MODELING, IDENTIFICATION AND CONTROL, 1990, VOL. 11, NO. 1, 43–47
doi:10.4173/mic.1990.1.3

Analysis of manufacturing based on object oriented discrete event simulation

EIRIK BORGEN†, HENNING NEERLAND†
and JAN OLA STRANDHAGEN†

Keywords: *Discrete event simulation, economical analysis, graphical modeling, object orientation.*

This paper describes SIMMEK, a computer-based tool for performing analysis of manufacturing systems, developed at the Production Engineering Laboratory, NTH-SINTEF. Its main use will be in analysis of job shop type of manufacturing. But certain facilities make it suitable for FMS as well as a production line manufacturing. This type of simulation is very useful in analysis of any types of changes that occur in a manufacturing system. These changes may be investments in new machines or equipment, a change in layout, a change in product mix, use of late shifts, etc. The effects these changes have on for instance the throughput, the amount of VIP, the costs or the net profit, can be analysed. And this can be done before the changes are made, and without disturbing the real system. Simulation takes into consideration, unlike other tools for analysis of manufacturing systems, uncertainty in arrival rates, process and operation times, and machine availability. It also shows the interaction effects a job which is late in one machine, has on the remaining machines in its route through the layout. It is these effects that cause every production plan not to be fulfilled completely. SIMMEK is based on discrete event simulation, and the modeling environment is object oriented. The object oriented models are transformed by an object linker into data structures executable by the simulation kernel. The processes of the entity objects, i.e. the products, are broken down to events and put into an event list. The user friendly graphical modeling environment makes it possible for end users to build models in a quick and reliable way, using terms from manufacturing. Various tests and a check of model logic are helpful functions when testing validity of the models. Integration with software packages, with business graphics and statistical functions, is convenient in the result presentation phase.

1. The SIMMEK project, organization and goals

Since 1985 a research project has been running at Production Engineering Laboratory, NTH-SINTEF. The project is financially supported by the Norwegian Council for Scientific and Industrial Research, NTNF. Its main goal has been to develop computer tools for performing analysis of manufacturing systems, based on simulation techniques. At early stages simulation of processes was also considered, but since 1986 discrete event simulation has been focused. In late 1988 a prototype of a simulator based on discrete event simulation was launched. This prototype is now being tested in a number of Norwegian companies, and the first commercial version called SIMMEK was available by the beginning of September, 1989.

Received 7 August 1989.
† SINTEF Production Engineering, NTH-SINTEF, Trondheim, Norway.
This paper was presented at SIMS '89 (Scandinavian Simulation Society), 31st Annual Meeting, Bergen, Norway May 31–June 2, 1989.

2. The use of discrete event simulation tools in Norwegian industry

During the first years of the project, most computer tools available for performing discrete event simulation were examined. Programming tools, both general purpose programming languages and simulation languages, were examined, as well as simulation tools which are often described as simulators. These were tools both for PCs and mainframe systems. Some of these tools were tested in the laboratory, others were tested in real life in Norwegian companies. Despite the number of tools available, and the willingness to look into new techniques, we found that simulation was often rejected as a method for performing analysis of manufacturing systems. During these tests, the following questions were asked;

What are the potentials of performing analysis of manufacturing with computer based discrete event simulation tools?

We found that the need for such analysis tools was large; a lot of investments and changes are implemented without proper analysis being done, and these changes are often crucial for the survival of a company.

What were the reasons why simulation of this type was not used?

The availability of most existing tools were limited. Programming and simulation expertise were necessary, experiments had to be done by experts, and were extremely cost and time expensive.

How should a simulation tool be used in larger scale in industrial companies?

The simulation tools must use advanced facilities in user interaction, i.e. windowing and graphical facilities, so that they will be easy to use and easy to learn to use. No programming should be needed, and manufacturing terms should be used in the modeling of manufacturing systems. The result presentation must be complete, and graphical presentation facilities available.

The answers we found to these three important questions made it obvious to us that the project should develop a simulation tool with all these facilities available. The rest of this paper will describe how this tool has been implemented, and how it is to be operated by the users.

3. The technical approach of SIMMEK

The Simulation Kernel in SIMMEK is based on discrete event simulation. It is event oriented in the way that it uses the traditional approach of constructing an event list to schedule how the model should be operated during the simulation. The different steps of operations are executed by stepping to the next event scheduled, letting it happen, and updating the system parameters. This is described in somewhat more detail in the §6. In the simulation kernel we have used much of the ideas from class DEMOS in SIMULA. DEMOS is a class in SIMULA developed to perform discrete event simulation experiments. In fact plans existed at the early stages of the project to develop a SIMULA/DEMOS program code generator. Through a modeling environment the user should identify the objects of different types, and a generator should transform this into SIMULA code. But this solution did not give the necessary flexibility in modeling. Changes in DEMOS concerning the reporting functions were also needed, and it is never a good solution to change program code in an existing system.

The modeling environment in SIMMEK is however process oriented. We find, which may be a surprise, no contradictions between a process oriented modeler and an event oriented simulation kernel. A process is taken to be 'what happens, in sequence, to one occurrence of one type of product'. One occurrence may be one single item, a batch, or a full order. One occurrence is in either case the entity in the model. A model will consist of a number of processes, normally one for each type of product or part. This process oriented model is transformed, with no loss of generality, to the event oriented simulation kernel. This can be done since a process of course is a sequence of events.

There is a true object orientation in the modeling environment. The objects may be of two main categories; the products already mentioned, and the resources. The product routings, in SIMMEK called the process plans, are taken to be the processes. Products and parts are born and die during the simulation run. The resources however are permanent. The processes of the products are a sequence of references to which resources they require, and for how long they need to possess them. The object orientation of the modeling environment is reflected in the internal structure of the Simulation Kernel, as it is shown in the section Internal Structure.

4. The system architecture

The full system will consist of seven modules; two modeling modules, a linker, a simulation kernel, an analysis part, and two modification modules. In the prototype version the two modification modules are not present. The two modeling modules are separate, and are called the Layout Modeler and The Product Flow Modeler. The Layout Modeler is an environment for describing the resource objects. Most of the resources; machines, transport units, operators etc., are all-time present although they may be 'down' for some scheduled or unscheduled period. Such down time for a machine is typically when the machine breaks down, or when it is stopped for maintenance. A special type of resource objects are called requisites. They are in fact consumable, but may be refilled or supplied by certain procedures.

The Product Flow Modeler is used to model the parts and products, also called the entities of the system. A full product description consists of three main elements; order data, cost data, and a process plan. The order data gives the number of work pieces in each order, the start up times of orders of this type, the arrival rate of orders of this type, etc. The cost data may be used to perform investment analysis with SIMMEK. The process plan is the key information, because it is used to describe the product's routing through the layout. In addition this module is used to specify which priority rules the jobs are to be selected at the resources, and whether the start up times are to be scheduled in a certain way, i.e. the rules the model should be operated by. Examples are FIFO or LIFO, scheduling by identification of critical resource, etc.

The Object Linker transforms the data in the two modelers into information that is processable by the Simulation Kernel. This module performs the tests required to be certain that the simulation can be run without complications.

The Simulation Kernel is used to perform the wanted number of replication runs of the model. This is where the model 'lives'. The Kernel produces a full trace, which may be examined by Edit™, and also an aggregated summary report.

The Analysis module produces far more complete reports from the trace. It tries to highlight what the system believes is the most significant results in each experiment. The results are presented in files by a format that that is readable by two software

packages; Cricket Graph™, and Statworks™. These packages can be used to perform statistical tests on the results, and to present the results in graphics.

The two modification modules, are still being specified and will hopefully be successfully implemented this year. They are meant to be a decision support tool giving advice on changes to make in models in order to produce better overall results.

5. The user interaction

The main idea when specifying the Front End has been to provide the users with a modeling environment as close as possible to how they see the real system they are modeling. Production planners and managers, floor managers and machine operators should be able to recognize on the screen the layout with the machines and operators, the products and their routings. Of course no programming should be needed. This is made possible by the creation of a number of symbols to represent the different objects, each with different parameters according to the type of object. SIMMEK takes advantage of the Finder System on the Apple MacIntosh™ with its windowing and input facilities. After a few hours inexperienced users are able to perform realistic experiments with the SIMMEK system.

When an element in the real layout is identified, and it is decided that it must be a part of the model, it must be put into the model layout. This is done very easily by pointing at a symbol of such an element in the Layout Modeler, and copying it into the model layout. These symbols contain parameters which are typical for the chosen type of element or resource; machine, operator, transporter, etc. The modeling route should not be set once and for all. The user should be free to choose where to start and where to finish, and be able to take a break no matter how incomplete the model is. But it is advisable to start with modeling the layout. This is so because when modeling product flow a lot of references to the resources in the layout is needed. If the layout is present, this is done simply by pointing and clicking the resource symbol. If not the full name of the resource must be written from the terminal. If the user gives values or attributes that are illegal or inconsistent, messages will be given. The parameter that is out of range is shaded, and must be corrected. Through the message field detailed information about what is wrong is given. The syntax and logics of a completed model will be checked by the Linker before the simulation starts. That is automatically performed, and messages are given on the screen. It is also possible to get aggregated schemas of the input data. These schemas show what objects have been put into the model, the load of each resource, etc. This makes it easy to reveal modeling mistakes or overload of the model, and is extremely important in the validation aspect. The results are presented in files in Cricket Graph™ format, a format that is also readable by Statworks™. This gives the user the possibility to manipulate and combine results from many experiments. It is also possible to compare the results directly with results from the real system, in order to convince everybody of the model's validity. The most important and significant results may be extracted and presented graphically just the way the manager wants them.

6. The internal structure

The job of the Linker is to transform the model data into basic data elements or structures, which is input to simulation kernel. The object orientation in the model data is conserved during this phase. Each object, no matter if it is a resource (for instance a machine), or an entity (product), is transformed to a data structure. There are of course

different structures representing the different types of objects. A product has a certain structure and the process plan of the product data structure is represented as a linked list with pointers to resource data structures. These lists are dynamically expanded during the transformation phase, and have no limitations in number or size. From this follows that the size of the models and objects are limited only by the size of the computer.

The order data information is used to create occurrences of the different product data structures. Each occurrence will either have a pointer to it from the event list, or it will be in the event list, or it may be in a queue waiting for a resource. It follows from this that the event list is also a linked list. Each element in the list contains the time the event is scheduled, and either the product itself or a pointer to the product scheduled. This depends on whether the event is an acquirement (pointer) or a release (the product itself) of a resource. The clock function of the kernel is a step function through these linked lists. These lists may be viewed as lists of activities, but the activities are limited by the events start and finish.

7. Hardware and software specifications

The prototype is implemented on a MacIntosh™ computer, Plus, SE, or II. It uses the Finder™ for modeling functions and storing data. The Linker and Kernel is implemented in Lightspeed™ C. The results are presented in files in Cricket Graph™ format. Statworks™ is also used for statistical analysis, and Edit™ for examining the trace.

8. Conclusions

We believe that the SIMMEK simulator will survive in the jungle of tools for analysis of manufacturing based on discrete event simulation. Tests performed in companies show that the time to learn to use the system is far shorter than any tool on the market in Norway. It has also proven its simulation speed in benchmark tests. With its freedom in the modeling phase, and still with excellent means of model validation and result presentation, it has been received with enthusiasm in many companies.