Proceedings of the 1994 Winter Simulation Conference
ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila

HIERARCHICAL ABILITIES OF DIAGRAMMATIC REPRESENTATIONS OF DISCRETE EVENT SIMULATION MODELS

Vlatko Čerić

Faculty of Economics
University of Zagreb
Kennedyjev trg 6, 41000 Zagreb, CROATIA

ABSTRACT

Diagrammatic models are a particular class of conceptual models which enable representation of models in two dimensions. Besides, this type of models typically use a limited nimber of symbols and thus enable easier model comprehension as well as better communication between modellers and their clients. One of the powerful features of diagrammatic modelling is its capability to enable hierarchical modelling. Advantages of hierarchical modelling are to enable easier modelling of complex systems by using top-down model design and permitting information hiding. In this paper technical capabilities of several diagrammatic modelling methods (augmented Petri nets, activity diagrams and activity cycle diagrams) for using hierarchical modelling are demonstrated. Top-down model development with diagrammatic modelling techniques is illustrated on a rather comprehensive example of modelling airport passenger building operations. Arguments for using hybrid model development are presented as well.

1 DIAGRAMMATIC MODELLING IN DISCRETE EVENT SIMULATION

Diagrammatic modelling methods are one of the mostly used and developed class of conceptual modelling methods in discrete event simulation. They consist of a set of symbols connected in diagrams. Each type of symbol has a specific meaning related to a particular notion in discrete event simulation like event, resource, activity, condition, dynamic entity etc.

There are several reasons which brought popularity to diagrammatic modelling methods (Martin, 1987; Evans, 1988; Kreutzer, 1986). Some of them are of general character, applicable to modelling of any type of systems, while the other are specific for discrete event simulation modelling:

- Conceptually close objects can be represented physically adjacent, bringing to light strength of connections in the system;
- Interactions between objects are shown in two dimensions, enabling much easier comprehension of a model than the forced sequential ordering of objects in procedural representations. This is due to a parallelism of human visual system which enables fast visual processing of the whole model or its significant parts;
- Syntax and semantic of diagrammatic modelling methods are often rather simple, which helps the easier and faster model design and understanding;
- Hierarchical model decomposition is possible in most diagrammatic methods, which again assists both modelling of complex systems and model understanding;
- Most diagrammatic models enable manual simulation of system dynamics - it can be done by moving counters which represent objects by following the rules of one of possible simulation strategies. This feature can help in model validation, and is also useful as a simulation learning tool.

Besides all these, diagrammatic modelling methods exemplify the fundamental feature of simulation models as imitation of real system structure and operation. This increases transparency of models and enable them to be the extraordinary tools of communication between modellers and their clients.

Two overviews of discrete event simulation diagrammatic methods have been recently made by (Pooley, 1991a; Čerić and Paul, 1992), describing a set of the mostly used methods.

The most important and mostly used representatives of diagrammatic modelling methods in discrete event simulation are the following five methods: augmented

Petri nets (Evans, 1988; Törn, 1981; Törn, 1985), activity cycle diagrams (Pidd, 1992), activity diagrams (Hughes, 1984; Pooley and Hughes, 1991), event graphs (Schruben, 1983) and GPSS block diagrams (Schriber, 1974). This selection of methods is based on the intensity of use of the methods, their modelling power as well as availability of documentation and literature. A comparison of four of these methods using a set of criteria is given in (Čerić and Paul, 1992).

2 HIERARCHICAL MODELLING APPROACH

Design of complex systems is confronted with a problem of describing system objects, their characteristics and interactions in a concise and understandable way. One of the basic strategies for accomplishing this task, devised both by nature and by human beings, is a *hierarchical approach* (Simon, 1981). This box-within-a-box approach develops model elements from higher levels into a more detailed description on lower hierarchical levels.

Diagrammatic modelling seems to be a particularly appropriate tool for hierarchical model design. Since diagrams are drawn in two dimensions, objects having complex internal structures can be represented as simple icons - details of these objects can be developed separately (on lower levels) in the diagrammatic form too. Some parts of these objects may be rather complicated themselves and can be further developed in separate diagrammatic models, etc. Such an approach was used for centuries in design of geographic maps, architectural drawings etc. Use of hierarchical approach in diagrammatic modelling in discrete event simulation will be discussed in the following section.

Hierarchical model design strategy offers two basic model development techniques: abstraction (aggregation) and analytic refinement (Balmer, 1987). *Abstraction* is used for aggregation of number of lower level model elements into a single higher level element. It enables simplification of complex model structure, and is a basic technique used in a *bottom-up strategy* of model building. *Analytic refinement* is a technique of developing a detailed description of a single modelling block on the higher hierarchical level. It enables development of a model starting with a rough system description, and is a base for a *top-down strategy* of model building.

In a powerful *top-down strategy* of stepwise model refinement higher model levels are developed first, resulting in a simple description of a system using just a few model units. These model units are regarded as *modules* which are further refined on lower levels, etc., until a satisfactorily final level of system details is developed. This approach to model development is particularly appropriate if modules coincide with parts of physical or logical system structure (Pooley, 1991b). Hierarchical design strategy enables specialized working groups to develop parts of models independently and in parallel with other working groups. This approach results in less errors, better ability of model understanding and validating as well as faster model development.

Although top-down strategy appears to be the most powerful and mostly used model building strategy, both bottom-up strategy and the combination of the two strategies are also often used in model design. Hierarchical approach successfully supports each of these approaches.

Another important feature of hierarchical approach is that it enables *information hiding* (Murphy and Blake, 1989). This means that local contexts of modules are not identified on higher hierarchical levels. The only information known on a higher level is function of the element (module), described by its name and input-output relations with the other model elements. Moreover, if anything changes in description of lower level modules, higher level representation can stay intact. Information hiding is strongly related to principles of abstraction and simplicity, and supports model security and reliability (which are major issues in software development process).

Hierarchical approach is also very useful in *model evolution*, which happens in various phases of model life. Models have to be modified in order to reflect the changed view or design goals of clients, as well as new insight of modellers. Instead of dealing with a model as a whole, we can localize necessary changes to relevant parts of the model only. Another advantage of hierarchical modelling is in improved *communication with users*, who can understand model more easily by concentrating on the part of model of interest.

Besides natural relation of hierarchical modelling with *modular programming*, one other possibility in computer-based development of diagrammatic models is that libraries of standard diagrammatic models could be developed for purpose of assisting reuse of frequently needed submodels (Pooley, 1991b).

To resume, the main advantage of hierarchical modelling are to enable:

- easier and less error-prone modelling of complex systems;
- top-down model development, enabling parallel submodels design;
- information hiding;
- improved communication with users;
- modular programming, including creation of a library of reusable modules.

3 HIERARCHICAL MODELLING WITH DIAGRAMMATIC TECHNIQUES

In a diagrammatic simulation modelling approach models are described as networks of interconnected icons. Some of these icons may describe a cluster of actions which are further developed in detail on next lower level diagram. Some of the icons on lower level diagrams may themselves be a conglomerate of actions, hence we can unfold them further in detail, etc.

In this section we shall demonstrate the use of hierarchical modelling with activity diagrams, activity cycle diagrams and augmented Petri nets. The hierarchical abilities of event graphs and GPSS block diagrams will be discussed too. A more detailed treatment of hierarchical modelling with diagrammatic modelling techniques in discrete event simulation is given in Čerić (1995).

3.1 Activity diagrams

Hierarchical modelling approach in activity diagrams was suggested by Pooley (1991b), who uses the following modelling terminology. A model containing all details is called a 'flat' model, while processes in it are called 'atomic processes'. Processes containing groups of atomic processes are called 'compound processes', and diagrams containing such compound processes are called 'configuration diagrams'. Only process symbols (boxes) and links are used in configuration diagrams.

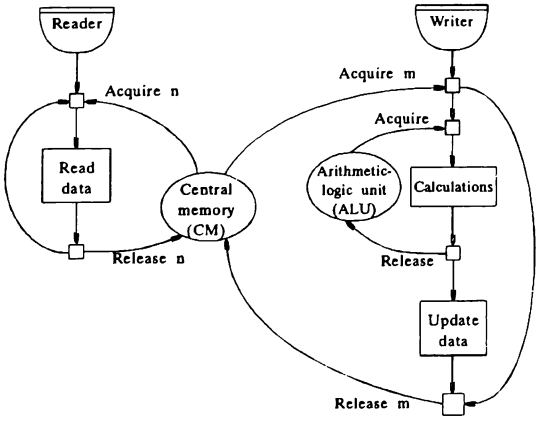

Figure 1: Activity diagram of a reader-writer process

We present hierarchical abilities of activity diagrams on the example of a reader-writer system, being an expanded version of a system described in Pooley (1991b). This model, shown on Figure 1, contains reader and writer processes appearing in computer environments. After a reader is generated he or she submits a request for n units of central memory (CM), in order to be able to read data from CM. After reading is over, he releases all CM used by him. After a writer is generated, he first submits a request for m units of CM, and after they are allocated to him he requires an arithmetic-logic unit (ALU). When ALU is obtained necessary calculations can be made (using both ALU and CM). After that ALU is released and data update operation in CM can take place. Finally, all units of CM used by the writer are released.

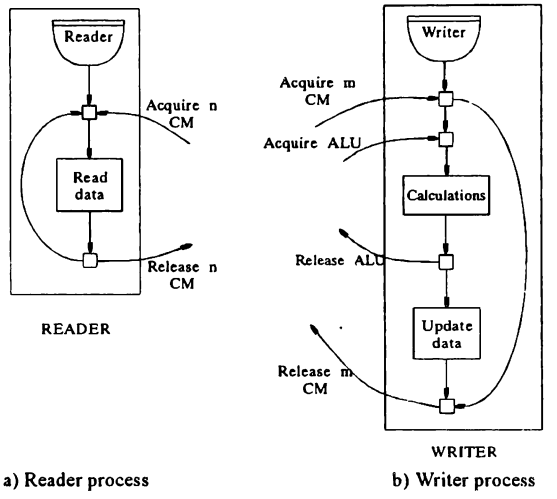

Figure 2: Activity diagrams of reader and writer processes

We shall form two macro symbols ('compound processes') here. Figure 2 shows a READER and WRITER processes containing all actions done by reader and writer, respectively, but excluding all computer resources used (CM and ALU). A compact higher level diagram ('configuration diagram') using these 'compound processes' is shown in Figure 3. Again, any changes in reader and writer processes will affect only 'compound processes' models, while the 'configuration diagram' will remain intact.

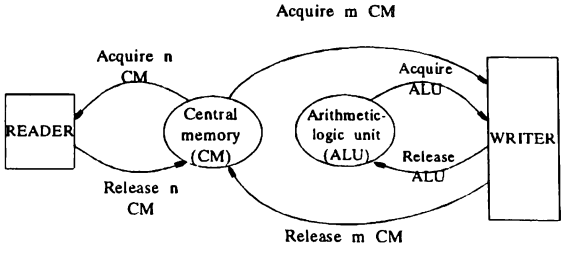

Figure 3: Activity diagram of a reader-writer system with abstraction of READER and WRITER processes

3.2 Augmented Petri nets

The idea of hierarchical modelling approach was expressed already for original Petri nets (Peterson, 1977). This idea was advocated (Törn, 1981; Törn, 1985) in the context of augmented Petri nets (simulation nets) too. Namely, a group of transitions and places can be regarded either as a transition or as a place on a higher hierarchical level. Such a group will be regarded as a transition if its elements communicating with the rest of the model are transitions, and *vice versa*.

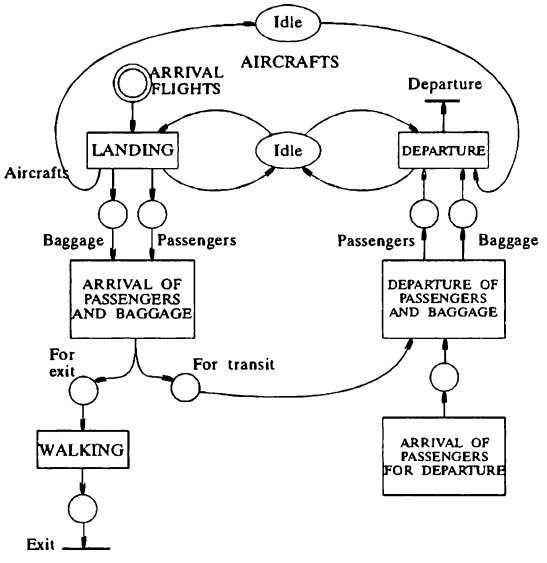

Figure 4: High level augmented Petri net (APN) model for airport passenger building operations

We shall demonstrate hierarchical abilities of augmented Petri nets on the part of the comprehensive model of the airport passenger building operation. The model will be restrained to international arrivals and departures. Two main processes in this system are arrival and departure flights. Both of them are using the same runway(s), and arriving aircrafts are used for the subsequent departure flights. After landing, arrival flights are followed by passenger and baggage processing. After this, some passengers exit from the system while the others are transit passengers to some of the departure flights (processing of their baggage is not modelled here). Passengers coming to the airport for departure flights are processed too, as well as their baggage, till the moment they reach appropriate departure aircraft.

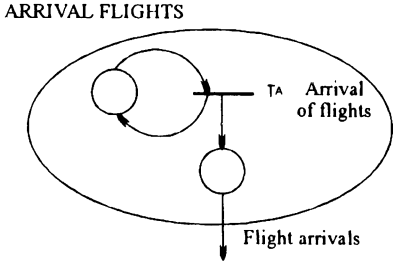

Figure 5: APN model of ARRIVAL FLIGHTS module

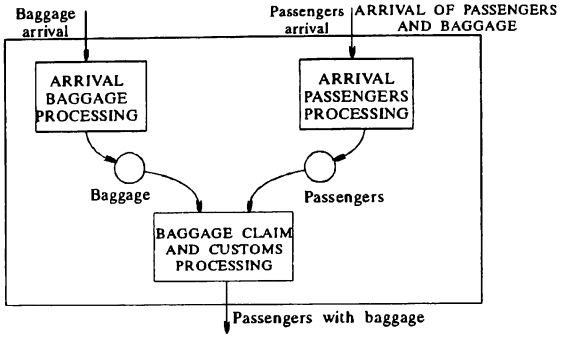

Figure 6: APN model of ARRIVAL OF PASSENGERS AND BAGGAGE module

A high level (zero level) model using augmented Petri nets (APN) diagrammatic technique is shown in Figure 4. Two of the first level modules are a 'macro' place ARRIVAL FLIGHTS, shown in Figure 5, and 'macro' transition ARRIVAL OF PASSENGERS AND BAGGAGE, shown in Figure 6. This second module can be further decomposed in third level modules, one of them shown in Figure 7.

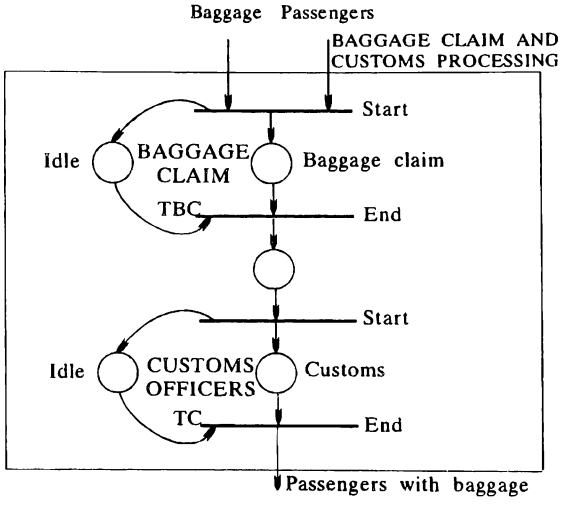

Figure 7: APN model of BAGGAGE CLAIM AND CUSTOMS PROCESSING MODULE

This model is clearly demonstrating the diagrammatic methods ability of multiple-level hierarchical decomposition. A complete model of the airport passenger building operation is presented in Čerić (1995).

3.3 Other methods

There are just a few reports on using *activity cycle diagrams* for hierarchical modelling (Szymankiewicz et al, 1988). This fact is rather curious having in mind extensive use of this method for modelling, as well as its good coverage in literature. Anyway, activity cycle diagrams could be quite conveniently used in hierarchical modelling. Namely, any sensible group of symbols in activity cycle diagram will contain activities, so that any 'macro' symbol could be understood as a macro activity. Hierarchical modelling abilities of activity cycle diagrams are demonstrated in (Čerić, 1995).

In the *event graphs* technique the only symbols which could be candidates for hierarchical abstraction are events. However, since events are happening in a moment of time, any connection of a group of events which are not happening at the same moment (either because of a fixed time delay or because of a necessity to wait till some condition is fulfilled) could not be regarded as the event itself. In view of above, event graphs seem inappropriate for hierarchical modelling.

GPSS block diagrams technique could in principle be used in hierarchical modelling. Practical obstacle here is that symbols of this type of diagrams were conceived as a one-to-one correspondence to GPSS blocks (statements). However, it is possible to use general principles of hierarchical modelling and develop useful higher-level abstraction symbols for this modelling technique (Čerić, 1995).

4 TOP-DOWN MODEL DEVELOPMENT

It was claimed that hierarchical modelling can lead to a successful top-down model development. The course and power of this approach was demonstrated in previous section on a more comprehensive Petri net model of the airport passenger building operation.

This example in its complete form apparently illustrates the gains obtained by using the hierarchical top-down system decomposition. If we try to draw the complete model at once, we will have serious problems in doing this correctly and efficiently. Besides, it would lead us to a large model with all atomic elements included, and this is not easy for comprehension.

5 HYBRID MODEL DEVELOPMENT

Top-down model design is not the only one by which this model could be developed. Some of the 'atomic' system elements can often be constructed directly, and they can subsequently be aggregated into higher level 'compound' elements or modules. This can be done in a few hierarchical levels. This type of development is known as the *bottom-up* model design approach. However, some other parts of that same model could more easily be developed in earlier phase with crude 'compound' elements, which are later refined into 'atomic' elements following the *top-down* model design approach.

Such *hybrid model development*, using both top-down strategy (analytical refinement) and bottom-up strategy (aggregation) in design of the same model is a very flexible strategy. It doesn't renstraint a model development team to just one direction of model development, but rather leaves team members to apply the approach that best suits their experience and their current level of knowledge on a particular part of the system.

In design of comprehensive models such hybrid model development is likely to be the best suited modelling approach in existance.

For both top-down and bottom-up model development hierarchical modelling approach can be followed to achieve hierarchically structured final model.

6 CONCLUSIONS

Hierarchical modelling is a powerful model construction approach used in many modelling areas. We have discussed hierarchical modelling abilities of several discrete event simulation diagrammatic modelling methods, and demonstrating them for activity diagrams and augmented Petri nets. It was shown that this approach leads to easier and more reliable model construction, as well as localization of requests for model change. A popular top-down model design strategy was successfully used on a rather comprehensive example of airport passengers buiding operation. Our intention is to apply hierarchical modelling to computer based diagrammatic modelling using augmented Petri nets.

ACKNOWLEDGEMENTS

I want to acknowledge a personal help of Professor Ray Paul, as well as of the Brunel University Computer Science Department, for providing their valuable

resources and good working atmosphere during a three month visit to Brunel University in 1993. I also greatly appreciate help of both The British Council and the Commission of the European Communities, whose research and visiting grants over the past few years have enabled this research.

REFERENCES

Balmer, D. W. 1987. Hierachical Modelling in Discrete Event Simulation, *Proceedings of the United Kingdom Simulation Council Conference.*

Čerić, V. and R. Paul. 1992. Diagrammatic Representations of the Conceptual Model for Discrete Event Systems. *Mathematics and Computers in Simulation* 34: 317-324.

Čerić, V. 1995. *Diagrammatic Modelling for Discrete Event Simulation.* To be published.

Evans, J. B. 1988. *Structures of Discrete Event Simulation: An Introduction to the Engagement Strategy.* Chichester: Ellis Horwood.

Hughes, P. H. 1984. DEMOS Activity Diagrams. Notat nr 1, FAG 45080 Simulering, Host 1984, Norges Tekniske Hogskole, Institutt fot Databehandling, Norway.

Kreutzer, W. 1986. *System Simulation: Programming Styles and Languages.* Sydney: Addison-Wesley.

Martin, J. 1987. *Recommended Diagrammatic Standards for Analysts and Programmers: A Basis for Automation.* Englewood Cliffs: Prentice-Hall, .

Murphy, J. S. and K. G. Balke. 1989. *Software Diagramming: a New Design Paradigm.* New York: McGraw-Hill.

Peterson, J. L. 1977. Petri Nets. *Computing Surveys* 9: 223-252.

Pidd, M. 1992. *Computer Simulation in Management Science*, 3rd Ed. Chichester: Wiley.

Pooley, R. J. 1991a. Towards a Standard for Hierarchical Process Oriented Discrete Event Simulation Diagrams. Part I: A Comparison of Existing Approaches. *Transactions of The Society for Computer Simulation* 8: 1-20.

Pooley, R. J. 1991b. Towards a Standard for Hierarchical Process Oriented Discrete Event Simulation Diagrams. Part III: Aggregation and Hierarchical Modelling. *Transactions of The Society for Computer Simulation* 8: 33-41.

Pooley, R. J. and P. M. Hughes. 1991. Towards a Standard for Hierarchical Process Oriented Discrete Event Simulation Diagrams. Part II: The Suggested Approach to Flat Models. *Transactions of The Society for Computer Simulation* 8: 1-20.

Schriber, T. J. 1974. *Simulation Using GPSS.* New York: Wiley.

Schruben, L. 1983. Simulation Modeling With Event Graphs, *Communications of the ACM* 26: 957-963.

Simon, H. A. 1981. *The Sciences of the Artificial*, 2nd Ed. Cambridge, Mass.: The MIT Press.

Szymankiewicz, J., J. McDonald and K. Turner. 1988. *Solving Business Problems by Simulation.* London: McGraw-Hill.

Törn, A. A. 1981, Simulation Graphs: A General Tool for Modeling Simulation Designs. *Simulation* 37: 187-194.

Törn, A. A. 1985. Simulation Nets, a Simulation Modeling and Validation Tool. *Simulation* 45: 71-75.

AUTHOR BIOGRAPHY

VLATKO ČERIĆ is an Associate Professor in the Faculty of Economics at University of Zagreb, Croatia. He received B.S. and M.S. degrees in physics from University of Zagreb in 1969 and 1972 respectively, and he received Ph.D. degree from University of Belgrade in 1985. His research interest is in discrete event simulation, particularly conceptual modelling and complex systems modelling. He got a Fulbright lecturer/research grant for 1994/95, and is visiting Prof. Andrew Seila in The University of Georgia at Athens for that period.