Proceedings of the 2016 Winter Simulation Conference
T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick, eds.

DISCRETE EVENT SIMULATION ON THE MACINTOSH FOR BUSINESS STUDENTS - AGPSS AND ALTERNATIVES

Ingolf Ståhl

Department of Economic Statistics
Stockholm School of Economics
Box 6501
SE-Stockholm, SWEDEN

ABSTRACT

The paper first discusses the importance of discrete event simulation (DES) in the business school curriculum. It next notes how small Macintosh lap tops have become increasingly popular among business students. We next discuss what DES software is available on the Mac, first directly, then indirectly by running DES software for Windows in some way on the Mac. Noting that there is not much simple DES software on the Mac, but yet a great demand for such software from many business students, we turn to the transfer of one pedagogical software system, aGPSS, from Windows to the Mac. We here first give a brief historic background of aGPSS. Next we discuss some of the problems encountered when transferring aGPSS to the Mac. The paper ends with a brief discussion of some pedagogical aspects of using aGPSS on the Mac in the teaching of basic management science.

1 INTRODUCTION

As a teacher of simulation at a business school, I have often been asked: Why do you teach discrete event simulation, DES, at a business school? Why are you not content by just simulating financial flows in a spreadsheet? The answer is that I have found DES, implying dynamic stochastic simulation, to be of great importance in a business curriculum for several reasons.

First, I have had DES replace, or at least complement, many analytical/optimization parts of Operations Research or Management Science methods, such as queuing theory, inventory theory, PERT/CPM and parts of Decision Analysis. My students have liked this, since this has implied a greater focus on solving problems, and fewer methods to be learnt and forgotten.

Furthermore, DES has provided my students with a better understanding of the physical processes in a firm. DES is one way of giving business students an introductory understanding of some problems in the areas of production economics, material handling, inventory management, etc. Hence, in an introductory course on production economics for business students, I have had DES play an important role. Closely related to this is the demonstration of the connection between the physical activities and the consequential financial flows. For this kind of simulation, a general-purpose simulation system is of greater interest than a system focused entirely on manufacturing.

Stochastic simulation is also required for handling the uncertainty that is the core of financial theory. We can just think of how we answer the following questions: How much will we sell next year: 100,000 units for certain or 80,000 - 120,000 units? When will customer X pay: Within 30 days for certain or with 80 percent probability within 60 days? What will the $/€ ratio be a year from now: 1.10 for certain or between 1 and 1.2? In all three cases, the last answer, indicating uncertainty, seems more reasonable. If all future payments could be forecast with certainty, all corporate debt would be as safe as government bonds and there would then not be any need for different types of financial instruments and hence no need for financial theory. Against this background, it seems strange that most simulation of the future financial position of a corporation, e.g. cash forecasts, is done using deterministic simulation, without any uncertainty, by ordinary spreadsheets. Instead most financial simulation **should** be stochastic.

Finally, there is a need for **dynamic** simulation in the form of DES, allowing us to follow each major payment, regardless of when it takes place. This can be illustrated by Figures 1 and 2 of a cash forecast of a small corporation. These diagrams have been produced by a simple aGPSS simulation program for cash forecasting, presented in Born and Ståhl (2013). The program deals with an importer who buys and sells certain machines. It pays the producer in cash directly for each unit, but provides the customers with credit. Orders arrive according to an exponential distribution, while customers' payment times vary according to an Erlang distribution. Our students can write this program after ten hours of study.

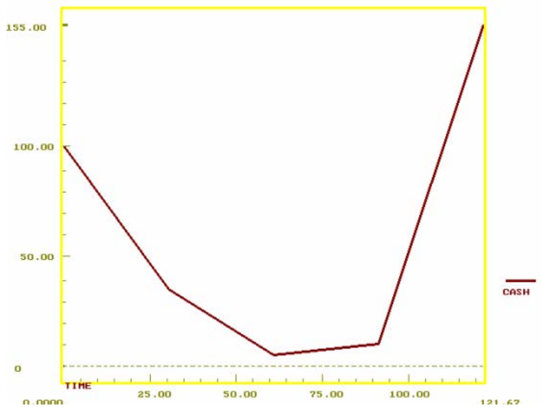


Figure 1: End of the month cash graph.

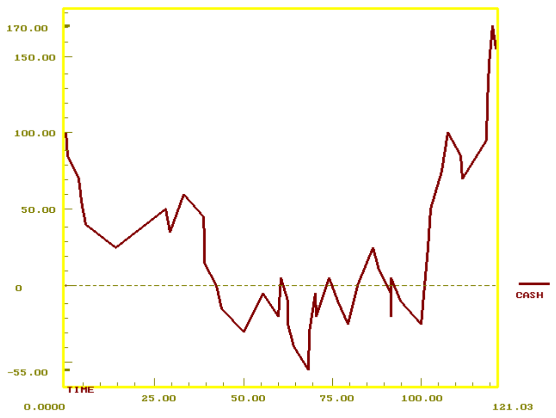


Figure 2: Dynamic cash graph.

We see that the two graphs give completely different impressions. From Figure 1 it appears that there would be enough cash for the corporation and hence no liquidity problems. The financial problems, with negative cash several times, are clearly seen in Figure 2, where we can follow payment by payment. In Figure 1 these problems are not perceived at all, since by chance there is a cash surplus at each end of the month. This illustrates the need for a dynamic, discrete-events approach to corporate financial planning. As discussed in Ståhl (1993), the need for this type of dynamic simulation for cash flow forecasts is especially large, when a couple of hundred large payments constitute more than half of the payments of the company, which is true for many smaller corporations.

Among other important applications of DES that are of interest in a business school, I should mention the establishment of optimal equipment life, overbooking and price differentiation in the airline industry, bidding on stocks of an oil company with uncertainty about the success of an oil exploration process, valuation of perfect information in decisions under risk and a model of stocking of perishable goods in a supermarket. Another example illustrates discrete/continuous simulation as applied to business problems with the aim of doing market forecasts and evaluations of high tech stocks. Another model deals with the evaluation of European options, which for the case of constant volatility can be solved analytically, but for the case of volatility varying over time requires simulation. Other applications include simulation based costing and the use of laptop based simulation models for sales support (Ståhl 2016).

2 DIFFERENCE IN DES FOR BUSINESS AND FOR ENGINEERING STUDENTS

It should be noted that there is in general a substantial difference between a DES course for business students and one for engineering students. First of all there is a difference in background, in particular as regards experience in programming. While most business students have little experience in programming, many engineering students, especially those in computer engineering, are experienced programmers.

Production engineering students, who use DES for manufacturing planning and plant layout, e.g. in engineering shops, are furthermore different from most business students as regards how they rate the importance of animation. Animation is important in many manufacturing situations, where the animation of all processes can be carried out in proportional time. Proportional time implies that the time compression ratio (time-in-reality/time-on-the-screen) is constan**t**, during at least a substantial part of the simulation. If, one process takes 5 minutes in reality and 5 seconds on the screen, then another process taking 60 seconds in reality will in the case of constant time compression ratio take 1 second on the screen. For many other animations, like service processes, one must on the other hand fake the times in order to make a "good looking" simulation. Animation is hence most interesting in situations where one can animate with the same time compression ratio for all processes. In situations where one is forced to choose between the three evils of either running a boring animation for a long time without much happening, distorting reality by having different time compression ratios for different tasks or having the animation very jumpy, one might instead often do just as well without animation.

Animation might furthermore not be very informative, when the simulation mainly deals with invisible processes, such as message flows, cash flows and accounting features like costs and profits. One must then choose symbols that often do not lead to much better model understanding than one could obtain by presenting graphs and text.

Another important difference between business students and manufacturing students is that business students are more interested in applications where one wants to analyze the effect of uncertainty. This implies that one must run the simulation many times. One then often wants to determine the confidence levels, e.g. between which lower and upper values 95 percent of all results would come, if one ran the simulation a million times, instead of e.g. just 100. When using animation for testing a plant layout, one is mostly content just running the model once or a few times. For this reason, business students are in particular interested in DES software that focusses on repeated runs instead of animation.

3 THE MAC IS INCREASINGLY USED BY BUSINESS STUDENTS

Both from my personal experience with business students in Sweden, Norway and Latvia and from studies of articles on the Web, I have come to the conclusion that the Mac is used by more and more business students. In the three schools where I have been teaching for many years, I have noticed two trends. Several decades ago most simulation work was done on Windows based PCs in the computer labs of the school. Gradually, more and more students did the work on their own laptops. First most lap tops were Windows based, but gradually more and more students have turned to the Mac, in the last years frequently of the Mac air type. At my business school, SSE in Stockholm, 75 percent of the students have stated that they have a Mac.

On the web I have found several articles claiming that around 70 percent of new college students are bringing Macs with them to school. There has during some years been an annual increase of about 10 percent in the market share for the Mac from a year ago (OSX 2010). Some articles discuss the reason for this increase. Some think that an information cascade can explain this Mac domination phenomena. More and more incoming college students choose to buy a Mac, because older students at the school are using a Mac, rather than a PC. Although data suggest that a PC is more compatible with most software products and accessible to more information, college students still dominantly choose Macs over PCs, just because of this powerful social conformity effect (Cornell blog 2012).

The strong position of the Mac might seem surprising in view of the higher price of the Mac, e.g. around $1000 for the smaller Mac air, compared to most ordinary Windows laptops, often at half the price or less. One reason for this lack of price insensitivity is that with an annual college cost, e.g. at some private colleges of $40,000, the price of a Mac, to be spread out over several years, seems almost insignificant. There is also the argument that the Mac, e.g. the Mac air with the 11.6 inch (29.5 cm) screen, has some advantages, like a low weight (1.08 kg), which is important when carrying it between classes. The higher price is according to some of our students even an advantage, since it gives the Mac a prestige status. It should finally in this context be mentioned that, according to some claims on the Web, the dominance of Mac is greater for business students than for engineering students (PhysicsForum 2014).

4 DES SOFTWARE ON THE MAC

Against this background of heavy use of Macs among business students, not the least our own, it has been of interest to see what kind of software for DES there is on the Mac. Our starting point is the Simulation Software Survey in the October 2015 issue of ORMS Today (Swain 2015). This is a list of 55 discrete simulation software products from 31 vendors, who have provided the data for this survey.

One of the questions asked the vendor is on which operating systems the software runs. For this paper, a question of interest is whether the simulation software could be used to build up simulation models in Windows or on the Mac. While some of the products seem to concern running already constructed models, e.g. on the Web, a total of 51 of the products seem to allow the building of simulation models. All are stated to do this in Windows. In contrast, for only 7 of the products, the vendor has stated that the software runs on the Mac. These are, besides aGPSS (just transferred to the Mac and to be discussed later), AnyLogic, DPL, ExtendSim, JaamSim, SIPmath and TLS. Of these, DPL and SIPmath do not seem to deal with DES and TLS is stated to be specific for the mining and oil industry (Swain 2015).

Besides aGPSS, I have some knowledge of AnyLogic and ExtendSim of Imagine That Inc., not the least through meeting with some key persons behind these products. My first contact with ExtendSim was already at an Apple educational conference in San Jose in the late 80s. ExtendSim was then the only DES product presented at this conference. Extend later moved to Windows. The current version of ExtendSim runs on Mac OS X 10.6.8 (and earlier versions), but it does not run on Mac OS X 10.7 (and later versions). Imagine That Inc. is currently rewriting ExtendSim in a newer development environment that will natively support Intel-based Macs. There is not yet a definite date when this process will be completed.

AnyLogic claims to be the only software that combines discrete event, system dynamics, and agent-based simulation methods. AnyLogic allows users to easily develop 2D and 3D animation. Prebuilt object libraries allow users to create models by dragging and dropping elements from palettes. The libraries include e.g. road traffic, pedestrian dynamics and railways. Models can be exported to a website, where they can be shared with others. AnyLogic is Java based and is quite advanced, which perhaps can be seen from Figure 3 on the next page, showing the workspace of AnyLogic. This figure is presented at the start of their free textbook *AnyLogic 7 in Three Days* (Grigoryev 2015), which is available on the Web, also in German, Japanese and Chinese. The core of this book are four practical examples, covering all the main software functionality. Two examples, a small job shop and an airport, deal with DES. There is also a larger book of 614 pp. (Borshchev 2003). There is also a free AnyLogic Personal Learning Edition, which is downloadable from the AnyLogic website (AnyLogic 2016). There are for this version certain size restrictions, e.g. on the number of dynamically created agents, 50. The downloadable file is larger than 400 Mbyte, which indicates that that AnyLogic is quite a complex system.

It should here be stressed that AnyLogic is a quite advanced professional software, used for many types of large applications of many different types. It can hence not to be directly compared with the aGPSS software, to be discussed further below. aGPSS has a much more limited main purpose, namely to be very easy to learn and be suitable for giving business students a tool for carrying out project work in a company as part of a course on simulation, intended to require roughly one month of student time.

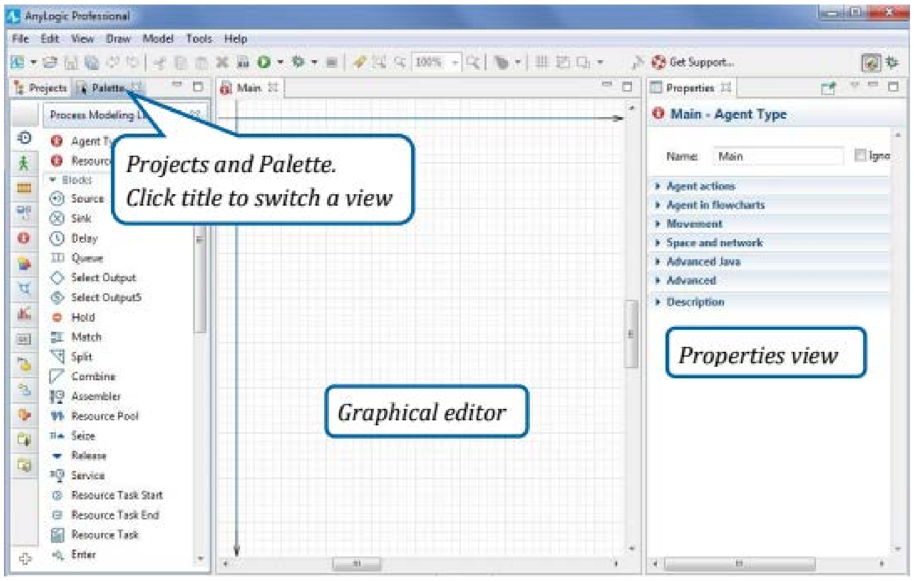


Figure 3: Workspace of AnyLogic.

JaamSim from Ausenco is a free and open source discrete event simulation software licensed under Apache 2.0. It includes a drag-and-drop user interface, interactive 3D graphics input and output processing, and model development tools and editors. The latest version of the software can be downloaded from the JaamSim website. The downloadable file of the software is only around 7 Mbytes. A starting screenshot is presented in Figure 4 below. The source code is published on GitHub. A manual for JaamSim can also be found on the JaamSim website. This manual of 118 pages can, however, not be used as a tutorial, since it lacks concrete model examples (JaamSim 2016).

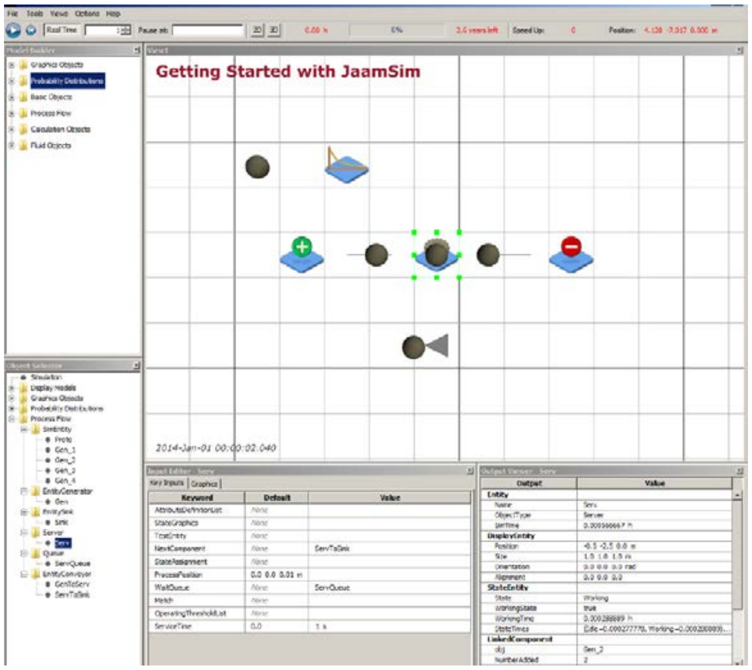


Figure 4: Screenshot of the JaamSim Basic Example.

Seeing that there are so few DES software products that run directly on the Mac, while there are so many that run under Windows, a natural question arises as to the possibility of running Windows products in some way on a Mac. Why should one bother to create a special version for the Mac, if students can run the simulation system on Windows installed on the Mac? Looking more closely at this issue on the web, I found that there are three main possibilities to run Windows software on the Mac, mainly using some kind of emulator (MacWindows 2016).

The first possibility is to use a tool called *Boot camp*, which is a built-in utility on Mac, which lets you run Windows applications on your computer. Shortly after the first Intel-based Macs arrived on the market, Apple released this tool, which lets Mac owners install and boot their machines natively into Microsoft Windows alongside an existing OS X installation without running two operating systems concurrently. One major drawback of Boot Camp is, however, that you have to reboot your Mac every time you want to switch between Mac and Windows. This is one of the primary reasons why many Boot camp users have switched to the second method to be discussed next.

This second possibility is to run a so called Virtual machine software. The most popular of these seems to be *Parallels.* This software can cost a little more than $50.00, but you must also get a copy of the Windows operation system. In contrast to Boot camp, there is no need to reboot, so you can conveniently access Windows and Mac programs simultaneously. It also allows you to choose various views, including full Windows view, full screen mode and coherence mode. Similar software products are *VMware Fusion* and *VirtualBox*, freely available as Open Source Software. If one plans to run several Windows software systems on one's Mac, this might be of interest. For our students, who might just want to run only one DES software on their Mac, this has not seemed as an attractive alternative. They would then have to buy and install both Windows and another application software on top of the DES software, thus roughly doubling installation time and software costs.

A third alternative is to use *CrossOver* of Codeweavers. This is a software that runs thousands of different Windows programs on Mac. It allows for an easy switch between Mac and Windows programs without rebooting, without using a virtual machine and without purchasing a Windows license. CrossOver translates the commands of the Windows software, so it can run like a Mac product. It is, however, a big process to recreate all the Windows operating system commands and there does not seem to be any general DES products that run well on CrossOver. On a Web site (CodeWeavers 2016) one can check whether a software can run on the Mac using CrossOver. I have checked 16 different general DES products. Only for Arena of Rockwell Automation did I get anything, but a NO answer. Arena is given a 3 star rating, saying that there is a limited functionality with CrossOver, but a 5 star rating is recommended for flawless usage.

It should in this context finally be stressed that my main skepticism as regards the use of these three Windows emulating software systems for use by our business students in an introductory simulation course does not concern performance, e.g. in terms of execution speed, but rather that I foresee that our type of students would find it quite bothersome to acquire, install and learn to use such additional software.

5 HISTORY OF AGPSS

My conclusion from the section above is hence that there are few alternatives for simple DES on the Mac for my kind of business students, who shall do projects in a company after only 12 – 20 class room hours of a simulation system. It has hence appeared suitable to transfer a Windows based DES software, which we for decades have used to teach such students, to the Mac. Before going into details about the transfer process, I shall give some more information about this software, aGPSS.

aGPSS is the most modern version of GPSS, General Purpose Simulation System, which was developed by Geoffrey Gordon at IBM and was made available to the public in the fall of 1961. Although it was not the very first computer software system for discrete event simulation, it very soon became the dominant simulation software. The very first Winter Simulation conference, held in 1967, was called "Conference on Applications of Simulation Using GPSS". In 1972, GPSS was ranked in the tenth position among what were judged to be the world's thirteen most important programming languages of any type (Sammett 1972). The popularity of GPSS, in particular within education, was further enhanced by the appearance in 1974 of T. Schriber's famous Red book, *Simulation Using GPSS* (Schriber 1974). Surveys show that GPSS might still be among the most used simulation systems (Dias *et. al.* 2007). Within education, especially in Europe, its relative position is even stronger.

The fundamental aspects of GPSS are discussed at length in three papers presented at the WSC, *GPSS Turns 40: Selected Perspectives* (Schriber *et. al.* 2001), *GPSS - 40 Years of development* (Ståhl 2001), and *GPSS – 50 years old, but still young* (Ståhl *et. al.* 2011), all available on the WSC web. Gordon's original GPSS (1) was followed by GPSS 2 (in 1963), GPSS III (in 1965) and GPSS/360 (in 1967). As discussed in Ståhl (2001), GPSS (1) and GPSS 2 were quite similar, while GPSS III implied a big change from GPSS 2. GPSS/360, the system described in the Red book, was, in turn, quite similar in basic structure to GPSS III.

I started my teaching of simulation in the mid-80s when teaching the main course in Management Science at SSE (the Stockholm School of Economics). The textbook used was MacMillan and Gonzales (1973), containing a chapter of 77 pages on GPSS, written by Schriber. To learn more about GPSS, I acquired his Red Book. This book has a total of 27 case studies, which all refer to practical problems, but have a simple program code.

I wanted my students to use GPSS for making their own small models. At this time, GPSS was in Sweden available only on IBM mainframes, which SSE lacked, so our students had to walk a mile to another college. The students enjoyed GPSS, but did not like this walk. Hence, when SSE acquired a mini-computer, which lacked a GPSS system, I decided to make my own "mini-GPSS" system. This started as a small subset of GPSS/360. The system was programmed in FORTRAN, the best programming language at this time. It was a proper subset of GPSS/360, with two exceptions. It avoids a major critique against IBM GPSS, namely that strange results are obtained, if a GENERATE block is immediately followed by a SEIZE block. There are separate internal waiting lists for **each** server instead of one single current event list, leading to higher efficiency.

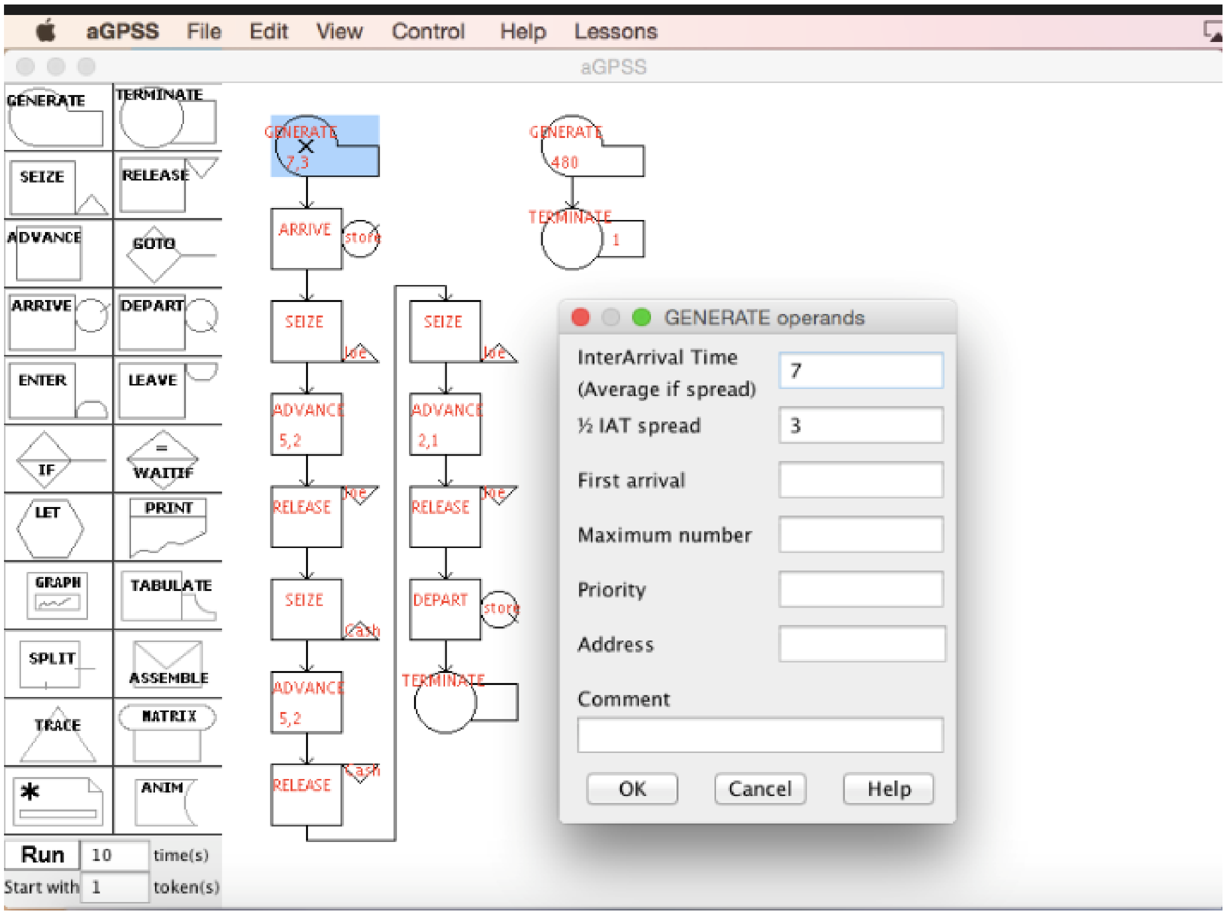
In 1983-85, I was visiting professor at Hofstra University, NY, where I taught an elective full-semester course in simulation five times. This course was devoted almost completely to GPSS, which was first run on an IBM main frame. I had, however, already started to transfer my mini-GPSS to the new IBM PC micro-computer. The new system, called micro-GPSS, could be used in 1984. Since the whole course was devoted to simulation, I found it important to have the students make a simulation project in a company, with a focus on modeling, data collection, etc. As I expanded this subset of GPSS, I found it important to include features that the students needed in their projects and yet keep GPSS easy to learn. The primary guide on what to include was the 27 case studies in Schriber (1974). I could rewrite not only all of these 27 case studies, but also about 99 percent of all programs in other GPSS textbooks, with roughly the same amount of code in our GPSS as in standard IBM GPSS.

Back in Sweden, I introduced GPSS as a quarter of a full-semester course taken by all 300 students at the SSE. Here I used several teaching assistants and got a great amount of feedback on what was difficult to learn and also what was difficult to teach. Based on this feedback, I simplified the syntax in several respects. micro-GPSS changed from a pure subset of GPSS/360 to a stream-lined version. The syntax of many blocks was changed to conform with well-known programing logic like that of BASIC. Statistics collection was furthermore greatly simplified and features that could lead the students to make difficult-to-find logical errors were eliminated. I also developed a system with 500+ error messages. To encourage replications, I had our GPSS make replications of the runs by just one single command, and also automatically carry out a statistical analysis of these repeated runs, e.g. of confidence levels. I also made it easy to create functions based on empirical data and easy to change these functions as new observations are made.

In the late 1980s, micro-GPSS was also transferred to the Mac and presented at an Apple educational conference at San Jose. In 1990, a textbook (Ståhl 1990), based on micro-GPSS, was published by Prentice Hall and students in many countries then ordered the software. This micro-GPSS was, just like GPSS/H, a purely text-based system, but there was an increasing student demand for building the models using a **G**raphical **U**sers **I**nterface. In 1997, a new Swedish Foundation for Computers in Education was looking for university developed software that could be placed on the Web. This was a chance to get financing of roughly $150,000 to have a GUI-based micro-GPSS put on the Web. In 1999 we presented the first tentative versions of such a system, developed in Java. This was first called WebGPSS. As discussed below, it later became the basis for aGPSS.

The models are built in the client applet on the user's PCs. The users here build up the program in the workspace, exemplified in Figure 5 below. Blocks are chosen by clicking on a symbol in a menu to the right, leading to the symbol appearing in a block diagram. This point-and-click method allows for a faster model build-up than the drag-and-drop method. By next clicking on a block in the block diagram, a dialog with syntax explanations is opened to allow for the input of the operands of the block.

The execution of the code was carried out by a server, located on a remote central computer, using the "engine" of micro-GPSS. The server produced output files in ASCII format, sent over the web to the user's computer, where a GPSS applet turned these files into tables, histograms and graphs. This output, shown under several tabs, is more readable and understandable than the output generated by standard GPSS. It is easy to print and save each result tab. Besides the block diagram and block dialogs, our GPSS also provides a single easy-to-read program listing, which is clear and compact and allows for short comments. This listing also makes it easy for the teacher to correct and mark the student programs.



Run 10 time(s)
Start with 1 token(s)

GENERATE operands

InterArrival Time (Average if spread)	7
½ IAT spread	3
First arrival	
Maximum number	
Priority	
Address	
Comment	

Figure 5: aGPSS-WebGPSS workspace, block diagram and block dialog on the Mac.

Figure 5 on the preceding page shows examples of the main features of aGPSS/WebGPSS. In the workspace we see a model of a bookstore in which customers first go to the seller Joe to choose a book and next to the cashier to pay for the book. Finally, they return to Joe to pick up the book. We note that Joe in this case can be in two different places in the model. This allows for much simpler program code than in an animation based model, where Joe can be in only one place.

To the left we see the block dialog of the GENERATE block. The block dialogs allow for the input of operands for the blocks and contain most of the syntax information needed.

WebGPSS was made generally available on the Web, run on a Swedish server, and was used not only in a few Swedish colleges, but also in several high schools, in particular for individual projects in the senior high school year. It should be noted that since WebGPSS was an applet on the client, it was runnable not only on Windows, but also the Mac. With the advent of WebGPSS, students using our GPSS, mainly in Sweden and the US, both those running on PCs and on Macs, switched from the text based versions of micro-GPSS to the new WebGPSS.

The execution of the simulation program on the central server would, however, within a few years prove problematic. More and more students used WebGPSS. With new facilities for repeated runs for calculating confidence levels and making simple optimization, student runs became more computer time consuming. These developments lead to server overloading, when many students ran WebGPSS simultaneously. Hence, a strong demand arose for a stand-alone version for Windows, to be run without the web, and with both the server and client units on the same single PC. This new version, mainly a Java application, roughly 35 Mbytes in size, replaced the Web version a decade ago. It was then found suitable to change the name to aGPSS, stressing that it is meant to be the first simulation system for beginners. To get this new aGPSS available fast, it was made available only for Windows, leaving the Mac users without aGPSS.

There has, however, among our students been an increasing demand for a new Mac version. During the last few years, when I have been teaching simulation with GPSS in Sweden, Germany, Latvia and Norway, I have from an increasing number of students got a request for an aGPSS version for the Macintosh. Such requests have come also from users of aGPSS, studying with other teachers or doing self-studies, notably in the US, India and Japan.

6 TRANSFERRING AGPSS TO THE MAC

The main aim of the transfer process was to keep the Mac version as similar as possible to the old Windows version. This has not been completely possible.

One difference has been regarding animation. As seen in Figure 5 on the preceding page, there is an ANIM block to the right towards the bottom of the symbol menu. Under *File* in the main menu one can also ask for automatic animation. Both ANIM and this automatic animation are features that were introduced into aGPSS fairly recently. They both allow for the production of files that can then be used by the Proof animation system (Ståhl et. al. 2011). Since Proof is not available on the Mac, a Mac user must transfer the thus generated files to a Windows system to get animation. Since few of our business students have used this animation systems in their project work, this does not seem as a major problem.

With aGPSS no longer running only under Windows and on a separate server, we have a new aGPSS icon. One click on this icon is now all that is needed to start aGPSS, This is a lot simpler than the earlier procedure requiring clicks on both a server and a client button. Since on the Mac there is no letter symbol in the upper left hand of the block dialogs, but instead three colored dots, we can in the Windows version keep *W* (for **W**indows) in this corner of all dialogs. There are also other automatic changes in the menu systems. The main menu of aGPSS on the Mac, as seen in Figure 5, contains *aGPSS File Edit View Control Help Lessons,* but the corresponding menu in Windows does not need *aGPSS*. In both this menu and the dialogs, the three colored dots of the Mac system replace the Windows square icons for closing, size change and hiding.

Under *Lessons*, there are a total of 18 lessons, which contain a summary of the first 18 lessons in Born and Ståhl (2013). Moving to the Mac also involved changing some of these lessons, so that they can be used for both systems. Some smaller changes were also made in the Help files.

Another difference has been regarding the text in the block dialogs. The main cause of this is that with the same Java program there will be larger letters used on the Mac, sometimes with the result that the words in the dialogs do not fit any longer. Hence, we had to rewrite some of the texts in the dialogs of 8 of the 22 blocks. For example, in the GENERATE block dialog we replaced "Average if ½ spread" with "Average if spread". Similar changes were done in several of the other dialogs. This will eventually affect also the revised textbook, where 26 of the 431 figures will have to be replaced (Ståhl and Born 2016).

A final difference concerns how a matrix in form of a CSV file, e.g. from Excel, is read by the aGPSS system. In our earlier Windows system, the file had to be loaded into a special directory, c:\GPSS, where one also had to put the program that reads the matrix data. This did not work on the Mac. Instead one pastes all the matrix data directly after the final END in the model in *Text Editing* dialog. The model then reads this data directly and one does not have to bother having a directory c:\GPSS. This is clearly much simpler and it has therefore now also been introduced into the Windows version.

7 SOME PEDAGOGICAL ASPECTS OF USING AGPSS

I shall end the paper by pointing at some lessons that I have learnt from teaching DES to business students for over four decades and which still have guided me, not the least in the transfer of aGPSS to the Mac.

One lesson is that is important for users of Mac with e.g. the small 10.6 inch screen that the workspace with the block symbol menu and the block dialogs as simple and readable, as seen in Figure 5. This is also very important when projecting the aGPSS models on a moderate size screen in a class room with many students. Also those in the back rows musts be able to read everything. In order to keep this workspace simple, I have reduced the number of block types and simplified the block dialogs (Ståhl 2003). Only if one does all simulation on a computer with a large screen, e.g. over 20 inches, or only has a few students or a very large projection screen, then all the details shown e.g. in Figures 3 and 4 can be easily read. One must acknowledge that many situations in education are quite different from those in business.

Another important lesson is that it for the learning process is efficient to avoid overwhelming the students with difficult technical details. Based on feedback from our students, we have constantly tried to simplify aGPSS, without decreasing its power. In contrast to other software developers, we have not been interested in adding new spectacular features.

I have rather wanted to ensure that students at a very early stage get to write programs that are simple, but not trivial. The best learning results have been obtained when I have proceeded step by step, with simple examples in the beginning, so that no student is left behind at an early stage and loses the possibility to catch up, and when students do not have to learn a new concept every time that a new and different thing is to be done. Students have liked to find that the new aspects can be handled using already known concepts, even if the programs thereby become slightly longer.

This has also been the strategy behind the instructional lessons associated with aGPSS. As seen in Figure 5, the menu contains at the right the text *Lessons*. Clicking on this text, one can reach the first 18 lessons, which are abbreviated forms of the first 18 lessons in the textbook (Ståhl and Born 2016). This book contains 50 lessons, covering in 440 pages **every** aspect of aGPSS, easy for both the teachers and students to master. We have found that the teach ware is as important as the software when it comes to enabling students to do simulation modeling.

This is all connected with my basic goal of giving my students at least some idea about discrete event simulation, DES, in order to create informed buyers of simulation services. Knowledge of simulation is also important for being able to "sell" the idea of making a larger simulation project to top-management. It should be stressed that my goal has not been to teach a specific simulation system as such. The simulation software used is only a tool to allow the students to produce simulation models. It is fairly unlikely that a student in the future will work with the specific software learnt in the course. It is more important to have the student open to several types of simulation software, also those providing the model in text code.

In most of my courses I have made the students work through the whole simulation process as regards some concrete problem, usually in a company. In this way, the student actively learns the whole process, from formulating the question to be answered, delimiting the problem system, outlining the model graphically, gathering data, coding the program, verifying, validating and documenting, running the program a sufficient number of times, doing a statistical analysis for drawing significant conclusions, and presenting the results in a form suitable for a potential user, with a focus on implementation.

The students have first made a reasonably valid simulation model of the present set-up. They have gathered data (on items like arrival and service times and on waiting lines) from the real system and then compared the output data (e.g. on waiting lines) from the tentative model with real data to validate the model. If not validated, the model has to be adjusted. Finally, the students have provided and tested a suggestion for an improvement of the system. An important aspect of this course is hence that less time was spent on learning simulation software and more on the other aspects of the simulation process. The best way to learn all these aspects is through these simulation projects.

I have in such courses had good experience of students, in groups of two or three, doing project work in different Swedish and Norwegian corporations. I have guided some 1000+ student projects. They have dealt with a great variety of subjects, for example in banking, telecommunications and retailing. Quite a few projects have involved "sales support simulation models", where the simulation model was run on a laptop, interactively with a client, e.g. in order to determine the optimal configuration of a machine. (Ståhl 2016). Many of the project programs have had continued use in the corporations. The work on the project, of a couple of man weeks, has in many cases led to extended work in the form of a Master's thesis and/or in several cases landed the student a job in the project company. In many cases the aGPSS model has been seen as a sort of prototype model, which has then been expanded by a team in the company, using a more complex simulation system (Ståhl 2002).

Also in the cases when our students have not continued with simulation, the project work has according to most student evaluations proved valuable. The students, often with little prior business experience, but now having some expertise in a field often missing in the company, have been able to interact with managers and have gained not only communication skills, but also increased self-confidence.

REFERENCES

AnyLogic. 2016. http://www.anylogic.com/downloads

Born, R. and I. Ståhl. 2013. *Modeling Business Processes with a General Purpose Simulation System. Part 1 - Introduction.* Stockholm: SSE.

Borshchev, A. 2013. *The Big Book of Simulation Modeling. Multimethod Modeling with AnyLogic 6.* St. Petersburg: AnyLogic.

CodeWeavers. 2016. http://www.codeweavers.com/compatibility/search/

Cornell blog. 2012. https://blogs.cornell.edu/info2040/2012/11/05/macs-for-the-win-for-college-students/

Dias, L., G. Pereira and A. Rodrigues. 2007. "A Shortlist of the Most Popular Discrete Simulation Tools." *SNE - Simulation News Europe* 17(1).

Grigoryev, I. 2015. *AnyLogic in Three Days*. http://www.anylogic.com/free-simulation-book-and-modeling-tutorials

Jaamsim. 2015. http://jaamsim.com/docs/JaamSim%20User%20Manual%202016-02.pdf

OSX. 2010. http://osxdaily.com/2010/08/05/70-of-college-freshman-use-macs/

McMillan, C. and R. Gonzales. 1973. *Systems Analysis – A Computer Approach to Decision Models.* 3rd edition. Homewood, IL: Irwin.

MacWindows. 2016. http://www.macwindows.com/emulator.html

PhysicsForum. 2014. https://www.physicsforums.com/threads/windows-vs-mac-for-engineering.758594/

Sammett, J.E. 1972. "Programming Languages: History and Future." *Communications of the ACM* 15:601-611. New York: ACM.
Schriber, T.J. 1974. *Simulation Using GPSS*. New York: Wiley.
Schriber, T., S. Cox, J. Henriksen, P. Lorenz, J. Reitman and I. Stahl. 2001. "GPSS Turns 40: Selected Perspectives." In *Proceedings of the 2001 Winter Simulation Conference,* edited by B. Peters, J. Smith, D. Madeiros and M. Rohrer, 565-576. Piscataway, New Jersey: Institute of Electric and Electronic Engineers, Inc.
Ståhl, I. 1993. "Discrete-event Simulation for Corporate Financial Planning". In *Proceedings of the 1993 Winter Simulation Conference,* edited by G.W. Evans, M. Mollaghasemi, E.C. Russell, and W.E. Biles, 1264-1169. Piscataway, New Jersey: Institute of Electric and Electronic Engineers, Inc.
Ståhl, I. 1990. *Introduction to Simulation with GPSS - on the PC, Macintosh and VAX.* Hemel Hempstead, UK: Prentice Hall International.
Ståhl, I. 2001."GPSS – 40 Years of Development". *Proceedings of the 2001 Winter Simulation Conference*, edited by B. Peters, J. Smith, D. Madeiros and M. Rohrer, 577-585. Piscataway, New Jersey: Institute of Electric and Electronic Engineers, Inc.
Ståhl, I. 2002. "Simulation Prototyping." In *Proceedings of the 2002 Winter Simulation Conference,* edited by E. Yücesan, C-H. Chen, J.L. Snowdon, and J.M. Charnes, 572-579. Piscataway, New Jersey: Institute of Electric and Electronic Engineers, Inc.
Ståhl, I. 2003. "From 44 to 31 to 28 to 22 and now to 18 – Less Becomes More in GPSS." *Simulation and Visualisierung 2003,* edited by T. Schulze, S. Schlechtenweg and V. Hinz. Magdeburg: SCS.
Ståhl, I., J. Henriksen, R.Born and H. Herper. 2011. "GPSS 50 Years Old, but Still Young." In *Proceedings of the 2011 Winter Simulation Conference,* edited by S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, and M. Fu, 3952-3962. Piscataway, New Jersey: Institute of Electric and Electronic Engineers, Inc.
Ståhl, I. 2016. *100 aGPSS Models - Cases and Exercises*. Stockholm: SSE.
Ståhl, I. and R. Born. 2016. *Business Modeling with a General Purpose Simulation System – In Windows and on the Mac.* Stockholm: SSE.
Ståhl, I., J. Henriksen, R. Born and H. Herper. 2011. "GPSS 50 Years Old, but Still Young." In *Proceedings of the 2011 Winter Simulation Conference,* edited by S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, and M. Fu. Piscataway, New Jersey: Institute of Electric and Electronic Engineers, Inc.
Swain, J. 2015. "Simulation Software Survey." *ORMS Today* 45(5):36-49.

AUTHOR BIOGRAPHY

INGOLF STÅHL is Professor Emeritus at the Stockholm School of Economics, Stockholm, of a chair in Computer Based Applications of Economic Theory. He has taught GPSS for forty years at universities in Sweden, Norway, Germany, Latvia and the USA. Based on this experience, he has led the development of the aGPSS educational simulation systems. His email address is <ingolf.stahl@hhs.se>. His aGPSS system is at <http://www.agpss.com/>.