2003-01-01

Simulation Activity Diagrams

John Ryan
Technological University Dublin, john.ryan@dit.ie

Cathal Heavey
University of Limerick

Simulation Activity Diagrams

John Ryan Dr.
DIT, john.ryan@dit.ie

Cathal Heavey Dr.
UL

— Use Licence —

Attribution-NonCommercial-ShareAlike 1.0

You are free:

- to copy, distribute, display, and perform the work
- to make derivative works

Under the following conditions:

- Attribution.
  You must give the original author credit.
- Non-Commercial.
  You may not use this work for commercial purposes.
- Share Alike.
  If you alter, transform, or build upon this work, you may distribute the resulting work only under a license identical to this one.

For any reuse or distribution, you must make clear to others the license terms of this work. Any of these conditions can be waived if you get permission from the author.

Your fair use and other rights are in no way affected by the above.

SIMULATION ACTIVITY DIAGRAMS

John Ryan*
School of Hospitality Management and Tourism, Faculty of Tourism and Food,
Dublin Institute of Technology,
Cathal Brugha Street, Dublin 1, Ireland.
Tel: 01 – 4027562
E-mail: john.ryan@dit.ie

Dr. Cathal Heavey
Department of Manufacturing and Operations engineering,
Schrodinger building,
University of Limerick, Limerick, Ireland.

ABSTRACT
This paper presents a technique, Simulation Activity Diagrams (SAD), developed to lessen the modelling burden during the initial requirements gathering phases of a simulation project. The technique also allows the capture and visual communication of detailed information, to manufacturing personnel, which may otherwise be lost in detailed programming code.

KEY WORDS
Process modelling, Simulation, Visual communication, requirements gathering

INTRODUCTION

Most operations systems can be viewed as Discrete Event Systems (DES) e.g. manufacturing systems, business processes, supply chains, etc, these systems are complex and difficult to both understand and operate efficiently. One of the most commonly used tools for the analysis of such systems is simulation (Harpell, Lane & Mansour, 1989). Simulation in theory has great potential to assist in the understanding and efficient operation of these systems, however it has not had the penetration in the market that was predicted in the 1980's. Many reasons have been put forward for this such as, poor salesmanship, poor education and time commitments within an organization (Keller, Harrell & Leavey, 1991). However another reason may be the heavy burden placed on the model developer. Simulation modelling can often become a very heavy programming task with the inner workings of a system only being visible to those who are intimately involved in the programming tasks. While simulation can provide quantitative information, there is a lot of other valuable information as regards operations management that is lost within simulation code. Such code may contain detailed information as regards part routings, operations, resource configurations, processing times and so on. The SAD technique presented in this paper has been developed specifically to support the requirements gathering phase of a simulation project. It is a posit of this work that the information gathered at this phase of a simulation project would be a valuable resource to those involved in process improvement projects if it were available in a correct and easy to understand manner. While there are various process modelling techniques available to a model developer such as Activity Cycle Diagrams (ACDs) (Ceric and Paul, 1992), Petri Nets (Petri, 1962), Event Driven Process Chains (EDPCs) (Keller, Nuttgens & Scheer, 1992), Role Activity Diagrams (RADS) (Ould, 1995), the IDEF 3 process modelling technique (Mayer et al, 1995), the IEM technique (Mertins, Jochem & Jakel, 1997) etc, none of these techniques can adequately represent the detailed information contained within a simulation model. While the SAD technique adapts a number of the modelling approaches of some of the aforementioned techniques the overall modelling approach is significantly different. The SAD technique endeavours to model complex interactions such as those that take place within an actual detailed simulation model of a real system. To achieve this the technique uses a number of SAD modelling primitives to represent the various events that are listed in a simulation event list. To also represent more complex interactions the SAD technique introduces the concept of an action list, which

is used to represent detailed actions that collectively can make up any event within a simulation event list.

SAD ACTION LIST

A discrete event system consists of a series of discrete events, the outcomes of which when grouped together ultimately decide the progress of a particular system. In a simulation engine these events are stored in an event list and executed in order of their time of occurrence. The SAD technique graphically represents every event in a simulation event list by means of an activity.

"An activity is any event that causes the change of state of a discrete event system"

However an event in a simulation model can often represent more than one event or task. Often model developers group such events together to lessen the programming burden. This, while understandable can often lead to difficulties in relation to non simulation personnel understanding simulation models. To overcome this an activity can be subdivided into a series of what are defined as actions.

"An action element represents the individual task or tasks that have to be performed within a system at a particular instance"

This approach allows an activity or event to be further subdivided into it's various individual elements or tasks. In other words an activity in a SAD model can be considered to be a list of actions that have to be executed in order for the activity to be fully completed. Figure 1 shows an activity consisting of three actions, which are executed as follows.

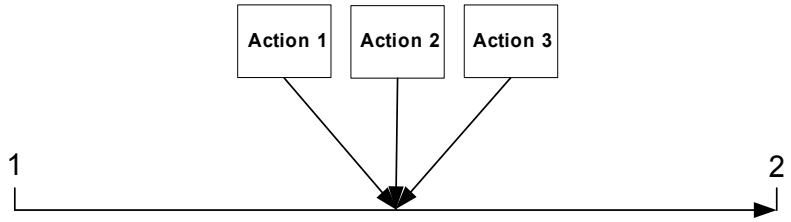

Figure 1 SAD Actions

The system is in state 1, before it can transition to state 2 all actions, 1,2 and 3 must be executed. In this way an individual activity is considered a separate mini event list or action list within the SAD model. These actions are executed from top to bottom and from left to right ensuring that each criterion is satisfied. Only when each action has been executed, can the full activity be executed and the system transition successfully to state 2. Such an activity could be used to represent a simple simulation model mimicking a simple system. The system modelled may be as follows, an entity in state 1 has to be processed before transitioning to state 2. To represent this the simulation model would release an entity from state 1 after which it would seize the entity and delay it for a period of time. In this way representing the time taken to execute actions 1,2 and 3. In other words seizing the entity for the period of time taken to execute activity A. The completion of this delay period of delay, or activity A, allows the transitioning of the entity to state 2. In terms of the simulation model this may be represented by the release of the entity whereby it may exit the system or move onto further stages of waiting or processing.

Taking this approach a SAD becomes a graphical representation of the various events to be executed as per a simulation event list. Each of these events is represented in an SAD by an activity. This activity is then further graphically represented by an action list. This will now be further developed in the following section by the introduction of a series of modelling primitives that may be used in the detailing of such an activity.

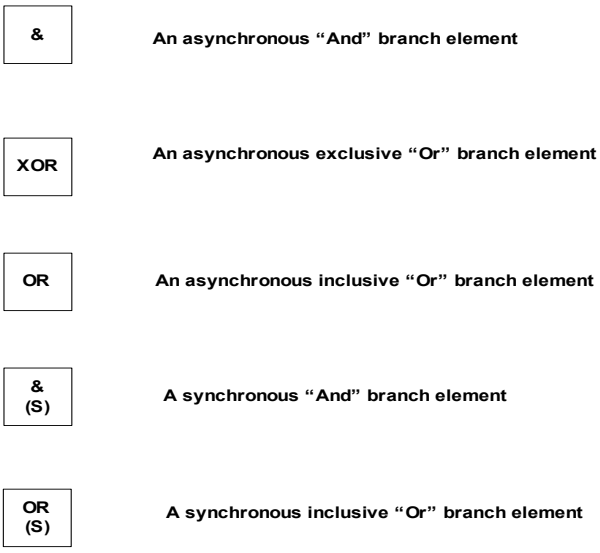

Figure 2 SAD Branching elements.

SAD MODELLING PRIMITIVES

Within most systems, actions such as those in Figure 1 are rarely executed without a number of other types of resources being used. These resources are briefly introduced below;

- Primary resource element- A primary resource element represents any resource within a discrete event system which facilitates the transformation of a product, physical or virtual, from one state of transition to another.
- Queue resource element - A queue modelling element represents any phase of a discrete event system where a product, virtual or physical, is not in an active state of transformation within the system.
- SAD entity element - An entity element represents any product, physical or virtual that is transformed as the result of transitioning through a discrete event system.
    - Entity state element - An entity state represents any of the various states that a physical object or component explicitly represented within a system transitions through during physical transformation
- SAD Informational element - An informational element represents any information that is used in the control or operation of the process of transition by a product through a discrete event system.
    - Informational state element - An informational state represents any of the various states that information used in the operation or control of a discrete event system transitions through during the support of the operation of the physical transformation
- Auxiliary resource element - An auxiliary resource represents any resource used in the support of any phase of transition of an entity within a system.
    - An actor auxiliary resource represents any auxiliary resource used in the direct support of the execution of an action or actions within the process of transitioning a system from one state to another.
    - A supporter auxiliary resource represents any auxiliary resource used in the direct support of an actor auxiliary resource in the execution of an action or actions within the process of transitioning a system from one state to another.
- SAD Branching Elements - Most discrete event systems are complex in nature and rarely if ever linear. To account for the representation of such situations the SAD technique uses a number of branching elements. Figure 2 shows the various types of branching elements used in the SAD modelling technique.

- Link Types - Links are the glue that connects the various elements of a SAD model together to form complete processes. Within the SAD technique there are three link types introduced known as entity links, information links and activity links. The symbols that represent each type are shown in Figure 3
- SAD Frame Element - The SAD frame element provides a mechanism for the hierarchical structuring of detailed interactions within a discrete event system into their component elements, while also showing how such elements interact within the overall discrete event system.

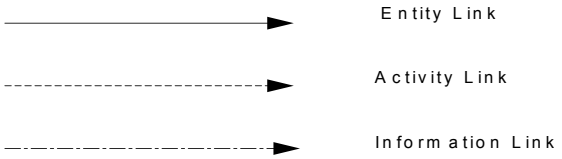

Figure 3 SAD Link Types

As can be seen elements such as SAD entity and informational elements are also subdivided into constituent state elements. The SAD auxiliary resource element is also subdivided into constituent actor and supporter auxiliary resources, to allow a model developer to distinguish between an operator and other auxiliary resources.

SAD MODEL STRUCTURE

An SAD model is logically executed from left to right and from the centre auxiliary resource area to the extremities of the model and is structured as follows. At the centre of the model are located the actors and supporters also known as auxiliary resources. These are the supporters for both the information and physical models. This is advantageous for the purposes of communication during the requirements gathering phase of a simulation project as the persons with whom the simulation model developer will be communicating will generally be a supporter within the process. Therefore each SAD model will be developed from the perspective of the persons interacting with the system.

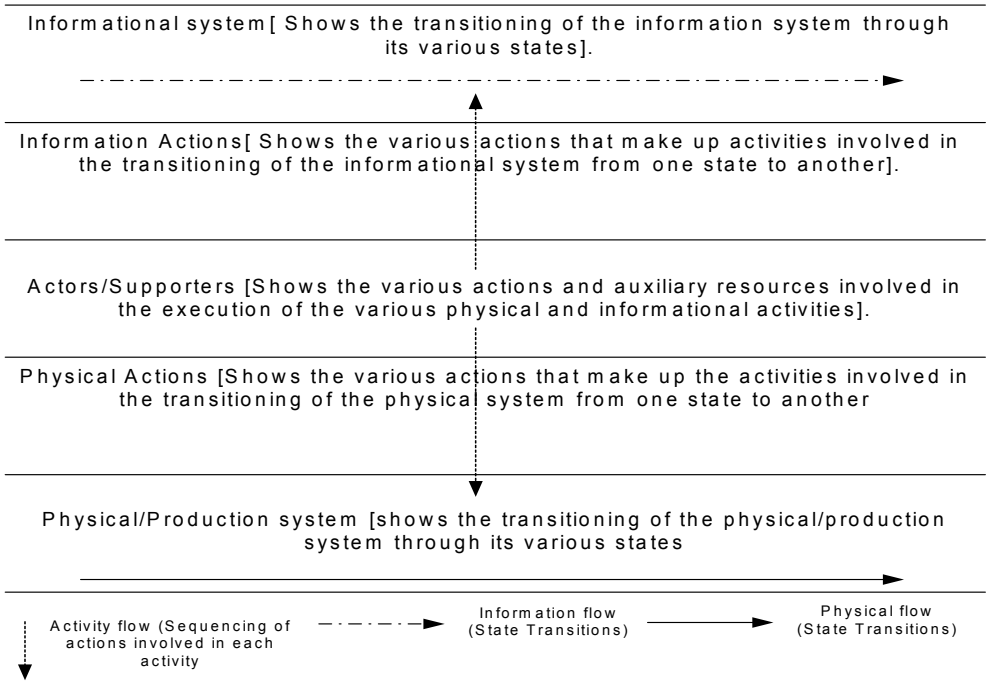

Figure 4 SAD Model structure

The interconnecting areas between both models contain the actions to be executed. A series of these actions and the associated interactions with other SAD modelling elements make up an action list. A

series of these activities in turn make up a sequence of transition for a product of family of products within a discrete event system. Figure 5 shows a simple SAD model for both a physical and informational system.

In this instance the physical system transitions from an entity state 1 to either an entity state 2 or an entity state 3 based on the result of the three actions, “A”, “B” and “C” executed prior to the exclusive asynchronous or fan out branch, “XOR”.

Within the informational model in Figure 5 the system is at an informational state, “A”, and has two states that it can transition to, “B” or “C”, based on the results of a series of 3 actions, “D”, “E” and “F” that are carried out on the primary resource element, “Resource Z”. The logical sequence of these actions along with the location of the execution of such actions is shown within the information actions section of the model. Here the 3 aforementioned actions are shown as is the logical sequence of their execution and where theses actions are carried out. All of this is shown by means of the various branch types and activity links. Again the centre of each SAD model is the section that contains the auxiliary resources, both actor and supporter types. In this section the resources that are used to support the execution of the actions are shown.

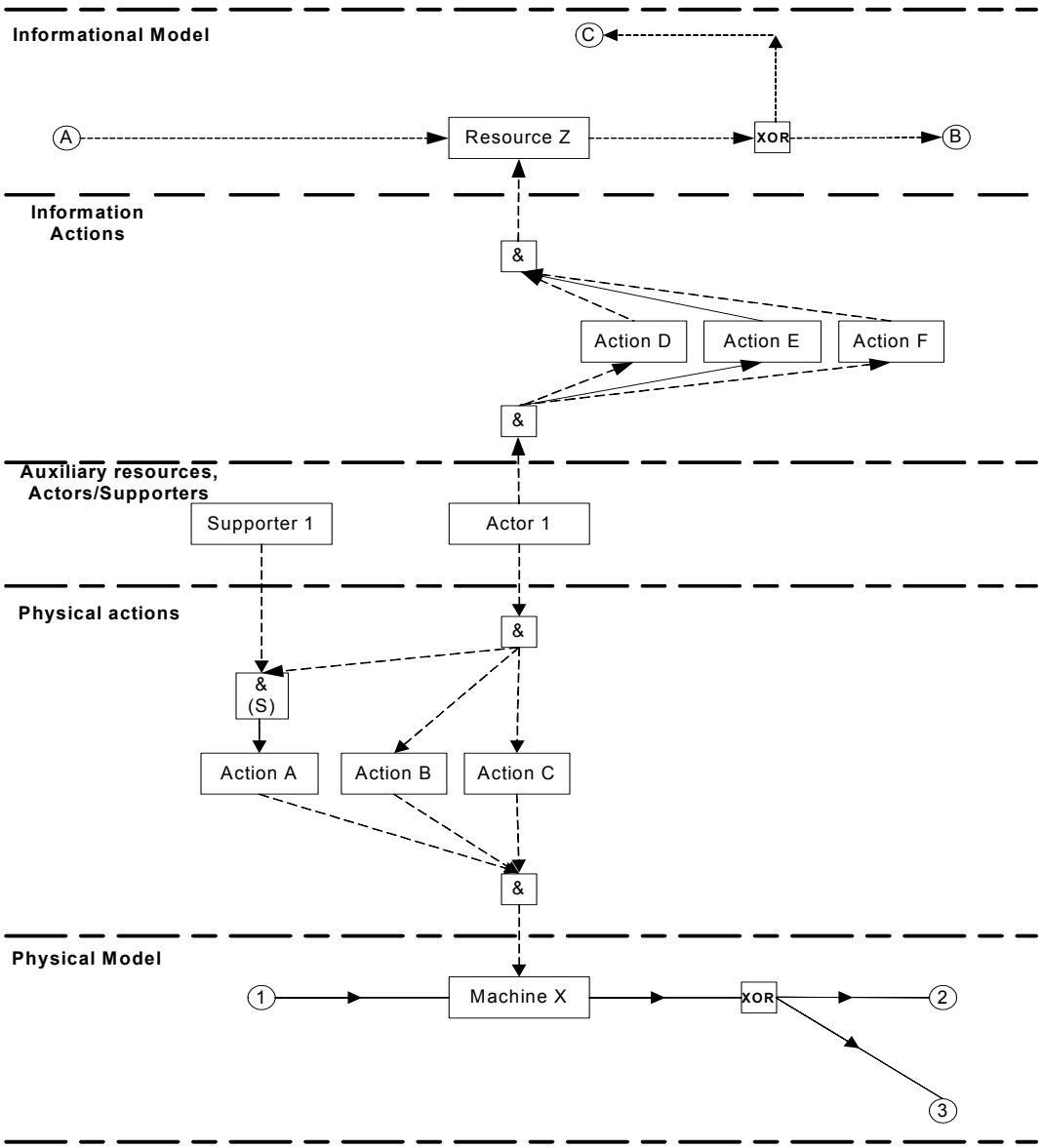

Figure 5. An Extended SAD.

ANIMATION OF SAD MODELS

Thus far the modelling elements used to develop a SAD model have been introduced. However such graphical models are capable of only representing a certain amount of detailed information and knowledge. Often complex discrete event systems contain detailed information and knowledge related to process interactions that cannot be captured well by such graphical representations. To provide a means of making such information available to a model user the SAD technique uses animation and a structured language.

The animation of SAD models is based on the use of the SAD branch modelling elements and allowing a model developer to use these branching elements as a structured language around which can be built a detailed textual description of each section of a SAD model. These same textual descriptions can then be presented to a model user during the animation of a SAD model. Such animation allows for the explanation and representation of any ambiguities that may arise around any aspect of a SAD model.

CONCLUSIONS

The SAD technique endeavours to model complex interactions such as those that take place within an actual detailed simulation model of a real system. To achieve this the technique uses the various SAD modelling primitives to represent the various events that are listed in a simulation event list. To also represent more complex interactions the SAD technique introduces the concept of an action list, which is used to represent detailed actions that collectively can make up any event within a simulation event list. The SAD technique also allows for the modelling of both a physical and informational system that may make up discrete event system along with interactions between each. Such a modelling approach allows for the modelling of a modern discrete event system and in turn a simulation model of the same. Finally the use of animation and structured text within the SAD technique allows for the removal of any ambiguities that may arise within a complex model. As a result of these modelling approaches the SAD technique uses a set of high level modelling primitives that are capable of representing complex discrete event systems. Such an approach increases the users access to the information and knowledge that would otherwise be lost in detailed simulation code. While also placing a low modelling burden on the model developer and promoting the visualisation and communication of detailed information in a user friendly manner for models users.

REFERENCES

Ceric, V. and Paul, R., *Diagrammatic representations of the conceptual simulation model for discrete event systems-.* Mathematics and Computers in Simulation, 1992. **34**: p. 317-324.

Harpell, J.L., Lane M.S., and Mansour A.H., *Operations research in practice: A longitudinal study.* Interfaces, 1989. **19**(3): p. 65-74.

Keller, L., Harrell, C. and Leavey J., *The three reasons why simulation fails.* Industrial Engineering, 1991. **23**: p. 27-31.

Keller, G., Nuttgens, M. and Scheer, A.W. *Semantische Processmodellierung auf der Grundlage Ereignisgesteuerter Processketten (EPK)*. in *Veroffentlichungen des Instituts fur Wirtschaftsinformatik*. 1992. Saarbrucken: University of Saarland.

Mayer, R.J., et al., *Information Integration for concurrent engineering (IICE) IDEF3 process description capture method report.*. 1995, Knowledge Based systems Incorprated (KBSI): college station.

Mertins, K., Jochem, R. and Jakel, F.W., *A tool for object-oriented modelling and analysis of business processes.* Computers in Industry., 1997. **33**: p. 345-356.

Ould, M.A., *Business processes: Modelling and analysis for re-engineering and improvement*. 1995: Wiley.

Petri, C.A., *Kommunikation mit Automaten*. 1962, University of Bonn: West Germany.