Proceedings of the 1989 Winter Simulation Conference
E.A. MacNair, K.J. Musselman, P. Heidelberger (eds.)

Teaching Simulation with Pascal_SIM

Robert M. O'Keefe
Department of Decision Sciences and Engineering Systems
Rensselaer Polytechnic Institute
Troy, NY, USA

Ruth M. Davies
Department of Accounting and Management Science
University of Southampton
Southampton, England

ABSTRACT

Pascal_SIM is a tool kit for producing simulations in Pascal, developed specifically for teaching discrete event simulation. We have gained considerable experience with Pascal_SIM, and have produced a text called *Simulation Modelling with Pascal* (published by Prentice Hall International) that is based upon these routines. This tutorial is an introduction to using Pascal and Pascal_SIM as a basis for teaching simulation.

1. INTRODUCTION

Despite the large and growing number of simulation programming languages and packages, there are some good reasons why using Pascal to teach simulation may be advantageous. First, since Pascal is the language of choice for teaching programming, students may already have a good grounding in the language. Programming simulations can reinforce good programming practice, and further develop skills in Pascal.

Second, students can be exposed to the inner details of simulation modeling, including time advance mechanisms, variate generation and animation. Third, and most important, students can employ all common world views (event, activity and process based) and not graduate from a course in simulation under the mistaken belief that the world view implicit in the language used is the only way to structure a simulation model.

2. Pascal_SIM

Pascal_SIM is described fully in *Simulation Modelling with Pascal* (Davies and O'Keefe, 1989), and briefly in O'Keefe and Davies (1986). It is a set of Pascal constants, types, variables and routines which largely conforms to the ISO standard, and has been used with Turbo Pascal (versions 3, 4, and 5), VAX/VMS Pascal, Pro Pascal, and many others.

Pascal_SIM contains a minimal set of routines numbering 50 in all. Originally put together in 1983 and used in research projects, the number of routines grew considerably thereafter. In 1985 we cut the package back to an absolute minimal set. Numbering less than 600 lines of code, students familiar with Pascal can quite easily grasp the details of the entire package, and then enhance, extend or alter as necessary.

The package uses extensively the pointer facilities of Pascal. This enables it to provide temporary entities, which can be created and disposed of, and to provide flexible list processing facilities such that entities can be inserted or extracted from any point in the list. The procedure for controlling the time advance mechanism and for removing entities from the top of the calendar is called the executive, and Pascal_SIM includes three different executives for the three different world views: three phase, event method and process view.

The package provides for the use of 36 random number streams and facilities to sample from histograms and common distributions. It is thus possible to use multiple runs using different random number streams or to use the method of common random number streams for variance reduction. Results can be collected into state or time weighted histograms. The animation facilities are very simple and text based, providing portability between different computers and different versions of Pascal.

3. TEACHING SIMULATION

Those teaching simulation obviously need to explain the theory and techniques of discrete event simulation. Students also require practical examples and 'hands on' experience using software. We have taught simulation to undergraduate and postgraduate students in England at the Polytechnic of the South Bank, London, University of Kent at Canterbury, and more recently the University of Southampton, and additionally in the USA at Virginia Tech. Our approach is generally as follows.

After an introduction to the principles of simulation, we explain and demonstrate the time advance structures of discrete event simulation with some examples to show how these can be programmed in Pascal_SIM. These initial examples are all deterministic. When students comprehend the time advance mechanisms and are able to develop simulations using one particular world view (normally the three phase approach), we introduce, in turn, distribution sampling, the presentation of results, queue priorities and queue disciplines, variance reduction and animation. At each stage the students use case studies, examples and exercises in Pascal_SIM. In more advanced courses we would then present techniques for dealing with complexity in simulations, cover the process view in some detail, and introduce other software, probably a common simulation programming language such as GPSS.

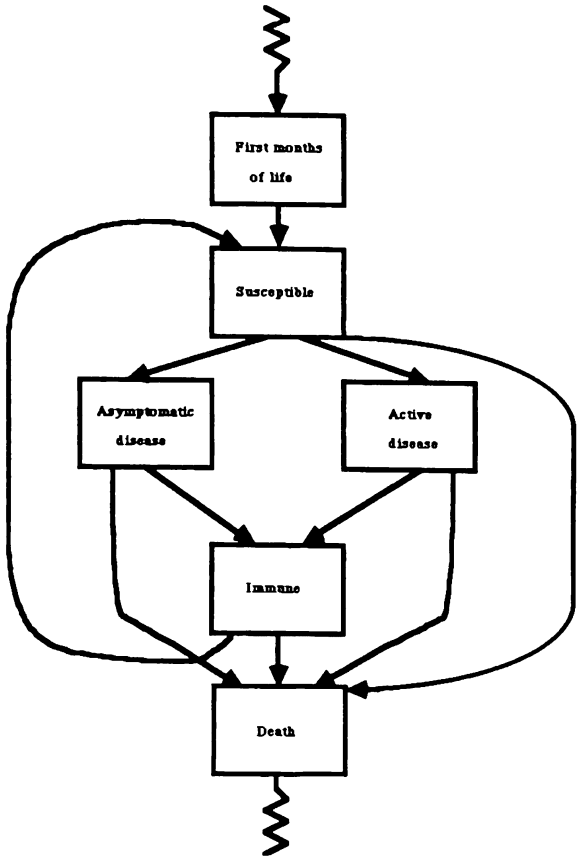

Figure 1: Activity diagram showing the state of health of patients in a village where the disease of trachoma is prevalent.

Using Pascal_SIM, good programming practices such as top-down design and modularity can be reinforced. An event or a process segment is a procedure, and can have local variables and procedures. This is particularly important for Computer Science students schooled in structured programming methods. The major advantage in using Pascal_SIM both for ourselves and our students is that it is easy to learn, extremely flexible and the routines can be easily extended to more complex structures (such as those for the health problems investigated by one of the authors and discussed in Davies and Davies, 1987).

Throughout, we make use of *activity diagrams* as a means of describing a model. Examples are shown in Figures 1 and 2. The diagrams may be closed, in which case it becomes the well known activity cycle diagram, or open. Added to this we emphasize listing of all queues, variables, assumptions, etc.; coupled with the diagram, this becomes a semi-formal specification of a model. We have also made use of pseudo-code to provide a stepping stone from this specification to a working program.

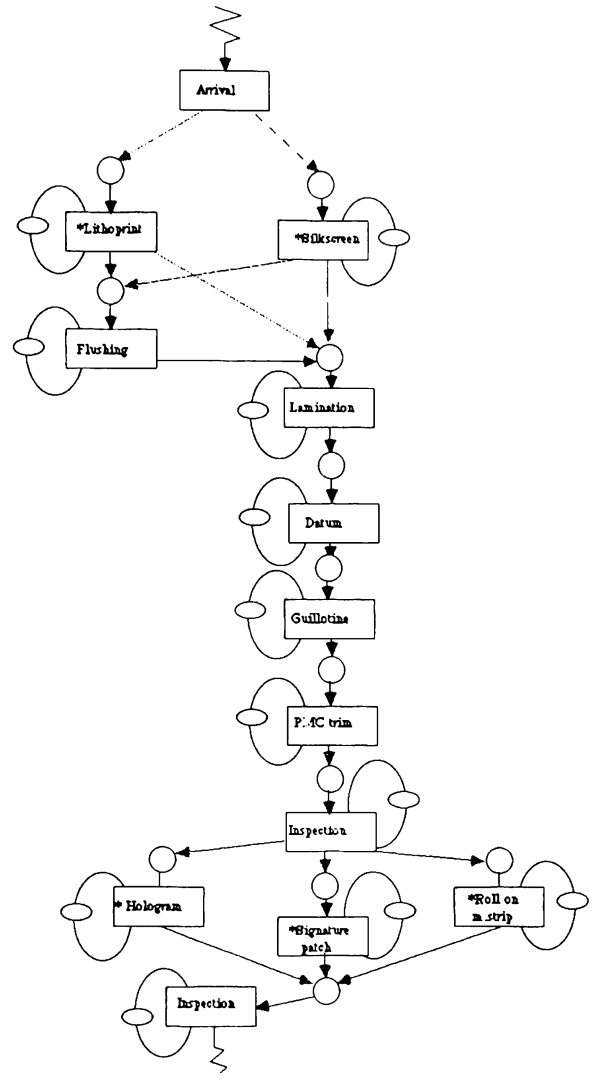

Figure 2: Activity diagram showing the production of laminated plastic cards. Hatched or dotted arrows indicate paths that are sometimes, but not always, followed. Asterisks indicate that more than one of the marked activities, at a particular point in the flow chart, may be performed in any order.

4. STUDENT PROJECTS

The authors have had experience of providing course work and supervising simulation projects over several years. Many of these involved students in collecting data and modeling systems in local hospitals, banks, garages and road junctions. Some provided a basis for cases studies that appear in the text book.

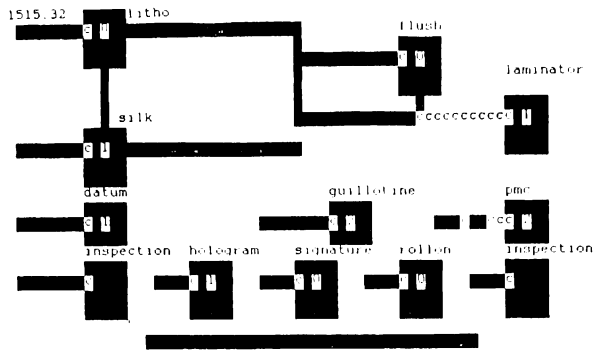

Figure 3: Visual display of the simulation modeling the production of laminated plastic cards. The letter 'c' indicates a batch of plastic cards. The number in the top left hand corner shows simulated time in minutes.

To show the type of work that has been done with Pascal_SIM, two case studies are presented here. Both were undertaken as a three month project required as part of the MSc in Operational Research at the University of Southampton. The projects are on this course are usually of a practical nature, many taking place in an industrial setting. In the academic year of 1987/88, although all the students learnt Pascal and followed a course about discrete event simulation, none were familiar with Pascal_SIM.

Three students engaged in projects requiring simulation techniques chose to use Pascal_SIM, largely because it was much more flexible than the alternatives available to them. Each project is very different from the process flow examples, normally grounded in manufacturing, that are prevalent in so many courses on simulation. All three students, with the help of the text book, were able to code working simulations within a couple of weeks. They then improved them, adding animation, more complex logic and output analysis as appropriate. One of the students was doing preliminary work for a larger project in which she was to model the spread of and possible treatments for AIDS. The other two projects are described here.

4.1. Modeling The Spread of Trachoma

Trachoma is a contagious disease that is prevalent in Africa and can cause corneal scarring and, hence, blindness. A person who has had the disease before can be re-infected after a period of immunity. The disease may, especially in younger children, be asymptomatic, but nevertheless may affect the eyesight in exactly the same way as an 'active' attack. Those who live in an area where particular strains of the disease are prevalent are increasingly likely to loose their eyesight as they become older; vaccination programs are available which induce immunity to the disease but only for a limited period of time. The purpose of the project was to write a simulation model which could be used to evaluate the benefits of different treatment and vaccination programs.

The student chose to describe the system quite simply. The only entities used represented people of the village, who were deemed to either be sighted or else to have lost their eyesight completely. Figure 1 shows the way in which entities pass through the following states: susceptible to disease, with active disease, with asymptomatic disease, and immune. When a simulated person 'died' the entity was removed from the simulation. The model kept track of the number of times each person passed through an active or asymptomatic disease state, and this number influenced the probability of eyesight loss after a disease phase.

If there was a vaccination program, it was assumed to take place at regular intervals and provide a period of immunity on each occasion. A treatment program with antibiotics, on the other hand, reduced the lengths of the periods of infection for each person but did not affect the periods of disease immunity. The program showed the effect of these programs on the numbers of patients with severe visual loss in the different age groups.

This was a simulation with a single type of entity whose choice of states and time in each state depended on various probability distributions. There were no constraints and hence no queues or conditional events. Each scheduled event set the time for the next change of state (i.e., the next scheduled event). Where the simulation branched and there was uncertainty as to which state an entity would go to next, the simulation identified the time to all possible next events and chose the shortest. For example, at the beginning of a period of immunity the next event might be a change to a period of susceptibility, renewed immunity (as a result of a vaccination) or death. If the program determined that a vaccination program was due before the person became susceptible again or was due to die, then the next event was the start of a another period of immunity.

The screen display provided some very simple graphics showing how the numbers in each state change as the simulation proceeds. At the end of each simulation run, histograms summarized this information, showing the trends over time.

4.2. Production of Credit Cards

The third student tackled a production line problem, the production of plastic laminated credit cards, which at first sight appears to have a classic process view structure. Items wait in queues to be processed on a series of different machines. However, complications arose from the fact that the items are always dealt with in batches, and activity times are a function of setup times and batch sizes. Furthermore, some batches require particular processes which are not required by others (see Figure 2).

The student was asked to look at the system because delivery dates were not being met. The main purpose of the simulation was thus to examine ways of reducing queue lengths and throughput times, and it was used to explore both the effects of providing additional machines to relieve bottlenecks, and also the effect of different queue disciplines. A comparison was made between FIFO queueing and the SPT (shortest processing time) rule where priority is given to the job with the shortest estimated processing time on the next machine.

There was a considerable amount of past data available with which to test the system, including: previous batch sizes, the processes they required and the dates they were requested. The data were used to derive distribution functions of arrival times and process times. The simulation, programmed in Pascal_SIM, provided a simple animation (see Figure 3 for a screen dump) to validate the simulation and show the build up of queues at different machines. It also provided for repeat runs with different random number streams so that statistical analyses could be performed on the resulting process times and queue lengths.

In running the simulation with FIFO queues, it became apparent that the laminator was a major bottleneck. The simulation showed that the effect of providing a second laminator would reduce the overall processing time by, on average, approximately 91 minutes. This was more than 50% of the total processing time of half of the batches (the smaller ones). The effect of changing from FIFO to SPT whilst there was only one laminator also, as might be expected, had a dramatic effect. The overall processing time was reduced by, on average, 38 minutes. However, when the number of laminators was increased to two, it no longer made any difference which queue discipline was used.

This student was able to use Pascal_SIM to make an analysis of the problem and to provide some very useful results for the company concerned. The most difficult aspects of the problem were the problem formulation and the data analysis. The non-standard aspects of the simulation structure, the rather complex activity time functions and the non-FIFO queue discipline, proved straightforward to program in Pascal.

5. USE OF ANIMATION

The animation facilities in Pascal_SIM use text based facilities (see Figure 3) that are available on most computer terminals. In the interests of portability, they do not take advantage of EGA or VGA facilities, for example. Nevertheless, the principles of using simulation graphics remain exactly the same and can be taught using the relatively simple set of procedures in Pascal_SIM. Both technical problems, such as how fast to advance the animation relative to the model, and design problems, such as what to show on the screen and what to leave off, can be considered.

Despite their limitations, students find the graphics facilities of Pascal_SIM extremely seductive. The authors have found that very few students would consider simulation course work or a project complete without a colorful moving picture indicating how the simulation is progressing. The ever present danger is that the animation is seen as an end in itself, and important experimental techniques such as replications for valid samples and variance reduction are completely overlooked. In order to teach these important techniques we have found that animations sometime have to be expressly forbidden.

6. CONCLUSION

Our general approach to teaching simulation with Pascal is to emphasize good programming practice, producing well designed code from a specification. The use of graphics and output analysis are taught as complementary, and neither is allowed to dominate the other. Pascal and Pascal_SIM are a very appropriate foundation for a course in simulation for some students, particularly those with good programming skills. Even if they go on to use more advanced specialist packages and languages, they learn about the inner details of simulation programming, and are exposed to all common world views.

ACKNOWLEDGEMENTS

Thanks are due to Mark Nicholson and Femke Galle, former students of the Mathematics faculty of the University of Southampton, who kindly allowed us to discuss their work.

REFERENCES

H. Davies and R.M. Davies (1987) A simulation model for planning renal services in Europe, *Journal of the Operational Research Society* **38**, 693-700.

R.M. Davies and R.M. O'Keefe (1989) *Simulation Modelling with Pascal*, Prentice-Hall, Englewood Cliffs, NJ, USA.

R.M. O'Keefe and R.M. Davies (1986) Visual discrete simulation with Pascal_SIM, in *Proceedings of the 1986 Winter Simulation Conference* (J.R. Wilson, J.O. Henriksen and S.D. Roberts, eds.), IEEE, Piscataway, NJ, 517-525.

AUTHORS' BIOGRAPHIES

BOB O'KEEFE is an associate professor in the Department of Decision Sciences and Engineering Systems at Rensselaer Polytechnic Institute, Troy, New York. He was previously a member of the Board of Studies in Management Science at the University of Kent at Canterbury and visiting assistant professor in the Department of Computer Science at Virginia Tech. He has a BSc in Computer Studies and Operational Research from the University of Lancaster, and a PhD from the University of Southampton. Research interests include Artificial Intelligence and simulation, Visual Interactive Simulation and the methodology of decision support and expert systems.

Robert M. O'Keefe
Department of Decision Sciences and
Engineering Systems
Rensselaer Polytechnic Institute
Troy, NY 12180-3590
USA

RUTH DAVIES is a lecturer in the Department of Accounting and Management Science at the University of Southampton. She has been working on the application of statistics, operational research and computing to health care, for a number of years. Current research interests include the application of simulation techniques to planning resources for patients with chronic diseases and the use of simulation in integrated environments. She has a BSc in Mathematics from the University of Warwick, an MSc in Neurocommunications from the University of Birmingham and a PhD from the University of Southampton. She has held research positions in the Universities of Reading and Southampton and has recently been a senior lecturer at the Polytechnic of the South Bank, London.

Ruth M. Davies
Department of Accounting and Management Science
University of Southampton
Southampton, SO9 5NH
England