Integration of animation and explanation in a modeling and simulation environment

Brahim Belattar [*] **, Mahieddine Djoudi** [**]

***** Laboratoire d'Informatique et de Simulation Appliquées, Université de Batna, Route Boukhlouf , 05000, Algérie, **E-mail :** belattarb@univ-batna.dz
****** Laboratoire IRCOM-SIC , UFR SCIENCES - Bat. SP2MI , BP 30179 - 86962 Futuroscope Chasseneuil Cedex, France , **E-mail :** djoudi@sic.sp2mi.univ-poitiers.fr

Abstract : In this paper, we present a framework of a modeling and simulation environment (MOSE) based on artificial intelligence (**A.I.**) techniques and graphical interfaces. This environment is centered around a system called SYSAMSE in charge of animating discrete event simulation models and a knowledge based system (KBS) in charge of automating simulation models construction. Its main features are to help users in building simulation models and to understand and learn about them through animating and explaining their dynamic behavior during experimentation. First, we show that traditional techniques like simulation trace, graphic animation or interactive debugging are insufficient to understand dynamic behavior of simulation models. We show that explanation capability is completely neglected in existing MOSE. The way of integrating explanation and animation in our environment is discussed and implementation requirements are pointed out.

Keywords : Simulation, Modeling, Graphical interface, Knowledge based systems, Animation, Explanation.

Résumé : Dans cet article nous presentons l'architecture d'un environnement de modélisation et de simulation (EMS) basé sur une interface graphique et des tchniques d'intelligence artificielle. Cet environnement s'articule d'une part sur un système d'animation graphique dédié à l'animation graphique de modèles de simulation à événements discrets et d'autre part sur un système expert dédié à l'automatisation du processus de construction d'un modèle de simulation et à l'explication de son comportement dynamique au cours d'une simulation. Nous montrons que l'aspect explication est totalement négligé par les EMS existants. L'insuffisance des outils de trace, d'animation graphique ou de déboguage interactif est mise en évidence. Une approche d'intégration de l'explication et de l'animation graphique dans l'EMS que nous proposons est présentée et les perspectives d'implémentations sont discutées.

Mots clés : Simulation, Modelisation, Interface Graphique, Systèmes à Base de connaissances, Animation graphique, Explication.

1. Introduction

It is known that the Modeling process is one of the most difficult and crucial task in each simulation project. There has always been a desire of software developers to automate the construction of simulation models. The goal is to provide the modeler with a system that helps him in defining his problem and that automatically generates the appropriate simulation program. The major elements of such a system are a user interface and an automatic code generator. The user interface serves as an assistant to the modeler in defining the problem specification while the automatic code generator uses this specification to write the simulation

program in a target simulation language. Three approaches are commonly used to assist the modeler in defining the simulation model, or problem specification : natural language interface, interactive graphical interface, and interactive dialogue interface. Among these three types of interfaces, interactive graphical interfaces **(IGI)** are by far the most popular.

This paper attempt to give a framework of an Intelligent Modeling and Simulation Environment (**IMSE**) based on an IGI, and artificial intelligence techniques (**AI**). The IMSE is centered around a system called SYSAMSE in charge of animating discrete event simulation models and a knowledge based system (**KBS**) in charge of automating simulation models construction. Its main features are to help users in building simulation models and to understand and learn about them through animating and explaining their dynamic behavior during experimentation.

2. Integrating AI techniques and IGI

Many authors had shown the benefits of using AI techniques in simulation (see for example [OKE86], [SHA88]). We agree that such techniques can effectively lead to powerful and expressive environments if used intelligently.

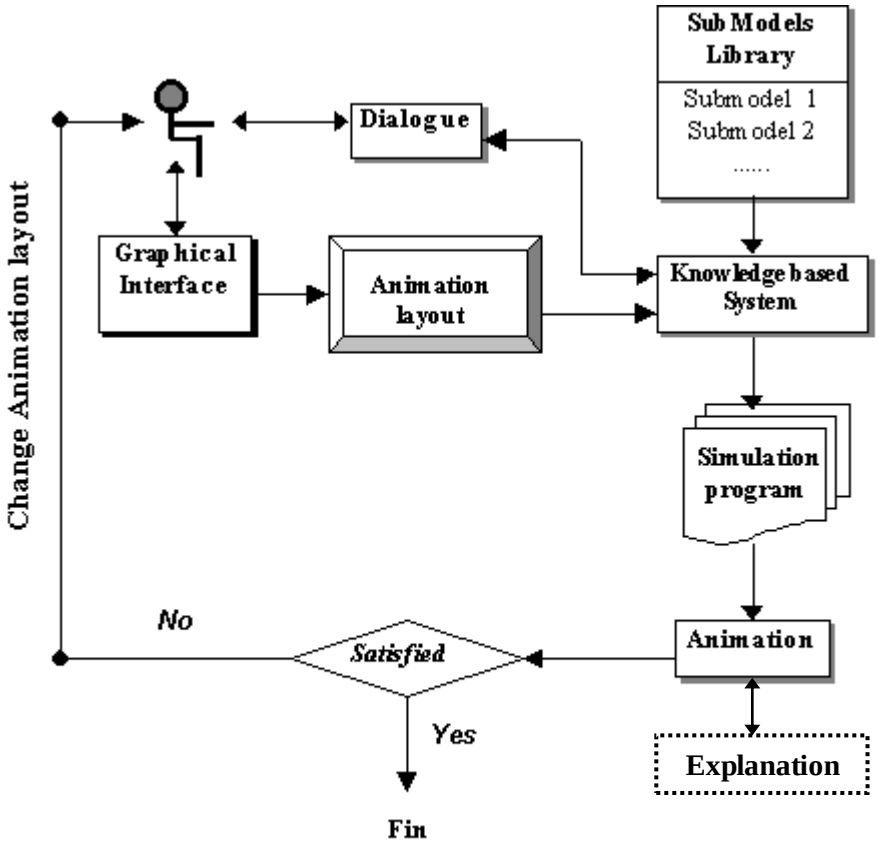

Figure 1. The IMSE Architecture

We propose a new approach which aim to combine AI techniques and IGI in an effective modeling and simulation environment. The main components of IMSE is a knowledge based system (KBS) and an IGI (see Figure 1).

The KBS have two main functions. It identifies the system components by checking the animation layout and produces an internal specification of the model. It then generates simulation program (expressed in SLAM II simulation language) from the internal specification of the model. The knowledge required by this environment can be classified in two categories, namely : (1) modeling knowledge, and (2) target simulation language knowledge. We had shown in [BEL91 b] a way to express all these categories of knowledge. The knowledge on animation layout structure can be thought as a set of production rules that are used by the inference engine to identify real system components from the internal structure of an animation layout. Taking into account the layout structure supported by SYSAMSE, components to be searched will be only made up of resources and queues. The knowledge on Modeling with SLAM II language can be expressed as a set of production rules that define how to represent a combination of resources and queues with SLAM II concepts. The target simulation language knowledge can be expressed as a set of production rules that transform the internal specification of the model into a simulation program made up of SLAM II statements (see [BEL91 b] for more details).

3. Related work

The IMSE we propose is based on an animation system called "SYSAMSE" for animating discrete event simulation models. [for SYSAMSE architecture see BEL91a]. The user interface is menu driven and provides a set of functions which mainly allow to develop and update simulation models, to build and check animation layouts, to bind between simulation models and animation layouts, and to perform animation. The initial version of SYSAMSE was tied to SLAM II simulation language [PRI86]. Briefly said, animation with SYSAMSE is built in four steps. the first step is the building of a simulation model and is supported by a separate module. The second step is the building of an animation layout which serves to show graphically on the screen the dynamic changes in the model during simulation and is also supported by a separate module. The third step is the binding between a simulation model and an animation layout. The last step is the invocation of the animation process which stands a continuous dialogue with the simulation system SLAM II in order to acquire the necessary data for updating the layout. It is important to note that an animation layout is built using icons created by the modeler itself. Because of the general purpose orientation of SYSAMSE, an icon may represent a resource, an entity , or a queue. In its initial version, SYSAMSE uses an animation layout only as a support for animating simulation results. We had proposed in [BEL91b] another approach to use an animation layout as a problem specification that serves as input to the automatic code generation process. The main advantage of such an approach is that from the modeler point of view, the animation layout becomes also the simulation model. Presently, We aim at extending this approach by integrating explanation capability in conjunction with animation and model reusability.

4. Explanation in KBS

Explanation is a topic that had been extensively used in knowledge based systems in order to explain and justify the inference strategy they adopt to reach a solution. Meanwhile, there is a lot of attention paid by researchers to the practical benefits of using such a feature in simulation environments as we do in IMSE. In KBS, the purpose of explanation is to inform users so that they have better understanding of problem and its solution and to explain why the solution given or the question asked is not what was expected by the user. Many works on this

subject exist and is essentially focused on expert systems. Explanations in early expert systems were simply based on the rules and conditions used in proofs or attempted proofs and questions permitted were also very limited. On the other hand, explanation had not yet been recognized as a necessary feature in a simulation environment and this justify a lack of work on this topic.

5. KBS and simulation systems : What is different ?

As reported by [OKE 86] and [SHA88] there exist similarities between simulation systems and expert systems. Many simulation models uses knowledge as opposed to data. Examples of such knowledge is ranking rules in a queues (FIFO,LIFO,...), resource selection rules, queues selection rules, branching (conditional or probabilistic,...). This knowledge is often embedded in simulation model and is not generally coded separately as in expert systems. Meanwhile, it is important to point out that simulation systems and expert systems do not operate similarly. An ES uses a knowledge base (KB) which contains the set of facts, rules and other necessary information. This KB is stored in the form of an AND/OR graph. An inference mechanism (also called inference engine) solve a problem by doing a tree search until a set of assertions is found that provides a solution. In the case of deep-first search method, if there is many rules at the same level of hierarchy, each having its condition evaluate to thrue, there is only one rule that will be chosen and fire. Another difference is that the time do not serve to drive the inference mechanism except in ES oriented to real time applications.

On the other hand, simulation systems are based on a clock handling the simulation time, an event calendar handling an events list sorted in ascending order of their beginning time. At each step, the clock is initialized with the beginning time of the first event in the calendar and this event is initiated. In the case of multiple events having the same beginning time, all these events will be initiated at this time. Hence, backtracking simulation events which is an operation that will be used by explanation, seems to be very difficult with simulation systems. We can also observe that the time is an important variable in simulation because it serves to control its progression.

6. Explanation in simulation

In the simulation area, explanation have not yet been adopted as an effective tool in helping end-users to understand and learn about simulation models. From our point of view, the integration of explanation ability in simulation environments is motivated by the following observations : **(1)** during experimentation with a simulation model, many decisions are made by the simulation system (SS) but do not appear with simulation results, (**2)** graphical animation of simulation models portray dynamic changes in system state but does not provide explanations about decisions made internally by SS or about state changes that have not been preplanned when building animation, (**3)** trace Execution is intended to be used by expert users to debug simulation models. It does not help in understanding system behavior because user must elaborate explanations to all questions about system being simulated only from a scratch of text, (**4) i**nteractive simulations allow answering '***What If...?***' questions but not '***Why ... ?***', '***Why not ...?***', or '***How ... ?***' questions which can effectively help in understanding the reasons that cause the system to reach a certain state. Let us give two examples in order to justify our motivation. The first example (Figure 2) is expressed in SLAM II concepts and depict a single queue of waiting entities associated to a group of identical reso(or machines). The resource selection rule used is to route an entity to the resource having the **Largest Idle Time** (LIT).

Therefore, the resource selection algorithm is based on the values of **resources idle times.** With SLAM II, such values **are never provided** with simulation results and there do not exist a subroutine or function to get them. It is only possible to know the last time at which the status of a given resource has changed. A resource idle time must then be calculated dynamically from the current simulation time and the last time at which the status of this resource has changed. Hence, we are facing a situation in which a decision (routing an entity to a resource) is made by the simulation system on the basis of an information which is not communicated with the simulation results.

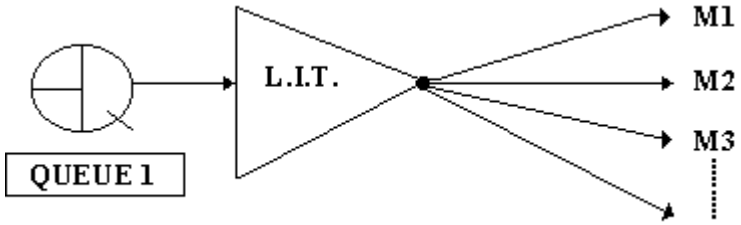

Figure 2. Resource selection example with SLAM II

The second example (Figure 3) is expressed in SIMAN concepts and depict a group of resources each having a queue associated to it. A block named PICKQ serves to dispatch an arriving entity to one resource of the group. The resource selection rule used is to route the entity to the resource having in its queue the smallest number of entities (SNQ). The queues selection algorithm is then based on the values of **queues capacities.** With SIMAN, these values are provided on demand using a special command (SHOW NQ(..)) of the interactive debugger. Hence, user have to search himself explanation to his question about the situation he faces. When using animation, these values can be portrayed on the screen if and only if they had been preplanned when building animation. In the case where all values are portrayed, user must still build explanation to his question about the situation he faces simply from watching

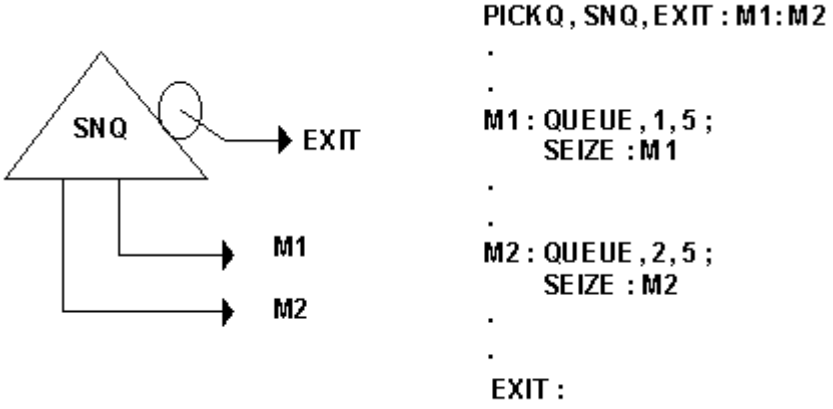

Figure 3 : Example of queue selection with SIMAN

animation. In the example of Figure 3, when an entity is routed to a queue (or machine to which is associated the queue), the user must have the possibility to issue one of the two following question styles : (**1)** Why ***entity*** is being routed to queue number **1** ?, (**2)** Why ***entity*** is being

routed to queue number **1** and not to queue number **2** ?. The first question style translates a situation that is not globally well perceived by the user. In this case, explanation will have to produce a response including queue selection rule (QSR), numbers of entities in each queue before routing takes place and a justification that the algorithm used by the SS in selecting a queue according to the QSR specified gives well the queue number **1**. A response to this style of question could look like the following :

Explain : ***Why*** *entity is being routed to queue number* **1** ?
Action attempted : **Select a queue** with respect to **SNQ** criterion
Time of selection : **10.00E+3.0**
Capacity before routing :

queue number **1**	**:**	10
queue number **3**	**:**	25
queue number **2**	**:**	30
queue number **4**	**:**	75

Explaining : the entity has been routed to queue number **1** because at time of routing ***queue number 1 contains*** *the smallest number of entities* **: 10**
Explain : **.....**

In the case of the second question style, the user has introduced in its question an element of precise comparison. One can therefore limit the explanation only to the two queues number 1 and 2 because the user has the possibility to ask a same question with respect to other queues. A response to this second style of question could look like the following :

Explain : ***Why*** *entity is being routed to queue number* **1** and **not** to *queue number* **2** ?
Action attempted : **Select a queue** with respect to **SNQ** criterion
Time of selection : **10.00E+3.0**
Capacity before routing :

queue number **1**	**:**	10
queue number **2**	**:**	30

Explaining : the *entity* ***has not*** *been routed to queue number* **2** *because at the time of routing queue* number **2** ***contains more*** *entities than queue number* **1** **: 30 > 10**
Explain : **.....**

We must point out that it would be preferable to list queues capacities in accordance with the QSR specified. For our example, capacities are listed in an ascending order because the QSR is to select the queue with the smallest number of entities. Hence, we are sure that the first queue appearing in the list is the queue that match the rule. In the case of another QSR their will be another type of sorting. For example, if the QSR is to select the queue with Largest Number in Queue, queues capacities would be presented in descending order. As one can see, explanation has to take into account the context of the question. In the first style of question, it was necessary to explain why the "***Why***" of the question is possible. In the second style of question, it will be necessary to explain that the "***Why***" of the question is impossible.

These examples shows that classical features as trace execution , graphical animation or

interactive simulation are unable to portray all about simulation models. Hence explanation is necessary. It can be used to help a wide range of users, including those who merely which to observe the simulation model, those who which to learn about it, those who which to gain confidence in its validity and those who which to obtain some basis on which to criticize it and thereby justify their rejection to it.

6. Implementation requirements

The architecture proposed in Figure 1 shows that explanation module have to operate in conjunction with animation. We think so because during animation users are more curious and tend *to discover situations which seem abnormal for them and hence are tempted to ask questions about what is happening*. Explanation must give the possibilities to users to ask questions about elementary components of the model like entities, queues, resources taken individually. It must also give the possibilities to users to ask questions about situations implying several model components taken together. When watching an animation on the screen, users do not only want to understand why a certain situation arises but also how it arises. It is then important to provide them this possibility of asking. Therefore a " ***How.....*** **?**" probe will cause the system to explain the source of the events that occurs before system reaches a given situation. It is also necessary to plan feedback from a user. This will allow him to reject an explanation if he fell not satisfied. Such situation can arise when an explanation is not well understood or when the model do not operate as expected. In the first case, explanation module must have the capacity to reinterpret the question asked and to reformulate a new explanation to the same question. In the second case, constructive suggestions or advice to review the simulation model must be given to user. Explanation must also take into account the responses given to previous questions in order to capture the essence of question being asked and permit more meaningful responses. Another important feature is to adopt a temporal reasoning since time is an important variable in simulation. Hence , temporal relations between simulation events must be kept. and managed. Taking into account simulation time will allow questions specific to time to be asked. Some examples of such questions are : (**a)** What happen **at** time **t** ?, (**b)** What **was** happen **before** time **t** ?, (**c)** What **was** happen between time **t1** and **t2, (d)** What **will** happen **at** time **t, (e)** What **will** happen **after** time **t,** etc.

At the time, SYSAMSE was completely recoded using Java© programming language. This strategy was influenced by two major factors : **(1)** advantages of Java language (reusability of code, platform independance, distribution on the Web, etc.), **(2)** simulation languages written in JAVA like SIMJAVA© or SILK© are freely distributed for academic users (fortunately we have both languages). We are also working on the way of formalising the various knowledge needed by our IMSE and how to express it in an uniform manner in order to simplify the inference mechanism. The approach of making the explanation module part of the KBS must also be justifed versus making it as a separate module like in Figure 1.

7. Conclusion

In this paper, we have presented a framework of an intelligent Modeling and simulation environment which combine practical benefits of IGI and AI techniques. Important features of such an environmemt are : the same graphical representation of a system to be simulated is used for automatic simulation program generation purpose and for animating simulation results. Another important feature is that the IMSE provides an explanation module which gives to a user the possibility to ask questions about situations that he does not globally perceive or that he thinks that they would be arise otherwise. When observing animation on the screen, users are

stimulated and become more curious to know enough about what is happening and why it is happening so. Hence, combining animation and explanation in the same environment will simplify the understanding of simulation models especially for the inexperienced users.

References

[BEL91a] B. Belattar, "Conception et Réalisation d'un système d'animation graphique pour modèles de simulation à événements discrets", Thèse de Doctorat, UCB Lyon I - FRANCE, 1991.

[BEL91b] B. Belattar, H. Pierreval, A. Dussauchoy, "An Intelligent Modeling Environment Based on A Graphical Interface", 3rd International Symposium on Systems Research Informatics and Cybernetics, Baden-Baden (Germany), August 1991.

[BEL91c] B. Belattar , A. Dussauchoy, "SYSAMSE : A System for Animating Discrete Event Simulation Models", 3rd International Symposium on Systems Research Informatics and Cybernetics, Baden-Baden (Germany), August 1991.

[BEL97d] B. Belattar , M.E. Laraba, "Apports de l'explication en simulation", Actes de la conférence MOSIM'97, 5-6 juin 1997 , Rouen, France, pp. 527-537

[CLE80] A.J. Clementson, "ECSL/CAPS : Detailed Reference Manual", University of Birmingham, U. K , 1980.

[GIL86] A.R. Gilman, "A tutorial on SEE-WHY and WITNESS", Proceedings of the 1986 Winter Simulation Conference, 1986 , pp. 178-183, 1986.

[KLE86] B. Kleine, "A SIMFACTORY tutorial", Proceedings of the 1986 Winter Simulation Conference, pp. 193-196 , 1986.

[OKE86] R. O'keefe, "Simulation and expert systems - A taxonomy and some examples"; Simulation January 1986, pp. 10-16

[PEG86] C.D. Pegden, "Introduction to SIMAN", System Modeling Corporation, Calder Square P. O. Box 10074, State College, Pennsylvania 16805-0074, 1986.

[PRI86] A.A.B. Pritsker, "Introduction to Simulation and SLAM" (3rd edition), Halsted Press, New York, 1986

[SHA88] R.E. Shannon, "Knowledge Based Simulation Techniques for Manufacturing"; International Journal of Production Research, Vol. 26, No. 5, pp. 953-973, 1988

[SHAR 94] G. Sharma, R.G.S. Asthana, S. Goel, "TO knowledge based simulation approach for train operation and planning Flight", Simulation 62, N°6, June 1994, pp. 381-391

[STA85] C.R. Standridge, "Performing Simulation Projects with the Extended Simulation System TESS", Simulation December 1985, pp. 283-291, 1985

[ZEI 96] B.P. Zeigler, Tae H. Cho, Jerzy W. Rozenblit, "TO knowledge based simulation Environment for Flexible Hierarchical Manufacturing", IEEE Transactions on Systems, Man, And Cybernetics, Vol. 26, N°1, January 1996 pp. 81-90

[ZHA89] Zhang, Shou-Xiang, Schroer, B.J. and F.T. Tseng "An Automatic Programming Approach to Simulation Prelaunch Activities", Simulation, July 1989, pp. 23-29