Proceedings of the 1999 Summer Computer Simulation Conference (SCSC 1999).
July 1999. Chicago [SCS]. ISBN 1-56555-173-7

PROBLEM ORIENTED MODELLING AND SIMULATION

Hans P.M. Veeke and Jaap A. Ottjes,
Faculty of Mechanical Engineering and Marine Technology
Delft University of Technology
Mekelweg 2, 2628 CD Delft, the Netherlands
e-mail: H.P.M.Veeke@wbmt.tudelft.nl , J.A.Ottjes@wbmt.tudelft.nl

KEYWORDS
Industrial processes, Transportation,
Model design, Discrete simulation,
Process-oriented

ABSTRACT

This paper proposes to combine the 'systems approach' technique with process-oriented modelling and simulation. The process-oriented modelling concept, in which a system is decomposed into a set of component classes, each with a specific data structure, processes and mutual interactions fits well to the general concept of 'systems approach'. The advantages of this are that during the design phase the modelling can be 'problem oriented' and furthermore a hierarchical structure can be designed. Combining this approach with a modern software platform provides opportunities for flexible modelling an reuse of component classes. As an example the modelling of part of a container terminal is shown. An absolute demand on a simulation tool for practice is its speed and the capability to manage large-scale models. A simple benchmark has been worked out in a process-oriented package 'TOMAS', the kernel of which is a fast sequencing mechanism for managing parallel processes.

INTRODUCTION

Industrial systems with a large number of actors, complex control systems and a high demand for flexibility and scalability make heavy demands upon modelling and simulation. This paper is mainly concerned with developments in which logistic simulation models are an integrated part of a design process. This implies that when the modelling process starts, all questions are either still incomplete or not yet explicit. Consequently, during the design process models continue to evolve as insight into the processes increases and the questions become more detailed (van Dijk *et al.* 1996). The authors' experience of modelling large-scale industrial systems for production and transport by using systems approach as a general modelling method and process oriented simulation extends over many years. This approach allows rapid adaptation of the models to new demands and parameterisation of the equipment and controls of the models, but has the disadvantages of requiring programming and of poor visualisation facilities. In industrial simulation, the current trend is towards the use of graphically oriented simulation packages. Most new simulation tools focus on providing visual facilities to model designers and there is only a limited possibility to use a programming environment. They can be used in many practical cases, especially when control demands are simple or standard and allow fast prototyping and easy visualisation (Lilegdon *et al.*1994). However, when these tools are used in situations requiring complex controls, the user may encounter problems and is committed to artificial program-constructions or is forced to employ a programming language for coding specific features. The adaptation of models appears to be difficult and, if programming is needed, additional skills are required.

SYSTEMS APPROACH

The systems approach is a technique that is widely used to investigate organisational and logistic processes. This approach facilitates structural improvement and/or effective design of logistic systems (in 't Veld 1998). The advantages of using the systems approach are that:

- it forces designers to think in terms of processes, functions and controlling mechanisms, thus providing a structured model and a natural way for all members of the design team to communicate with each other about the model;
- the use of the black box-approach, which is an essential part of the systems approach, ensures explicit definition of input- and output flows and global results can be formulated by simply observing these flows, even without simulation;
- by defining levels of aggregation, it is possible to obtain a balanced, hierarchically structured model for analysis and design. This has the additional advantage of recognising at which level which questions have to be answered.

In view of the above, simulation is only part of the systems approach. For this reason, we make a clear distinction between modelling and simulation (Veeke 1982)

MODELLING AND SIMULATION

The main goals of modelling are to define the system under investigation in terms of process descriptions and procedures and to formulate the right questions and the options that may be chosen. Simulation is the logical sequel to this with the following objectives:

- give quantitative support to the modelling by showing the consequences of all options formulated by modelling;
- to give qualitative support to the modelling by revealing unexpected problems. Such problems frequently appear in situations where control issues are being investigated. During the design of a simulation model the use of a control-principle, for example, often results in unexpected behaviour or indicates the lack of data.

From these arguments it becomes clear that during an analysis or design process modelling and simulation have to be used interactively. When simulation is used only after all decisions have been made, it is not accorded its true value. Thus, there is a danger that that simulation leads only to 'streamlining the existing' or that the results of simulation can no longer be used because the design decisions have already been taken. The next example of the design of a container terminal illustrates the modelling concept.

DESIGN OF A CONTAINER TERMINAL.

In this example a part of the design process of a container terminal is described. During this design process we want to gain insight in how much equipment is needed.

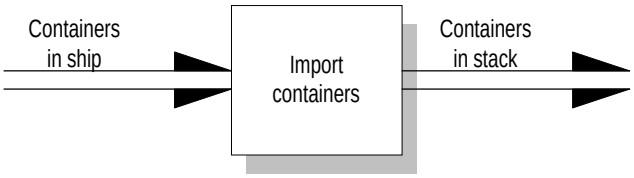

Figure. 1. First modelling level of the import process.

The example is restricted to the seaside-import process. Containers must be unloaded from a ship and put into a stack-area. The highest level black box is shown in fig. 1. This level is rather abstract, still we can argue that we always need three functions to perform the import-process: unload containers from a ship, transport them to the stack and finally put them into the stack. So zooming into the black box 'import containers' gives figure 2.

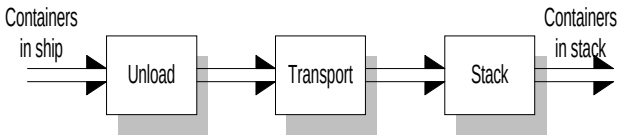

Figure 2. Second modelling level of the import process.

At this level, some important questions must be answered:

a. What kind of equipment do we use? We can choose for equipment which is able to handle all the functions, but we also can choose for equipment specialised on each function separately.
b. In case of separate equipment for all functions, what about the interface between it. Is there a direct coupling or is a buffer function needed.

If we can't or won't answer these questions right now, we have to model this case in the most general way. So between these primary functions 'transfer'-functions are inserted. This demonstrates also the strength of modelling in terms of 'functions': each function may become empty after choosing in a later stadium of the design process.

In this example we choose for a chain of processes for each container with no buffering in between and we decide to use QC's (Quay Cranes) for the unloading the container ship, AGV's (Automatic Guided Vehicles) for transport and ASC's (Automated Stacking Cranes) for the stacking of containers. The AGV's have no facilities to pick up or put down containers themselves. They are served by QCs and ASCs. These choices necessitate separate 'transfer'-processes between unload and transport, and between transport and stack. This is shown in figure 3.

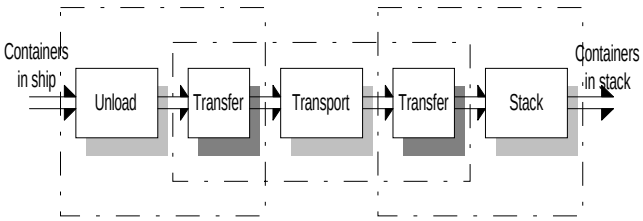

Figure. 3. Third modelling level of the import process.

The dotted lines indicate the processes for each equipment-type involved. The transfer processes not only demand technical provisions, but also influence the logistic process, because two equipment-types are needed for each transfer-process. Therefore synchronisation is needed between equipment processes to avoid excessive waiting times.

The next modelling step is to describe the processes of each equipment-type. To that end we have to add the equipment-flows as shown in figure 4.

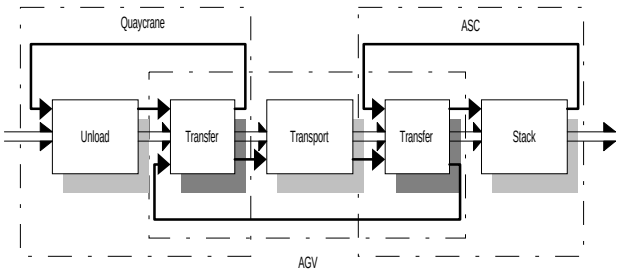

Figure. 4. Fourth modelling level of the import process

Now we will focus on the individual equipment processes. The AGV-system is taken as an example,
see figure 5. The job control will not be elaborated here.

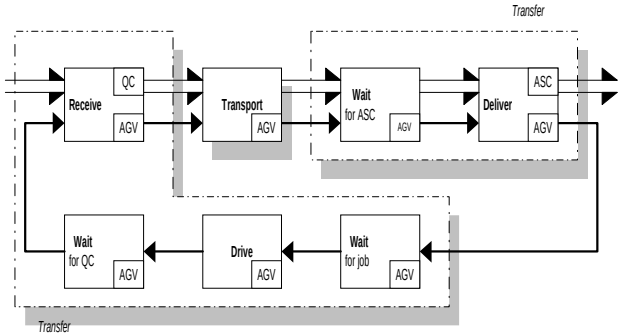

Figure. 5. Fifth modelling level of the import-process: the AGV-process.

In each black box the equipment-types involved have been indicated. We now have a complete view of the AGV-process for handling import-containers. The minimum number of AGV's needed can easily be calculated by using averages of technical handling times and transport times and ignoring waiting times. On the other hand the waiting processes become clear explicitly. They are determined by the quality of the job-control-process. Several questions can be derived from this model. How to control the terminal to minimise the waiting times. How many AGV's are needed then? What is the effect if AGV's would have their own pick-up and put-down facilities? If we buffer containers at the QC's, what will be the consequences for the AGV-system? In this specific example it might be necessary to zoom into the traffic control of the AGV system. A complete model of the automated container terminal is obtained if the other equipment processes are added and if a similar model for the export container flow has been developed. Answering questions involving waiting times and job control methods demand simulation of the model.

This modelling approach fits highly with the process oriented approach of simulation and thus with tools supporting this method.

TOOL REQUIREMENTS

According to systems approach, the 'process-approach' plays a central role during the modelling phase. Therefore, it is quite logical that a simulation tool should support this approach. Tools supporting the process approach are therefore preferable to tools that use the 'event-approach'. Because defining a 'process' is a logical way to arrange the events occurring in a system and to maintain a natural way of description, the arrangement of events is essential. In cases of event-simulation, this arrangement is usually determined by the simulation tool itself (and not well documented). With the process-approach, the sequence of events within the same process is completely defined. The sequence of events in the various processes is managed by the active components of the model.

In this respect the concept differs from that usually understood by process oriented simulation, in which it is considered equivalent to flow orientation (Robert and Dessouky 1998). However all the advantages of object orientation remain valid.

In accordance with these principles a 'Tool for Object-oriented Modelling and Simulation', (TOMAS), has been developed. This may be considered as a 'toolbox' that can be used in a standard Delphi-environment. TOMAS is object-oriented, supports modelling and is able to simulate. It will support for hierarchical modelling and zooming facilities. TOMAS is a follow up of the package 'Must' (Must simulation software1992), which is still being used for modelling large-scale industrial systems,
(Ottjes *et al.* 1996).

The main classes in TOMAS are:

- Components. Each component has its own 'process-method' and methods to generate events (hold, passivate, standby...) during the execution of the process-method. A component can also influence the processes of other components (by means of StartProcess, ResumeProcess, CancelProcess...) and, finally, a component possesses methods to enter queues, leave queues etc.
- Queues. A queue is a set of components. In the case of a 'waiting' queue, statistical data are gathered automatically. The use of queues appears to be very useful for modelling the control system.
- Collections. A collection is used to assemble values of quantities to be measured. A collection provides online

support for the graphical representation of these quantities.

Visualisation of the simulation results is of growing importance.
Several online visualisation purposes may be distinguished:
1. Debugging and verification
2. Display intermediate results
3. Presentation

Points 1 and 2 are well served if the actual status of the simulation, such as queue status, histograms and user defined specific status information can be shown on screen in a flexible way by dynamic tracing. When the research topic involves the physical movement of components, as in the case of traffic control studies, animation is indispensable. Point 3 may demand both dynamic tracing and animation.

The simulation tool TOMAS is being developed in a standard Delphi-environment so all facilities of Delphi are available including the possibility to implement classes developed as general reusable 'Delphi components'. Animation is supported as a separate toolbox, using the windows thread mechanism.

PROCESS MODELLING AND SIMULATION SPEED

The principle of the process modelling will be illustrated by solving the fastest route problem. This is not intended to be an alternative to the Dijkstra algorithm but rather as an illustration of the process modelling, the dynamic generation of components, the parallel processing of components and to obtain simulation speed performance data.
The informal model will be sketched in terms of component classes, attributes and processes.

Model purpose: find fastest route between start Node and destination Node in a network.

Component classes
nodeClass:
Attributes: set of tracks (trackSet)
trackClass:
Attributes: node 1, node 2
carClass:
attributes: array of passed nodes (passedTrackArray), beginNode, trackToDrive, endNode

Process of a car
- hold driving time (sampled from distribution)
- add trackToDrive to passedTrackArray
- draw track on screen
- if endNode=destinationNode then report and terminate simulation
- create a new car for each track in trackSet of endNode which has not yet been passed
- copy own passedTrackArray to passedTrackArray of all new cars
- activate all new cars
- terminate self

Initialisation:
- create network: This means: read from a file the coordinates of all nodes and the connecting tracks, assign node 1 and node 2 of all tracks and put the nodes into the trackSet of the correct node.
- decide on startNode and destinationNode
- create the first car at the startNode and activate it.

In applied simulation speed is of the utmost importance so a benchmark was therefore developed. As a benchmark network a 100 x 100 nodes "Manhattan shaped" network was used that consequently contained 10.000 nodes and 19.800 tracks. The first car was started in the middle of the network, the end node being the middle node of the lower edge of the network. The travelling time for each track was sampled from a negative exponential distribution with an average of 1. The simulation was terminated after the first car reached the destination node. The run was executed on a Pentium-2 processor running at 266 MHz. The results are shown in figure 6. The Delphi/TOMAS code for the definition of the car class and its process description is shown below.

```
CarClass     = Class(TomasComponent)
  PassedTracks     : Array[1..MaxTracks] Of
                     TrackClass;
  NrOfPassedTracks : integer;
  StartNode        : NodeClass;
  TrackToDrive     : TrackClass;
  EndNode          : NodeClass;
  Published
  Constructor Create (cName: String;
   BeginNode: NodeClass; DoTrack: TrackClass;
   Creator: CarClass);
  Procedure Process; Override;
  Procedure Multiply(FromNode: nodeClass);
  Procedure ShowPath;
  End;

Procedure CarClass.Process;
  Begin
  EnterQueue(CurrentCarSet);
  Hold(TravelDis.Sample);
  Inc(NrOfPassedTracks);
  PassedTracks[NrOfPassedTracks]:=TrackToDrive;
  {check if in node  B}
  If (EndNode=DestinationNode)
   AND (Winner=Nil) Then
    Begin
    Winner:=Self;
    End;
```

```
  If Winner= nil then Multiply(EndNode);
  LeaveQueue(CurrentCarSet);
  Terminate;
  End;

procedure CarClass.Multiply(FromNode:NodeClass);
  Var
    hQ: TomasQueue;
    hTrack: TrackClass;
  Begin
  hQ:=FromNode.TrackSet;
  hTrack:=hQ.GetFirstComponent;
  While hTrack <> Nil Do
    Begin
    If NOT(hTrack.Passed) Then
      Begin
      Inc(CarCount);
      NewCar:=CarClass.Create('Car_' +
       hTrack.GetName,FromNode,hTrack,Self);
      NewCar.StartProcess(CurrentTime);
      hTrack.Passed:=True;
      End;
    hTrack:=hQ.GetNextComponent(hTrack);
    End;
  End;
```

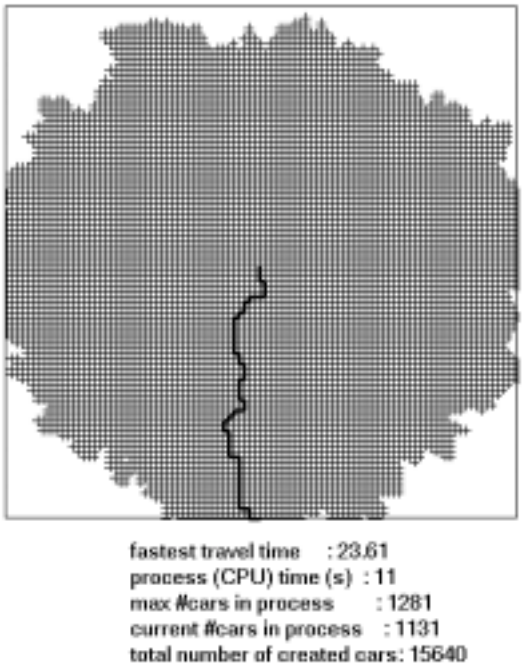

Figure 6. Result of the fastest route model. The nodes have been numbered from the upper left to the lower right part of the network area. As a starting node was node5050 in the middle of the network was taken and the as destination node9950. As soon as the first car reaches the destination node the others do not multiply any more and terminate. This explains the "loose ends" in the plot.

CONCLUSIONS

The process-modelling approach conforms well to the general modelling concepts of systems approach. This makes it possible to develop problem-oriented hierarchical models if there are no restrictions imposed by the simulation tools to be used in the model implementation phase.

The process modelling approach is object-oriented in nature. This means that modern object-oriented platforms can be used as a basis for modelling. The interface facilities combined with the inheritance, extensibility of the objects allows the user to find the balance between coding ‘own’ classes and selecting and/or extending predefined classes.

A simulation tool is being developed to support the process-oriented approach. It is based on a time-sequencing kernel to manage the concurrent processes of the model. Industrial models require that this mechanism should be fast. A simple benchmark is used that gives quantitative performance data.

REFERENCES:

Dijk Johannes N. van, Jobing Marcel J., Warren James R., Seeley Douglas, Macri Riccardo, "Visual Interactive Modelling with SimView for Organizational Improvement", *Simulation* vol 67:2, pp.106-120, August 1996

Lilegdon William R., Martin David L., Pritsker A. Alan B., "FACTOR/AIM: A Manufacturing Simulation System, *Simulation* vol 62:2, pp.367-372, june 1994

Must, " *Simulation Software User and Reference Manual*" v:5.50 (1992) Upward Systems, Rijswijk, The Netherlands

Ottjes, J.A., Duinkerken, M.B., Evers, J.M., Dekker, R. “Robotised inter terminal transport of containers” , *Proc. 8th European Simulation Symposium 1996* Genua [SCS] pp. 621-625, ISBN 1-56555-099-4 Vol I

Robert, C.A., Dessouky, M. “An Overview of Object-Oriented Simulation” *Simulation* vol:70:6, pp. 359-368. 1998.

Veeke H.P.M., "Process simulation as a management Tool, *Proceedings of the IASTED International Symposium Applied Modelleing and Simulation*, Paris, 1982.

in 't Veld, J., "*Analysis of organisation problems*", Educatieve Partners Nederland BV, 1998, ISBN 90 11 045947 (in Dutch)