The Structured Design Method of Simulation Programs

Yukio KANEDA*

Abstract

A new design technique of discrete event simulation programs is proposed. The new world view "actor" is introduced. We regard a system is composed from several modules and actors play active roles of them.

By describing these actor routines, we can organize the simulation program easily.

1. Introduction

In proportion to increasing the complexity of the simulated system, the simulation program of it becomes complex and hard to program and debug.

We propose a new structured design method of discrete event type simulation programs. In case of writing a simulation program, it is very convenient if we can easily reflect the structure of the simulated system to the structure of its simulation program. However, in conventional simulation languages all event routines are arranged uniformly under the main routine. We introduced a new world view "actor" and made it easy to reflect the structure of the simulated system to the structure of its simulation program.

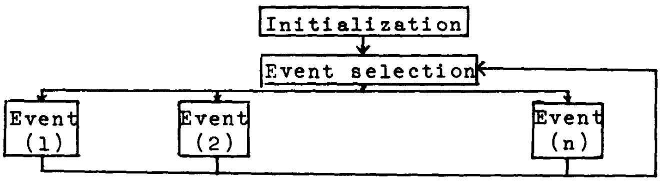

Fig.1 The control structure of the discrete event simulation program.

This paper first appeared in Japanese in Joho—Shori (Journal of the Information Processing Society of Japan), Vol. 17, No. 9 (1976), pp. 782~788.

* Faculty of Engineering, Kobe University

2. Problems of the conventional simulation languages

Conventional discrete event simulation languages have the following shortcommings.

1) All event routines are arranged uniformly under the main control routine. So it is very difficult to compose the program which explicitly reflect the object system's structure.

2) There are no effective facilities which prevent the ripple type errors.

3. Structured design method

We often find the simulation program is composed of the several event routine groups of which event routines are connectd relatively tight and that connections between these groups are soft.

Consequently, we can divide the movements of the control between event routines into two types.

1) Control movement between tightly connected event routines.

2) Control movement between softly connected groups.

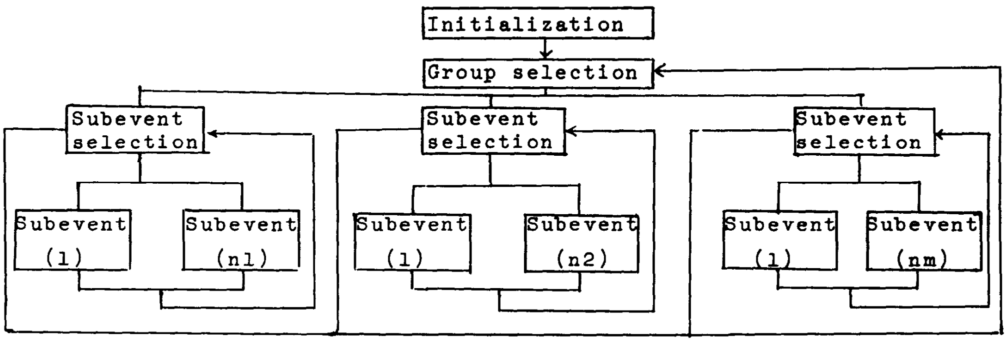

Fig.2 The control structure of a simulation program which is composed of several subevent routine groups.

3.1 Concept of an actor

We consider the system is organized of a set of actors each of which works concurrently and communicates with one another. Every actor routine is a description of one of the basic activities of the system. When an actor routine is activated (called), it recognizes its own situation and executes some activities.

These activities include following two functions.

1) To change the system environments which are represented by the system variables.

2) To send an activation signal to other actor.

3.2 Structure of the simulation program

A group routines which describe an actor are composed of several subevent routines and one control routine. The control routine rules locally its subordinate subevent routines. The complete simulation program is composed of these actor routines and one main control routine which rules these actor routines.

3.3 Control of routines

Simulation is made by dynamically calling these actor routines or subevent routines with each other. We categolize these calls into following three types: (1) calls bewteen subevent routines in an actor, (2) calls between subevent routines on account of time laps and (3) calls between actors.

Type (1) is controlled by the subevent selection part and type (2) is controlled by the main routine. All future subevents are sorted in order of their occurence time. The main routine removes the earliest subevent and updates current time and calles that subevent. Type (3) is also controlled by the main routine. When an actor activates another actor, it returns control to the main routine with the target actor number. The main routine checks its regularity and calls that actor.

4. Example

This section shows how to model a time sharing system by the actor concept.

4.1 Model system

Our time sharing system is consist of TTY terminals, one CPU, one main memory and one disc device.

Fig.3 Time sharing system

4.2 Model building

The simulation model is consist of three actors.

1) Actor(1) simulates the CPU and the main memory part. It has four subevents.

(1) Subevent(1) requests for a main memory area.

(2) Subevent(2) requests for the CPU.

(3) Subevent(3) releases the CPU for I/O.

(4) Subevent(4) releases the CPU and the main memory area for the end of job.

2) Actor(2) simulates the behaviour of the disc, and has two subevents.

(1) Subevent(1) is for initiating the disc I/O.

(2) Subevent(2) is for terminating the disc I/O.

3) Actor(3) simulates the behaviour of the users at terminals and has three subevents.

(1) Subevent(1) indicates an arrival of a user.

(2) Subevent(2) makes parameters for a job.

(3) Subevent(3) processes the end of a job.

5. Conclusion

The merits of our design method are followings.

1) It is easy to test the correctness of the program.

2) We can limit the size of actors as we can understand their dynamic behavior easily.

3) Each actor is relatively independent from others, we can make their side effects and ripple errors minimumn.

4) The system is divided into several inter independent actors, the modification of the program is easy.

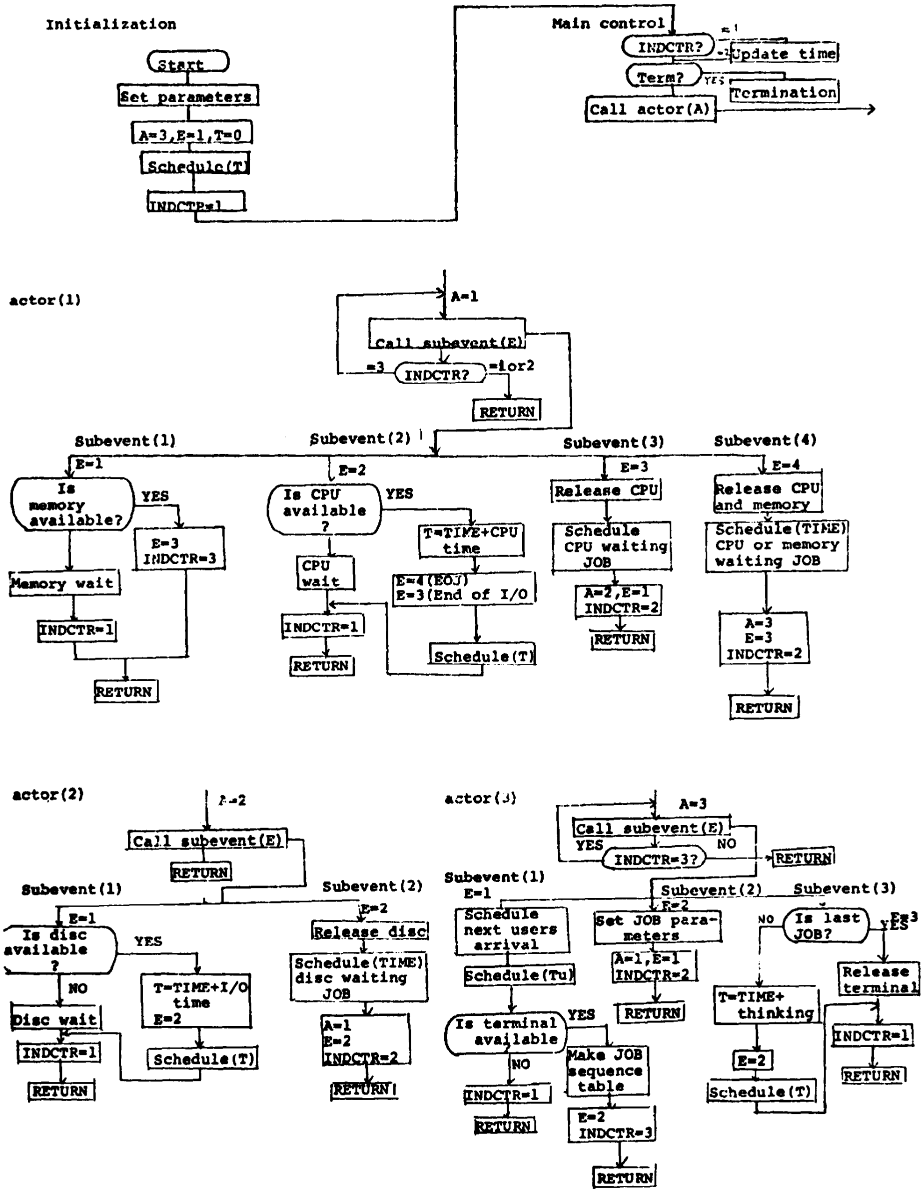

Fig.4 Control structure of the simulation program.