SOFTWARE FOR REMOTE PARALLEL SIMULATION

Alexander Mikov, Elena Zamyatina, Anton Firsov

Abstract: This paper describes distributed/parallel simulation system Triad.Net and software, which allows geographical distributed users to participate collaboratively and remotely in simulation experiments and to observe simulation model behavior via Internet.

Keywords: Simulation, remote access, portal

ACM Classification Keywords: I.6.7. Simulation Support Systems

Introduction

At present, the simulation remains the universally recognized method of scientific investigations in various spheres of knowledge. Thus it is extremely important to elaborate software able to construct simulation models, to control the simulation experiments, to analyze and to present the simulation results in a proper way. Due to the problems complexity we need more and more the distributed/parallel simulation systems using resources of several processors or computers [1, 2 and 3].

There are three important issues noted by R.Fujimoto и K.Perumalla [4] which promote the parallel information technology in the sphere of simulation:

- availability of graphic interface;
- availability of remote access to the parallel computer resources;
- availability of an adequate software allowing the remote users to operate jointly via Internet with one and the same simulation model.

The report presents language and program tools of the simulation system Triad.Net allowing the remote investigators to collaborate with other researchers interacting with the same simulation model and observing the simulation model behavior within the simulation run.

Description of the Simulation Model in Triad.Net

CAD system Triad [5,6,7] was developed in Perm University and was intended to design built-in computer systems.

Let's present Triad simulation model brief description. The exhaustive information is presented in detail in [5]. So simulation model μ={STR,ROUT,MES} consists of three layers, where STR is a layer of structures, ROUT – a layer of routines and MES – a layer of messages. The layer of structure is dedicated to describe the physical units and its interconnections, but the layer of routines presents its behavior. Each physical unit can send a signal (or message) to another unit. So each object has input and output poles (P_{in} – input poles are used for sending messages, P_{out} – output poles serve for receiving messages). A message of simple structure can be described in the layer of routines, but a message of complex one – only in the layer of messages. Many objects to be simulated have a hierarchical structure. So its description has a hierarchical structure too. One level of its structure is presented by graph P = {U, V, W}. P-graph is named as graph with poles. V is a set of nodes, presenting the physical units of object to be designed, W – a set of connections between them, U – a set of external poles. The internal poles are used for information exchange within the same structure level; in contrast the set of external poles serves to send signals (or more complex information) to the objects, which are situated

on higher or underlying levels of description. Special statement *out (*out <message> through <pole name>) is used to send a message. A set of routines is named ROUT- routine layer.

Special algorithms – routines – define the behavior of a physical unit and are associated with some particular node of graph P = {U, V, W}. Each routine is specified by the set of events (E-set), the linearly ordered set of time moments (T-set) and the set of states {Q-set}. State is specified by the values of local variables. Local variables are defined in a routine. The state is changed only if an event occurs. One event schedules another event. So simulation system Triad.Net is a discrete-event one. Routine (as an object) has input and output poles (Pr_{in} and Pr_{out}) too. An input pole serves to receive messages, output – to send them. An input event e_{in} has to be emphasized among the other events. All of the input messages are processed by the input event, an output messages – by the normal event.

Simulation system Triad.Net is advanced Triad system, but it is the distributed/parallel one. Conservative and optimistic algorithms were designed in Triad.Net (mathematical model is given in [13]). Besides, Triad.Net is characterized by the following [8]:

1. Triad language includes the special type of variables – type “model”. There are several operations with the variable type “model”. The operations are defined for the model in general and for each layer. For example, one may add or delete a node, add or delete an edge (arc), poles, union or intersection of graphs. Besides, one or another routine (routine layer) using some rules can be assigned to the node (structure layer). The behavior of the object associated with this node would be changed. Besides, there’s no need to retranslate the model. Thus a simulation model can be described by linguistic structures or built as a result of a model transformation algorithm.
2. Simulation model is hierarchical, so each model in a structure layer can be associated with some substructure.
3. Model analysis subsystem has to provide a user with the possibility to formulate not regulated request. This approach to the information collection allows to avoid information superfluity or its insufficiency. One can change the set of collecting data within the simulation run, model remains invariant. Model analysis subsystem has to posses smart software tools to analyze the simulation run results and to recommend the policy for the following simulation runs.
4. Simulation algorithm has to be effective and scalable, therefore the objects have to be mapped to the several computers of network (or several processors) so that to minimize the communication time and to consider the processor workload.
5. The system has to be object-oriented (inheritance and reusable code have to be provided).

Graphics editor of simulation model forms xml-document. This document includes object descriptions, the description of routines, the description of complex messages and information procedures. Information procedures are intended for data collection, we shall describe them in detail later. Besides, xml-document contains the information about a simulation model object distributed over several computers [17].

Xml-language usage increases simulation system flexibility, providing interoperability and code reusability [8]. Notice that a rich variety of simulation systems uses xml language nowadays [10, 11, 12 and so on].

Triad.Net can be related to “monolithic” [1] simulation systems because of the fact that it has an environment for simulation experiment carrying out and programming means for simulation results analysis and its representation. On other hand Triad.Net is a component-oriented simulation system because each object can be represented as a separate component too.

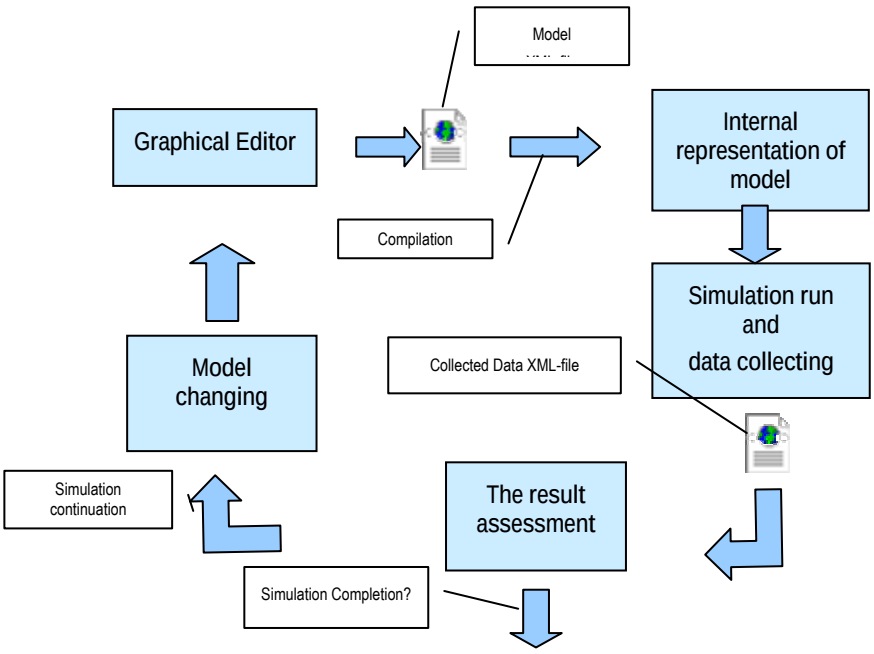
Information Procedures

The objects acting under some definite scenarios and synchronizing algorithm could be called as simulation algorithm. The special objects (information procedures and conditions of simulation) performing the collection, processing and data analyses within the simulation run make up a research algorithm.

Information procedures observe only that model elements (events, local variables, input and output poles) which are indicated by a researcher. Information procedures act as attached sensors. If a researcher decides at some time of simulation run to observe other model elements or to fulfill some other processing of data being collected, he could make some definite directives to change the set of information procedures attached to a model.

Information procedures are the only programming tools providing simultaneous access to the different objects. It is these information procedures that allow a researcher to interact with a model during the simulation run. And a remote interaction is possible too. Model information collected and processed in the information procedures is analyzed in a special program tool "the conditions of simulation". "The conditions of simulation" determine whether the imitation run to be finished.

Simulation system Triad.Net includes some language tools to describe the information procedure syntax. The description of information procedures includes: the description of the beginning part, made prior to a simulation experiments, the description of final part (made after the experiment), the description of an information procedure body, the description of setting and interfacing parameters. The body of information procedure contains collected data processing algorithm. The information procedure starts to follow this algorithm when: the observed variable is changed or observed event takes place or a message is detected on the input pole or a message is sent from the output pole. Researcher receives model description as xml-document using graphic editor. The results of the simulation run are presented as an xml-document too. Thus simulation run results can be processed by Triad.Net integrated tools or by standard tools used for xml-documents processing.



1. Simulation in Triad.NET

As it mentioned above information procedures and special program tool "conditions of simulation" make up research algorithm. Research algorithm functions separately from simulation algorithm and satisfies the

requirements (the requirement 3). Research algorithm and simulation algorithm can function as concurrent run units.

Visualization

In either case (parallel or not parallel simulation) visualization has become indispensable program tool of a researcher. In addition simulation model debugging gets new motivation for visualization tool design in the case of parallel simulation [4]. Graphic visualization makes the decision of this problem and the problem concerning simulation run speed up (the requirement 4) more transparent.

Most of commercial simulation systems (not parallel) have friendly graphic interface, but it is not true for the parallel ones. Nevertheless the works are underway [4, 13, and 14].

User-friendly interface supporting all simulation phases and remote access to parallel resources through the mediation of Internet can only improve the parallel simulation model employment. This is of particular importance for the researchers because of the fact that they very often do not have an access to powerful computer resources. Moreover, the specialists in simulation and in other knowledge domain (biology, telecommunications and so on) are not good specialists in parallel and distributed programming on frequent occasions. These researchers need user-friendly graphical interface because it can hide the special features of parallel and distributed programming. The remote access via Internet should be of particular assistance in collaborating work on a single project for geographically distributed researchers. The project Jane [4] is a very interesting elaboration of this kind. Jane is an interactive simulation framework; it is based on client-server architecture and intended to help users to remotely and collaboratively interact with parallel simulations over the Internet. Server is written in C, and client – in Java. Jane supports clients written in Java also.

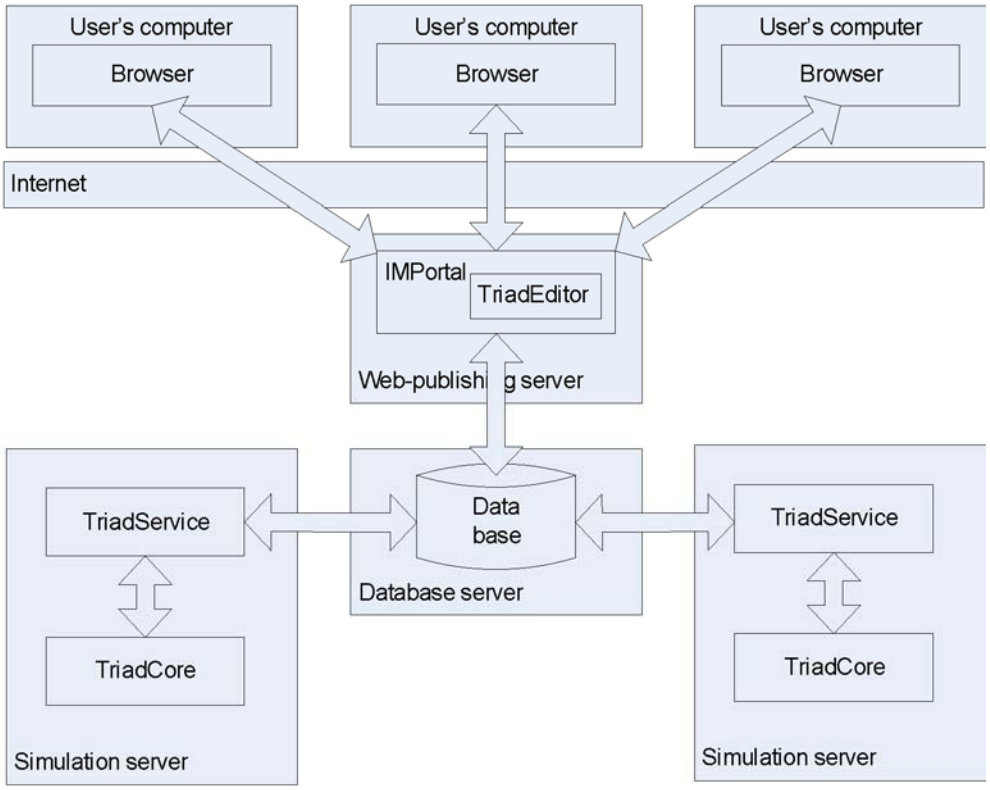
The Jane simulation framework developers decided to avoid tightly integration of graphical interface software and parallel simulation one. That is because the parallel simulation software is inherently complex and it is important to avoid the additional complexity which could appear as a result of tightly coupling parallel simulation kernel and graphical interface. Jane simulation framework was developed as a neutral one because it may be incorporated in Ted and RTI software for remotely visualizing parallel simulation run, debugging and representing simulation run results.

Triad.Net distributed simulation software applies special software tools – information procedures and conditions of simulation- to visualize the behavior of model components during simulation run and simulation results (the results of information procedures functioning - integrated statistical characteristics of simulation model, for example). Besides, information procedures are a very convenient tool for an interaction of a model developer and simulation model within the simulation run.

Issues of Elaboration of Triad.Net Simulation System

When elaborating simulation system Triad.Net we applied .Net technology. Nowadays the Triad.Net software is written in C#. Since .Net technology and xml (the model is represented as an xml-document, and the yields of information procedure is represented as an xml-document) permit to develop rather flexible component–oriented simulation system. Moreover program code can be reused [15,16].

The information procedures are used in graphical editor. Graphical editor includes information procedures, intended to visualize and animate the data being collected during the simulation run (nowadays we carry out the developing special software in VRML providing visualization and animation) and provides the remote access to a simulation model [9,18].



2. Triad.NET architecture for remote access

As a result the simulation system Triad.Net includes such components:

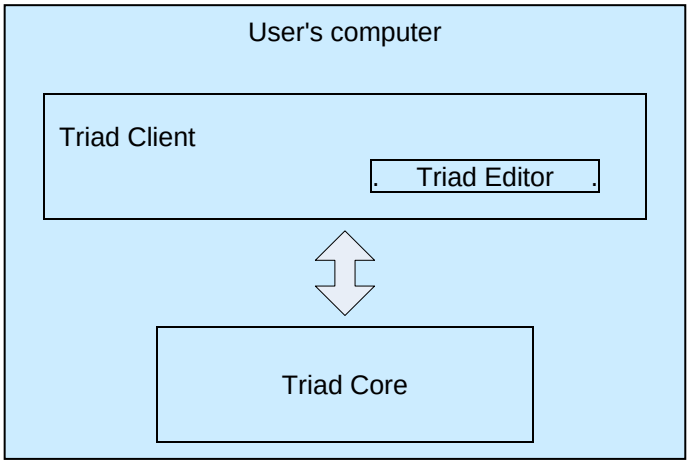
1. IMPortal is a metadata based Internet-portal. It can be used separately as a self-dependent application. One can adjust portal structure in order to change portal content (or build it again in spite of the fact that its content is not concerned with the simulation model). IMPortal is a frame of Internet –portal. Administrator defines its content.
2. TriadCore is a library of types and contains the descriptions of basic structures. These structures are used in the simulation model. The list of structures:
   - BaseObjectClass is a class of basic object descriptions; all classes of objects inherit its properties and methods.
   - BaseObject is a class of basic object descriptions; all objects of simulation system inherit this basic object.
   - BaseSpy is a class of basic information procedures; all information procedures inherit this basic information procedure.
   - ModelRunner is a class for model launch.

   Channel, Port, Message, Event and other basic classes of lesser importance, their functions are clear from their names.
3. TriadEditor is a module with functions realizing user graphical interface for simulation model editing. TriadEditor is the user control Windows elements and can be built in other Windows applications. One may represent Triad.Net simulation model as xml-document using adequate property of this component.

TriadEditor is used in IMPortal module providing collaborative investigations for geographical distributed model developers and researchers.

4. TriadClient is a Windows application to be used if one doesn't have an opportunity to develop and examine the simulation model over Internet.
5. TriadService is a Windows service for simulation model execution.

Model developer has a local and remote access. Separate simulation model components accommodate different servers and begin to execute their algorithms. There is now limit for the amount of servers.



3. Triad.NET architecture for local work

There are two versions of software in the simulation system Triad.Net providing local and remote access to a simulation model. Model developer can use local version if he has no access to Internet. Local version consists of Windows application which uses TriadEditor component and Triadcore – the library of classes.

TriadEditor component provides the possibility to develop, edit and launch a simulation model. It is able to be built-in equally in Web-page and usual Windows application. Microsoft Framework 2.0 has to be included in a client computer software.

Bibliography

1. Ferenci S, Perumalla K., and Fujimoto R. An Approach to Federating Parallel Simulators //Workshop on Parallel and Distributed Simulation, May 2000 (http://www.cc.gatech.edu/ computing/ pads/papers.html)
2. pcl.cs.ucla.edu/projects/parsec
3. www.isi.edu/nsnam/ns/
4. Perumalla K., and Fujimoto R. Interactive Parallel Simulations with the *JANE* Framework//Workshop on Parallel and Distributed Simulation(http://www.cc.gatech.edu/ computing/ pads/papers.html)
5. Mikov A.I. Formal Method for Design of Dynamic Objects and Its Implementation in CAD Systems // Gero J.S. and F.Sudweeks F.(eds), Advances in Formal Design Methods for CAD, Preprints of the IFIP WG 5.2 Workshop on Formal Design Methods for Computer-Aided Design, Mexico, Mexico, 1995. pp.105-127.
6. Mikov A.I. Simulation and Design of Hardware and Software with Triad// Proc.2nd Intl.Conf. on Electronic Hardware Description Languages, Las Vegas, USA, 1995. pp. 15-20
7. Миков А.И. Определение характеристик ожидания в однолинейной системе по моментам исходных распределений// Изв. АН СССР. Техническая кибернетика, №3, 1980

8. Миков А.И., Замятина Е.Б., Фатыхов А.Х. Система оперирования распределенными имитационными моделями сетей коммуникации. //В кн. «Материалы Всероссийской научно-технической конференции «Методы и средства обработки информации МСО-2003».1-3 октября 2003 г.
9. Миков А.И., Замятина Е.Б. О разработке исследовательского портала «Имитационное моделирование»//В кн. «Материалы Всероссийской научно-технической конференции «ИММОД-2003».23-24 октября 2003 г.
10. Syrjakow M., Syrjakow E., Szczerbicka H. Towards a Component-Oriented Design of Modeling and Simulation Tools// Proceedings of the International Conference on AI, Simulation and Planning in High Autonomy Systems (AIS 2002), Lisbon, Portugal, April 7-10, 2002
11. Thomas Wiedemann. Next Generation Simulation Environments Founded on Open Source Software And XML-based Standard Interfaces //Proceedings of the 2002 Winter Simulation Conference (E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds.), pp.623-628.
12. Thorsten S. Daum, Robert G. Sargent, A Web -Ready HiMASS : Facilitating Collaborative, Reusable, And Distributed Modeling And Execution Of Simulation Models With XML//Proceedings of the 2002 Winter Simulation Conference E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds.,pp 634-640
13. Вознесенская Т.В. Математическая модель алгоритмов синхронизации времени для распределённого имитационного моделирования.//В кн. Программные системы и инструменты. Тематический сборник факультета ВМиК МГУ им. Ломоносова №1., стр.56-66
14. Bagrodia, R., The Maisie Visual Programming Environment, Department of Computer Science, University of California, Los Angeles, http://may.cs.ucla.edu/projects/mvpe/.
15. Миков А.И., Замятина Е.Б. Система имитационного моделирования с XML интерфейсом.// Сб. трудов международной школы-семинара «Современные проблемы механики и прикладной математики», Ч1, т1, Воронеж, 2003, с.244-247
16. Миков А.И., Замятина Е.Б. Система имитации с удалённым доступом.//В кн. «Материалы третьей междисциплинарной конференции с международным участием (НБИТТ-21), 21-23 июня 2004, Петрозаводск, стр.73
17. Миков А.И., Замятина Е.Б., Осмехин К.А. Метод динамической балансировки процессов имитационного моделирования.//В кн. «Материалы Всероссийской научно-технической конференции «Методы и средства обработки информации МСО-2003».1-3 октября 2005, стр. 472-478
18. Замятина Е.Б., Муллаханов Р.Х., Фирсов А.Н. On Approach of Researching Portal Development //В кн. «Информационные технологии и телекоммуникации в образовании и науке», 15-22 May, 2005, Turkey, pp.243-246

Authors' Information

Alexander Mikov – Director of the Institute of Computing, Bukireva str., 15, Perm, Russia, e-mail: Alexander_Mikov@mail.ru

Elena Zamyatina – Perm State University, Bukireva str., 15, Perm, Russia, e-mail: e_zamyatina@mail.ru

Anton Firsov – Perm State University, Bukireva str., 15, Perm, Russia, e-mail: a_firsov@mail.ru