module" (OM) and its subordinate subprograms. They operate for a specified time on a given set of input time series and produce a specified set of output time series, under control of the "operations supervisor" (OSUPER). All of the pieces of time series involved in this internal operation have the same interval and duration. They are therefore viewed as written on an "internal scratch pad" (INPAD), resident in the memory of the computer (Fig. 2-1). The operating module receives the scratch pad with some rows filled with input and, after its work is done, returns control to the supervisor with another set of rows filled with output. The operating module may overwrite an input row with its own output. The computing module being executed, together with the options being invoked, will determine the number of rows required in the INPAD. For example, simulation of the hydraulic behavior of a stream requires relatively few time series (eg. inflow, depth and outflow) but the inclusion of water quality simulation adds many more time series to the list. Now, the total quantity of memory space available for storage of time series is also fixed (specified in a COMMON block) by the options in effect; this is the size (area) of the INPAD. Since both the size (N*M) and number of rows (M) in the INPAD are known, the "width" (no. of intervals,N) can be found. The corresponding physical time is called the "internal scratch pad span (INSPAN)."

| Row Number | Time Interval Numbers | | | | | | | |
|---------------|-----------------------|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | — | N |
| 1 | | | | | | | | |
| 2 | | | | | | | | |
| 3 | | | | | | | | |
| 4 | | | | | | | | |
| 5 | | | | | | | | |
| — | | | | | | | | |
| — | | | | | | | | |
| M | | | | | | | | |

NOTE: there is one time series per row.

Figure 2-1 Logical structure of the internal scratch pad

The "get time series" function prepares the input time series. This work is done by a subroutine group called TSGET. It obtains the correct piece of a time series from the appropriate file, aggregates or disaggregates it to the correct time interval, multiplies the values by a user specified constant (if required), and places the data in the required row of the internal scratch pad. Subroutine group TSPUT performs the reverse set of operations. TSGET and TSPUT are sometimes bypassed if a required time series is already in the INPAD when the operation is started, or if the output is being passed to the next operation via the internal scratch pad.

2.2 Time Series Storage

The time series used and produced by an operation can reside in four types of storage.

1. The Watershed Data Management (WDM) File

The WDM file has replaced the TSS as the principal library for storage of time series. As far as the computer's operating system is concerned, it consists of a single large direct access file. This space is subdivided into many data sets containing individual time series. Each is logically self-contained but may be physically scattered through the file. A directory keeps track of data sets and their attributes. Before time series are written to the WDM file, the file and its directory must be created using the interactive program ANNIE, which is documented separately.

2. The Hydrologic Engineering Center Data Storage System (DSS)

The DSS is the primary hydrologic data storage system of the U.S. Army Engineers Hydrologic Engineering Center (HEC). It is similar in design and function to the WDM file. A DSS file consists of a single, large, direct-access file containing many individual data sets that are identified by unique identifiers called "pathnames". A pathname is a string of characters from 5 to 80 characters long, and similar in construction to a file pathname on computer disk. Creation and maintenance of DSS files is typically performed by a utility program such as DSSUTL, which is documented separately.

3. Sequential Files

These are ASCII, formatted disk files with a constant logical record length. Time series received from agencies such as the National Weather Service are typically stored in sequential files.

4. Internal Scratch Pad (INPAD)

If two or more operations performed in sequence use the same internal time step, time series may be passed between them via the INPAD. Successive operations may simply pick up the data written by the previous ones, without any external (disk) transfer taking place. This is typically done when time series representing the flow of water (and constituents) are routed from one stream reach to the one next downstream.

2.3 Time Series Management For An Operation

Any operation involves a subset of the activities shown in Fig. 2-2. The operating module expects a certain set of time series in the INPAD. The operations supervisor, acting under user control, ensures that the appropriate input time series are loaded from whichever source has been selected, and informs the computing module of the rows in the INPAD where it will find its input. Similar arrangements hold for output of time series.

2.4 HSPF Software Hierarchy

The hierarchy of functions in HSPF is shown in Fig. 2-3. Some explanatory notes follow.

The "Run Interpreter" is the group of subprograms which reads and interprets the "Users Control Input." It sets up internal information instructing the system regarding the sequence of operations to be performed. It stores the initial conditions and the parameters for each operation in the appropriate file on disk and creates an instruction file which will ensure that time series are correctly passed between operations, where necessary.

The "Operations Supervisor" is a subroutine which acts on information provided by the Run Interpreter, invoking the appropriate "application" or "utility" modules. It provides them with the correct values for parameters and state variables by reading the files created by the Run Interpreter.

Operating modules are either "application modules" or "utility modules." They perform the operations which make up a run. Each time one of those modules is called, an operation is performed for a period corresponding to the span of the internal scratch pad (INSPAN). The Operations Supervisor ensures that the correct module is invoked.

"Service subprograms" perform tasks such as reading from and writing to time series storage areas, adding T minutes to a given date and time, to get a new date and time, etc.

The "Time Series Management System" (TSMS) consists of all the modules which are only concerned with manipulation of time series or the files used to store time series. It includes the WDM management functions, and TSGET and TSPUT.

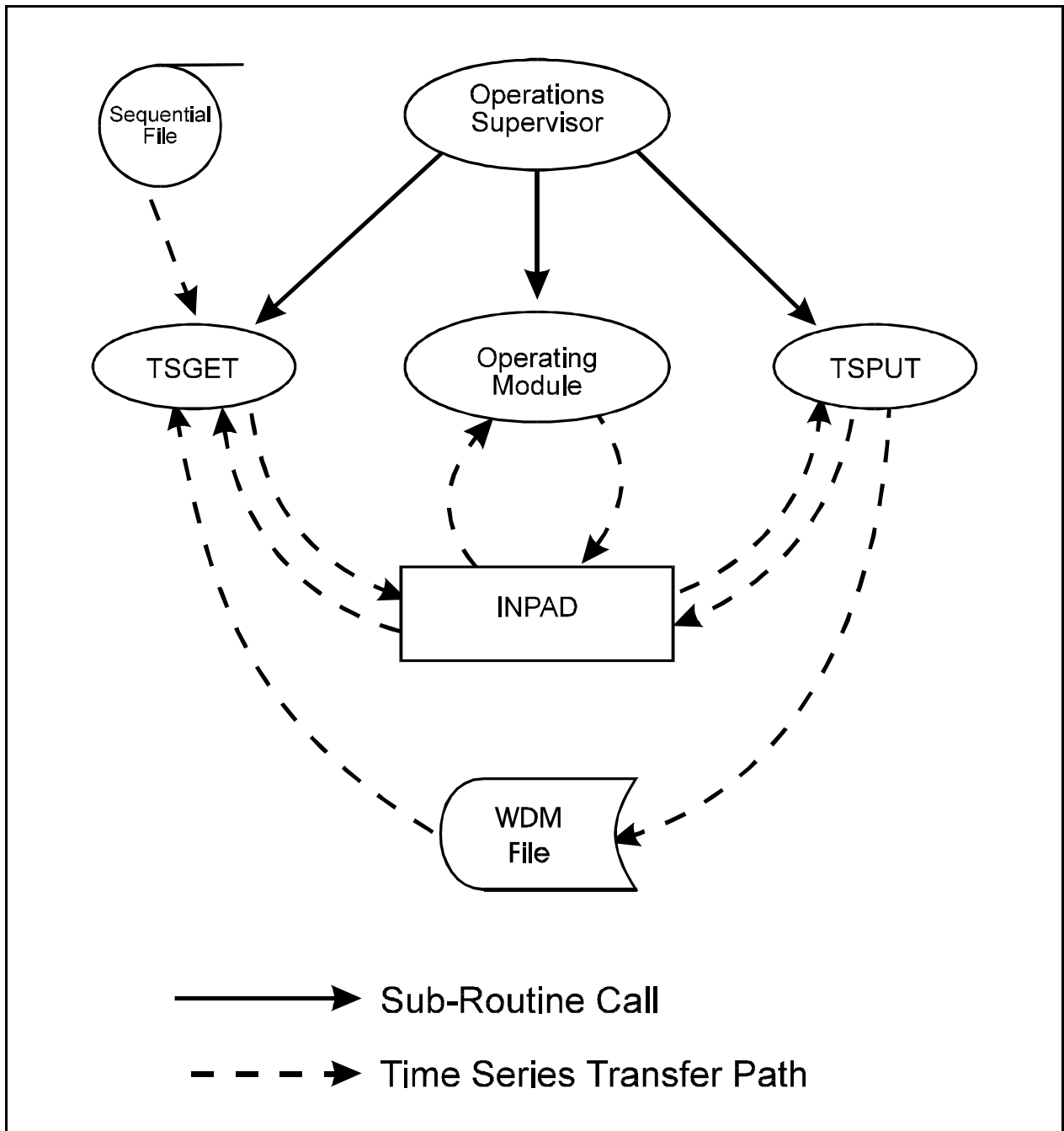


Figure 2-2 Activities involved in an operation

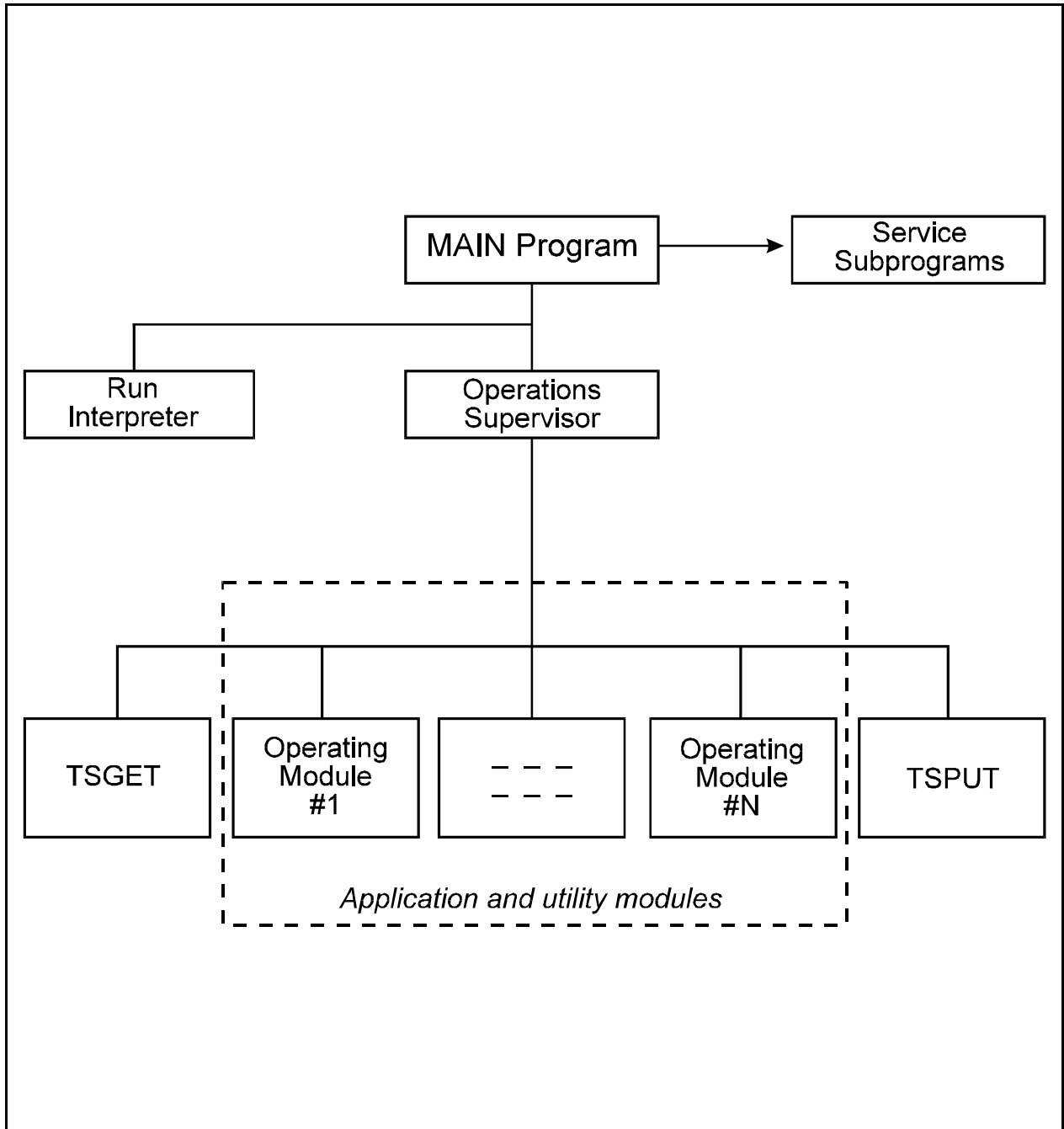


Figure 2-3 Overview of HSPF software

3.0 STRUCTURE OF A JOB

3.1 Elements of a Job

A "JOB" is the work performed by HSPF in response to a complete set of Users Control Input. It consists of one or more "RUNs". A RUN is a set of operations which can be performed serially, and which all cover the same period of time (span). The operations are performed in a sequence specified in the Users Control Input. To avoid having to store large quantities of intermediate data on disk, operations may be collected in a group in which they share a common INPAD (INGRP).

3.2 Groups Of Operations

In most runs, time series have to be passed between operations. As described in Section 2.2, each operation can communicate with four different time series storage areas: the WDM file, DSS file, the INPAD, and sequential files. This is illustrated in Fig. 3-1.

Potentially, any time series required by or output by any operation can be stored in the WDM file, DSS, or a sequential file. The user simply specifies the exact origin or destination for the time series, and the HSPF system moves the data between that device and the appropriate row of the INPAD. This system can also be used to transfer data between operations. However, it does require that all transferred data be written to the WDM file, DSS, or a sequential file. This may be very cumbersome and/or inefficient and it is better to transfer data via the INPAD, where possible.

To transfer data via the INPAD, operations must share the same pad. This means that all time series placed in the pad have the same time interval and span. This requirement provides a logical basis for grouping operations; those sharing a common INPAD are called an INGRP (Fig. 3-1). The user specifies the presence of groups in his "Users Control Input (UCI)." A typical sequence of input is shown in Fig. 3-2.

The user also indicates (directly or indirectly) in the control input the source and disposition of all time series required by or output by an operation. If the user indicates that a time series must be passed to another operation then the system assumes that the transfer will be made via the scratch pad. If they are not in the same INGRP there is an error. Without a common INPAD, the data must go via the WDM file or DSS. The structure of the Users Control Input is documented in Part F.

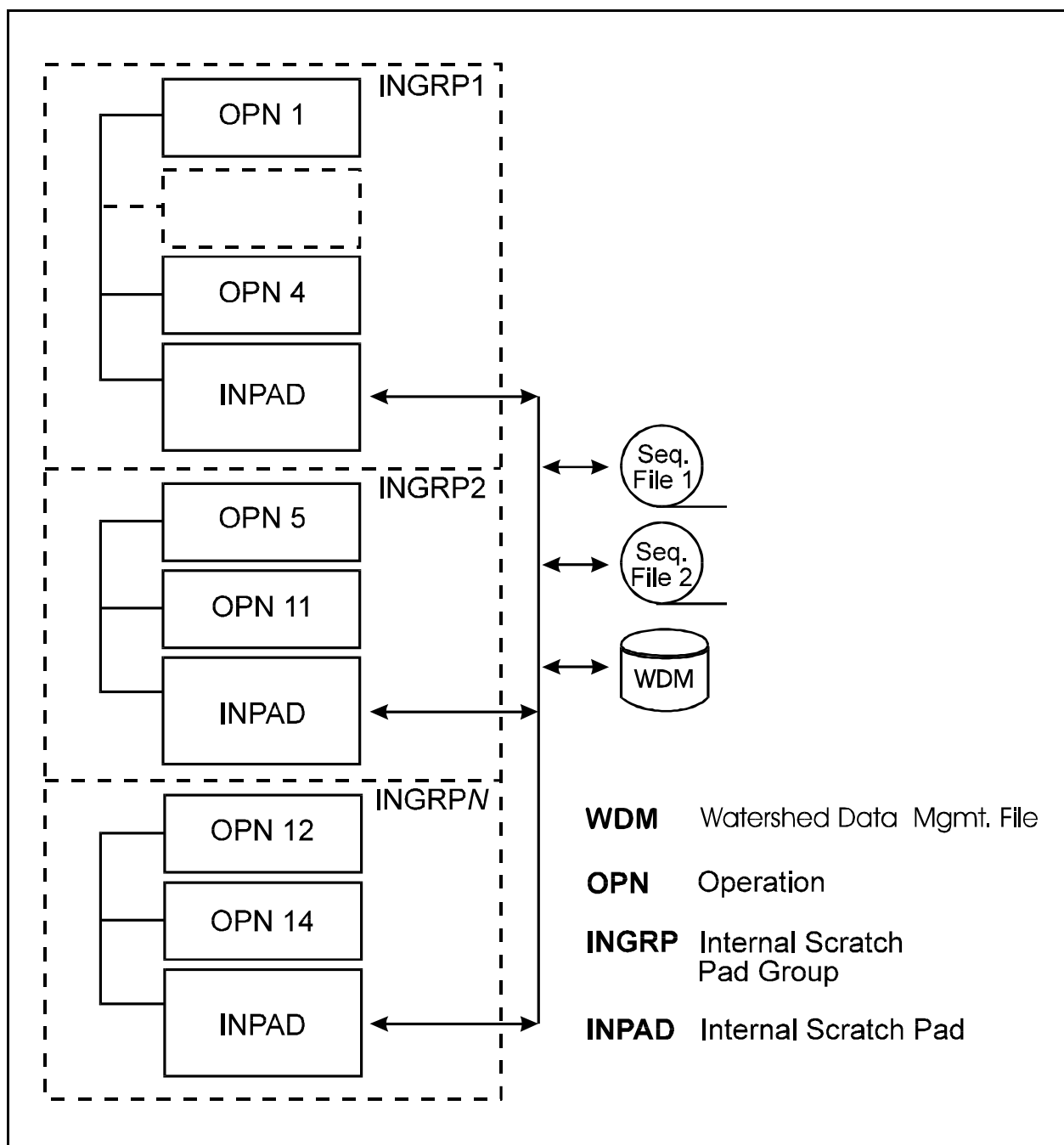


Figure 3-1 Schematic of data flow and storage for a single run

General Principles

The sequence of events in a run is as follows (refer to Fig.3-1).

- a. Operation 1 is performed until its output rows in the INPAD are filled.
- b. Data are transferred from those rows to other time series storage areas, as required. If any of these data are not required by other operations in INGRP1, their INPAD rows are available for reuse by other operations in INGRP1.
- c. Steps (a) and (b) are repeated for each operation in INGRP1.
- d. Steps (a), (b), and (c) are repeated, if necessary, until the run span is complete.
- e. The INPAD is reconfigured and work on operations 5 through 11 proceeds as in steps (a-d) above. The step repeats until all INGRP's have been handled. The run is now complete.

Note that reconfiguration of a scratch pad implies that its contents will be overwritten.

```
OPN SEQUENCE
  INGRP                                INDELT = 00:30
    COPY      1
    PERLND    1
  END INGRP
    PERLND    2                                INDELT = 00:30
    PERLND    3                                INDELT = 00:20
  INGRP                                INDELT = 00:30
    COPY      2
    RCHRES    1
    RCHRES    3
    RCHRES    5
    RCHRES    20
    RCHRES    22
    RCHRES    23
    RCHRES    7
    RCHRES    8
    RCHRES    50
    RCHRES    100
    RCHRES    200
  END INGRP
  INGRP                                INDELT = 00:10
    DURANL    1
    PLTGEN    1
  END INGRP
END OPN SEQUENCE
```

Fig. 3-2 Extract from typical Users Control Input,
showing how grouping of operations is specified

4.0 CONVENTIONS USED IN FUNCTIONAL DESCRIPTION

The primary purpose of the Functional Description (Part E) is:

1. to describe the functions performed by the various subprograms
2. to explain the technical algorithms and equations which the code implements.

Subprograms are described in numerical order in the text. This system provides a logical progression for the descriptions. General comments regarding a group of subprograms can be made when the "top" subprogram is described, while details specific to an individual subordinate subprogram can be deferred until that part is described. For example, a general description of the PERLND module (Section 4.2(1)) is followed by more detailed descriptions of its twelve sections, ATEMP (Section 4.2(1).1) through TRACER (Section 4.2(1).12).

5.0 METHOD OF DOCUMENTING DATA STRUCTURES

5.1 Structure of Data in Memory

The way in which we arrange the variables used in our programs is important. We structure them, as far as possible, using techniques like those used in Structured Program Design. We try to group data items that logically belong together.

Most of the variables in an Operating Module are contained in the Operation Status Vector (OSV). The OSVs for the application modules are shown in the Programmer's Supplement (Johanson, et al. 1979). The format used to document a data structure is similar to that used to declare a "structure" in PL/1. We do this because the technique is logical and convenient, not because of language considerations.

5.2 Structure of Data on Disk Files

The HSPF system makes use of two different types of disk-based data files:

1. Watershed Data Management (WDM) file and HEC Data Storage System (DSS) files contain time series data input and output.
2. The message/information file (HSPFMSG.WDM), is a read-only, binary file that contains information used by the program, such as keyword names, input formats, parameter defaults and limits, and error/warning messages.

6.0 METHOD OF HANDLING DIAGNOSTIC MESSAGES

HSPF makes use of two kinds of diagnostic message; error messages and warnings, which are printed to the Run Interpreter Output file during both the Run Interpretation and simulation phases of a run. These messages are all stored on the "message/information" file: HSPFMSG.WDM. This system for storing the messages has at least two advantages:

1. Because the messages are not embedded in the Fortran, they do not normally occupy any memory. This reduces the length of the executable code.
2. The files are easier to maintain than if the messages were embedded in the code. A user can obtain a listing of the contents by "exporting" data sets from the message file using the ANNIE program's Archive function.

Each message has been given a "maximum count". If the count for a message reaches this value, HSPF informs the user of the fact. Then:

1. If it is an error message, HSPF quits.
2. If it is a warning, HSPF continues but suppresses any future printing of this message.

In addition to the above features, the Run Interpreter has been designed to:

1. Stop if 20 errors of any kind have been detected. This gives the user a fair number of messages to work on, but avoids producing huge quantities of error messages, many of which may be spurious (say, if the code could not recover from early error conditions).
2. Stop at the end of its work if any errors have been detected by it. Thus, HSPF will not enter any costly time loop if the Run Interpreter has found any errors in the User's Control Input.

PART C

STANDARDS AND CONVENTIONS

This section has been omitted.

PART D

VISUAL TABLE OF CONTENTS

This section has been omitted.
