Simulation Modelling Practice and Theory 19 (2011) 785–800

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

The extended activity cycle diagram and its generality

Donghun Kang, Byoung K. Choi *

VMS Lab., Department of Industrial and Systems Engineering, KAIST, 335 Gwahak-ro, Yuseong-gu, Daejeon 305-701, Republic of Korea

A R T I C L E   I N F O

Article history:
Received 10 May 2010
Received in revised form 8 November 2010
Accepted 9 November 2010
Available online 18 November 2010

Keywords:
Activity cycle diagram (ACD)
Modeling power
Extended ACD
Three-phase rule

A B S T R A C T

The simplicity of the two-symbol activity cycle diagram (ACD) has made it easy to learn and use to describe discrete event systems in a natural way. But as the complexity of the model increases, the ability of the ACD to provide a full description of the system under consideration becomes more limited. This paper presents an extended ACD with arc attributes and its formal definition to increase the modeling power of the ACD. The validity of the proposed extensions is shown by proving the generality of the extended ACD, which has not yet been done on the ACD. In addition, this paper proposes a model specification for the simulation execution of the extended ACD models which we call the activity transition table (ATT). At last, the simulation software tool is presented to support model implementation, model execution, and experimentation for the extended ACD models.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

The *activity cycle diagram* (ACD) is a method to describe interactions among objects in a system. It provides the graphical modeling notations to explain a series of activities in diverse real-life circumstances. The core idea of the ACD was introduced by Tocher to solve the congestion problem at steel plants in a general framework; this took the form of the *flow diagram* [1] with the *three-phase rule* [2].

Objects in a system can be classified into two classes: (1) transient object or *entity* that receives the services and leaves the system; and (2) resident object or *resource* that serves the entities. In the ACD, the behavior or life cycle of an entity or resource in the system is represented by an activity cycle, which alternates between active and passive states. The passive state of an entity or resource is called a *queue* in a circle, and the active state is called an *activity* in a rectangle as shown in Fig. 1. The arc is used to connect the activity and the queue or the queue and the activity.

The activity represents the interaction between an entity and resource(s), which usually requires a *time delay* for completion. The token is used to represent the state of a queue or an activity. A queue cannot contain different types of tokens. All activity cycles are closed on themselves [3].

Fig. 2 shows an ACD model for a single-server system. This model consists of three activity cycles: one for an entity of "jobs" and the rest for the resources of "generator" and "machine". A job is generated at the interval of t_a time unit by a generator and stored in a queue "B" waiting for its processing on a machine. A ready-to-process machine serves a job for t_s time unit if a queue "B" has at least one job. The resource can usually perform one or more different activities in any sequence. All the activity cycles in Fig. 2 are closed.

The three-phase rule [2] provides a fundamental method to handle the flow of time in a discrete event simulation:

Phase A: Advance the clock to the time of the next (bound-to-occur) event.

* Corresponding author. Tel.: +82 42 350 3115; fax: +82 42 350 3110.
E-mail addresses: donghun.kang@vmslab.kaist.ac.kr (D. Kang), bkchoi@kaist.ac.kr (B.K. Choi).

1569-190X/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.simpat.2010.11.004

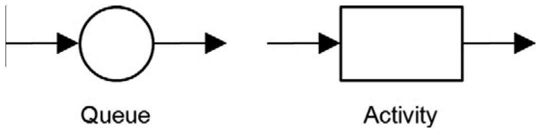

Fig. 1. Basic graphical modeling notations of the ACD.

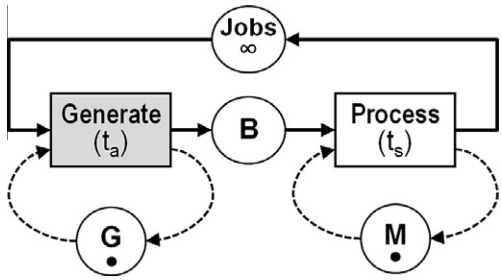

Fig. 2. ACD model of a single-server system.

Phase B: Terminate any activity bound to end at this time.
Phase C: Initiate any activity of which the condition in the model now permits.

The ACD represents the state flows of entities and resources in a system, while the three-phase rule is based on the event that denotes a state change of a system when it occurs. In phase B, the activities, which are bound to occur, end with the release of resources and entities (into output queues), which is called a bound or *bound-to-occur (BTO) event*. In phase C, any *conditional event* that satisfies the beginning condition represented by the availability of entities and resources is initiated by acquiring them.

Assume that the single-server system in Fig. 2 starts with a bound-to-occur (BTO) event ("Generated") for a "Generate" activity at time zero. At phase A, the simulation clock is advanced to zero because "Generated" is the only available BTO event and the time-to-occur of this event is zero. At phase B, a "Generated" BTO event is executed to terminate the "Generate" activity by adding a job token to a queue "B" and returning a resource token to a queue "G". At phase C, any activity of which all input queues have one or more tokens is initiated. Currently, every queue in the model has exactly one token. Therefore, the conditional events for the "Generate" and "Process" activities are now initiated by updating the token values of the input queues. Then, the BTO events for the "Generate" and the "Process" activities are scheduled to occur at $0 + t_a$ and $0 + t_s$ time. This process continues until it satisfies the end-of-simulation (EOS) condition.

Formally, the ACD model M can be defined as follows:

$M = (A, Q, I, O, \tau, \mu, \mu_0)$, where
$A = \{a_1, a_2 \ldots a_n\}$ is the finite set of activities,
$Q = \{q_1, q_2 \ldots q_m\}$ is the finite set of queues,
$I = \{i_a \subseteq Q \mid a \in A\}$ is the input function, a mapping from activities to sets of queues,
$O = \{i_a \subseteq Q \mid a \in A\}$ is the output function, a mapping from activities to sets of queues,
$\tau = \{\tau_a \in R_0^+ \mid a \in A\}$ is the time delay function,
$\mu = \{\mu_q \in N_0^+ | q \in Q\}$ is the finite set of the number of tokens for each queue,
$\mu_0 = \{\mu_1, \mu_2 \ldots \mu_m\}$ is the finite set of initial token values for each queue.

In the ACD, the activity represents the active state of entities and resources or the interaction between them, which results in a change of states in the system whenever it begins and ends. Therefore, the relationship between queues and activities is defined as a function of activities. The input function I represents the queues directly connected to an activity and output function O defines the queues directly connected from an activity. The time delay function τ defines the time that a resource(s) takes to serve an activity to the entity (ies). On the contrary, the time of the entities or resources in a queue is undefined, because it cannot be decided until at least one succeeding activity of a queue has satisfied its beginning condition.

An ACD model for a single-server system shown in Fig. 2 can also be represented formally as shown in Table 1. For example, the "Process" activity, $a_2 \in A$, has the input queues of "B" ($q_5 \in Q$) and "M" ($q_6 \in Q$) queues. This is defined by the element i_2:$\{q_5, q_6\}$ of the input function I.

This paper presents an extended ACD model with arc attributes to increase the modeling power of the ACD models. The generality of the extended ACD model is also demonstrated with two approaches: (1) conversion of the extended ACD model

Download English Version:

https://daneshyari.com/en/article/491821

Download Persian Version:

https://daneshyari.com/article/491821

Daneshyari.com