A Computer-assisted Performance Analysis and Optimization (CPAO) of Manufacturing Systems based on ARENA® Software

Said Taktak[1], Wafik Hachicha[2] and Faouzi Masmoudi

[1]*Higher Institute of Industrial Management, B.P. 1164, Sfax 3018, Tunisia*

[2]*Mechanics, Modeling and Manufacturing Research Unit (U2MP), ENIS, Tunisia*
Higher Institute of Industrial Management, B.P. 1164, Sfax 3018, Tunisia

[3]*Mechanics, Modeling and Manufacturing Research Unit (U2MP), Engineering School of Sfax, University of Sfax, B.P. 1173, Sfax 3038, Tunisia*

[2]*wafik_hachicha@yahoo.fr*

Abstract

Simulation project requires highly qualified multidisciplinary staff rarely available in a Small and medium-sized enterprise (SME). The aim of this paper is to develop a computer-assisted performance analysis and optimization (CPAO) to help a SME manager which is considered in this paper as an inexperienced user in applying a simulation project without using explicitly the ARENA® software. After the design of the suitable simulation model with ARENA® software by an expert simulation modeler, the inexperienced user of CPAO can operate the process of simulation and optimization easily and simply. Major manipulations include the following. (1) The setting of possible configurations. (2) The statistical analysis and graphical analysis of simulation results. (3) The improvement and the optimization of some criteria. The developed CPAO application is carried out in two steps. Firstly, the Unified Modeling Language (UML) is employed for the CPAO design phase. Secondly, Visual Basic Administration (VBA) language is exploited to develop various User Forms dialogues with the inexperienced user, ARENA software, Ms Excel and Ms Access. Finally, for the simulation optimization technique, the simulated annealing (SA) algorithm is adopted. To demonstrate and validate CPAO results, an illustrative example which is considered a manufacturing lines system with buffer stocks design problem is fully detailed.

Keywords: *Computer application, discrete event simulation, ARENA®, UML, VBA, simulated annealing, manufacturing lines, stock capacity design*

1. Introduction

Many real world problems in management and optimization are very complex and mathematically intractable so that simulation is the appropriate tool for system analysis and performance evaluation. Computer simulation requires developing a program that mimics the behavior of a system as it evolves over time and records the overall system performance. As the technology of computer hardware and software advances, simulation has emerged as an essential tool in academic research. Simulation has been applied to various sectors, such as manufacturing and business [1, 2], services and supply chain management [3], etc. In fact, the simulation method has the advantage of being applicable whatever the complexity of systems. Despite, the simulation is rarely used in many companies of the underdevelopment countries.

The literature on manufacturing systems simulation consulted reinforces our conviction that simulation is a technique that still has a lot of underexploited potentialities mainly in real world applications.

Although, the improvements in simulation software which have reduced model development time, simulation project still requires a highly qualified multidisciplinary staff (modeling and computer programming, advanced statistical analysis, expertise in manufacturing systems and management, project management etc.) sometimes not available in a Small and medium-sized enterprise (SME). In fact, most complex simulation projects need to be considered essentially as a task of software engineering. Moreover, the simulation is a tool for decision support and does not give a solution but it allows the comparison and then a choice of possible configurations. Problems arise mainly in the planning of simulation experiments and statistical analysis of the different simulated results.

Thus, the idea of this research is related to the potential improvement of computer simulation use through the development of computer-assisted applications. The main objective of this paper is (1) to assist an inexperienced user in applying the process of discrete event simulation, and (2) to simplify the setting of possible configurations, the statistical analysis and graphical analysis of simulation results, systems improvement and optimization of some criteria thought User Forms with the user and ARENA® software. To achieve these goals, object-oriented (OO) design is applied through UML (Unified Modeling Language) and VBA (Visual Basic for Application) which is integrates with the SIMAN simulation language is used.

The remainder of the article is organized as follows. Section 2 illustrates the relayed literature review. Following this, Section 3 describes the conceptual design of CPAO. Section 4 presents the developed CPAO implementation. Section 5 contains principals User Forms thought Illustrative example. Finally, Section 6 concludes with brief remarks.

2. Related Literature Overview

Simulation can help users by contributing in design, in management and in the decision-making of production systems. It is able to model all kinds of company processes: physical, informational and decisional. Simulation models can be built at all hierarchical (operational, tactical, and strategic) and detailing levels (machine, cell, shop, *etc.*). Moreover, the whole manufacturing system life cycle can be modeled and simulated (design, analysis, implementation, operation). Today however, not all the possibilities of computer simulation are currently used. Computer simulation is limited in its uses and its applications [4]. According to our literature review, there is no research works that are similar to the proposed approach in this paper. All reviewed articles proposed computer application in order to build manufacturing simulation model.

Habchi and Berchet [4] propose an approach for modeling and simulation of manufacturing systems. This approach, which models and simulates both operation and control subsystems, is based on the interaction of three main kinds of concepts: the product processing system, the moving entity, and the systemic decomposition and analysis of manufacturing systems. An algebraic specification of these concepts and their modeling are proposed using UML. Anglani, *et al.* [5] present a procedure to develop flexible manufacturing system simulation models, based on the UML analysis/design tools and on the ARENA® simulation language. They proposed a definition of a systematic conceptual procedure to design flexible manufacturing system simulation models and of a set of rules for the conceptual model translation in a simulation language. Under the same manufacturing system kind, Madan *et al.* [6] present a framework for the determination of an efficient level

of simulation model fidelity, which will achieve acceptable output accuracy with minimum resources and thereby reduce model building effort and computation time. Firstly, they formally define different levels of model fidelity using building blocks available in OO modeling, where an operation at a higher level is either decomposed into more detailed operations or subjected to more constraints at a lower level. Secondly, five models with different fidelities are defined. Then, simulation models that conform to these OO models are constructed. Finally, Boesel *et al.* [7] propose a framework for simulation-optimization software for JGC Corporation of Japan which statistical error control within a heuristic search procedure is implemented.

3. The Conceptual Design of CPAO

As shown in Fig.1, the developed computer application ensures a dialogue between an inexperienced user, simulation software ARENA®, Ms Access and Ms Excel. ARENA® software which is developed by System Modeling Corporation [8-9], is a graphical transaction-oriented language for the discrete-event simulation; it is one of the most commonly used simulation languages at this moment [4, 5, 9, 10, 11]. In the ARENA® software, the language functionality is incorporated in the building blocks, called modules, with which simulation models can be implemented. Basing on type and level of functionality, these modules are grouped together in panels called templates.

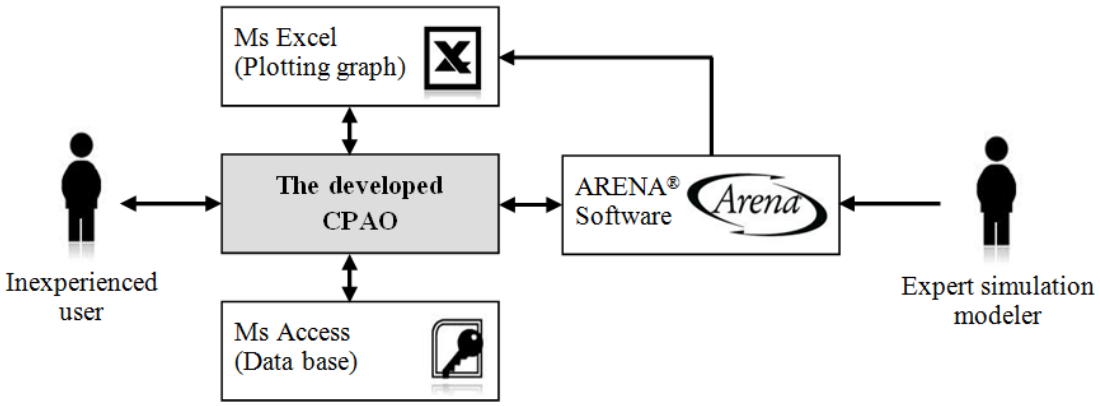

Figure 1. The Developed Computer Application Data Exchanges

In the developed CPAO, there are three intervening: (1) Expert simulation modeler who builds simulation model, (2) Inexperienced user (SME manager) who prepares specifications and uses the CPAO tool, and (3) CPAO software developer who implements CPAO as described in this paper.

The choice of the UML to design the developed computer application is mainly due to the complex nature of the problem. UML is a most useful method of visualization and documenting software systems design. UML uses object oriented design concepts and it is independent of specific programming language. In recent years, UML has emerged from earlier methods for analysis of oriented objects systems, and in 1997, become recognized and accepted as a potential notation standard by the Object Management Group (OMG) for modeling multiple perspectives of information systems. UML is a simple, expressive, extensible, and visual modeling language. UML is based on the object-oriented paradigm, and enables the extraction of architecturally significant elements of a model with respect to different viewpoints, independent of the system's scale. UML defines a set of basic diagrams

that provide multiple perspectives (structural/static and behavioral/dynamic) of the system under analysis or development.

UML diagrams include Use Case diagrams, Class diagrams, Sequence diagrams, Collaboration diagrams, State Transition diagrams, etc. The utilization and understanding of this notation can be very complex and in this paper only a subset of these diagrams will be addressed. More information about UML can be found in [12]. Three main elements are used in the UML for the representation of the different aspects of an information system development. These are Uses Cases diagrams, collaboration diagrams, and Classes diagrams.

3.1. Uses case Diagram

Use Cases represent the high-level functionality of the system in development, describing "what the system should do". Although the Use Cases try to capture what users want from the system, they do not specify how the system should do it; this is left to the system design phases. Individual Uses Cases do not contain much information, hence Use Case scenarios must complement its description, and these ideally define what the Use Case should achieve throughout its functionality. The Use Case scenarios are composed of the flow of events, such as details about user actions, software actions and reactions, constraints, needs for graphical interface, relationships with other Use Cases, etc. The Use Case description provides helpful information when specifying the properties (attributes and methods) of the classes needed to perform such a Use Case.

Three elements are identified in a Use Case, namely: (1) Actor, which represents an external stimulus to the software system; (2) Action, which represents a capability requested of the system; and (3) Subject, which represents the item acted upon by an Action requested of the system. When a UML Use Case has been identified, a UML Use Case diagram can be modeled representing the relationships between Actors and their Actions. Fig. 2 shows the principal used Use Case diagrams which include the following uses case: (1) consult simulation results, (2) compare different configurations, and (3) optimize configuration.

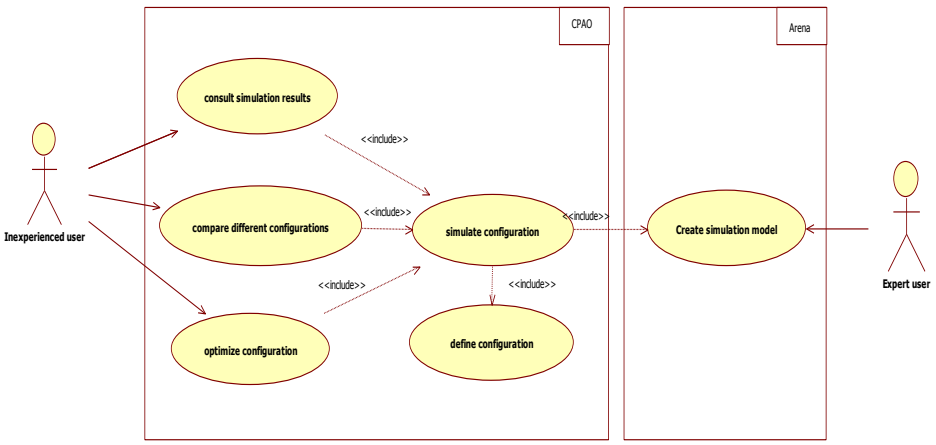

Figure 2. Use Case Representation of the CPAO

Even after agreement is achieved on the functionality of the system, this functionality may be refined since a better understanding of it is gained by the initial identification of the main structure of the system. To achieve this goal, a Sequence diagram was proposed for each Uses Case.

3.2. Sequence diagram

Sequence diagrams provide a dynamic high level representation of the system. They capture and represent interactions required between objects, through their methods, emphasizing the time ordering of messages. These diagrams represent mainly the behavioral aspects of objects, showing what methods (functions) are required to satisfy a specific Use Case

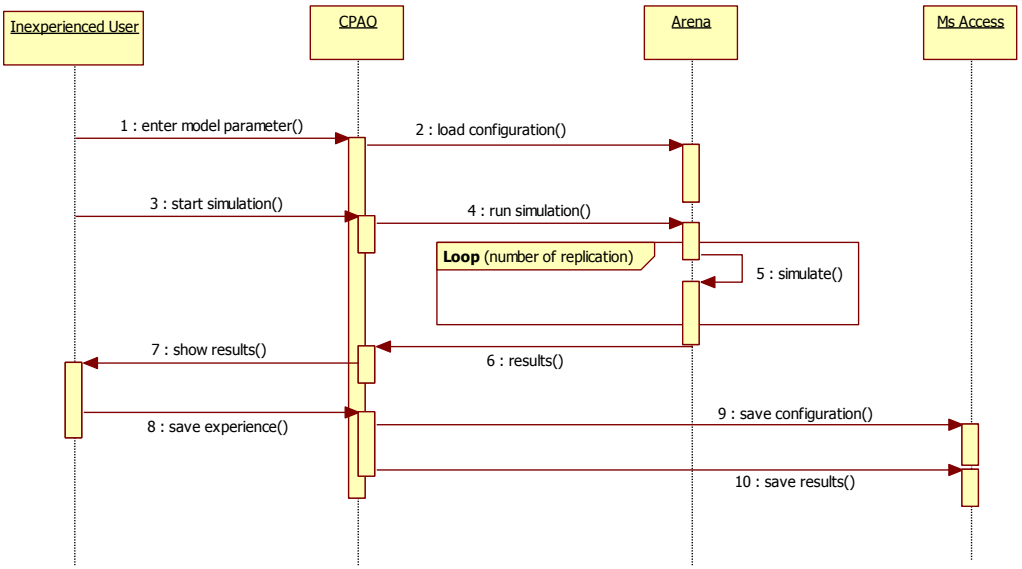

Figure 3. UML Sequence Diagram Associated to "Consult simulation results"

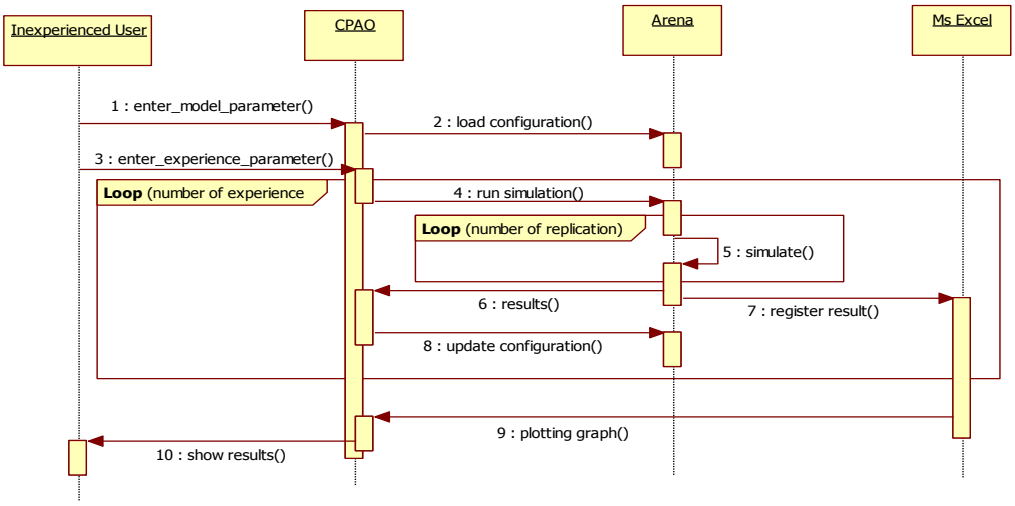

Figure 4. UML Sequence Diagram associated to "Compare different configurations"

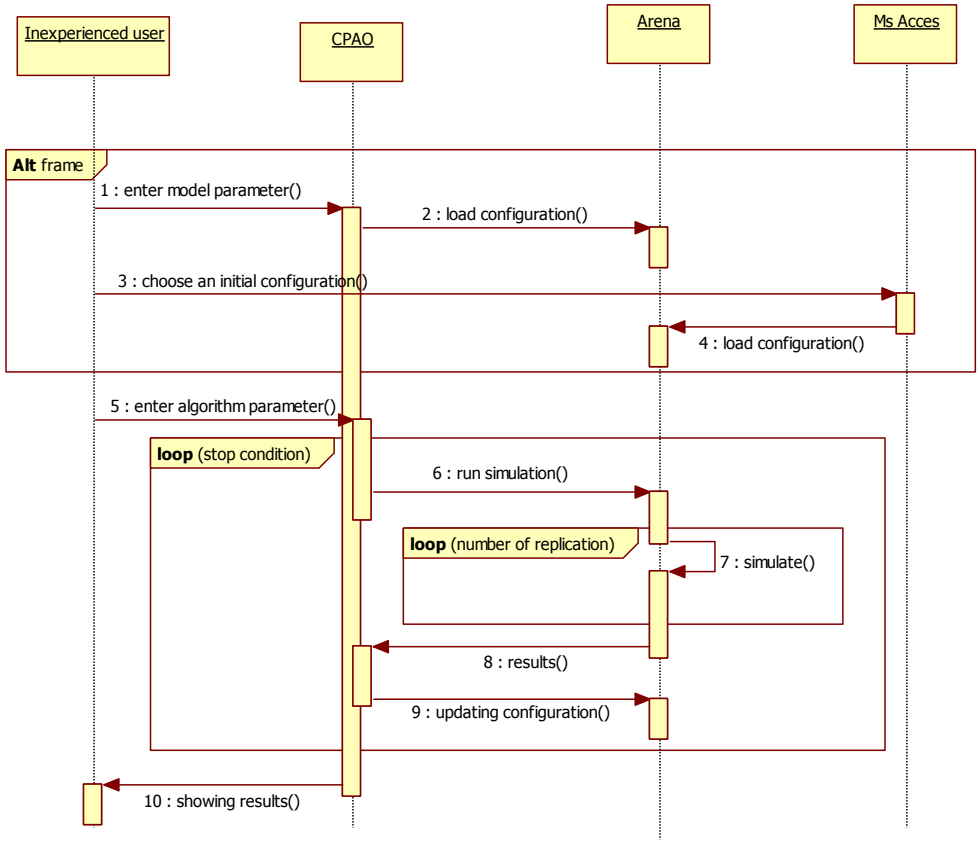

Figure 5. UML Sequence Diagram associated to "Optimize configuration"

3.3. Class Diagram

The fundamental concept of OO languages is the *object.* An object consists of a set of *attributes*, or variables, which describe its internal state, and a set of *operations* that the object can perform, such as to update or query the state, or perform calculations. An object is an *instance* of a *class*, which can be thought of as a set of objects with similar properties, i.e., the same attribute and operation names, and the information about the properties is normally provided at the class level. At a given time, there may be several instances of each class, and the instances may have different attribute values. A class is shown as a box with the name inside it. Alternatively, the box may be divided into three compartments, containing the class name, the attributes, and the operations, respectively. Objects are also shown as boxes; with the difference that the object name is underlined to show that it is an instance, rather than a class.

Fig. 6 presents the UML class diagram for the Implementation of CPAO. It contains six classes which are configuration, module, parameters module, results, process results and system results.

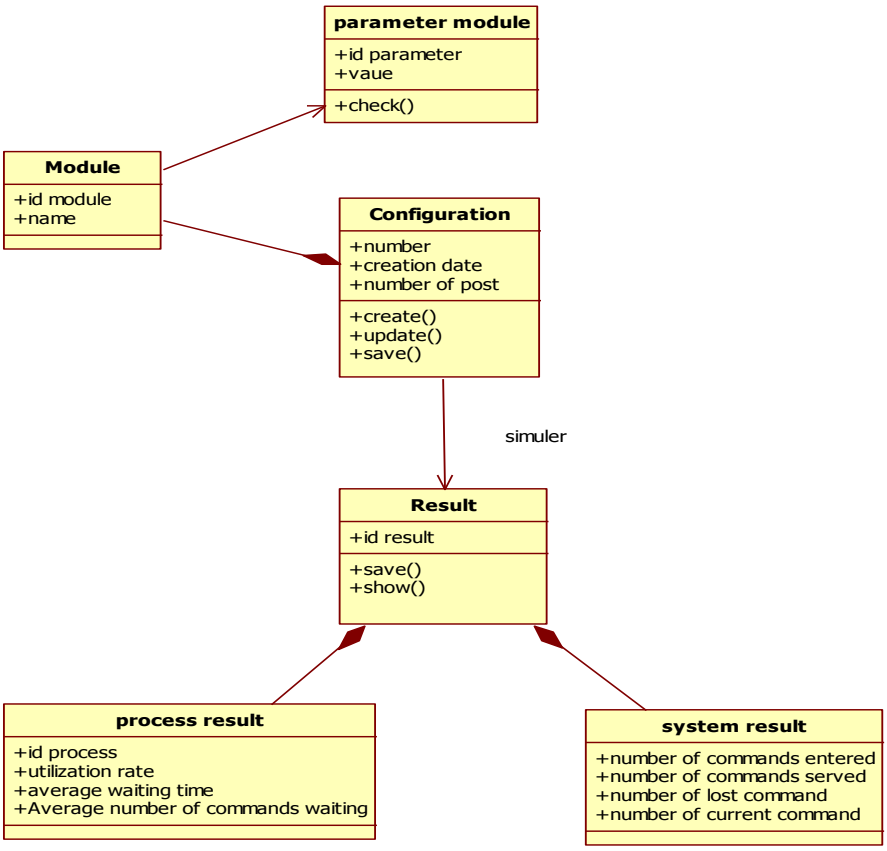

Figure 6. UML Class Diagram for the Implementation of CPAO

4. The Developed Application Implementation

4.1. VBA Language

Visual Basic for Applications (VBA) is an event-driven programming language which was first introduced by Microsoft in 1993 to give Excel 5.0 a more robust OO language for writing macros and automating the use of Excel. Actually, each Office application supports VBA [11, 13]. The user interface is identical in all theses application that supports VBA. Pressing the Alt-F11 key on any of these application loads the editor illustrated in Fig. 7. The structure of the VBA language is beyond the scope of this presentation but documented in numerous texts, such as in [13].

The new VBA user has much to learn but help is always as close as the F1 and F2 keys. Pressing the F2 key in the Microsoft Visual Basic editor provides lists of available constants, functions, and properties. Of these three constructs only the value of a property can be changed and then only if it has not been declared as read only. Actually, VBA is quite easy to learn because the built in editor automatically checks the syntax of each code line as it is entered. The Debug Compile feature checks that all variables have been defined prior to attempting execution. Finally even during model execution, many VBA errors can be interactively debugged and corrected without restarting the execution of the Arena model.

This interactive debugger also includes a breakpoint feature and the ability to view the current variable values as the code is executed.

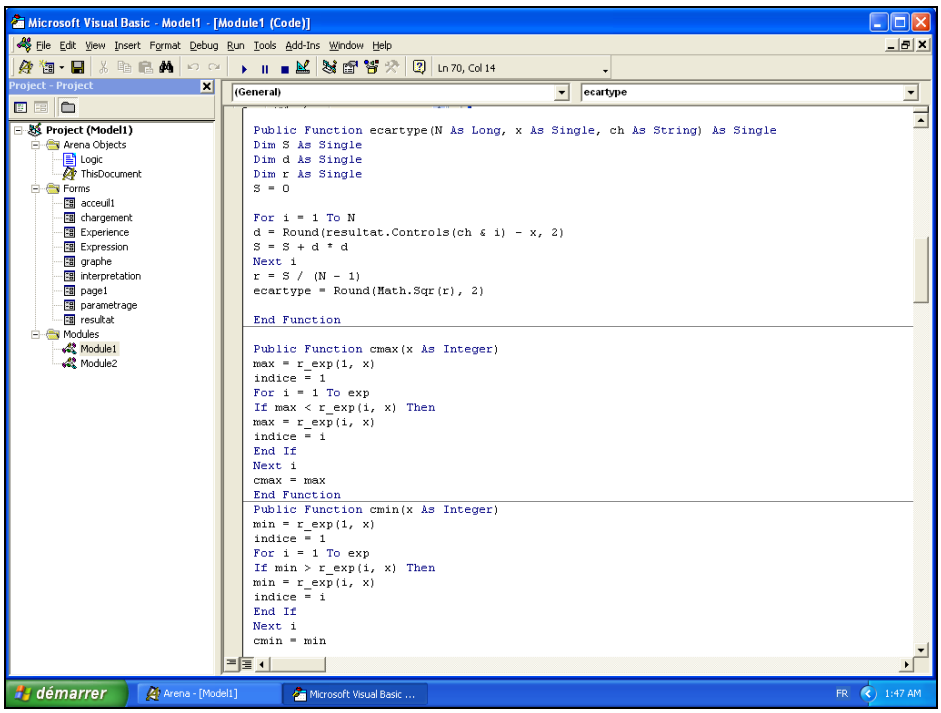

Figure 7. Example of the VBA Editor Window

4.2. Simulated Annealing as the Optimization Algorithm

As mentioned above, simulation models provide insight on the behavior of real systems. This insight can be used to improve system performance by ad hoc changes to the system design parameter values, or the simulation model may be analyzed repeatedly to find a set of design parameters that provide the best simulated performance. "Simulation optimization" is defined as a repeated analysis of the simulation model with different values of design parameters, in an attempt to identify best simulated system performance. The design parameters of the real system are set to the "optimal" parameter values determined by the simulation optimization exercise, rather than in an ad hoc manner based on qualitative insights gained from exercising the simulation model.

Several classes of simulation optimization problems and solution methodologies have been proposed and analyzed in the literature. Many comprehensive literature reviews, that discuss foundations, theoretical developments and applications of simulation optimization techniques, have been written on this topic such as in [14-15]. In brief, as indicated by [15], simulation optimization Methods can be classified under two main headings: local optimization (such as Statistical Selection Methods, Metamodel Methods, Stochastic Gradient Estimation, etc.) and global optimization (such as evolutionary algorithms, simulated annealing, tabu search, Bayesian/sampling algorithms, etc.). In this paper, the simulated annealing algorithm was adopted.

Kirkpatrick et al. [16] initially presented the simulated annealing algorithm, which attempts to solve hard combinatorial optimization problems through controlled randomization. Since then the algorithm has been applied to many optimization problems in a

wide variety of areas. Moreover, simulated annealing algorithm has been adopted in simulation optimization methods as a global optimization.

The pseudo-code for the general procedure for implementing the simulated annealing algorithm is presented in Table 1. The algorithm evolves from an initial solution S_0 for the problem. In the inner cycle of the algorithm, repeated while $n \leq L$, a neighboring solution S_n of the current solution S is generated. If S_n is better than S ($\Delta \leq 0$) then the generated solution replaces the current solution, otherwise the solution is accepted with a certain probability ($p \leq e^{-\Delta/T}$). Clearly, the probability of acceptance is high if the performance difference is small and T is large. The key to simulated annealing is to specify a cooling schedule, by which the temperature is reduced so that initially inferior solutions are selected with a high enough probability so local optimal are escaped, but eventually it becomes small enough so that the algorithm converges. In others words, the value of the temperature T decreases in each iteration of the outer cycle of the algorithm, which diminishes the probability of accepting as current solution worst solutions. Obviously, during the algorithm the best solution found (S^*) is always kept and the generation of neighboring solutions obliges that two consecutive solutions must be different ($S_n \neq S_{n-1}$). The most important characteristic of this algorithm is the possibility of accepting worst solutions, which can allow it to escape from local minima.

Nonetheless, the performance of the algorithm depends on the definition of several control parameters as follow. (1) The initial temperature (T_0) should be high enough that in the first iteration of the algorithm the probability of accepting worst solutions is, at least, of 80% [16]. (2) The most commonly used temperature reducing function is geometric: $T_i = a_i T_{i-1}$ ($a_i < 1$, and is constant). Typically, $0{,}7 \leq a_i \leq 0{,}95$. (3) The length of each temperature level (L) determines the number of solutions generated at a certain temperature (T). (4) The stopping criterion define when the system has attained a desired energy level

Table 1. Simulated Annealing Algorithm in Pseudo-code

Select an initial temperature $T_0 > 0$;
Select an initial solution, S_0, and make it the current solution, S, and the current best solution, S^*;
repeat
    set repetition counter n =1;
    repeat
    generates solution S_n in the neighborhood of S;
    calculate $\Delta = f(S_n) - f(S)$;
    if $\Delta \leq 0$ then $S = S_n$;
    else $S = S_n$ with the probability of $p = e^{-\Delta/T}$;
    if $f(S_n) \prec f(S^*)$ then $S^* = S_n$;
    $n = n + 1$;
    until $n >$ number of repetitions allowed at each temperature level L;
    reduce the temperature T;
until stop criterion is true.

Some of the most common criteria are based on (1) the total number of solutions generated; (2) the temperature at which the desired energy level is attained (freezing temperature); and (3) the acceptance ratio (ratio between the number of solutions accepted and the number of solutions generated). Naturally each of these control parameters must be

refined according to the specific problem on hand. Two other important issues that need to be defined when adapting this general algorithm to a specific problem are the procedures to generate both the initial solution and the neighboring solutions.

5. The Developed User Forms thought Illustrative Example

The case study adopted in this section consists of a production line that includes five workstations in line. The processing time of each workstation is shown in the Table 2 and the order between arrivals time at the manufacturing system follow the Expo (6) min.

Table 2. Workstation's Processing Time of the Case Study

Workstation	Processing time (minute)
Workstation 1	Expo(7)
Workstation 2	Expo(5)
Workstation 3	Expo(8)
Workstation 4	Expo(7)
Workstation 5	Expo(5)

In the CPAO software, the user can select the number of workstations and the type of problem to be studied by simulation. In Fig. 8, the number of workstations is taken equal to 5 and the type of problem is determining the best storage capacity system.

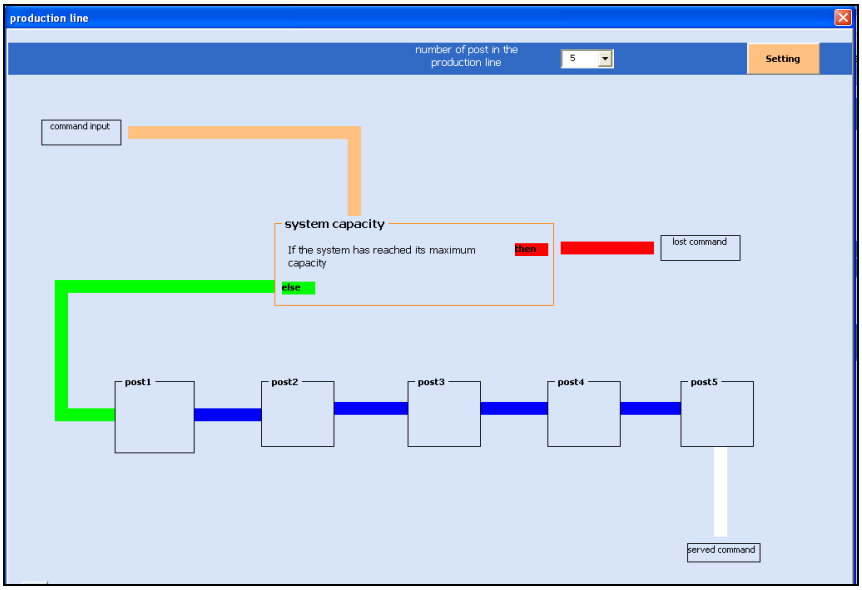

Figure 8. User Form of the Manufacturing System Architecture

After that, the user just clicks on the button to the right and top to move to the setting user form as shown in Fig. 9. In this user form, the user gives the value for each parameter of the simulation model such as processing times, unit costs, etc. In addition, simulation setup is introduced at this step: The number of replications (called also number of experiences) is fixed at 10. Each replication length is 1000 minutes.

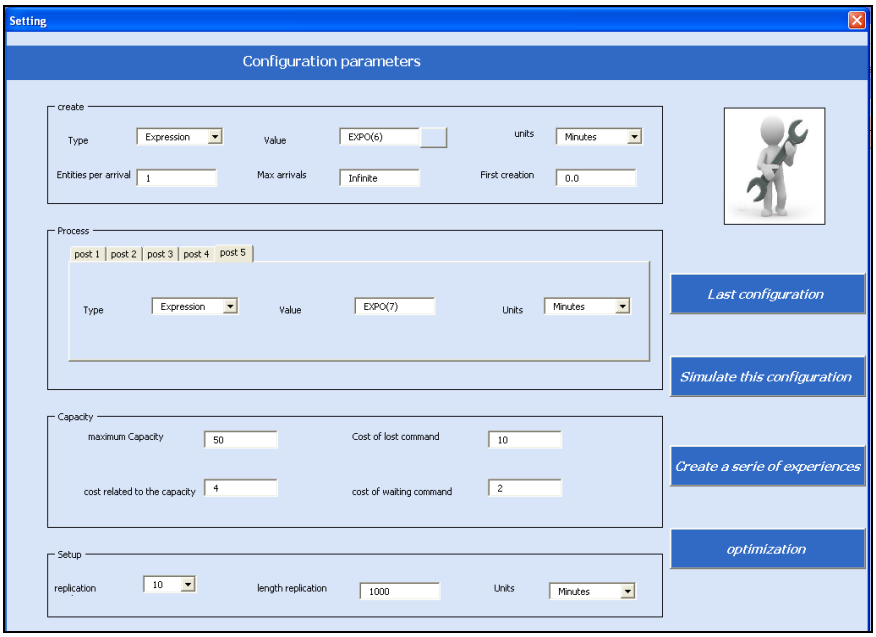

Figure 9. User Form of the Configuration Parameters and Simulation Setup

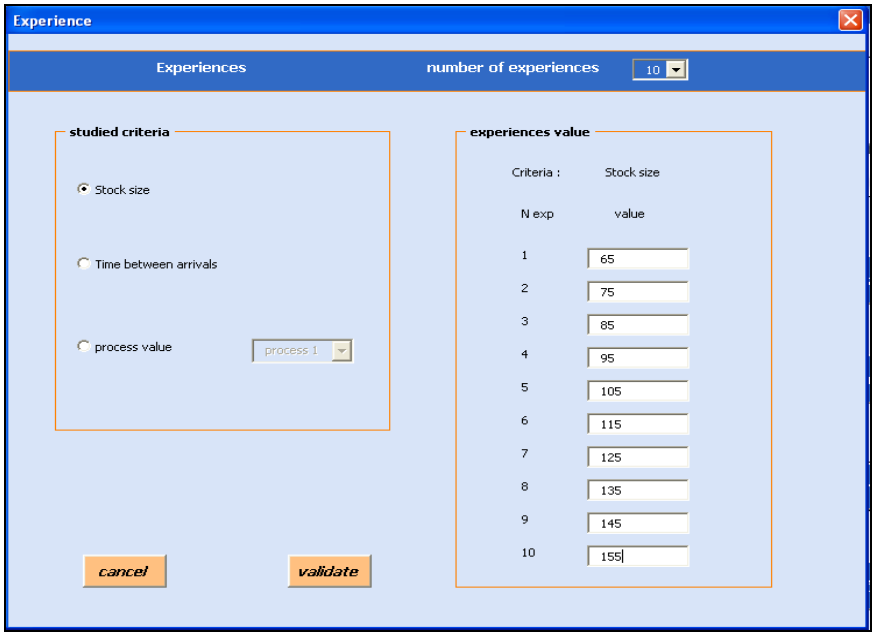

Figure 10. User Form of the Operational Parameters

Then, the user can create a series of experiences as presented in the Fig. 10. There are three studied criteria: (1) Stock size design, (2) Time betweens order arrivals, and (3) Workstation parameters. One objective of the application of simulation is to search for a set of operational parameters so that system performance is improved. In fact, user indicates the possible operational parameters levels. In this case, stock size levels range from 65 to 155.

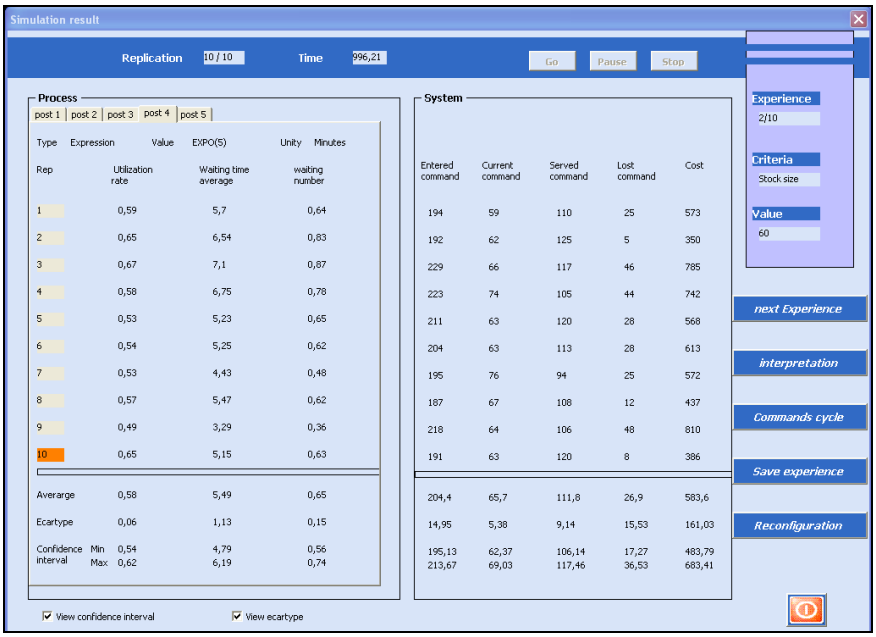

Figure 11. User Form of the Simulation Performance Measures

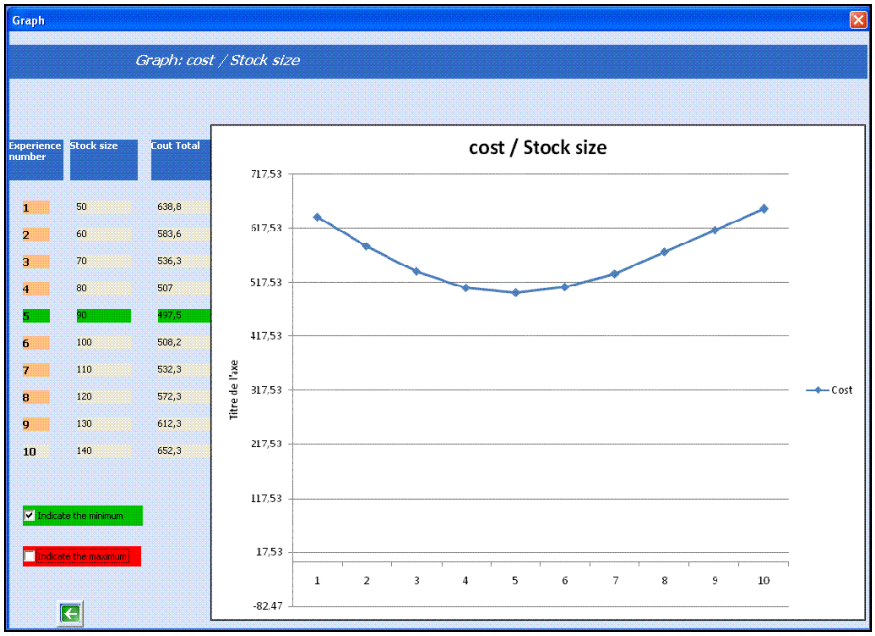

Figure 12. User Form of the Simulation Graphics

After validating the previous steps, the user views the different numerical results of performance measures related to each workstation and to the manufacturing system and this in the same window as shown in Fig. 11. Moreover, the average, standard deviation, and confidence interval for each performance measure are simply available. For more simplicity data analysis, user can consult many figures which show the evolution of each performance measure for the different levels of parameters. For instance, Figure 12 presents the graphical evolution of the total cost according to the stock size.

As mentioned in Fig. 9, the user can pass directly to the optimization user Form as presented in Fig. 13. After the entry of all algorithm parameters as explained in 4.2 subsections which are an initial solution, the initial and the minimal temperatures, the level, the reduction coefficient, and the differential neighborhood.

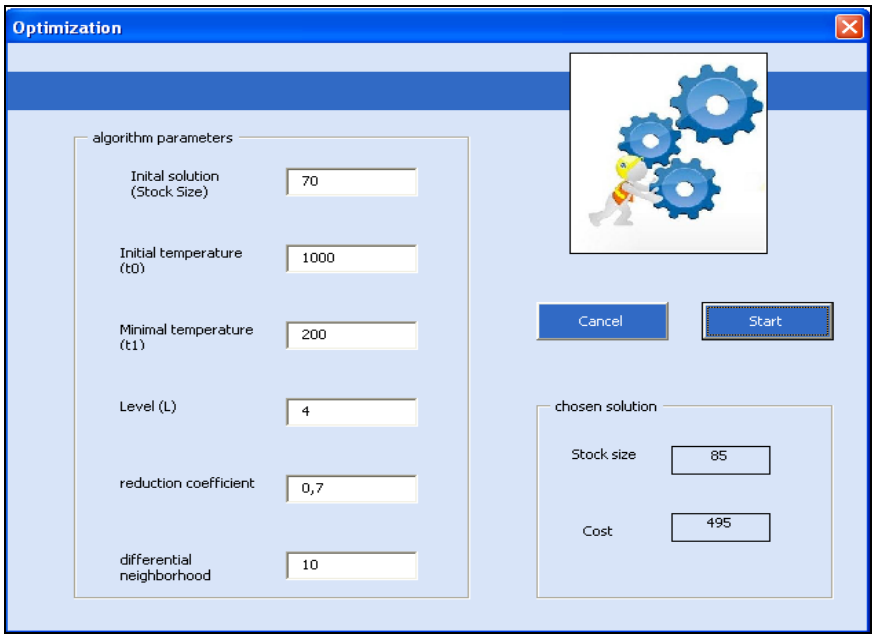

Figure 13. User Form of the Optimization Simulation Parameters and Results

The simulated annealing algorithm is implemented using the VBA language. The user should press the start button to see in the same window the optimal solution. The final solution of the problem is a stock size equal to 85 and therefore a cost is about 495 units.

9. Conclusion

To assist a medium-sized enterprise manager to conducting a simulation project, a computer-assisted performance analysis and optimization (CPAO) tool has been developed and described in this paper. The contribution of this work is moving towards the rigorous proposed methodology, and not to case study results. The methodology of design and development of CPAO application is presented through a specific case of the production lines and the buffer stock design problem. A perspective of this work is to make extensions to the functional production system and to the cellular manufacturing systems.

References

[1] M. Jahangirian, T. Eldabi, A. Naseer, L.K. Stergioulas, T. Young, Simulation in manufacturing and business: A review, European Journal of Operational Research, 203, **(2010)**, pp. 1-13.

[2] F.T.S. Chan, H.K. Chan, A comprehensive survey and future trend of simulation study on FMS scheduling, Journal of Intelligent Manufacturing, 15(1), **(2004)**, pp. 87-102.

[3] R. Bandinelli, M. Rapaccini, M. Tucci, F. Visintin, Using simulation for supply chain analysis: reviewing and proposing distributed simulation frameworks, Production Planning & Control, 17(2), **(2006)**, pp. 167-175.

[4] G. Habchi, G., C., Berchet, A model for manufacturing systems simulation with a control dimension, Simulation Modelling Practice and Theory, 11, **(2003)**, pp. 21- 44.

[5] A. Anglani, A. Grieco, M. Pacella, T. Tolio, Object-oriented modeling and simulation of flexible manufacturing systems: a rule-based procedure, Simulation Modelling Practice and Theory, 10, **(2002)**, pp. 209-234.

[6] Madan, Monish; Son, Young-Jun; Cho, Hyunbo; Kulvatunyou, Boonserm, *Determination of efficient simulation model fidelity for flexible manufacturing systems*, International Journal of Computer Integrated Manufacturing, 18, **(2005)**, pp. 236-250.

[7] J. Boesel, B.L. Nelson, N. Ishii, A framework for simulation-optimization software, IIE Transactions, 35, **(2003)**, pp. 221-229.

[8] A.M. Law, W.D. Kelton, Simulation Modelling and Analysis, second ed., McGraw-Hill, New York, **(1991)**.

[9] C.D. Pegden, R.E. Shannon, R.P. Sadowski, Introduction to Simulation Using ARENA, McGraw-Hill, New York, USA, **(1995)**.

[10] G.L. Kov_acs, S. Kop_acsi, J. Nacsa, G. Haidegger, P. Groumpos, Application of software reuse and object-oriented methodologies for modelling and control of manufacturing systems, Computers in Industry, 39,**(1999)**, pp. 177-189.

[11] M.S. Seppanen, Developing industrial strength simulation models using visual basic for applications (VBA),1, 2000, 77-82, Winter Simulation Conference (WSC'00) - Volume 1, **(2000)**.

[12] G., Booch, J., Rumbaugh, I., Jacobson, The Unified Modelling Language User Guide, Addison-Wesly, Longman, Reading, MA, **(1999)**.

[13] K. Getz, M. Gilbert, VBA developer's handbook™, **(1997)**, Sybex, Inc., San Francisco.

[14] S. Ólafsson, Chapter 21: Metaheuristics, S.G. Henderson and B.L. Nelson (Eds.), Handbook in OR & MS, vol. 13 **(2006)**, North Holland.

[15] E. Tekin, I. Sabuncuoglu, Simulation optimization: A comprehensive review on theory and applications, IIE Transactions, 36, **(2004)**, pp. 1067-1081.

[16] S. Kirkpatrick, C. Gelatt, M. Vecchi, Optimization by simulated annealing. Science 220, **(1983)**, (4598), pp. 671-680.

Authors

Mr. Saïd Taktak is a PhD student in Computer Science at Faculty of Economics and Management of Sfax (FSEGS). He obtained his Master in Information Systems and New Technologies (2011). He is a researcher at Multimedia, InfoRmation systems and Advanced Computing Laboratory (MIRACL) Sfax-Tunisia.

Dr. Wafik Hachicha is an Industrial Engineer (1999) from National Engineering School in Tunis (ENIT) Tunisia. He obtained his PhD in manufacturing management (2009). He is a researcher at Mechanics, Modeling and Manufacturing Research Unit (U2MP) in Engineering School Sfax-Tunisia. His research activities deal with the modeling, analysis, optimization, and the simulation of manufacturing system and supply chain system. He is an associate professor at the Higher Institute of Management of Sfax-Tunisia.

Prof. Faouzi Masmoudi obtained his PhD in Computer-Integrated Manufacturing (1988) from ENSAM Paris-France. He is a Professor at the National School of Engineers of Sfax in Tunisia, and a Researcher at the Mechanics Modeling and Production Research Unit (U2MP) in Engineering School Sfax-Tunisia. The activities of research are: modeling and simulation of manufacturing systems and design and layout of the cellular systems.