Proceedings of the 1998 Winter Simulation Conference
D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannan, eds.

CRITERIA FOR SIMULATION SOFTWARE EVALUATION

Jalal Nikoukaran
Vlatka Hlupic
Ray J. Paul

Department of Information Systems and Computing
Brunel University
Uxbridge, Middlesex UB8 3PH, UNITED KINGDOM

ABSTRACT

In simulation software selection problems, packages are evaluated either on their own merits or in comparison with other packages. In either method, a list of criteria for evaluation of simulation software is essential for proper selection. Although various simulation software evaluation checklists do exist, there are differences in the lists provided and the terminologies used. This paper presents a comprehensive list of criteria structured in a hierarchical framework for simulation software evaluation consisting of seven main groups and several subgroups. An explanation for each criterion is provided and an analysis of the usability of the proposed framework is further discussed.

1 INTRODUCTION

Simulation has become a popular methodology (Doukidis and Paul, 1990) with a broad range of applications (Gogg and Mott, 1993; Shannon, 1992). As a result of this, many software packages have been developed for modelling simulation problems. The growing number and quality of simulation software (Haider and Banks, 1986; OR/MS Today, 1991; OR/MS Today, 1993), the cost of the package, the set up cost and the running cost (Grant and Weiner, 1986) and the fact that the complexity of simulation packages requires expertise for their evaluation make the selection of an appropriate simulation package a vital issue to simulation practitioners (Nikoukaran and Paul, 1998b).

The problem of criteria identification and their structuring in terms of a decision model is central to multi-attribute decision-making (Keeney and Raiffa, 1976). Most commonly, such models are developed in a hierarchical fashion, starting from some general but imprecise goal statement, which are gradually refined into more precise sub and sub-sub goals (Stewart, 1992). The analyst's role is to ensure completeness and avoid redundancies in structuring the objectives, to clarify the distinction between ends and means, and to structure the relationship between objectives into a hierarchy (McDaniels, 1996). The task of simulation software evaluation and selection, which is of a multicriteria decision making type, is usually time consuming. Deaver (1987) said "[The company] spent a year evaluating systems". As time is money, by reducing the time taken to evaluate packages we can reduce the cost of evaluation. This can be achieved by speeding up the process of learning and testing the features of a package. For this purpose a detailed knowledge of the features of simulation packages can be helpful, irrespective of the method used for the selection. The criteria for simulation software evaluation may be obtained using facilities such as related articles, experts' advise, vendors' information, software manuals, and by working with some simulation packages.

This paper addresses a list of criteria presented in a hierarchical framework for evaluating simulation software and it is structured as follows. Previous work on the subject and the need for a framework is discussed. The hierarchical framework is presented and an explanation is provided for each criterion and sub-criterion introduced. The usability of the framework is further discussed and conclusions are drawn.

2 EVALUATION CRITERIA FOR SIMULATION SOFTWARE

In contrast to the shortage of simulation software evaluation techniques and selection methodologies in the literature, many papers and books have stated their preferred list of important criteria for simulation software evaluation, however, the lack of a standard common list is apparent (Nikoukaran and Paul, 1998a). Various terminologies used by experts, sometimes without a clear meaning, may be a reason for not having a common list of criteria. A standard list of criteria, an explanation and an example for each criterion could overcome some of the pitfalls. Due to progress in the subject, new computer technology, and changes in the features of packages, it may not be possible to provide a standard list of criteria. But it could be possible to provide a dynamic list with a possibility of changing it or adding new criteria without much impact on the whole methodology of selection.

Experts have categorised their choice of criteria into several groups which varies from three to eleven (Nikoukaran and Paul, 1998a). The number of groups is important. If too many groups are introduced, it may be difficult to assign a criterion to a single group, as the criterion may belong to more than one group. If too few groups, selection may not be clear enough.

3 A HIERARCHICAL FRAMEWORK

Analysis of problems with multiple criteria requires the steps of identifying objectives, arranging these objectives in a hierarchy, and then measuring how well available alternatives perform on each criterion (Olson, 1996). We have classified the criteria for simulation software selection in a hierarchical structure as described below. The usability of this hierarchical form is explained in section 4.

The software, the vendor and the user are the important elements which form the elements of the highest level of the hierarchy. Software covers a wide range of issues. Considering the process of modelling a problem using a simulation package, we have defined the following sub-criteria: model and input, execution, animation, testing and efficiency, and output (Figure 1).

3.1 Vendor

This criterion is for evaluation of the credibility of the vendor, and to some extent his/her software (Figure 2).

Pedigree: The issues related to the history of the software and the vendor are categorised in this group.

An evaluation of this criterion could tell us, to some extent, how reliable the software and the vendor can be. Vendor history would determine several points. How long the vendor has been in this job. How successful and famous the vendor is in the market. How many experts the company has. How many customers they are servicing. Whether the vendor is supplying any other software and how successful they are. These issues will give an indication of the reputation of the supplier. Similar issues could be related to the reputation of the simulation software offered by the vendor. How long it has been in the market. How many users are using it. How successful and famous it has been. If there are any independent references about it, specially those that describe achievements provided by its use.

Documentation: Good documentation will enable the user not to be dependent on the supplier for every minor problem. The availability of a user manual with indexes and reference cards with important information and the main commands, and tutorials which can help the user learn how to use the package are features of good documentation. A few examples, which the user can go through, will help the user to learn the software more quickly. Providing an introduction to simulation and some statistical background can help the uninitiated. A good troubleshooting guide is very important.

Support: Software without proper support may not be trusted. The availability of training courses will help the user to learn the package faster. Maintenance and update of the software with the possibility of converting older version files to the new version are important. Free or inexpensive technical support and consultancy by the vendor and availability of a toll-free telephone would be desirable. Having a homepage, INTERNET discussions, user group meetings, and newsletters are good means of communication between the users and the vendor.

Pre-purchase: It would be helpful to ask the supplier to give an on-site demonstration. Most vendors provide demo disks which are useful. A one-month free trial of the software is a valuable opportunity for the user to find out the suitability of the software for his/her needs.

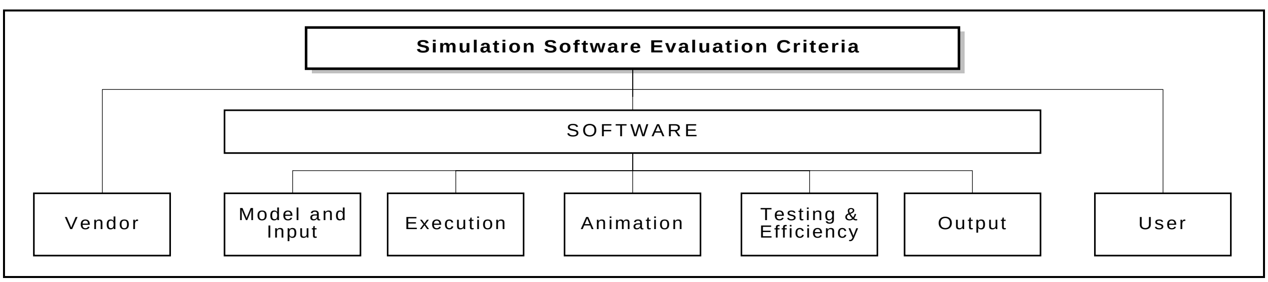

Figure 1: Main Criteria Groups of the Hierarchy

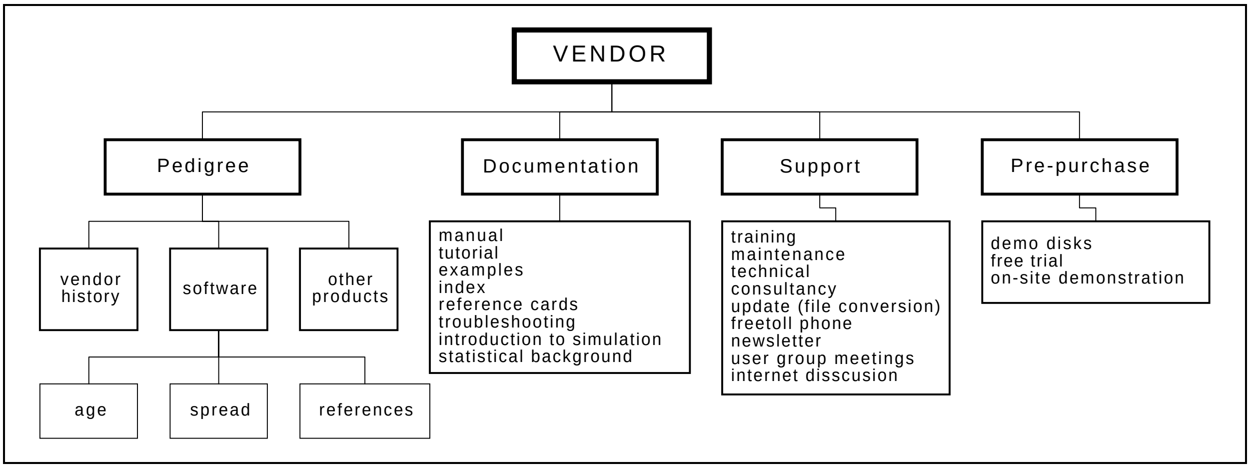

Figure 2: Criteria Related to Vendor Group

3.2 Model and Input

This category of criteria includes issues related to a model, its development and data input (Figure 3).

Model building: In this sub-criteria group we consider facilities which help the user in model development. Means such as mouse, keyboard and scanner may be used as tools for this job. A model could be made graphically or by entering codes. A user-friendly package speeds up the process of model development by providing necessary options from the menu panel. The package may provide modelling assistance. Prompts and dialogue boxes advise on the action that should be taken next. Modularity allows the user to develop the model in separate modules step by step. Each module can be tested and debugged separately and then linked together. The merging of models when a previously made model is going to be a sub-model for a larger model is useful. This option would be further enhanced if a library of reusable modules and pre-existing generic models were available. It should be considered whether any formal logic such as activity cycle diagrams, flow diagrams or network diagrams is needed for model development. The use of formal logic may help in understanding the problem better, but it may be time consuming. Some packages provide a hierarchical model building option. This option makes it possible to have access to more detailed sections of a model at a lower level by selecting a certain element in a higher level.

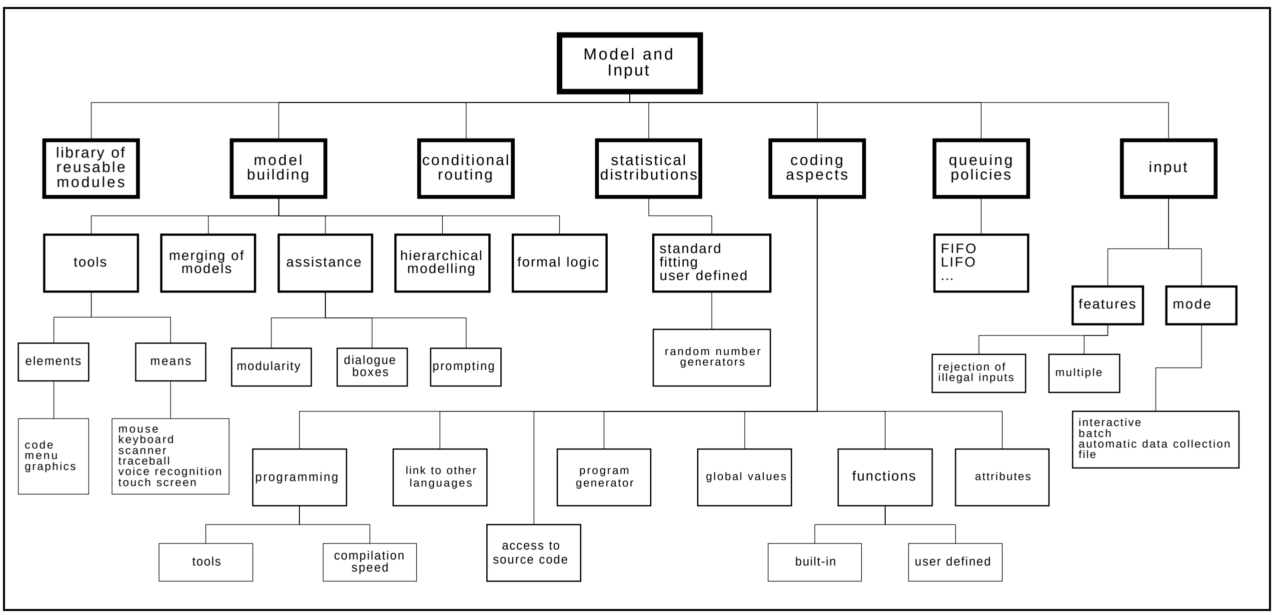

Figure 3: Criteria Related to Model Development and Data Input

Input: Data input could be interactive, batch, by reading from a file, or automatically collected from a system. Rejection of illegal inputs will prevent many of the errors which may occur during the model run.

Statistical distributions: When data is going to be input to the system as a statistical distribution, the software should be able to provide some standard statistical distributions such as normal, exponential, gamma, and rectangular distributions. An option to let the user define a different distribution is necessary. The ability to fit the data into a distribution is a good feature for software. A variety of different random number streams is necessary for replications of experiments. Users may be allowed to define their own random generators. The software may support different queuing policies such as FIFO, LIFO, By attribute, Minimal value, and Maximal value.

Coding aspects: This sub-criteria group provides flexibility to the package. If the package allows the user to enter the code, the tools provided for this purpose by the package and the compilation speed are two important issues. It may be possible to link the package to other languages such as FORTRAN. Some packages include a program generator. A program generator provides program code for the simulation model, which could be modified. Access to the source code of the simulation software is useful when integration requires programming. A library of in-built functions and the possibility of defining functions by user further enhances this sub-criterion. Attributes and global variables are often used in programming.

3.3 Execution

This criteria group includes issues related to experimentation (Figure 4).

A package with multiple runs feature provides facilities for automatically running the model several times and changing the random number generator seed each time. A summary output of the multiple runs could be written in a file. The automatic batch run feature is similar to multiple runs, with further improvements to set the software to change the values of some variables before each run automatically.

The warm-up period feature is for reaching the steady state of the system and then collecting statistics. The reset capability allows the user to reset the statistics of the model at some time during the execution of the model.

Start in non-empty state feature enables us to specify initial values for variables and attributes and determine the situation of the entities, queues, and activities.

Control of the speed of the model run is a desirable feature, which could reduce the model execution time. The ability to make an executable module of the model could be very helpful to some users. It would make it possible to run the model independently of the package.

3.4 Animation

Criteria for evaluation of animation deal with creation, running and quality of animation (Figure 5). Animation may come as an integral part of the package or it is added to the package.

Icons: Some packages provide a library of standard icons. The number and quality of these icons are important. Some packages have an icon editor. The possibility of creating new icons or importing them from other software packages such as CAD, bitmap, or a media control interface is another issue. It would be desirable to save the created icons in a library or add them to the library of standard icons. Icons could be 3 dimensional and coloured. It may be possible to change the colour of the icons or resize them.

Screen layout: This criterion deals with the issues related to the graphical presentation of the model appearance on the screen. The package may provide an editor for creation of the screen layout. It may be possible to use other software packages for this purpose. The screen layout could be multiple and the user could switch between screens. A virtual screen is a useful feature when the model display exceeds the size of the screen. It would be desirable to be able to print the screen layout.

Development: The same issues discussed in model development are applicable here, as well.

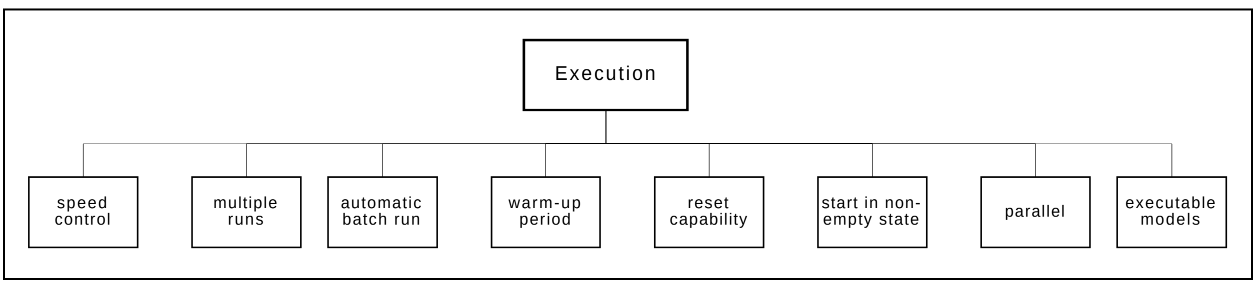

Figure 4: Criteria Group Related to Experimentation

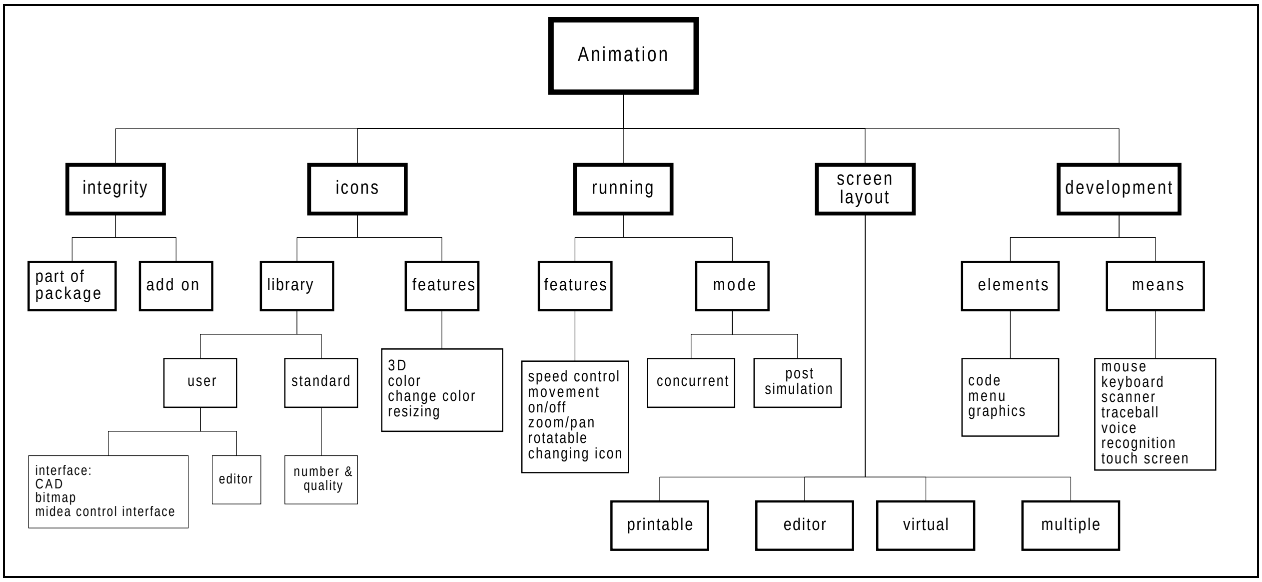

Figure 5: Criteria Group for Evaluation of Animation

Running: Animation could run with the model concurrently. This could lower the speed of the model run. On the other hand, there is a possibility of running the model first without animation, and then running the animation only. The features which could evaluate the animation better are control and change of speed of the animation run, the possibility of turning animation on and off, the possibility of zooming and panning, rotating icons and changing them during a run, and how smooth the movement of the icons is.

3.5 Testing and Efficiency

This category can be used to evaluate testability, debugging power and efficiency of a package (Figure 6).

Validation and verification: Many elements could be provided for this purpose. On-line help, on-line error messages, and an on-line tutorial save time. Logical error checks and error handling of a package are very important. The multitasking feature enables the performance of more than one operation at the same time. For example, editing a model while another model is running. Interaction feature allows interruption of the model run, change of the model, and continuing the execution of the model in order to see the effect of changes. A provisional exit to operating system within the package to do some operation is another feature. The step function lets the user run the model event by event and observe the changes in each state. The breakpoints can be used to determine some points of the time for the model to stop or start some other actions such as turn animation on or off.

Not many packages provide a backward clock facility. Running the model backward would help debug the errors which occurred during the model run, and which the program did not detect or could not stop at that time.

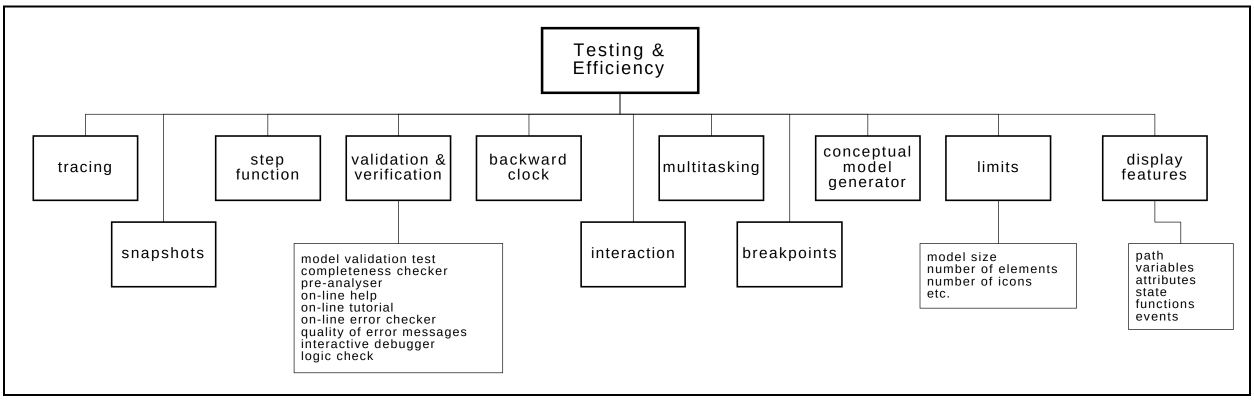

Figure 6: Issues Categorised under Testing and Efficiency Criteria Group

Conceptual model generator: A package could have the capability to produce a graphical representation of the model's logic such as an activity cycle diagram, a petri net, etc. This can help in the verification of the model.

Limitations: There are certain elements with a limit on them which are noticeable to the user. These are the size of the model, number of elements, number of icons displayed and in the library, the possibility of definition and length of entity names, specification of time units and length measures, etc.

Display feature: Some packages display the paths and the movement of the entities in the shape of different icons alongside the paths during the run. Dynamic display of the values of variables, attributes, and functions, and the state of the elements and the events, helps debugging.

Tracing: Trace files contain data collected about the state of the model for each task executed during the model run. Taking a snapshot records the values of particular variables at specified points during model execution. These data can then be used to generate statistics and plots describing model execution. Collecting snapshot data can slow down the model execution. It may be possible to turn snapshot data collection off until you are specifically interested in collecting the data.

3.6 Output

This criterion covers some important issues (Figure 7).

Reports: Simulation software usually produces some standard reports such as queue lengths, waiting times, and utilisation. It is an advantage if software allows the user to produce customised reports. These reports could be in a form presentable to managers.

Delivery: Simulation software may send its output to a file, a hardcopy device such as a printer or a plotter, or other software through an interface. An output file can be periodic, meaning that the output is saved every period of time. Access to output files can be helpful for output data manipulation.

Simulation software may integrate with other packages such as spreadsheets, statistical packages, data base management systems, CAD, and word processors to import or export data. The package could have facilities for storage, retrieve and manipulation of output data, input data, and data about the model.

Graphics: The simulation results could be presented in the form of statistical graphics such as histograms, bar charts, pie charts, and line graphs. These graphics could be displayed on the screen, dynamically changing with model run progress.

Analysis: Output analysis is an important issue. The package can provide statistics such as means, variances, and confidence intervals. A goodness of fit test could be applied to find out how much the simulation results are close to the real system.

3.7 User

The user criteria group deals with some specific user needs and circumstances (Figure 8).

The client should specify whether he/she wants a package for discrete event or continuous simulation or maybe both types. Packages are general purpose or special purpose oriented such as manufacturing, transport, communication, etc. A client who needs a package should evaluate the specific application related features in the alternative packages. We have not revealed the application specific criteria in this research.

The software can run on a PC, mini, main, or workstation. Portability lets the user develop a model on one machine and run it on another machine with a different configuration. The package may run on DOS, UNIX, OS/2, or Windows. Compatibility for software means that it can be used on more than one operating system. The user should specify if he/she wants a network version of the software. It is noticeable that a security device may limit the package to just one machine. (Particularly inconvenient for academic use).

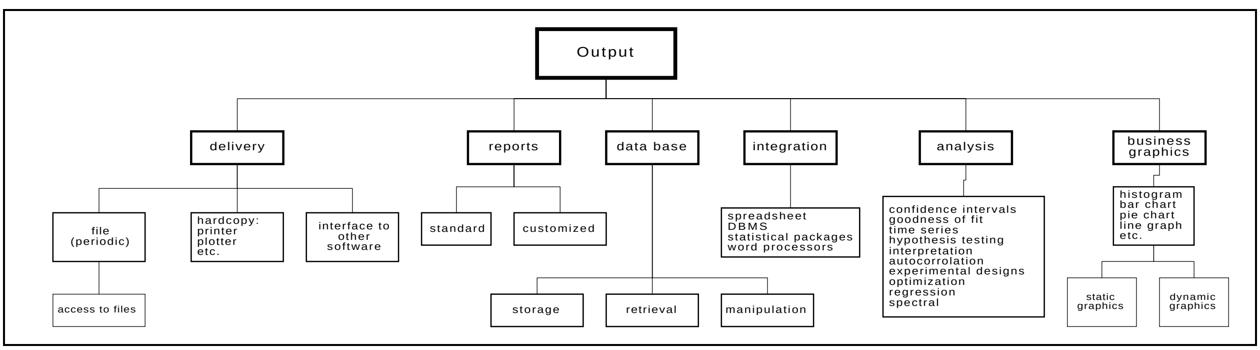

Figure 7: Criteria Group Related to Simulation Output

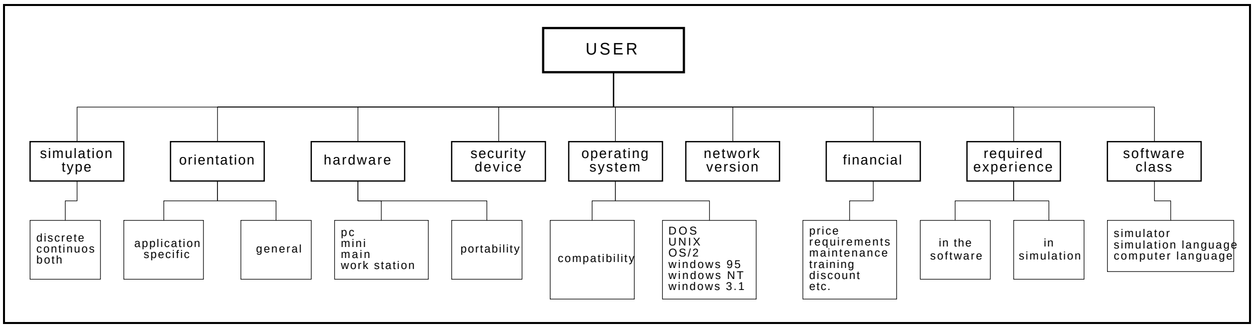

Figure 8: Criteria Categorised in User Group

Required experience: It would be helpful to find out if the package needs any previous knowledge and experience in simulation and software.

Financial: Obviously, one of the most important criteria is the cost of the software including the price, installation cost, cost of extra hardware requirements, and maintenance cost. Discounts such as educational discount and multibuy discount should be considered.

Software class: Simulation software can come in three types. It can be a general computer language such as FORTRAN, a simulation language such as GPSS, or a simulator such as WITNESS. Software class, or in other words type of package, may not be considered as an important criterion. In fact any simulation software which helps the client solve problems best would be suitable. It would not make any difference which class it belongs to.

4 THE USABILITY OF THE FRAMEWORK

The framework can be used as a tool to help the user test and evaluate features of packages. New features found in a package can be added to the hierarchy. In this way, the more packages we test, the more comprehensive the hierarchy will become (Nikoukaran and Paul, 1998b). Without such a framework it may not be possible to find out what features are not included in the package being tested. The data gained from this stage will form a checklist of the features, which does not show how good they are, but would be the basis for the evaluation of packages.

The hierarchy may provide the client with a better view of options and the ability to choose the appropriate one. The hierarchy is not designed for selection of simulation software for a particular application area. One can evaluate packages with respect to each branch of the hierarchy by comparing them, weight each branch according to the particular application area, find the overall value of each package as a single value and select the most appropriate one. In this case a change in one or two criteria will only affect the evaluation of the related branches and not the evaluation of all parts of the hierarchy. It should be mentioned that in comparing different features we consider issues such as ease of use, ease of learning, and quality depending on the type and nature of the feature.

There are different types of criteria included in the framework. Some criteria are in the form of numerical values, such as prices of packages, which are easily comparable. Some other criteria are not numeric or not quantifiable but they are comparable such as an icon editor, where we can test the editors and decide which one is preferable. For non-comparable criteria, such as the availability of a network version, whereas it can not be compared with a package without a network version, the client should consider only the packages which provide this feature. The evaluator should know what he/she is looking for. The hierarchy could tell an evaluator where to look for a particular issue and what to consider for evaluating a criterion.

The hierarchy is flexible to minor changes such as introducing a new criterion. Major changes, although their occurrences are rare, may cause a re-organisation of the hierarchy.

5 SUMMARY AND CONCLUSIONS

This paper presents a comprehensive list of criteria structured in a hierarchical framework for evaluating simulation software. Issues related to criteria for simulation software evaluation are categorised into seven main groups and several sub-groups. The hierarchy can be used for obtaining a better view of the features of simulation software and as a guide to test and analyse simulation modelling packages. With the help of a suitable evaluation technique, such as the Analytic Hierarchy Process (Davis and Williams, 1994), the hierarchy could be used to evaluate simulation software. Evaluation techniques and selection methodologies for simulation software have not been discussed in this paper. This is done in the selection stage when the user decides which criterion is more important than the others. For example, Hlupic and Paul (1995) listed criteria which have more importance for a package to be used for education. We conclude that not only can simulation software selection benefit from a comprehensive hierarchy of criteria but also areas for development of simulation software could be identified.

REFERENCES

Davis, L. and G. Williams. 1994. Evaluation and selecting simulation software using the analytic hierarchy process. *Integrated Manufacturing Systems* 5(1):23-32.

Deaver, R.A. 1987. Selecting a manufacturing simulation system, *CIM Review* 3(3):6-8.

Doukidis, G.I. and R.J. Paul. 1990. A survey of the application of artificial intelligence techniques within the OR society. *Journal of the Operational Research Society* 41(5):363-375. Reprinted in *Artificial Intelligence in Operational Research* (Macmillan, Basingstoke, 1992).

Gogg, T.J. and J.R.A. Mott. 1993. Introduction to simulation. In *Proceedings of the 1993 Winter Simulation Conference*, ed. G.W. Evans, M. Mollaghasemi, E.C. Russel, and W.E. Biles, 9-17. The Society for Computer Simulation International, San Diego.

Grant, J.W. and S.A. Weiner. 1986. Factors to consider in choosing a graphical animated simulation system. *Industrial Engineer* 31 August:37-40, 65-68.

Haider, S.W. and J. Banks. 1986. Simulation software products for analyzing manufacturing systems. *Industrial Engineer* 31 July:98-103.

Hlupic, V. and R.J. Paul. 1995. Manufacturing simulators and possible ways to improve them. *International Journal of Manufacturing System Design* 2(1):1-10.

Keeney, R.L. and H. Raiffa. 1976. *Decisions with multiple objectives: Preferences and value trade-offs.* New York: Wiley.

McDaniels, T.L. 1996. A multiattribute index for evaluating environmental impacts of electric utilities. *Journal of environmental management* 46:57-66.

Nikoukaran, J. and R.J. Paul. 1998a. Simulation software selection - a review. Submitted to *Simulation Practice and Theory.*

Nikoukaran, J. and R.J. Paul. 1998b. Simulation software selection “whys and hows”. *Yugoslav Journal of Operations Research* 8(1):93-102.

Olson, D.L. 1996. *Decision Aids for Selection Problems.* New York: Springer-Verlag.

OR/MS Today. 1991. Simulation software survey. *OR/MS Today* October:83-102.

OR/MS Today. 1993. Survey. *OR/MS Today* December:67-78.

Shannon R.E. 1992. Introduction to simulation. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 65-73, The Society For Computer Simulation International, San Diego.

Stewart T.J. (1992) A Critical Survey on the Status of Multiple Criteria Decision Making Theory and Practice, OMEGA 20 (5/6) pp.569-586.

AUTHOR BIOGRAPHIES

JALAL NIKOUKARAN holds a B.Sc. in Statistics from the London School of Economics and an M.Sc. in Operational Research from Lancaster University. He is currently on leave from the University of Yazd, Iran, undertaking research at Brunel University into simulation software selection. Jalal has more than eight years experience teaching Operational Research Techniques, consultancy in general computing, system design and programming, and has worked as a researcher with the Yazd University of Medical Science.

VLATKA HLUPIC is a lecturer in Simulation Modelling at Brunel University. She holds a B.Sc. (Econ) and an M.Sc. in Information Systems from the University of Zagreb, and a Ph.D. in Information Systems from the London School of Economics. She is researching into and has published extensively (over 100 papers), in simulation modelling software approaches to manufacturing problems. She has practical experience in the manufacturing and waste disposal industries, as well as having held a variety of teaching posts in England and Croatia. Her current research interests are in manufacturing simulation, software evaluation and selection, and in simulators and simulation languages.

RAY J. PAUL holds the Chair in Simulation Modelling in Brunel University. He previously taught Information Systems and Operational Research at the London School of Economics. He received a B.Sc. in Mathematics, and an M.Sc. and a Ph.D. in Operational Research from Hull University. He has published widely in book and paper form (two books, over 200 papers in journals, edited books and conference proceedings), mainly in the areas of the simulation modelling process and in software environments for simulation modelling. He has acted as a consultant for variety of United Kingdom Government departments, software companies, and commercial companies in the tobacco and oil industries.