Journal of Hydroscience and Hydraulic Engineering
Vol. 14, No. 2, November, 1996, 47-55

A RUNOFF SIMULATION WITH STRUCTURAL HYDROLOGICAL MODELING SYSTEM

By

Takuma Takasao

Department of Civil Engineering, Kyoto University, Sakyo-ku, Kyoto, 606, Japan

Michiharu Shiiba

and

Yutaka Ichikawa

Disaster Prevention Research Institute, Kyoto University, Uji, Kyoto, 611, Japan

SYNOPSIS

Regarding the runoff system as a set of dynamic elements which communicate with one another, we develop a new system for modeling the runoff system. In our system, the basic and common operations in runoff simulation, such as initialization of states, giving the values of parameters, giving and receiving data, setting time steps, are standardized and modeled as functions of the base model of runoff elements. The users have only to derive their models from the base model and do not need to write the codes for the basic and common operations as stated above. Our system is flexible enough that it can treat various types of communications among elements. As an example, a model for simulating the flow in a river network is shown.

INTRODUCTION

When creating a hydrological model of a basin, we may follow the method where (1) we divide the basin into some elements such as slopes, reservoirs, channel networks; or divide the water circulation in the basin into some elements such as rainfall, evapotranspiration, groundwater flow; (2) we prepare a library of element models which correspond to the elements; (3) we plug them into a total system model that is appropriate to the basin. We call such a method a structural modeling method. There are following advantages in this method.

First, because many element models have already been developed, when making a total system model, we have only to select some element models required, give the numerical values to parameters and state variables, and combine them according to the structure of the basin (Fig. 1). Second, even if a total system model must be updated because the modification of land use changes the hydrological characteristics of a part of the basin; or a new hydrological model is developed for some element, we have only to exchange the corresponding model to the element which needs to be replaced (Fig. 1).

Most hydrological simulations now use a numerical analysis by computer. If applied to the computer simulations, this method makes it possible to efficiently build and modify a total system

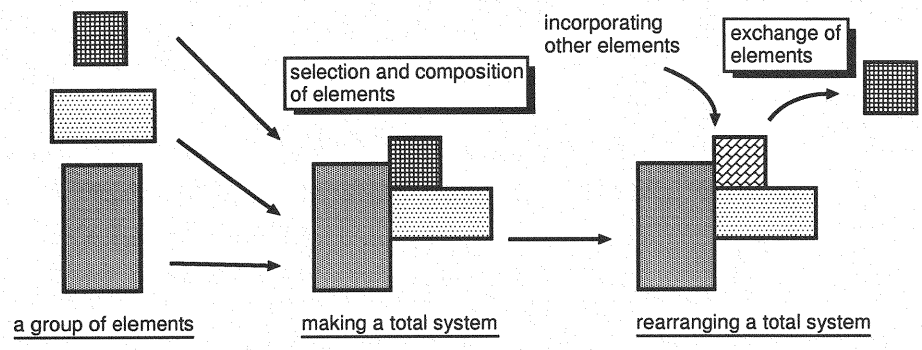

Fig. 1 Structure of a total system model

model. Therefore, we develop the software system — the structural hydrological modeling system — to apply this method to the hydrological computer simulations.

As a series of studies on the structural hydrological modeling system, we proposed its basic concept and developed the prototype edition in 1992. In 1993, we wrote the improved edition and made a library of element models. In 1994, we (6, 4) completed our system by improving the 1993 edition. This paper explains the outline of our system and shows a runoff simulation using the 1994 edition.

DERIVATIONS

Problems of Existing Models / Modeling Systems

A few existing hydrological models, like HEC–1, HSPF, and so on, include a library of models for specific elements, and a user can then select element models to suit his requirements (5). Such models have the advantage that the user is released from the laborious works of making and programming element models, so long as he uses the element models that the development organization provides.

However, when a user personally develops a new element model, and tries to connect it with a model which is provided by the development organization, a problem occurs. For a compatible connection between both models, the user must build and program his element model according to the method of data transfer among element models, and the way by which a total system model operates element models. This forces the user to understand the codes of the provided models in detail. Most of such models have repeatedly passed through numerous extensions since they were first developed, so their codes are usually complex and enormous. Therefore, it is practically impossible for a user to add his new model to the provided one. The revision of the provided model is also difficult for the similar reason.

Basic Requirements and the Way of Building the Structural Hydrological Modeling System

To realize the structural modeling method by computer, the structural hydrological modeling system must at least be able to combine a number of element models by defining the relation and the way of inputting / outputting among them, and must be able to smoothly operate a total system model. Moreover, even if some element models are exchanged or connections among them are changed in a total system model, this system must be able to smoothly operate the total system model.

Furthermore, the consideration in the preceding section implies that a user should be able to easily introduce his newly developed models into the system in order to make the system more practical. By summing up the above, we obtain the following basic requirements that the system should satisfy:

requirement 1: The system must standardize the way of inputting / outputting data and the procedures of operating element models.

requirement 2: The system must make standardized specifications open to the user.

requirement 3: The system must make it easy to build an element model according to the standardized specifications.

requirement 4: The standardized specifications must not restrict definitions of personal functions in an element model.

Some existing models / modeling systems can satisfy requirements 1 and 2. For example, such models / modeling systems have only to provide the manual in which the standardized specifications are described in detail. However, this does not resolve the faults of a modeling system stated in the preceding section. In addition, each user must write the code of his element model according to the standardized specifications by himself. Thus his model cannot guarantee complete reliability in regard to the use of the standardized specifications. Besides, when some changes are added to the standardized specifications, a user must rewrite the code of his element model according to the new specifications. In stating as such, it appears to be difficult for existing models / modeling systems to satisfy all of the four requirements.

Therefore, we adopt the following policies for building our system with particular attention to requirements 3 and 4:

policy 1: The standardized specifications of an element model are defined as functions of the abstract base element model (see Fig. 2).

policy 2: A user makes his element model by combining the model's proper functions with the base element model (see Fig. 3).

policy 3: All element models possess common functions that the base element model provides.

Policies 1 and 2 imply that a user can readily realize the standardized specifications in his element model. Also, because policy 2 means that each element model can be added to optional functions, requirement 4 can be satisfied. Furthermore, because policy 3 says that the improvement of the standardized specifications corresponds to the improvement and the exchange of the base element model, a user does not have to rewrite his element model even if the standardized specifications are changed.

To realize the policies above, we describe our system in an object–oriented computer language because it is equipped with the following useful programming concepts: 1. class, 2. inheritance, and 3. polymorphism, and they are advantageous to build our system. These concepts and their use are briefly explained below. For further details of these concepts, see, for example, Wiener and Pinson (7).

Class: A class is a package that contains several data and several functions that operate data. Data in a class are called data members and functions in a class member functions. An element model executes a hydrological computation by operating data such as state variables and parameters in the equations which express hydrological characteristics. So, taking state variables and parameters

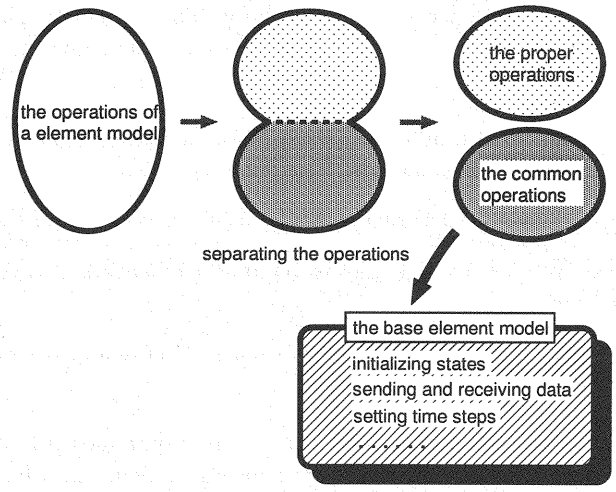

Fig. 2 Abstract base element model

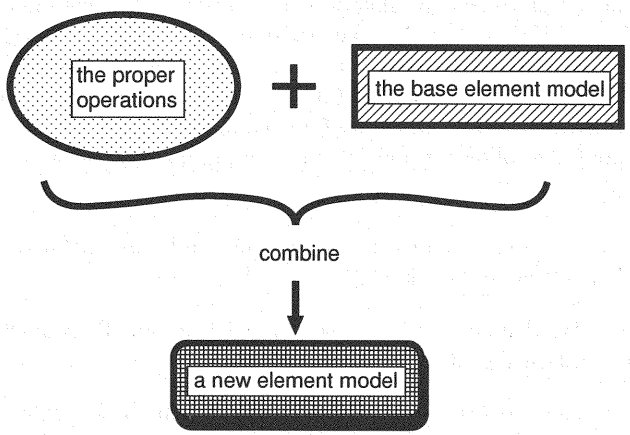

Fig. 3 Building of a new element model

as data members and equations as member functions, an element model can be expressed as a class.

Inheritance: Let us suppose that there are two classes, A and B, and that class B has some personal data members and member functions in addition to those of class A. Now, a user does not need to build class B independently of A. He can derive B from A. If only declaring that B is derived from A, he does not have to define the data members / member functions of A within the code of B. In the code of B, he has only to declare that B is derived from A, and add the data members / member functions specific to B. In this case, class A is called a base class and class B a derived class. By only declaring the derivation from A, B takes over the members of A. This is called inheritance (see Fig. 4).

In our system, we prepare the base element model as a base class from which a user derives his element model. The user can realize the standardized specifications in his model by only

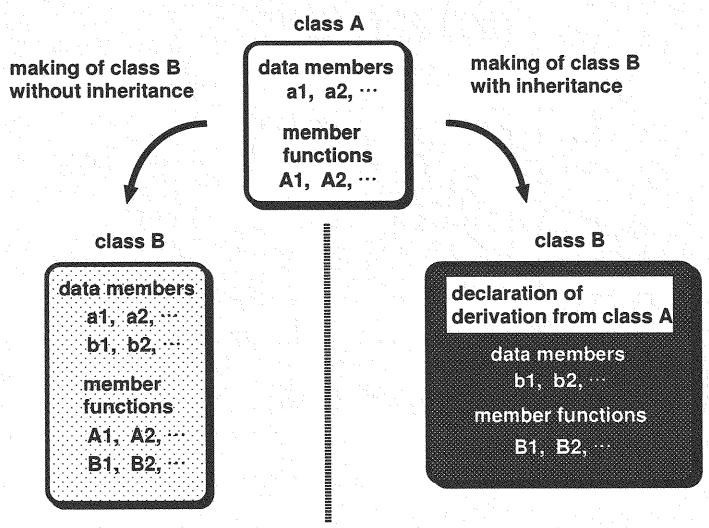

Fig. 4 Inheritance

declaring the derivation from the base element model. Then he can efficiently program his model.

Polymorphism: From the example above, it can be said that inheritance is a useful concept if both classes A and B are equipped with the very same function. However, even if it is found that both A and B should be equipped with a certain function, B may need to define the contents of its function to be different from A. For example, all element models should be equipped with the function of making a computation according to each hydrological characteristics, but the contents of the function are different respectively. However, "computing" is common to all element models, so if the procedures of computing are common among various element models, a user can more easily and more generally operate them.

In this way, even if the usage of a member function is standardized, each derived class can individually define the contents of the function. This is called polymorphism. For example, in C++, which is an object–oriented computer language, it is realized as the virtual function.

We describe our system in C++. C++ is only an extension of C so that it can deal with the object–oriented approach, thus its basic syntax is common to C. C is also a popular language in numerical analysis on the computer as well as FORTRAN. Because we want to provide our system for many users and assume that the users make their own element models, we think C++ is more appropriate than any other existing object–oriented languages.

Building the Structural Hydrological Modeling System

Outline of the system

In the following, see Fig. 5 when necessary. As we stated in the first section, we divide a basin into several sub–basin elements and combine those element models to make a total system model of the basin. Therefore, it is an element model that performs an actual hydrological computation, and a total system model has only to: (1) do interactive operations with a user, (2) transfer data to and from files, and (3) command its element models to compute, so it does not need to do personal hydrological computation in principle.

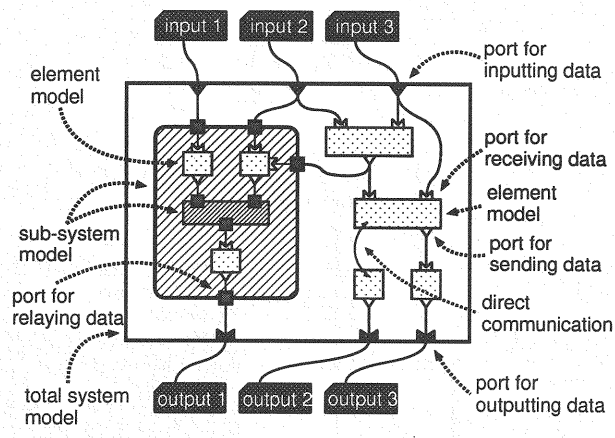

Fig. 5 General composition of a total model with the structural hydrological modeling system

Our system also uses the concept of "a sub–system model". A sub–system model expresses a sub–system within a total system, and defines connections among several elements and subordinate sub–systems within itself.

As an example of this concept's usefulness, let us suppose that we try to model water movement in soil. Water movement in soil consists of two dominant phenomena. One is interflow and the other is groundwater flow. If we make each element model for these two flows, and build a sub–system model by connecting them in advance, we do not have to connect them every time we make a model of water movement in soil.

Like a total system model, a sub–system model also does not perform its own hydrological computation in principle. It only commands its element models to compute.

In our system, we prepare the base models for a sub–system / total system as well as an element model. Thus a user can create his own sub–system / total system model as he likes and can efficiently operate them in standardized procedures.

Moreover, to unify the way of inputting / outputting data and to more freely transfer data among various types of models, we use the data pack model which represents a standardized data structure and the port model through which data are sent / received. Inputting data to an element model corresponds to receiving data at a port for receiving data, and outputting data from an element model corresponds to sending data from a port for sending data. In the same way, we use a port for relaying data when inputting to / outputting from a sub–system model, and use a port for reading / writing files for inputting to / outputting from a total system model. We prepare the base data pack model and the base port model, so a user can also make his own data pack / port model as he likes.

Usual data exchanges among element models in our system are carried out through ports. However, data exchanges through ports cannot be used in the case where we cannot know which data of a sender are necessary for a computation of a receiver until that moment, because it restricts data flow with one direction from a sender to a receiver. Therefore, we develop the method in which a receiver can inform a sender which data are needed. This method is called the direct communication method.

Developing a model by a user

In our system, a user can develop his own models as he likes deriving his models from various base models stated above. Here we explain the way of developing an element model which is the

core to model building. When developing an element model, a user has to:

1. derive his model from the base element model,
2. provide one or more ports for receiving / sending data,
3. define the functions which register ports,
4. define parameters and state variables,
5. define the functions which give the numerical values to parameters and give the initial values to state variables,
6. define the function which sends data at the beginning of the simulation,
7. define the function which determines the time step for the next computation,
8. define the function which judges whether or not to execute the next computation,
9. define the function which does computation and sends data, and
10. define the function which does a series of computational work until the simulation ends.

Flow chart of a simulation

Figure 6 shows the flow chart of a main program for a standard simulation. Because all element models are operated by the total system model which owns them, a user has only to describe some commands to the total system model in a main program.

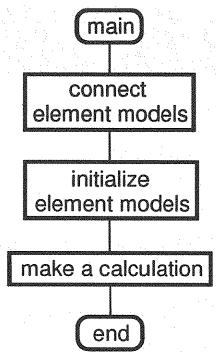

Fig. 6 Flow chart of a main program for a standard simulation

Example of a Simulation

As an example of a simulation using our system, a model for simulating the flow in a river network is shown below. A simulation of the flow in a river network has often taken advantage of the graph theory (e.g. (2)), but our system makes it possible to simulate the flow in a river network without the help of the graph theory. As will be described below, this simulation is also an example of the direct communication method. We supposed the virtual stream network shown in Fig. 7. We divided this into four stream elements (see Fig. 8), and created the total system model combining four element models (see Fig. 9). All the element models applied here route the flood flow using dynamic wave theory:

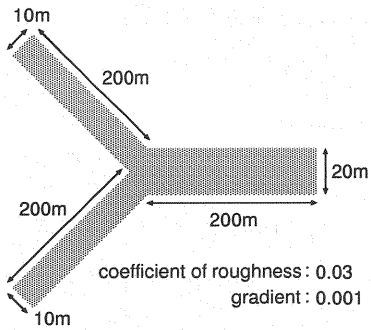

Fig. 7 Virtual stream network

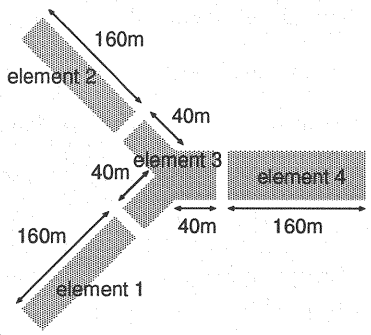

Fig. 8: Division of the stream network into four elements

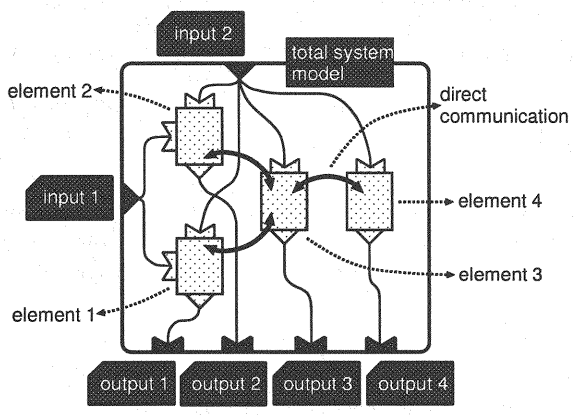

Fig. 9 Making of a total system model

Element 1, 2: Upper reach model

It receives a discharge hydrograph at a port for the upper boundary and a lateral inflow hydrograph at another port. In addition, it obtains the data which are necessary to compute state variables at the lower boundary by directly communicating with the element on the lower side.

Element 3: Confluence model

It receives a lateral inflow hydrograph at a port, and obtains the data which are necessary to compute state variables at both the upper boundary and the lower one by directly communicating with the element models on both sides.

Element 4: Lower reach model

It has a rating curve, which is a relation formula between the discharge and the depth at the lower boundary. It receives a lateral inflow hydrograph at a port, and obtains the data which are necessary to compute state variables at the upper boundary by directly communicating with the element model on the upper side.

When making these element models, we referred to Iwasa (1) and JSCE (3), and used the characteristic curve method to analyze the flood flow in these models. Figure 10 shows the result of the simulation.

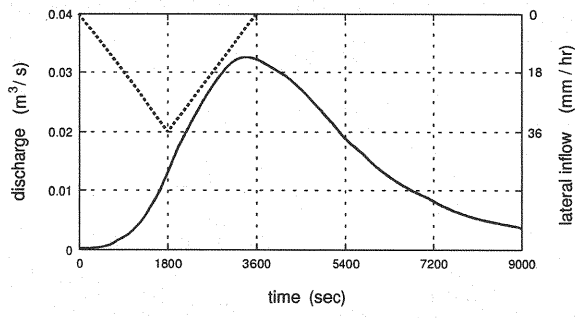

Fig. 10 Result of the simulation

Conclusion

In this paper, we explained the structural hydrological modeling system, and actually made a runoff simulation by creating and combining several element models.

In the future, we must develop a function by which an iterative calculation can be made among element models. Now, groups of processes that need an iterative calculation must be included in the same element model. This function will make it possible to more freely compose a total system model.

We distribute our system through the Internet. Access `ftp://rdp.dpri.kyoto-u.ac.jp/pub/` by anonymous ftp, please. We are glad if it could serve many people who make hydrological models.

REFERENCES

1. Iwasa, Y., K. Inoue, and T. Katayama : On the numerical methods for unsteady flow in open channel (in Japanese), Annuals, Disas. Prev. Res. Inst, Kyoto Univ., No. 19, B-2, pp. 187–200, 1976.

2. JSCE ed. : Numerical Analysis in the Field of Civil Engineering / Hydrodynamic Analysis (in Japanese), Saiensu-sha Co., Ltd., pp. 119–133, 1974.

3. JSCE ed. : Collected Formulas on Hydraulics (in Japanese), pp. 206–219, 1985.

4. Kishimoto, Y. : Structural modeling system for runoff systems and dynamic wave model (in Japanese), Graduation thesis, Department of Civil Engineering, Kyoto Univ., 1994.

5. Ponce, V. M. : Chapter 13 : Catchment modeling, Engineering Hydrology –Principles and Practices–, Prentice Hall Inc., New Jersey, pp. 389–451, 1989.

6. Suzuki, T. : Development of structural modeling system for runoff systems (in Japanese), Master thesis, Department of Civil Engineering, Kyoto Univ., 1994.

7. Wiener, R. S. and L. J. Pinson : Introduction to Object–Oriented Programming and C++, Addison-Wesley Publishing Company, Inc., 1988.

(Received December 25, 1995; revised September 17, 1996)