

**1st INTERNATIONAL SIMULATION
TOOLS CONFERENCE & EXPO
2013**

SIMEX'2013

EDITED BY

Jan F.M. Van Impe

and

Filip Logist

SEPTEMBER 18-19, 2013

BRUSSELS, BELGIUM

A Publication of EUROSIS-ETI

1st International Simulation Tools
Conference and Expo 2013

BRUSSELS, BELGIUM
SEPTEMBER 18-19, 2013

Organised by

ETI- The European Technology Institute

Sponsored by

EUROSIS - The European Simulation Society

Co-Sponsored by

FLEXSIM

Incontrol Simulation Solutions

aGPSS

University of Skovde

AnyLogic

Hosted by

TheHotel
Brussels, Belgium

EXECUTIVE EDITOR

**PHILIPPE GERIL
(BELGIUM)**

EDITORS

General Conference Chairs

Jan F.M. Van Impe, KU Leuven, Louvain, Belgium

Filip Logist, KU Leuven, Louvain, Belgium

Publications Chair

Yan Luo, NIST, Gaithersburg, USA

INTERNATIONAL PROGRAMME COMMITTEE

Comparative Studies

Christina Klüver, University of Duisburg-Essen, Essen, Germany

Academic Tools

Chrissanthi Angeli, Technological Institute of Piraeus, Athens, Greece

Adrian Florea, University of Sibiu, Sibiu, Romania

Chris Koh, Pepsico, Plano, TX, USA

Roberto Montemanni, IDSIA, SUPSI, Manno, Switzerland

Viorel Nicolau, University of Galati, Galati, Romania

Laurent Perochon, VetaGro Sup, Lempdes, France

Nabil Sahli, GUTech, Athaibah, Sultanate of Oman

Manuel Felipe Santos, University of Minho, Guimaraes, Portugal

Maria João Monteiro Ferreira Viamonte, Instituto Superior de Engenharia do Porto, Porto, Portugal

Commercial Tools

Valerian Croitorescu, University POLITEHNICA, of Bucharest, Bucharest, Romania

Houcine Hassan, UPV, Valencia, Spain

Fred Jansma, INCONTROL Simulation Solutions, Utrecht, The Netherlands

Muqem Khan, Northwestern University in Qatar, Doha, Qatar

Jan Pieters, Ghent University, Ghent, Belgium

Jan Studzinski, Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland

Open Source Tools

Jan F. Broenink, University of Twente, Enschede, The Netherlands

Jacinto Davila, Universidad de Los Andes, Merida, Venezuela

Paulo Goncalves, Instituto Politecnico de Castelo Branco, Castelo Branco, Portugal

Thomas Hanne, University of Applied Sciences and Arts Northwestern Switzerland, Olten, Switzerland

Axel Hummel, University of Leipzig, Leipzig, Germany

Frederik Ronsse, Ghent University, Ghent, Belgium

SIMEX 2013

© 2013 EUROSIS-ETI

Responsibility for the accuracy of all statements in each peer-referenced paper rests solely with the author(s). Statements are not necessarily representative of nor endorsed by the European Simulation Society. Permission is granted to photocopy portions of the publication for personal use and for the use of students providing credit is given to the conference and publication. Permission does not extend to other types of reproduction, nor to copying for incorporation into commercial advertising nor for any other profit-making purpose. Other publications are encouraged to include 300- to 500-word abstracts or excerpts from any paper contained in this book, provided credits are given to the author and the conference.

All author contact information provided in this Proceedings falls under the European Privacy Law and may not be used in any form, written or electronic, without the written permission of the author and the publisher. Infringements of any of the above rights will be liable to prosecution under Belgian civil or criminal law.

All articles published in these Proceedings have been peer reviewed

Selected papers of this conference are published in scientific journals.

Legal Repository: Koninklijke Bibliotheek van België, Keizerslaan 4, 1000 Brussels, Belgium
CIP 12.620 D/2011/12.620/1

For permission to publish a complete paper write EUROSIS, c/o Philippe Geril, ETI Executive Director, Greenbridge NV, Ghent University Ostend Campus, Wetenschapspark 1, Plassendale 1, B-8400 Ostend, Belgium.

EUROSIS is a Division of ETI Bvba, The European Technology Institute, Torhoutsesteenweg 162, Box 4, B-8400 Ostend, Belgium

Printed and bound in Belgium by Reproduct NV, Ghent, Belgium
Cover Design by Grafisch Bedrijf Lammaing, Ostend, Belgium

EUROSIS-ETI Publication
ISBN: 978-90-77381-78-6
EAN: 978-90-77381-78-6

PREFACE

Dear Participants,

On behalf of the Organizing Committee, we would like to welcome everyone to the 1st Annual Simulation Tools Conference and Expo (SIMEX 2013), hosted at TheHotel, Brussels, Belgium from September 18th-19th, 2013.

During these two days, we offer a very high-level scientific program, spanning a wide array of academic and commercial tools used in simulation and modeling. Furthermore, this event also offers you the possibility to try out these tools, so you can see them in action solving set problems and provide you with feedback on their use and applicability to your particular industrial and or commercial problems.

Three main speakers, from academia and industry, will be sharing their professional expertise. They are, Chris Koh from Pepsico, who will give a presentation on “Why simulation is Important; An Engineer’s Perspective”, Ingolf Stahl from the Stockholm School of Economics, who will give a presentation entitled: “An Easy to Learn System for Simulation of Business Problems” and Valerian Croitoresu who will give a talk entitled: “Commercial Simulation Software for Modeling and Analysis of Vehicle Powertrain Systems”.

We would also like to thank Kurt De Cock of Ghent University and Ingolf Stahl from the Stockholm School of Business for their introductory Problem solving courses using Flexsim and aGPSS, respectively.

We are indebted to all of you, without whom, the conference would not be possible. First of all, thanks go to all reviewers for critically evaluating the papers and keeping an eye on the scientific quality of the conference.

Moreover, we are grateful to the keynote speakers for accepting our invitation and for presenting their latest contemplations on significant simulation themes.

Furthermore, we would like to express our thanks to Mr. Philippe Geril, Secretary General of EUROSIS, who has taken the responsibility for most of the organizational matters. This new hands-on Simulation Tools Conference is only possible thanks to his hard work and never-ending enthusiasm.

Finally, our warm thanks go to all of you for submitting your research results, attending sessions and actively participating and discussing. We wish you all a stimulating and productive meeting.

General Conference Chairs
Jan F.M. Van Impe
Filip Lagast

Preface	VII
Scientific Programme	1
Author Listing	73

INVITED PRESENTATIONS

An Easy to Learn System for Simulation of Business Problems Ingolf Ståhl	5
--	----------

Commercial Simulation Software for Modeling and Analysis of Vehicle Powertrain Systems Valerian Croitorescu and Jan Anthonis	12
--	-----------

ACADEMIC TOOLS

Soft Computing Tools for the Analysis of Complex Problems Christina Klüver and Jürgen Klüver	23
--	-----------

A New Approach to 3D Simulation Technology as Enabling Technology for eRobotics Jürgen Rossmann, Michael Schluse, Christian Schlette and Ralf Waspe	31
---	-----------

Estimating Project Risks' Probability with Limited Statistical Data Michael Douloumpakis, Vrassidas Leopoulos, Elena Rokou and Konstantinos Kirytopoulos	39
--	-----------

COMMERCIAL TOOLS

Multi-Method Modeling Andrei Borshchev	47
--	-----------

Human-and Hardware-in-the-Loop Simulator for Studying Vehicle Longitudinal Dynamics Ionuț Stoica, Marius Bățăuș and Valerian Croitorescu	56
--	-----------

Simulation with ARENA of Two-Machine Flowshop Scheduling Using Modified Johnson Rule Fawaz Abdulmalek	61
---	-----------

Simulation Languages Facilities for Innovation Puhalschi Radu, Feher Szilard, Vasiliu Daniela, Vasiliu Nicolae and Irimia Petru Cristinel	66
---	-----------

SCIENTIFIC PROGRAMME

INVITED PRESENTATIONS

AN EASY-TO-LEARN SYSTEM FOR SIMULATION OF BUSINESS PROBLEMS

Ingolf Ståhl
Stockholm School of Economics
Box 6501
SE-11383 Stockholm, Sweden
ingolf.stahl@hhs.se

KEYWORDS

Education, Business, Decision, Discrete.

ABSTRACT

aGPSS is a stream-lined version of GPSS, the General Purpose Simulation System, which seen over the years is the world's most used program for dynamic simulation of systems facing uncertainty. aGPSS is intended to be the starting point in learning simulation, mainly for business students. aGPSS is the result of an educational development process, mainly at SSE, the Stockholm School of Economics, based on feedback from the 10,000 business students that we have taught over several decades. Students, often with little computer experience, have showed us what is difficult and, when they have made repeated mistakes, we have simplified the syntax.

We have also added a user friendly graphical user interface. By pointing and clicking on block symbols in a symbol menu, both a block diagram and program code can be constructed. We have recently added an automatic animation system, which directly produces an animation of the basic process using Proof Animation. A demo version of aGPSS can be downloaded from WWW.aGPSS.com.

WHY SIMULATION AT A BUSINESS SCHOOL?

This paper deals with a system for teaching simulation at business schools. It is hence suitable to start with answering the question why one should learn simulation at business schools. It should then first be mentioned that we with simulation refer to stochastic discrete event simulation.

The main reason for our focus on simulation in business schools is that simulation plays an increasingly important role in business and has now become the by far most important method within what is called Operations Research or Management Science.

The main reason for this development is, in turn, the extremely rapid development of computer technology. A computer for a given price has, and will probably within the foreseeable future, double in speed and capacity roughly every 18 months (an observation called Moore's law). This, in turn, implies that computers, for a given price, during the last two decades have increased in power more than a thousand times. This implies that if simulation earlier, because of high computing costs and time requirements, was

regarded as a method of last resort, simulation is now rather regarded as the first alternative to try.

Another factor contributing to this increased importance of simulation is the fall in prices for simulation software.

Because of these developments, computer simulation has come to replace many analytical/optimization parts of Operations Research or Management Science methods, such as e.g. queuing theory, inventory theory, PERT/CPM and decision theory. It appears appealing to cover many problem areas with **one** single easy-to-use method instead of relying on a great many more complex mathematical methods. There can then instead be a greater focus on **modeling**.

Simulation furthermore plays a major role for production planning and such simulation can give business students better understanding of the physical processes in a firm. Closely related to this is that one with simulation can demonstrate the connection between the physical activities and the consequential financial flows. This can be connected with cash flow forecasting, another important application area of simulation.

Furthermore, stochastic simulation is required to handle the uncertainty that is the core of financial theory. We can just think of how we want to answer the following questions: How much will we sell next year: 100,000 units for certain or 80,000 - 120,000 units? When will this customer pay: Within 30 days for certain or with 80 percent probability within 60 days? What will the €/\$ ratio be a year from now: 1.3 for certain or between 1.0 and 1.6? In all cases, the last answer, indicating uncertainty, seems more reasonable. In fact, if all future payments could be forecast with certainty, all corporate debt would be as safe as government bonds. Against this background, it is clear that financial simulation **should** be stochastic.

Finally, there is a need for **dynamic** simulation, i.e. discrete events simulation, allowing us to follow each major payment, regardless of when it takes place. This can be illustrated by two graphs of the cash forecast of a small corporation, Figures 1 and 2 on the next page. Figure 1 represents the static case, where only a limited number of time-points can be taken into regard, while Figure 2 represents the dynamic case, where **every** possible time-point can be included in the simulation. We see that the two graphs give completely different impressions.

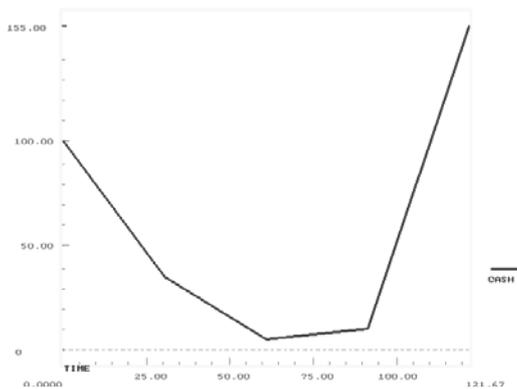


Figure 1: Static Cash Graph

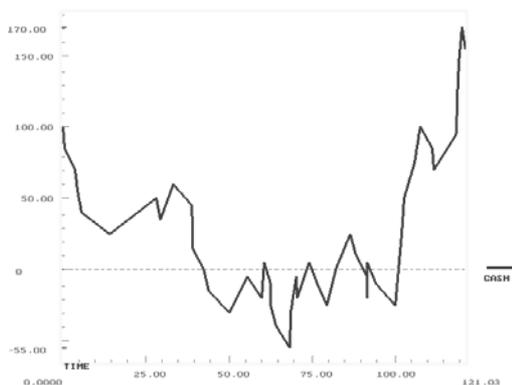


Figure 2: Dynamic Cash Graph

From figure 1 it appears that there would be enough cash for the corporation and hence not any liquidity problems. The financial problems, with negative cash a great many times, are clearly seen in Figure 2, where we can follow payment by payment. In Figure 1, these problems are not perceived at all, since by chance there is a cash surplus at each end of each month. This illustrates the need for a dynamic, i.e. discrete-events, approach to corporate financial simulation.

The two graphs above give a good answer to a question often asked, namely why we at a business school do not use only Excel. The main reason is that much of the dynamic simulation, as illustrated by Figure 2 above, is completely impossible to do in ordinary Excel, i.e. without resorting to Visual Basic. Standard Excel, generally with a maximum of 256 columns, does not allow the showing of all possible time points. Of the greatest importance is furthermore that Excel is built for "pulling" data from another cell and cannot assign value to a cell and can hence **not** have e.g. the sales of January by random paid either in February or March.

Among other important applications of simulation that are of interest in a business school curriculum, I should mention simulation based costing, forecasts of diffusion of new technology, determining optimum equipment life, supply chain management and lap-top based sales models, e.g. for dimensioning of equipment. It should finally be stressed that, with the rapidly increasing speed of cheap computers and the development of new methods for optimization (like taboo-search, genetic algorithms, simulated annealing, etc.), simulation allowing for any kind of more realistic functions

is being increasingly used also for optimization, except in certain standard cases, with linear and quadratic functions.

WHY SIMPLE SYSTEM AT BUSINESS SCHOOLS?

Before presenting our own system for teaching simulation at business schools, I would like to discuss why we have not used a commonly used commercial simulation system.

It is here important to discuss the purpose of teaching simulation at a business school, which I think is in many regards different from the purpose with teaching simulation to students of engineering, in particular of manufacturing systems. While a student of manufacturing engineering in many cases can be expected to directly use simulation in his/her first job and be employed to work full time working with the simulation system taught at the school, like Arena, Simio, etc., it is rare that the business graduate will work directly as a full time simulation expert.

The purpose of teaching simulation to the business student can rather be seen to fulfill three main goals.

1. As mentioned above, simulation can be used to replace many OR methods, like queuing theory, inventory planning methodology, decision theory, project management methods, etc.
2. The business students can be trained to be an informed buyer of simulation services, provided by e.g. earlier engineering students. By having some training in using a simulation software, the earlier business students will better understand both the potentials and limitations of simulation, in particular with regard to the economy of simulation activity.
3. As will be discussed more below, the main benefit for the business student of mastering a simple simulation system is that he/she can do a simulation project work in a company and, then go through all the steps of the simulation process, involving, among other things, data collection and presenting the results of the simulation study to people in the company. We have found that a simulation project with a clear focus on concrete questions has taught our business students more about general business than other types of student projects in companies.

Another big difference between the teaching of simulation to engineering students and business students concerns the background knowledge of the students. While most engineering students who start studying simulation have some experience with computer programming, e.g. in C++, the average business student can at best be expected to know some very rudimentary Excel, but very seldom any knowledge of VB. Most of our business students could not in a reasonable time learn to master the more advanced modern commercial simulation systems. Hence, the simulation package to be used for teaching business students should be quite different to that used by engineering students.

In order that the business student should be able to do the project work mentioned above, the time for learning the mechanics of the simulation system must be quite limited. In the cases when there is only time corresponding to a month of full time work, and the project work in the company, though mostly carried out in a group, requires around half of this time, not very much than around ten days can be spent on learning the mechanics of the simulation system. Time must also be left for reading about the other aspects of the simulation process. This really requires that the simulation system must be easy to learn.

STEPS TOWARDS A SIMPLE SYSTEM

We shall here look at how we, the developers of aGPSS, in the process of teaching over 10,000 business students have developed, step by step, a simple-to-learn simulation system, geared towards business students. We the developers include not only me, the author of the paper, but also, since 1991, the co-author of the aGPSS text books, Professor Richard Born of Northern Illinois University, and, more recently, the two simulation teachers, who have succeeded me in teaching aGPSS, namely Endre Bjørndal of the Norwegian School of Economics in Bergen and Edgars Jakobsons of the Stockholm School of Economics in Riga. It also includes several teaching assistants at SSE.

It should first be mentioned that, when I started out teaching simulation at the SSE in the mid-70s I used the IBM GPSS/360 system, which at that time was the system that was mostly used in business. I was then teaching a course in Management Science at SSE. This was a full-semester course, requiring roughly four full weeks of study. The textbook used was MacMillan and Gonzales (1973), containing a chapter of 77 pages on GPSS, written by Thomas Schriber. To learn more about GPSS, I acquired Schriber (1974), the "Red Book", and became quite fascinated by this. This book has a total of 27 case studies, which all refer to practical problems, but yet appear easy to understand.

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

In 1983-85, I was visiting professor at Hofstra University, NY, where I taught an elective full-semester course in simulation five times. This course was devoted almost completely to GPSS, which was first run on an IBM

computer using GPSS/360. I had, however, already earlier started to transfer my mini-GPSS from the PRIME to the new IBM PC micro-computer. The new system, called micro-GPSS, could be used at Hofstra in 1984.

Since the whole course was devoted to simulation, I found it important to have the students make a simulation project out in a company, with a focus on modeling, data collection etc. As I expanded my subset of GPSS, I found it important to include features that the students needed in their projects and yet keep GPSS easy to learn. The primary guide on what to include was the 27 case studies in Schriber (1974). I could rewrite not only all of the 27 case studies, but also about 99 percent of all programs in other GPSS textbooks, with roughly the same amount of code.

It should be stressed that this subset of GPSS, although it, as noted above, was almost as powerful as standard IBM GPSS contained far fewer block types, namely 22 to be compared to the 44 block types of IBM GPSS/360 and the almost 70 block types of GPSS/H (Wolverine Software), the successor of IBM GPSS,. In the limited time available to our students, 22 block types were quite enough. As reported on in Ståhl (2003), the choice of these 22 block types was greatly inspired by the selection of block types in the Red book (Schriber 1974).

When I was back in Sweden, GPSS was soon introduced as a quarter of a full-semester course taken by all 300 students at the SSE, many with little computer background. Here I used several teaching assistants. I got a great amount of feedback on what was difficult to learn and also what was difficult to teach. Based on this feedback, I started to simplify the syntax in several respects. Before 1985, micro-GPSS was a pure subset of GPSS/360, with the two exceptions mentioned above, but after this it became a strongly stream-lined version.

An IF block with "straight logic" replaced the TEST block, with IF Q\$Joe=4,BYE instead of TEST NE Q\$Joe,4, BYE of IBM GPSS. Further, WAITIF lock=U was used instead of GATE NU lock. With students tending to forget commas, GOTO BYE and GOTO BYE,0.15 were used instead of TRANSFER ,BYE and TRANSFER .15,,BYE. In order to simplify and make the concepts stronger, a single LET block replaced the ASSIGN, SAVEVALUE, PRIORITY and SELECT blocks. The collection of statistics was simplified by SEIZE Joe,Q replacing QUEUE JoeQ, SEIZE Joe, DEPART JoeQ. This really cut down the length of many programs. Furthermore, features that lead the students to make difficult-to-find logical errors were eliminated. With students reporting any unclear errors, I developed an extensive system for error reporting with 500+ error messages (Ståhl 2001).

My textbook *Introduction to simulation with GPSS* describing this streamlined micro-GPSS system was published by Prentice Hall (Ståhl 1990)

DEVELOPMENT OF A GUI-BASED GPSS

Up to 1998, micro-GPSS was, just like e.g. GPSS/H, a purely text-based system. We obtained feedback, however, from students that it was very desirable that input could also be made using a Graphical Users Interface, where the student from a menu of symbols could choose the building blocks of the program. Then in 1997, a new Swedish Foundation for Computers in Education was looking for university developed software that could be placed on the Web. This was a chance to get financing of roughly \$150,000 to have a GUI based GPSS put on the Web. In 1999 the first tentative versions of such a GUI-based system, WebGPSS, later to be called aGPSS, based on micro-GPSS, was presented, with the client developed in Java. The programs were first developed and translated into text code on the client PC. The code was then run on a remote server using the micro-GPSS engine.

On the client, blocks are chosen by clicking on a symbol in a menu, leading to a block diagram. The symbol menu has been gradually developed and today looks like in Figure 3 below. This point-and-click method allows for a faster model build-up than the drag-and-drop method. By next clicking on a block in the block diagram, a dialog with syntax explanations is opened to allow for the input of the operands of the block. The execution of the code is then carried out by the micro-GPSS “engine”, which produces output files in ASCII format, which aGPSS then turns into tables, histograms and graphs. The output, shown under several tabs, is more readable and understandable than the output generated by standard GPSS. It is easy to print and save each result tab.

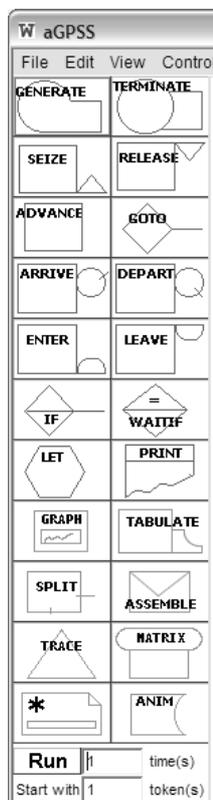


Figure 3: Symbol Menu of aGPSS

WebGPSS was made generally available on the Web and, being run on a Swedish server, it was used in some Swedish colleges and in several high schools, in particular for individual projects in the senior high school year.

Since 2004 WebGPSS has, however, only been run as a stand-alone version under Windows. As mentioned above, the code produced by WebGPSS was earlier run by use of the micro-GPSS engine on a remote server in Sweden, but when many students used this Web system at the same time, serious problems arose, in particular when students ran the same program many times to establish confidence levels. With the micro-GPSS engine programmed in FORTRAN, and not in Java, it could only run on the server and not the client, and hence the server easily got overworked.

With student users in many far away countries, there were also transmission problems. We hence decided that we had to have WebGPSS only running directly under Windows. The server module is then also run on the user’s computer. Because of this change, we have regarded the name **aGPSS** as more suitable than the old name WebGPSS.

RECENT DEVELOPMENTS

We have recently as control statements in aGPSS introduced facilities for experiments and optimization, for warm up and for antithetic random numbers. aGPSS allows for running the model many times with an automatic calculation of the limits within which the universal average will lie with, for example, 95% probability. It also allows for a limited amount of optimization with a graphic representation of these confidence intervals for each of many alternatives. In the case of a comparison between just two alternatives, it will determine, if one alternative is better than the other one with 97.5% probability.

We have also introduced a system for putting simulation output directly into Excel by a simple MATRIX block. Data can also be retrieved from Excel and put into an internal matrix.

Another recently introduced improvement in aGPSS is the possibility to place blocks in different columns. While other simulation systems mainly use a drag-and-drop buildup of the block diagram, aGPSS uses, as mentioned, a point-and-click buildup. The system originally placed a block just below the last inserted block, except for GENERATE blocks, always placed at the top. The disadvantage was that all blocks in a segment were placed in **one** single column.

The new system now allows the user to move the blocks to any desired position, in terms of integer numbers on columns and rows, by the arrow keys. The information about the **relative** position of a block that is not placed just below the earlier block is in the computer code indicated by an extra part of the comment. This has the advantage that all information about the placement of the blocks is in ASCII code, allowing for very compact code, yet containing all information necessary for the block diagram.

The most recent development has been the inclusion of **animation** with Proof of Wolverine Software.

We have first of all allowed for a block ANIM for the interface to Proof Animation..Clicking on this block, the dialog in Figure 4 is obtained.

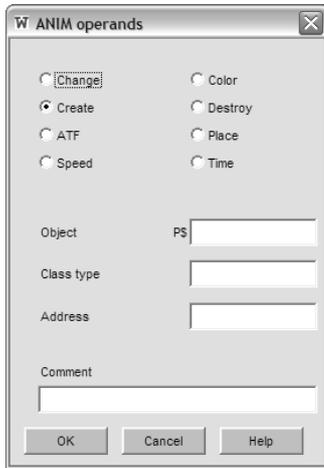


Figure 4: Dialog of ANIM Block

By clicking in one of the 8 radio buttons and writing in the three fields below, the students can in their aGPSS models insert blocks that produce the Automatic Trace Files, ATF, for Proof. The students will, however, when using the ANIM block, have to draw the layout files in the Proof system. This type of animation is intended for the most advanced students.

All students can, however, get animation **without** any ANIM blocks or user made layout files by using the newly developed system of **automatic animation** of aGPSS. Once one has constructed the block diagram of the aGPSS model, one just has to click on the control statement *Automatic animation with block animation*. This will cause the automatic generation of the relevant ATF and layout files, which then can be executed in Proof. This allows the student to follow the movement of the transactions through the system and thus better understand how the software works.

We shall exemplify both the new positioning system and the automatic animation system by a model, which deals with the following problem:

At a store, run by Boris and Naina, customers arrive at rate of 7 ± 3 minutes. A customer first goes to Boris and chooses a bottle. This takes between 3 and 7 minutes. Next he goes to Naina to pay for the bottle. This also takes 3 to 7 minutes. Finally, he returns to Boris to pick up his bottle. This takes between 1 and 3 minutes. He then leaves the store. There is one waiting line in front of Boris and one in front of Naina. A customer returning to Boris to pick up his bottle has to start at the end of this line again. The store is closed after eight hours.

The block diagram of this model, with the blocks positioned to make the model more readable, is shown in Figure 5. We here see clearly that the customers return to Boris the second

time. A snapshot of the automatic animation of this model, which provides information dynamically on the number of customers having come to each block, as well on the number of customers waiting at various blocks, is shown in Figure 6.

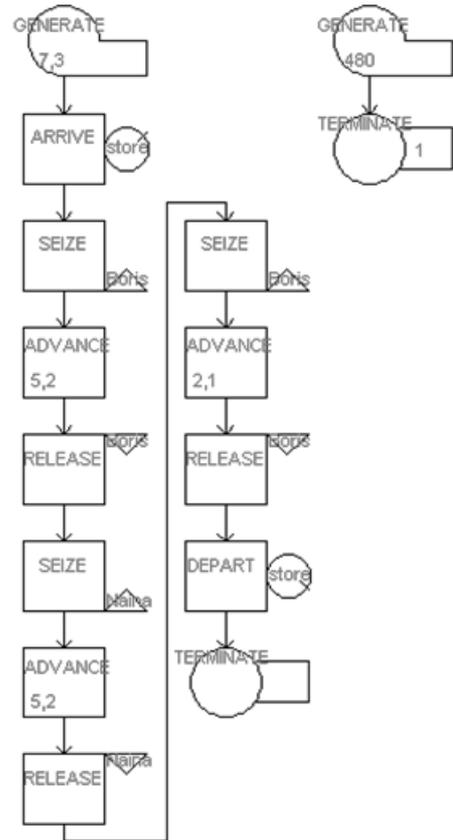


Figure 5. Block Diagram of Boris' Store

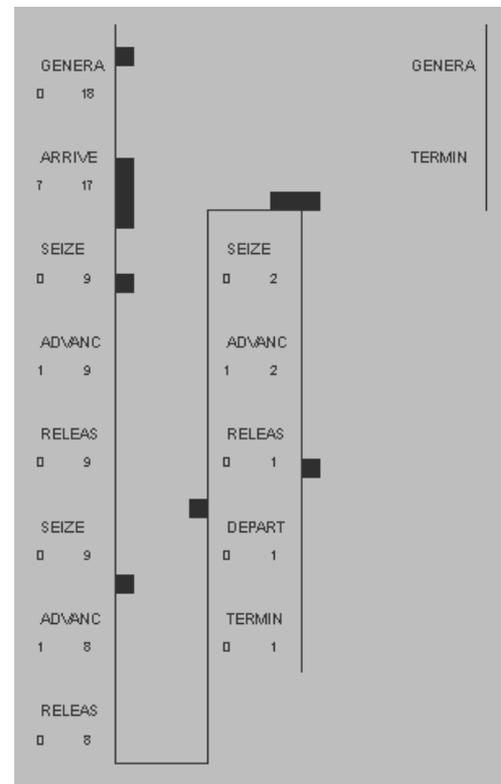


Figure 6. Automatic Animation of Boris' Store

DIFFERENT TYPES OF ANIMATION

After having presented the animation of Boris' store, we can also comment on one of the reasons why the aGPSS system allows for much easier learning and usage than many of the commercial systems with advanced animation capabilities, and which we therefore call **animation based systems**.

I shall first comment on two experiments. In Riga, I carried out an experiment with a class of students without simulation experience. Half of them had four hours of the animation based system Witness, the other half of aGPSS. At the end of each session, the students were asked to model the Boris problem in 45 minutes. While none of the Witness students could do this, all the aGPSS students could do so. The other experiment involved asking the software vendors at the Winter Simulation Conference during several years to solve the Boris problem. Vendors of block based systems solved it in less than 5 minutes, while vendors of animation based systems required more than 30 minutes (Ståhl 2002).

The reason for the difference can be explained by comparing the aGPSS block diagram in Figure 5 with a generalized diagram of an animation based system in Figure 7.

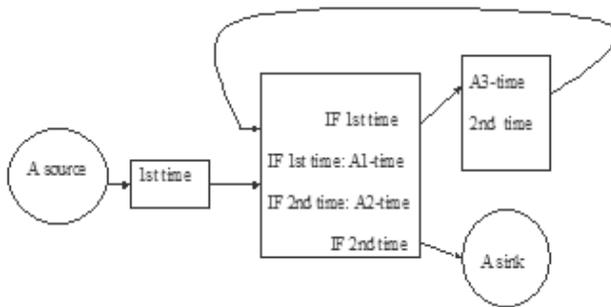


Figure 7: Logic for Boris' Store for Animation Based System

In the animation based system, each permanent server is in principle represented only **once**, since the server in the animation workspace must be in only **one** place. In the block based system, such a server can be represented in **many** different places in the model. Hence, we have SEIZE Boris in two places in the aGPSS model in Figure 5 and we hence do not need the special logic of Figure 7 to establish whether a customer comes the first or the second time to Boris.

CONCLUDING REMARKS

An important part of the continuous development of aGPSS has been the feed-back from the students, in the form of class discussions, errors in exams and project work. The system syntax has been simplified when several students have made the same mistake. Student feedback has also greatly influenced the error reporting system. This feedback has come not only from our own students in Sweden, the US, Germany, Norway and Latvia, but lately also from collaborating teachers in Japan and India. This development would not have been possible, if we would have had a policy of constant upwards compatibility, like e.g. GPSS/H has. In

contrast to commercial software vendors, we have not had to be concerned about a large existing customer base, since our main customers are students, who are novices to simulation.

We have tried to ensure that students at a very early stage get to write simple programs that are not trivial. We have proceeded step-by-step, with simple examples in the beginning, so that no student is left behind at an early stage and loses the possibility to catch up. We have always avoided pre-course knowledge requirements, e.g. of computer programming. Students have not had to learn a new concept every time that a new and different thing has to be done. Students have enjoyed finding that the new aspects can be handled using already known concepts, even if the programs become slightly longer.

Among other features that that our students appreciate are:

- The simple and fast build up of models by using a Graphical Users Interface with the point-and-click method, allowing for a faster build-up than the drag-and-drop method.
- The automatic generation of the most interesting statistics in an understandable form, such as simple tables, histograms and graphs.
- The automatic calculation of cumulative probabilities when defining empirical random functions. For each value, one just inputs the number of observations. aGPSS automatically transfers this into percentages and then cumulative probabilities. In contrast to other systems, the student does not have to do a lot of recalculations, if one decides to add one more observation.
- The ease with which replications of the runs are done, by just one command, and the automatic statistical analysis of these repeated runs, e.g. of confidence levels.
- The single easy-to-read and compact program listing, obtained automatically and allowing for short comments. This program listing, together with the block diagram, has made it easy for the students to study, discuss and document the logic of a model.
- The free or low cost of, software. From the site www.aGPSS.com one can download, free of charge, the smallest aGPSS version, large enough for running all of the 63 programs in the textbook Born and Ståhl (2013). Many aGPSS textbooks include a software version large enough to carry out projects of a sufficient size.
- From the site <http://www.wolverinesoftware.com> the student version of Proof animation can be downloaded free of charge.
- The availability of inexpensive textbooks covering **all** of aGPSS in less than 440 pages and hence **non-intimidating**. It can also serve as a manual (Born and Ståhl 2013; Born and Ståhl 2013b).
- The 300+ aGPSS models available, often helpful for the students when doing project work.

As teachers we have also appreciated that our GPSS classes on average have been given better ratings than other courses. The simplicity of the aGPSS syntax makes it possible for a teacher to have complete command of the whole system and to be able to answer any student questions. The workspace window, which is easy to project visibly to all students in a class room, facilitates teaching. The single compact program listing has been essential for making it easy to correct and mark the student programs.

It should finally be mentioned that although we see aGPSS primarily as an educational tool, it has also been used at several occasions out in companies, for building quite important models. In some cases this has been an outgrowth of a student project, where the student after graduation has been hired to extend his original simulation study. In some cases it has involved a project where the initial aGPSS model has been a **prototype** and where a much larger model then has been built in another, more advanced, system, but with the prototype playing an important role in explaining the simulation model to the decision makers in the company.

Among aGPSS projects that have had important practical applications, we can mention the following:

- The simulation of the exit behavior for 28,000 spectators going to the subway from the planned Olympic stadium in Stockholm. For the application to the IOC, it was important to assure that the new stadium and its exits would be up to standard.
- The establishment of costs for different products for a flexible manufacturing system at an IBM printer factory. This used a GPSS model with some 500 blocks. (Nielsen 1996)
- The construction of a sales support model, to be run on laptops by salesmen, of the Ericsson MD110 corporate switchboard to help the clients determine the number of agents and trunk lines.

REFERENCES

- Born, R. and I. Ståhl. 2013. *Modeling Business Processes with a General Purpose Simulation System – 1. Introduction*. Stockholm: SSE.
- Born, R. and I. Ståhl. 2013b. *Modeling Business Processes with a General Purpose Simulation System – 2 Extensions*. Stockholm: SSE.
- MacMillan, C. and R. Gonzales. 1973. *Systems Analysis - A Computer Approach to Decision Models*. 3rd ed. Homewood, IL: Irwin.
- Nielsen, S. 1996. *Omkostnadskalkulation for avancerade produktionsomgivelser*. Stockholm: EFI.
- Schriber, T. 1974. *Simulation Using GPSS*. NY: Wiley.
- Ståhl, I. 1990. *Introduction to Simulation with GPSS - on the PC, Macintosh and VAX*. Hemel Hempstead, UK: Prentice Hall International.
- Ståhl, I. 2001. "GPSS – 40 years of development". In *Proceedings of the 2001 Winter Simulation Conference* (Arlington, VA. Dec. 9-12). IEEE, Piscataway, N.J., 577-585.
- Ståhl, I. 2002. "Simulation Prototyping". In *Proceedings of the 2002 Winter Simulation Conference* (San Diego, Dec. 8-11). IEEE, Piscataway, N.J., 572-579.
- Ståhl, I. 2003. "From 44 to 31 to 28 to 22 and now to 18 – Less Becomes More in GPSS". In Schulze, T., S. Schlechtenweg and V. Hinz (eds.) *Simulation and Visualisierung*. Magdeburg: SCS.

BIOGRAPHY

INGOLF STÅHL is professor emeritus at the Stockholm School of Economics. His chair was in Business Administration, with a focus on computers for decision support, in particular on modeling business processes by simulation. His email address is ingolf.stahl@hhs.se and his aGPSS system is available at <http://www.aGPSS.com>.

COMMERCIAL SIMULATION SOFTWARE FOR MODELING AND ANALYSIS OF VEHICLE POWERTRAIN SYSTEMS

Valerian CROITORESCU¹
Jan ANTHONIS²

University POLITEHNICA of Bucharest
¹Automotive Engineering Department,
313 Splaiul Independentei st., 6th Sector,
060042, Room JC004, Bucharest, ROMANIA
²LMS International
68 Interleuvenlaan,
3001 Leuven, BELGIUM

E-mail: valerian.croitorescu@gmail.com, jan.anthonis@lmsintl.com

KEYWORDS

Powertrain, simulation, modeling, analysis, vehicle

ABSTRACT

The actual development of the powertrains involves more than one concern in regard with strict requirements from the market and from the authorities. A lot of obstacles can appear on the way of obtaining all the benefits that the vehicle needs. The developed model for each of the powertrains' subsystems uses more than one modeling and simulation software platforms. Starting from the different development stages, the difficulty consists in setting up the model and linking it with co-simulation platforms, taking into account most of its limitations and simulation runtime.

The powertrain virtual development represents a challenge to achieve also additional features. It is mandatory not to compromise other vehicle performances during the struggle to develop innovative systems.

Thereby this paper aims to present a modeling and simulation approach used for powertrain development, taking into account different sub-systems integration and using different modeling and simulation software platforms. The paper highlights the multiplicity of possible needed results that are being investigated using modeling and simulation.

INTRODUCTION

The automotive industry is following a continuous growth trying to response in a positive way to all of the market requirements. Also, new European standards will require automotive manufacturers to improve vehicles' systems in order to reduce CO₂ tailpipe emissions, to improve passengers' comfort and safety, to increase dynamic and energy performances and to be able to pursue prudent path towards autonomous driving. The automotive manufacturers are developing an array of other promising technologies in order to complete these and maybe the future requirements.

Being more difficult for incentive responses to customers demands with respect to fuel efficient vehicles, the powertrain development needs new concepts that avoid asking drivers to compromise.

The powertrain development represents a challenge in order to achieve also additional features. It is mandatory not to

compromise other vehicle performances during the struggle to develop innovative systems.

Powertrain development based on modeling and simulation is not a new approach for automotive industry, neither for automotive sub-systems manufacturers. Having its own seniority in terms of powertrain development, the modeling and simulation approach has its own limitations. It is mandatory to take into account the accuracy of the continuous improvement and computational performance of simulation tools.

The vehicle development process, within respect to the above mentioned few requirements, including also energy management, contains more than one level for designing the model. The primary level may be used for developing and optimizing the control strategies. Another level may include only partial physical aspects, trying to find an optimal balance between mathematical algorithms and physical data. Using all the previous levels, it is mandatory to add a more developed level as a need for the detailed physical subsystems.

Using numerical simulation for every design stage, for all the vehicles' systems, it was easily accepted and supported not only by all automotive manufacturers, but also by many engineering services companies that provides their research and results to industry.

The modeling results consist in a virtual mock-up vehicle which is used to simulate different processes during different cycles of functioning. Being a virtual device of a full-sized vehicle it is able to provide at least part of its functionality and enables testing of its architecture. Automotive industry is using virtual mock-ups vehicles as part of their development process. The virtual mock-up vehicles allow a realistic investigation very early in the vehicle development process, using the latest technologies in virtual reality. The virtual mock-up vehicle is a sum of information which describes its behavior. The model configuration, including architecture and parameters, can be easily changed and updated any time it is necessary.

The virtual powertrain development stages represent the key for reducing costs by a significant decrease of the experimental support and test number.

Using simulation allows accomplishing the advanced methods usage, tools and methodologies, which are efficiently integrated through the developments phases, from the early design stage to the final powertrain mock-up.

Simulation, based on numerical modeling, supports development stages by using mathematical algorithms and interpolation to analyze and optimize components and their performances.

Taking into account different methodologies for modeling and simulation and different design and development stages can only be achieved by using multiple simulation software platforms. In order to highline different simulation software and their possibility to be connected for vehicle powertrain development in this paper there were used two different modeling and simulation software platforms: LMS.Imagine.Lab.AMESim and MATLAB-Simulink.

It is obvious the simulation importance before developing the real mock-ups in order to simplify powertrain development stages and lowers the costs.

SIMULATION APPROACH

Engineering processes and developments require innovative approaches to mechanical and mechatronic systems. Current products are becoming more complex, and their complexity is increasing. Most command and control systems, and also the execution of those used as equipments are moving towards to engineering, due to their high complexity and the needed software to control them.

Physical mock-ups are difficult to build and also, they are hard to use due to their increasing costs and development delays. Therefore, as the powertrain development needs to evaluate multiple scenarios, the simulation environments are necessary.

LMS.Imagine.Lab.AMESim automotive applications

LMS.Imagine.Lab.AMESim allows developing and running different multi-disciplinary models of systems or sub-systems to analyze their complex behavior and to offer the possibility to create control systems from the early design stage.

The acronym AMESim comes from Advanced Modeling Environment for performing Simulations of engineering systems.

AMESim is based on an intuitive graphical interface through which the system may be displayed into the simulation model. AMESim is using icons which represent individual components of the systems. These icons are made as symbols that are easy to recognize or as standardized symbols.

AMESim offers a complete suite of 1D simulation, for multi-domain modeling and analysis, used on intelligent systems and forecasting their multi-disciplinary performances. Model components are described using validated analytical methods, which represent real engineering systems from hydraulics, pneumatics, electrics and mechanics. AMESim is widely used in the automotive industry, but many applications from other industries can be built, such as avionics, machineries and other engineering systems.

Building a simulation model requires the use of various specialized tools to access pre-defined components and libraries. The multiple libraries offer different components for which the parameters can be modified in order to respect the needed properties. The libraries offer many components

long before the existence patterns made using CAD software.

The graphical user interface allows connecting various components. It provides an easy to understand sketch of a complex system which allows in deep investigations for different design possibilities. Once the final sketch is done, it is possible to simulate the system, aiming the specific work stages process.

Using AMESim it is mandatory to follow several work stages: after selecting the required libraries and creating new categories, the users can add text, description as text, images, shapes, icons link etc. Behind each of the icons there are mathematical descriptions of the processes. The mathematical equations are defining dynamic behavior of various engineering systems, being implemented as simulation codes. The functional characteristics of each component are predefined, but it is possible to be set by each demand, being easy to change. The model is built up of several components, based on the developed mathematical equations. The construction of the model for simulation, basically consists of linking sub-models from various software's libraries. The sub-models are shown as icons. After linking the icons (which represent each component of the vehicle) in sketch mode and choosing the proper sub-model in sub-model mode, the parameters will be set. Each icon covers a fragment of C code, written using the specific equations for the system. After initializing the simulation there is possible to display multiple graphs, 2D and 3D, according to the selected variables.

An example of modeling that can be achieved using AMESim is represented by a clutch controller (fig. 1). The controller allows connecting and disconnecting the transmission according to the primary shaft rotary speed of the gearbox. This controller is using mathematical function that enables smooth coupling and uncoupling.

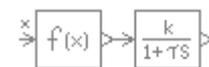


Figure 1: Clutch controller

It is possible to build more complex controls that can be saved as sub-models in the existing libraries or that can be part of new libraries. These sub-models will have the form, the appearance and as many ports as the user needs. An example of this kind of controller in presented in figure 2. This controller is used for a continuous variable transmission.

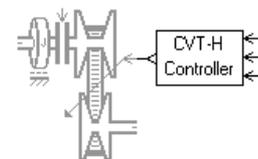


Figure 2: Continuous variable transmission and its controller

Starting from simple controllers it is possible to build more complicated ones, as the electronic control units for powertrains. Powertrains can be studied in detail using AMESim, being able to define, analyze and validate models for energy management, chosen architecture, components sizing.

A powertrain configuration is presented in figure 3, in which the electronic control unit was taken from the predefined libraries, being built by IFP (Institut Français du Pétrole). It allows powertrain functioning analysis. Each individual component can be studied also, taking into account the control, due to advanced real-time simulation interface.

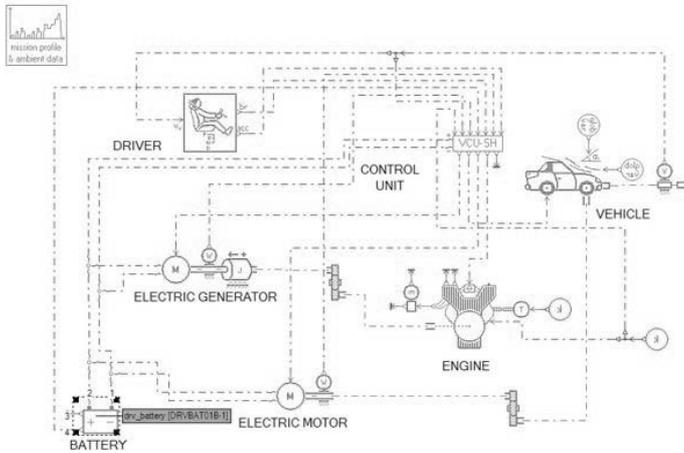


Figure 3: Powertrain configuration with control unit

Vehicle dynamics are also able to be investigated. By achieving appropriate models for the braking system, suspension, steering etc. there is an allowance to integrate and to control sensors and actuators, with different levels of detail that can easily make the transition to advanced real-time simulation (hardware in the loop) (fig.4).

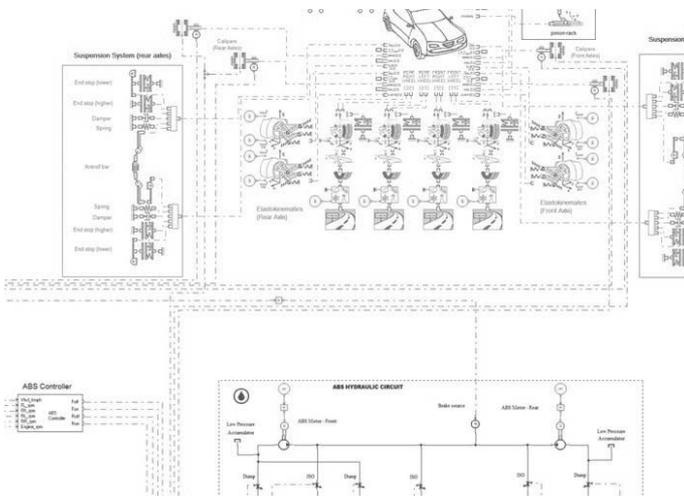


Figure 4: Vehicle dynamics analysis model with real-time simulation interface

A very complex model was developed to analyze and further optimize an unconventional powertrain. It consists in a hybrid powertrain configuration, that is using more than one controller and an electronic control unit especially designed. The electronic control unit works also as an interface between AMESim model and Simulink model (fig.5). Further details about this interface will be presented during **Co-simulation using AMESim and Simulink** chapter. The powertrain configuration includes beside the electronic control unit the driver, the vehicle, the transmission, the engine, the electric motor, the energy storage devices and auxiliary systems having their own parameters. The

parameters used in the simulation are taken from existing vehicle and components.

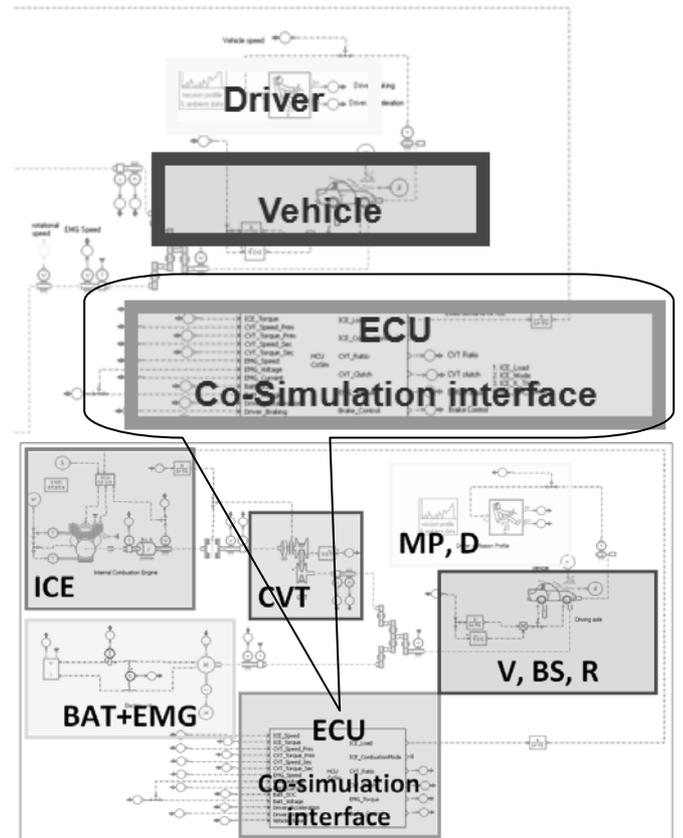


Figure 5: Complex powertrain configuration with co-simulation interface (ICE-internal combustion engine, CVT-continuous variable transmission, MP-mission profile, D-driver, BAT-battery package, EMG-electric motor/generator, ECU-electronic control unit, V-vehicle, BS-braking system, R-road)

Using a pre-defined mission profile for setting the vehicle control speed with regard to the selected driving cycle, it is possible to input the ambient data (wind speed, density and ambient temperature) as real parameters. Acceleration and brake control take into account the anticipation of the speed control.

The driver used in this model controls/calculates braking, acceleration and gear shifting using the vehicle speed. The driver is not able to calculate the gearbox ratio or clutch control.

The Electronic Control Unit (ECU) performs regulations for idle speed and maximum speed. Also the ECU controls the batteries (the state of charge), analyzing them in order to minimize the energy consumption. The ECU contains two different functional modules/levels, based on power demand and the hybrid mode selection. The ECU estimates the load demand based on current load due to road conditions, mass, speed, inertia. The hybrid modes are selected from predefined operating modes, based on power demand, driver's request and vehicle state.

The model for the electric motor can be used as an electric motor and a generator, being independent from the technology of the machine and converter. It is using the torque reference, the battery voltage and the rotary velocity for being able to compute the efficiency tables. It is

controlled by the ECU to deliver the motor torque and the losses.

The battery is an internal resistance model, which characterizes the battery as a voltage source and an internal resistance. The battery output voltage is calculated taking into account the equivalent internal resistance and the input current.

The transmission includes a continuous variable transmission to get the engine torque and to deliver it to the wheels.

The vehicle model is a simple vehicle load when considering no longitudinal slip between the tire and the ground.

MATLAB Simulink automotive applications

Simulink is a multi-domain simulation environment used for design of dynamic systems and integrated modeling based engineering. It provides a graphical working environment, interactive and customized set of libraries that allow users to design, simulate, implement and test time-varying systems, including communication elements, control, signal processing, video processing and images processing.

Being classified as general interest programming software [194], Simulink allows adding and expanding its capacity to more standard modeling domains by adding adjacent products. Simulink widens its availability for design, implementation, verification and validation.

Simulink is part of the MATLAB programming environment, providing access to a range of tools that enable the development of algorithms in order to analyze and visualize simulations, create packet processing scripts, customize the design environment and signals definition, parameters and test data.

Simulink provides a number of key features. Along with the ability to use pre-defined libraries and add other new libraries, containing new components, the interactive graphical editor helps for assembling and managing intuitive block. By using Simulink, there can be managed complex projects and there can be divided models into hierarchies of design components. Simulink menus offer the opportunity to be configured as needed.

Each Simulink developed model consist in an associated programming code that helps and supports the ability to operate via interfaces to other simulation and modeling environments. Simulink can access the functional blocks of MATLAB algorithms to be used for embedded systems. Simulink simulation has several modes and different ways of solving patterns. Simulation modes can be customized by changing the needed simulation parameters.

After completion the simulation stages Simulink offers the opportunity to examine the simulation results, to diagnose the performances and the unexpected behavior of the model, through which it has complete access to MATLAB. Also, there are several tools available for the needed analysis to ensure model consistency and identify possible modeling errors.

MATLAB codes can be imported into Simulink. Thus, MATLAB may be used for data analysis. In addition, the MATLAB code can be used to design integrated algorithms that can be implemented later with the rest of the generation model code (fig.6). There can be used codes and languages

developed in C / C + +, Fortran, etc. directly into the model, allowing to create custom blocks in the model.

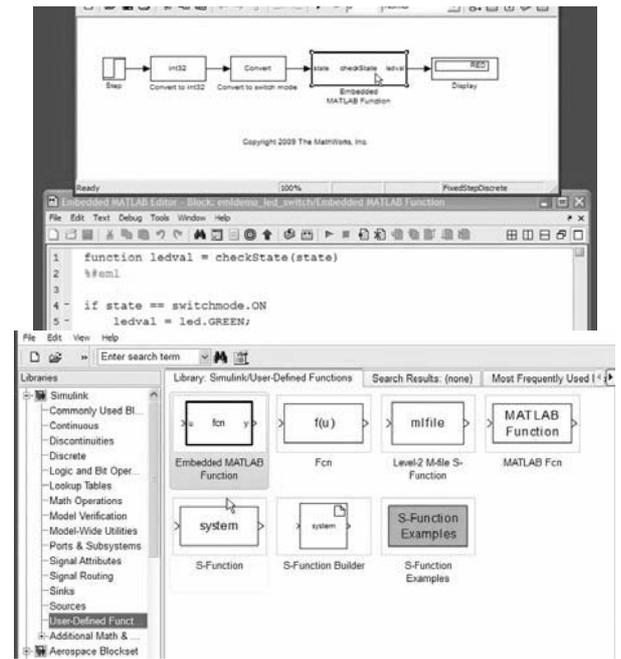


Figure 6: MATLAB Simulink algorithms integration

The models are organized following clear hierarchical levels, manageable subsystems and using reference models (fig. 7). Subsystems contain groups of blocks and signals grouped into a single block. Each subsystem can have its own graphical interface user-GUI set by the user, which hides the contents of the subsystem and change its image.

The model can be divided into subsystems. It is possible to model and simulate their behavior independently. Subsystems or components can then be saved and used in other models (fig.8).

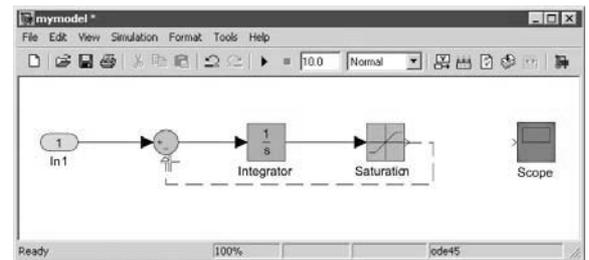


Figure 7: Simulink example model

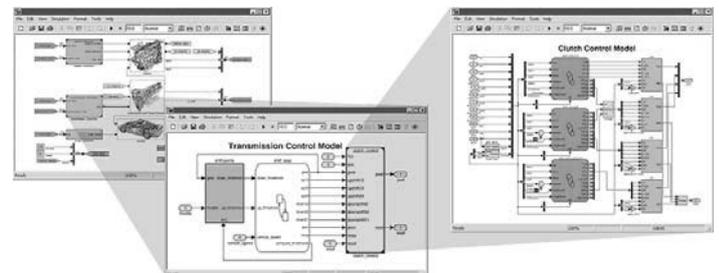


Figure 8: Vehicle model, sub-systems and blocks

Co-simulation using AMESim and Simulink

Both simulation environments allow special interconnection interfaces, being able to achieve real time simulation. The AMESim models can be easily imported and connected in Simulink environment and vice versa. Building an interface it is possible to use both solvers from AMESim and Simulink.

Co-simulation (as long as both environments are used to achieve the simulation-solver's together at the same time) is used as the simulation methodology that allows individual components and systems to be simulated by different simulation tools running simultaneous and exchanging information in a collaborative manner.

Several possible options for co-simulation are based on the technical requirements imposed by the simulation model closely related to be simulated. Thus, if the model created in Simulink is very complex, perhaps the best option is not to be imported into AMESim for simulation. If the model developed in AMESim is a requirement, the best option is to import the model created in Simulink into AMESim sketch. It is possible to develop the control unit in Simulink, having the physical model developed in AMESim. At the same time, it is necessary to be guided by rules. As an example: if the AMESim model is more complex or if certain features are required, the Simulink model will be imported and vice versa.

The co-simulation interface consists in the electronic control unit that was designed in Simulink (fig.9).

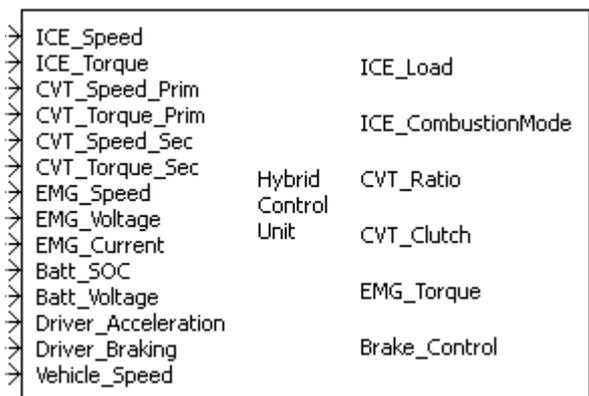


Figure 9: Co-simulation interface

The electronic control unit is designed in a modular manner. It is a separate model developed in Simulink, running on a computer in real time. It contains two working levels. The first level estimates the power demand and calculates the load demand in order to satisfy the proper driving conditions (road conditions, mass, inertia, speed, etc.). The second level switches the operation mode from the six available operation modes using the power demand, the vehicle status, the driver's demand and the requirements of the transition between modes. The electronic control unit includes several different controllers for the internal combustion engine, the electric machine, the continuously variable transmission and the braking system in order to achieve regenerative braking. Power demand is influenced by the driver and by the state of operation, taking into account the vehicle parameters. Based on this information, the current power demand (power needed to achieve current status driving) and power demand

(the difference between the power required to achieve the current status of the vehicle and the power required by the driver) are estimated and are used by the controllers and the operation modes.

The operating modes will be selected taking into account the driver's requirements with regard to the acceleration and braking, vehicle speed, battery charging status and power demand.

RESULTS

The approach used in this paper allows investigating and identifying the vehicle dynamics behavior – speed, acceleration, longitudinal slip, braking performances etc., fuel consumption, harmful emissions, battery state of charge, current consumption and many other needed parameters, defined as needed results from the beginning.

Simulating different models enables the user possibility to investigate a multitude of parameters.

In order to validate the model it is possible to compare the resulted vehicle speed curve with the control speed curve (fig. 10). The small differences between the control speed curve and the resulted speed curve shows that the developed model is quite close to the ideal model.

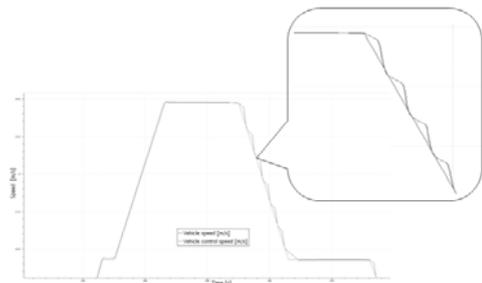


Figure 10: Resulted vehicle speed and the control speed

After simulating the models, the results that may be included on the same plot are the vehicle velocity, the battery state of charge and the fuel consumption (fig. 11). There can be discussed in the same time the three dependencies and the link between them. It can be easily noted that after more than half of the predefined cycle, the engine started and the powertrain went from electric drive to hybrid drive.

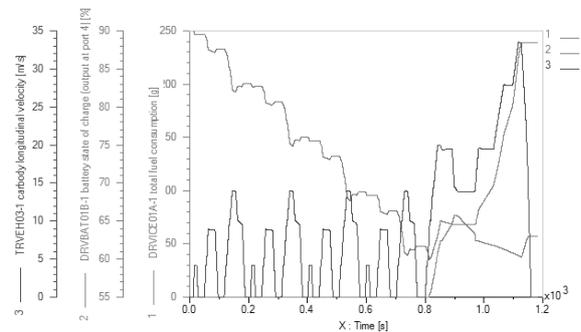


Figure 11: Total fuel consumption [g] / Battery state of charge [%] / Vehicle linear velocity [m/s] during a predefined driving cycle

Starting from two simulation models in which there are presented two different powertrain architectures, one of them a hybrid vehicle equipped with a gasoline engine and the

other a hybrid vehicle equipped with a Diesel engine, there can be presented the global efficiency curves (fig. 12). Both models were developed in AMESim. It can be noticed that the hybrid architecture equipped with the gasoline engine is more efficient during the predefined cycle than the Diesel architecture.

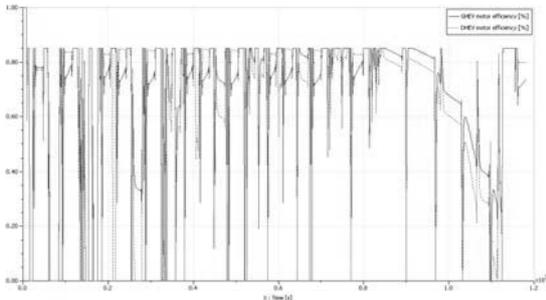
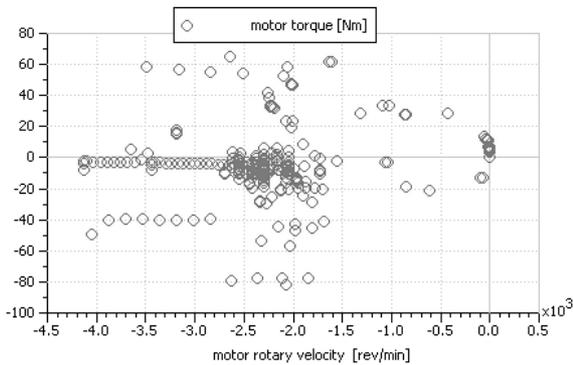
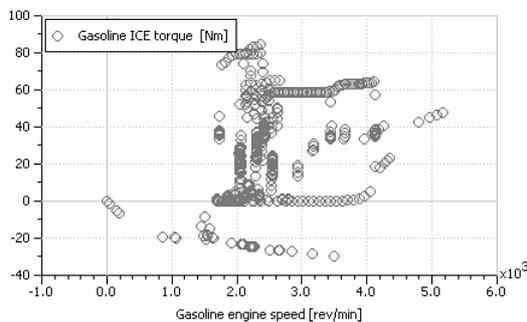


Figure 12: Global efficiency curves

A model that represents hybrid architecture is suitable for investigating the dynamic behavior and the energy parameters for the electrical motor and for the internal combustion engine (fig.13, fig.14). In figure 13 there are presented the torque evolutions with rotary velocity of the internal combustion engine and the electric motor, while in figure 14, there are presented the vehicle velocity, the engine power and the electric motor power during a predefined driving cycle.



- a -



- b -

Figure 13: a- Electric motor torque
b- Internal combustion engine torque

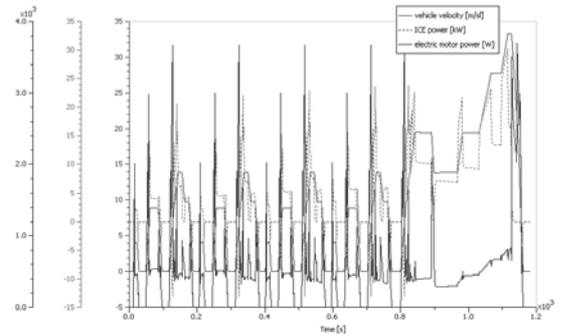


Figure 14: Vehicle velocity, engine power, electric motor power during predefined driving cycle

Complex models are difficult to investigate. In order to determine both dynamic behavior and thermal behavior of some components, it is recommended to develop functional models and thermal models separately for each component. The temperature evolution can be presented individually for each investigated component only if it has its own thermal model (fig.15). The heating and cooling processes for the some components during operation represent high interest for the energy losses.

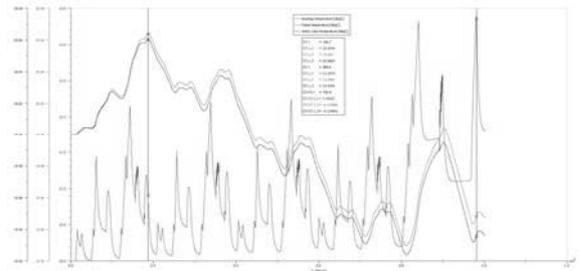


Figure 15: Temperature evolution for different components of the powertrain (stator's bearings, frame and stator from a electric motor used on hybrid architecture)

For a hybrid configuration there is also possible to investigate both thermal behavior and energy parameters for the batteries (fig.16).

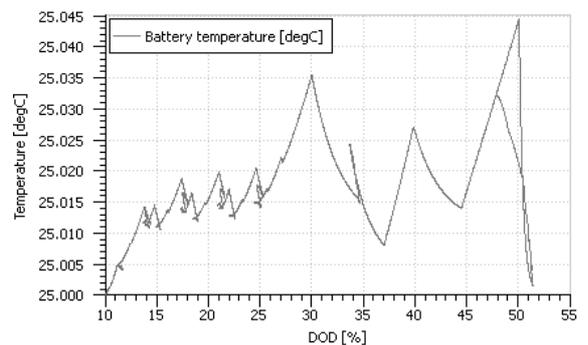


Figure 16: Battery temperature evolution with depth of discharge

For evaluating the energy parameters of the batteries there can be also investigated the state of charge, the power losses (as presented in Croitorescu 2013, based on thermal interpretation) while the vehicle is running during a predefined driving cycle (fig.17).

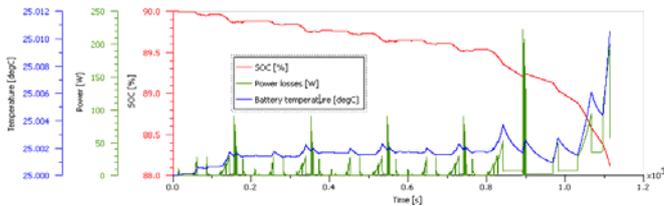


Figure 17: Battery state of charge, power losses and temperature

The above presented results are just a few from the multitude of possible investigations that may be done using AMESim platform and Simulink platform.

Starting from this information, the users are encouraged to develop more detailed models and to complete with more components the existing libraries from AMESim and to disseminate their contribution.

DISCUSSION AND CONCLUSIONS

The objectives of the paper includes, beside the engineering impact of powertrain architectures and their prerequisite control, the high demand for modeling and simulation in powertrain development using existing tools and environments.

Starting from known limitation that modeling and simulation approach has, the virtual powertrain development is able to optimize the behavior before the experiments will go on and to perform realistic tests using real-time platforms. Taking into account the model accuracy, the models are considered to be far from the real operating conditions. The engineers' requirements must be fulfilled especially at the latest development stages, such as final calibration for the engines or for the operating modes. The powertrain development mixes a lot of technical areas, using multiple domains. Dedicated simulation software platforms are able to meet all the requirements from the multiple domains, but it is difficult to couple and to investigate different components from a complex model. The modeling and simulation environments allow to develop from simple to complex models and to easily be used by others who are willing to develop them more. Simulating the models meets constraints that may create perturbations. The simulation time represents one of the important constraints, being relative depending on the models and their complexity.

The modeling and simulation approaches are reaching many design possibilities.

Using simulation it is possible to reach important information with regard to powertrain electrification. Therefore, this design allows reducing the severity of the trade-off between fuel consumption and performance, with marginal cost increases.

Modelling and simulation allow creating detail models using different approaches that can use form geometry design to mechanical functioning. All models have to be able to analyse different states for specific components and systems. Modelling and simulation provide facilities and services for a variety of engineering domains, covering some important tasks needed in the development mock-up phases, as: preliminary dimensioning for the powertrain architectures, based on performance, fuel economy, energy consumption, component dimensioning (internal combustion engine and

the electrical machine, heaters and coolers, batteries, transmissions etc.), numerical validation for specific used solutions (exhaust recirculation, superchargers, energy recovery devices, heat management, electrification systems etc.) and for real components (sensors, actuators, controllers etc.), and for hazard avoidance (explosions of batteries used on powertrains,)

Simulation and modeling are efficient to be used for Model in the Loop (MiL) and Hardware in the Loop (HiL) platforms for vehicle synthesis, design, calibration and control.

ACKNOWLEDGEMENT

The present paper is mainly a part of first author's PhD Thesis, being accomplished during his training period with full support of LMS International, Engineering Services.

REFERENCES

- Anthonis, J., Santos, F., Croitorescu, V., Van der Auweraer, H., "Multiscale thermal and NVH models for EV/HEV Integration of an SR-based Drivetrain", Proceedings of VPC, Pune, India 2011
- Aarnoutse, P., Kenda, R., Van der Auweraer, H., De Bruyne, S., Moncel, T., "Hardware in the Loop testing of a CVT for a Hybrid Vehicle", Virtual Powertrain Conference, 2010
- Belanger, J., Venne, P., Paquin, J.-N., "The What, Where and Why of Real-time simulation", Opal-RT, 2010
- Croitorescu, V., Anthonis, J., Vasiliu, N., "Thermal Modeling of an Electric Motor Used On Road Vehicle Powertrain", Proceedings of 10th ISC, Brno, Czech Republic, 2012
- Croitorescu, V., "Modern Drives Using Unconventional Energy Storage Devices – Hybrid Electric Vehicles", PhD Thesis, Bucharest, Romania, 2012
- Croitorescu, V., Anthonis, J., Vasiliu, N., "Thermal Modeling of an Battery Package Used On Road Vehicle Powertrain", Proceedings of 11th ISC, Ghent, Belgium, 2013
- Guzella, L., Sciarretta, A., "Vehicle Propulsion Systems – Introduction to Modeling and Optimization", Second Edition, Edition Springer Berlin Heidelberg New York, 2007
- Kain, S., Schiller, F., Dominka, S., "Methodology for Reusing Real-Time HIL simulation Models in the Commissioning and Operation Phase of Industrial Production Plants, Intelligent Mechatronics", ISBN: 978-953-307-300-2, 2011
- Miller, J.M., "Propulsion systems for hybrid vehicles", Institution of Engineering and Technology, Power and energy series 45, London, United Kingdom
- Sanchez, P.J., "Fundamentals of Simulation Modeling", Proceedings of the 2007 Winter Simulation Conference, Washington, 2007
- Van der Auweraer, H., Santos, F., Wickaert, K., Mas, P., Anthonis, J., "Vehicle integration of advanced hybrid and electric powertrain concepts", EEEV, Brussels, Belgium, 2011
- ELVA Project, Advanced Electric Vehicle Architectures - Societal scenarios and available technologies for electric vehicle architectures in 2020, ELVA Consortium 2011
- LMS Imagine.Lab AMESim – User's Guide, 2010
- MATLab 7.11, R2010b – User's Guide, 2010

WEB REFERENCES

- <http://www.systems-thinking.org/modsim/modsim.htm>
- <http://www.simulation-research.com>
- <http://zone.ni.com/devzone/cda/tut/p/id/10343>, Hardware-in-the-Loop (HIL) Test System Architectures

BIOGRAPHIES

VALERIAN CROITORESCU earned his Automotive Engineer Degree in 2007, as valedictorian. He prepared the diploma project in France, inside Ecole Nationale D'Ingenieurs De Tarbes. He also attended two master of sciences programs, at University POLITEHNICA of Bucharest: `Efficiency and Security in Automotive Engineering` and `Environmental Management`. His academic records include numerous awards and certifications. In September 2007 Valerian joined the academic staff of Automotive Engineering Department. In 2008, he founded "autojobs.ro", a specialized web portal in automotive industry having the goal to help and provide job seekers and employers the opportunity to be connected, followed by "aerojobs.ro" in 2012, aviation dedicated jobs web portal. In 2012, he earned a Ph.D. Degree at University POLITEHNICA of Bucharest, concerning Hybrid Electric Vehicles Development. A part of his Ph.D. thesis has been accomplished during a research stages at LMS International.

JAN ANTHONIS received the M.Sc. degree in mechanical engineering from the KU Leuven in 1994, and the PhD degree in 2000. From 2001 till 2007 he was post doctoral researcher at the Fund of Scientific Research in Belgium, developing researches on control and mechatronics with application in agro-machinery and agro-industry. From 2003 he is part time lecturer at the KU Leuven. In 2007 he became employee of LMS International, a spin-off company of the KU Leuven, founded in 1980, specialized in noise, vibrations and mechatronics in automotive, aerospace and other mechanical industries. Here he manages a research team on control and mechatronics.

ACADEMIC TOOLS

SOFT COMPUTING TOOLS FOR THE ANALYSIS OF COMPLEX PROBLEMS

Christina Klüver and Jürgen Klüver
Institute for Computer Science and Business Information Systems
University of Duisburg-Essen
45117 Essen,
Germany
E-mail: {juergen.kluever|c.stoica-kluever}@uni-due.de

KEYWORDS

Soft Computing, Shell, Tool, Cellular Automata, Evolutionary Algorithms, Fuzzy-Expert-System, Boolean Network, Neural Networks

ABSTRACT

We show a complex whole tool for the application of nature analogous or Soft Computing techniques respectively. The idea of this tool is to enable users to apply a certain nature analogous technique to his/her specific problem without having to construct an own program. The tool consists of five different shells, namely a shell for different Evolutionary Algorithms, a shell for Cellular Automata, a shell for Boolean Networks, two for Artificial Neural Networks, and a Fuzzy-Expert-System shell. The shells allow the users to insert not only different parameters but also problem specific rules that the users can construct themselves or take over from sets of predetermined rules and functions. The usage of these shells is demonstrated with two examples: The first is a cellular automaton model of a reminder department in large firms as decision support for the responsible managers how to optimize the reminding process; the second example is the selection of a suited model of procedure for project management.

1. INTRODUCTION

“Soft Computing” is a general name for rather different modeling techniques that probably better should be called “nature analogous” techniques (cf. e.g. Klüver et al. 2012). The common characteristic of these techniques is that they are heuristically orientated to certain natural models, in particular biological ones. Usually the general category Soft Computing contains modeling techniques like Cellular Automata (CA), Boolean Networks (BN), Evolutionary Algorithms (EA), Simulated Annealing (SA), which has its model in thermodynamics, and Artificial Neural Networks (ANN). The usage of Fuzzy methods in combination with expert systems is practically also always included.

The advantages of Soft Computing methods are on the one hand that several of them are logically equivalent to potential universal Turing machines, i.e. that they can be used to model any system that can be computationally modeled at all. Hence these methods can be used in a very general manner: Often it is necessary to apply for different problems different techniques. In contrast it is frequently possible to use for different problems, e.g., just the same type of certain neural networks. On another hand the mathematics of these modeling tech-

niques is in all cases comparatively easy to understand. One must not be a mathematician, physicist, or computer scientist to understand the basic logic of, e.g., neural networks or cellular automata. The intellectual task of a user, however, must always be to “translate” his problem into the logical structure of the according Soft Computing method the user will work with. Yet the user can then concentrate on his problem and not on a complicated mathematical technique.

A third advantage is that Soft Computing techniques allow a so-called “Bottom up” procedure. This means that the problem on hand should be decomposed in its logical units; these units then are combined according to the problem by the different rules of the Soft Computing technique. For example, if one has to compute the traffic flows on a highway in order to experiment with different solutions for avoiding traffic jams the single cars could be represented by the cells of a cellular automaton (e.g. Esser and Schreckenberg 1997; Stoica-Klüver and Klüver 2007); the rules of such a CA then determine the allowed velocity of the cars, the bans of overtaking if there are too many cars, and so on. An important parameter of an according simulation would be for example the number of cars on a certain length of the highway. Bottom up models are frequently easier for a user to construct than the usage of other techniques.

An additional possibility is the coupling of two or even more different techniques, namely the construction of so-called “hybrid systems” (Goonatilake and Keel 1995). For example, if one has a CA model of a certain domain, e.g. a social organization, and if one wants to improve this system then it could be coupled with an evolutionary algorithm to optimize the rules or other parameters of the CA. Yet although we have several times constructed such hybrid systems ourselves the tools described in the next sections “only” contain the different basic techniques; we intend to enlarge our tools with the possibility to couple them.

Our experiences with students of different fields like computer science, business administration, business administration technology, and communication science always were that the students could rather quickly understand how to model their specific problems with suited Soft Computing techniques. However, constructing own programs based on a specific technique is not always easy even for students of computer science. That is why we developed the different tools for each of the mentioned techniques to enable users constructing their own models without having to learn how to program their own software.

2. TOOLS AND TECHNIQUES

Our tools were developed in C# and JAVA as shells, i.e. as a frame for the users to base their modeling on the respective techniques. This means that the user, if he e.g. wants to use the optimization technique of a genetic algorithm, the user can define his parameters and sub algorithms (see below) and insert them into the shell. In addition he can insert also the size of his problem and so on. This can be made clearer by describing the different techniques and hence the tools that allow to use these techniques. By the way, we were of course not the first who developed shells for Soft Computing techniques – some web-references are given as examples. Yet as far as we know only our whole tool allows using all the different techniques with a simplified formal representation of the rules, or (fitness-)functions respectively.

2.1 NAOP – A Shell for Optimization Algorithms

The shell that contains the possibilities for constructing evolutionary algorithms is called NAOP – Nature Analogous Optimization Procedures – and consists strictly speaking of different sub shells, namely a special shell for Genetic Algorithms (GA), a shell for Evolution Strategies (ES), another shell for Simulated Annealing (SA), and two shells for two new evolutionary algorithms, which we developed, namely the so-called Regulatory Algorithm (RGA), and the Lamarck Algorithm (LA). When opening NAOP the user selects which technique he wants to use (Fig. 1). We shall describe the usage of NAOP with the example of the GA, which is the most known evolutionary algorithm.

A GA as all evolutionary algorithms simulates the biological evolution, namely the variation of the genome by mutation and recombination and by evaluating the results of the varying processes via a so-called fitness function or evaluation function respectively. Hence the usage of a GA demands that the problem must be represented as an artificial “genome”, usually a vector with numerical components. Frequently this vector is called an “individual”. The first step of the operations of a GA is to generate a “population”, i.e. a set of such vectors. Afterwards the elements of the population are changed by a) the recombination of two or more vectors, usually called “crossover”, and b) by mutation, namely the varying of one or several elements of the vectors. In orientation to the biological model the initial vectors are usually called the “first generation” and/or “parents”; the vectors resulting from crossover and mutation are called the “next generation” or “children” respectively. These children are evaluated via the fitness function, a part of the best elements are varied again, evaluated, and so on until either a predefined result – an optimum – has been reached or if the variations do not change the elements any more. In this case the GA has reached an attractor in the solution space.

Although the basic logic of a GA is rather easy to understand there are a lot of parameters a user has to take into account. The first step is the decision about the size of the vectors that represent the according problem and the specific coding of the components – binary, real numbers, or alphabetic and alphanumerical coding. The shell offers the according option that the user wishes. The second step and usually the most difficult one is the construction of an adequate fitness function. For example, if the user wishes to deal with the traveling salesman problem (TSP) then an adequate fitness function

would be the length of the route represented by a specific order of the components, i.e. the formal representation of the towns. The shell offers three possibilities for constructing an adequate fitness function:

a) The user selects a simple fitness function from a set of predefined functions. b) The user composes an own function from “building bricks” that the shell offers. c) For users, which are able to program in C#, the shell gives a platform to write the own fitness function, which is automatically coupled to the user’s GA.

The next steps are decisions about the two “genetic operators”, namely mutation and crossover. The user must determine a “mutation rate”, namely how many elements of a vector should be varied; in the case of real and alphabetic coding the user must define the sort of mutation he wishes. The shell offers the according options. In the case of crossover a “selection schema” must be chosen, i.e. which vectors should be selected for crossover – for example only the best ones and if so how many -, the “roulette wheel procedure”, and other possibilities; in addition the determination of a “marriage schema” is necessary, namely how many components of two or more vectors and which ones should be included when recombining the vectors by crossover. The shell offers several options for these two schemas.

The shell in addition enables the user to construct an “elitist” version of his GA. This means that the user can decide if several parents should be taken over in the next generation, in particular if some parents are still better than the best children. The user can determine those parents, i.e. how many and which parents should be preserved. Again the shell offers several options for this decision.

For the sake of brevity we name only these most important components of a GA although there are sometimes additional parameters, which must be considered; the shell allows this too. The other evolutionary algorithms the whole shell contains can be used of course the same way, i.e. the specific shell offers options according to the particular characteristics of the specific EA. We included in NAOP also Simulated Annealing, as we already mentioned, although it is strictly speaking not an evolutionary algorithm but a “thermodynamical” one. The chief parameters one has to take into regard when constructing a SA are offered by the SA shell the same way. Figure 1 shows the first page of NAOP:

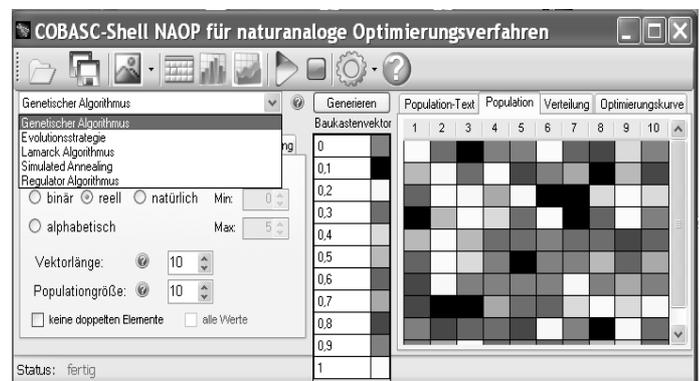


Figure 1: The introductory page of NAOP

At present we are working on an English version for NAOP.

2.2 The COBASC CA-Shell¹: A frame for constructing cellular automata

Cellular Automata (CA) are basically a set of “cells”, usually ordered on a two-dimensional grid. It is also possible to construct one- and multi-dimensional CA but the two-dimensional CA are by far the most used. Each cell on the grid, mostly visualized as a square, has a certain number of cells as “neighbors”, which are the “neighborhood” of the cell. A cell in form of a square has either a “von Neumann neighborhood” (named after John von Neumann, the inventor of this technique), namely the four adjacent cells at each side of the square, or a “Moore neighborhood”. In this case the neighborhood consists of the four cells adjacent at the sides and in addition the four cells at each corner of the first cell. Other neighborhood definitions are possible and of course the cells must not have the form of squares, but these mentioned definitions are those mostly used.

The basic logic of a CA is the following: Each cell is in a certain state, usually represented as a numerical value or a combination of different values. Hence mathematically a state is either a scalar or a vector. For example, if one wants to model a predator-prey system by a CA the state of a cell consists of different sub states like “predator” or “prey” or “plants”, “sex”, “age”, “state of hunger”, and so on. The so-called “transition rules” of a CA are logically of the form “if the cell in the center of a neighborhood is in state A, and if the cells of the neighborhood are in states B, C, ..., then the center cell changes its state into state F, or remains in state A”².

The transition rules always determine the state of the center cell for the next time step in dependency of the neighbor cells. Sometimes the rules must take into account each state of the neighbor cells separately. Alternatives are “totalistic” rules that aggregate the states of the neighbor cells, for example by summing them up. A totalistic transition rule then is e.g. of the form “if the center cell is in state A and if the sum of the neighbor states is equal or larger than X then the center cell goes into state F; else it remains in state A.” The rules can be deterministic ones or stochastic, i.e. they operate only with a certain probability.

The geometry of a CA is symmetric, i.e. each center cell is also a cell of the neighborhood of its neighbor cells. Hence the dynamics of a CA is in many cases very complex; one might say that CA represent a very transparent form of non-linearity (cf. Holland 1998). That is why particular CA are equivalent to universal Turing machines, i.e. each computable system can be modeled with a suited CA, despite its basic simplicity. In particular CA are an illustrative example for bottom up modeling.

The CA-shell gives the possibilities to construct two-dimensional CA (Fig. 2). We restricted the shell to this case because one- and multi-dimensional CA are mostly interesting only for pure basic research; the shell is for practitioners who want not or cannot program their own models.

In the first step of the usage of the shell the user has to determine the number of cells he wishes for his model – in princi-

ple as many as he wishes. The grid is always a square and hence the total number of cells is always a square number. Yet because the user can determine “empty” cells, i.e. cells that are in state 0 and therefore do not represent the units of the model the user is free to introduce an arbitrary number of units. The shell offers the option to use a von Neumann neighborhood, a Moore neighborhood, or an “extended” Moore neighborhood, namely defining up to 48 cells around the center cell as neighborhood. The user can decide if the cells that represent certain units of his model should be distributed at random on the grid or if he wants manually to insert these cells on special places of the grid. In addition the user can choose between a “synchronic” and a “diachronic” modus that determine the order in which the cells are taken into account when applying the rules.

The next step is the definition of the different possible states of the cells. The number and kind of states of course depend on the problem of the user. He is free to define the states as scalars or as multi-dimensional vectors. To make the CA logic easier for a user the shell advises first to define a (main) state, like e.g. “predator”, and afterwards sub states as “attributes” like “age” or “sex”.

The third step and usually not only the most important but also most difficult one is the definition of the transition rules. The user can choose between deterministic and stochastic ones. If he chooses the second option he has to insert the probability values for the different rules. The user can determine a general probability for all rules but also a specific probability value for each different rule. If the user needs totalistic rules the shell offers a “summing up” rule as described above. To be sure, this rule can be used also for defining totalistic rules using the mean value of the neighborhood cells.

The rules can be defined as logical conjunctions, i.e. taking into regard several different conditions that must be fulfilled, as logical disjunctions, and as “parenthesis” rules, namely combinations of conjunctions and disjunctions, for example “if $P \wedge (Q \vee R)$ then S”. If a user needs for his model many different rules it is sometimes difficult to see if the rules are logically consistent. Therefore the shell offers a “consistency check”, that is the information if there are logical contradictions between different rules, and if so, which rules contradict each other.

The next figure shows a two-dimensional CA after loading an existing file:

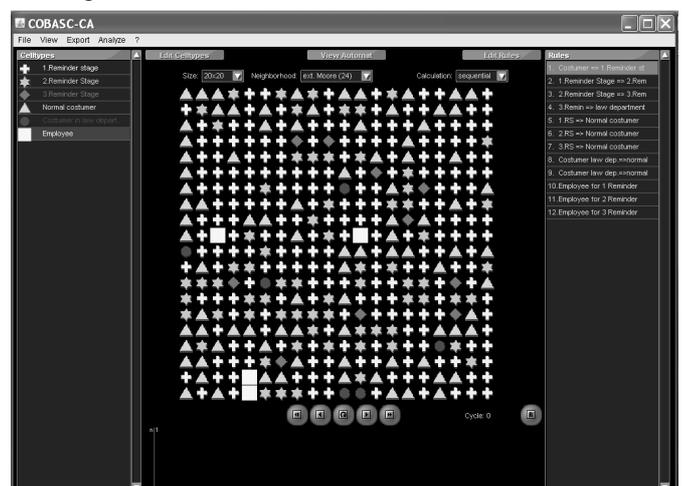


Figure 2: An introductory page of the CA-shell

¹ COBASC - Computer Based Analysis of Social Complexity - is the name of our research group that developed the different shells with several students of us

² The „Game of Life“, one of the most famous CA, gives an easy introduction into this logic; many examples of this CA can be found in the Internet.

Figure 2 shows the initial state of a model of a “reminding department”, i.e. the department of a firm that has to deal with customers who do not pay on time. The members of the department are represented by white squares; the other cells represent customers in different stages of reminding. The CA-shell has in the meantime since his construction been used by many users, in particular by students of business administration. We show below the example of a CA model constructed with the shell for analyzing the organization of sending reminders to customers of a firm.

2.3 The COBASC Boolean Shell

This shell gives a frame for the construction of Boolean Networks (BN). Boolean or logical networks are basically an extension of CA: The units or knots respectively of a BN are comparable with the cells of a CA; unlike the uniform neighborhood of a CA the geometry of a BN can be defined as inhomogeneous and asymmetric: The geometry might be inhomogeneous because a unit can be determined by two other units, only one unit, or more; in principle these determinations might be different for each single unit. The geometry might be asymmetric in contrast to the symmetric neighborhoods in CA because the neighborhood relation in a BN frequently is asymmetric, i.e. if unit A influences unit B then the converse is not necessarily true. The geometry of a BN is usually represented in an adjacency matrix. In most cases BN are binary coded; the rules of transition that are generally similar to those of CA are in this case basically the logical operators from propositional calculus.

The user can define his model by choosing the number of units, the number of the unit’s states and the specific transition rules. As the connections in a BN define the topology of it the user has to fill out the according adjacency matrix where the specific topology is represented; the shell offers of course an empty matrix (Fig. 3).

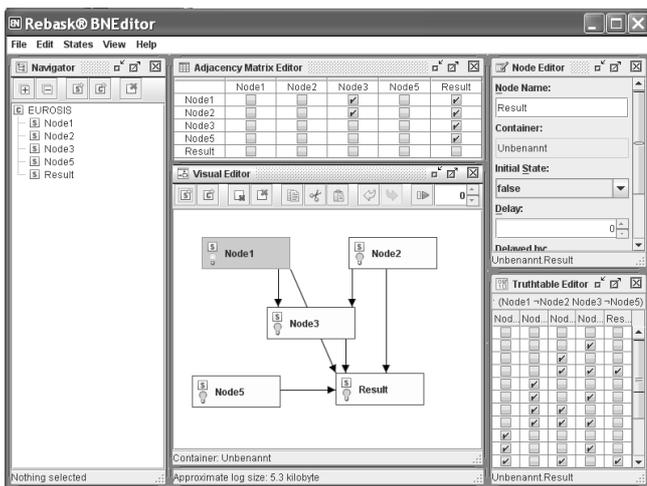


Figure 3: An introductory page of the Boolean shell

The figure shows a BN with five units; the initial state is in this case defined as an external activation of node 1.

In particular the user can define certain units themselves as a small BN: A BN unit may contain different sub units that themselves are connected and generate their specific dynamics via their own rules of transition. If a BN has some of these units with sub units the dynamics is “slowed down”, i.e. the according unit operates on a connected unit only when its sub

units had been activated and performed their specific “sub dynamics”. This possibility is in particular useful if one wants e.g. to model complex organizational processes where frequently the whole process is dependent on the execution of some sub processes, e.g. some sub projects.

A BN is in particular useful for decision support, e.g. if one wants to know how the logical relations between parts of complex problems depend on and influence each other.

2.4 The Neural Network Shell

Neural Networks (NN) are one of the most known modeling techniques of Artificial Intelligence and are orientated to the model of the brain. They consist of connected units or (artificial) neurons respectively that are in specific “activation states”, usually coded as real numbers. The connections are “weighted” with numerical values; the weights change the information flows between the neurons according to their size. Hence the activation state of a neuron is determined by a) the activation states of the sending neurons – similar as in the case of BN – and b) the varying of these values by the weights. In addition, so-called activation functions determine the exact input for the receiving neurons, for example by summing up the inputs to a neuron.

The most interesting characteristic of NN is their ability to learn, i.e. to change their structure according to the specific problems. As the weighted connections in a NN are represented in a so-called weight matrix, which is basically an extended adjacency matrix, learning of NN is mainly the variation of the weight matrix, i.e. the changing of the topology of the NN. Other forms of learning are possible and have been analyzed but the topology variation is by far the most common method. The topology variation is generated by “learning rules”, i.e. algorithms that compute the variation of the different weights. Frequently a network consists of different layers, i.e. sub sets of neurons that either form the input layer of the NN with the neurons that receive the input from the user or the output layer, which contains the neurons that show the results of the network’s operations. In addition many networks have one or several “hidden layers”, i.e. neurons that are neither input nor output neurons.

There are different types of learning possible. The most important ones are the supervised learning and the self-organized learning type. In supervised learning the network receives a “target”, namely a vector, which determines the learning process: the values of the output layer must become as equal as possible to the values of the target vector’s components as desired result of the learning process. Hence the topology of the supervised learning network must be changed in order to vary the values of the output layer as long as the values of the target vector have not been reached. This kind of learning is used in particular for the case of pattern association, useful for example for automatically recognizing faces or voices; in these cases the network receives as input, e.g., the pattern of a perceived face and has to associate this input with those faces it has already learned and which are most similar to the input.

Self-organized learning is performed without an external feedback like a target vector. The network generates by application of the according learning rules an internal structure, by which the network can order the inputs given to him. One might say that by self-organized learning the NN generates an explicit order of data that were only implicitly ordered before

the operations of the network. The most known example of human self-organized learning is the famous model of Piaget where the learner constructs a cognitive “schema” via the process of accommodation. Accordingly self-organized learning networks are primarily, although not exclusively, used for tasks in the field of data mining and generally data systematizing and classification.

When using the NN shell the user first has to decide if he wants to apply supervised or self-organized learning, i.e. he has to decide which sub shell he wants to use. If he chooses the first option he has to determine the number of neurons, the number of layers, the specific activation functions, and the learning rules the network should operate with. The shell offers different activation functions like the so-called linear function, the sigmoid function and the tangent hyperbolic function tanh. The user receives an advice, if he wishes so, which activation function is suited best for the specific network. Afterwards the user decides which learning rule he wants to use. The shell offers the so-called Delta rule for two-layered NN, the standard Backpropagation rule for three- or more-layered networks, and in addition two new learning rules, namely the Enforcing Rule Supervised (ERS) and a variation ERS 2 that were developed by us.

In the next step the user has to decide if the initial weight values should be generated at random by the shell – possibly within certain limits – or if he wants to manually insert these values. Finally he has to insert a target vector, according to his problem, and has to define a “stopping criterion” for the learning processes. This can be done either by defining a largest number of learning steps or by defining the maximal allowed distance between the output layer and the target vector. In the second case the learning process is considered as successful if the difference between the output layer and the target vector is equal or smaller than the maximal allowed distance (Fig. 4). This is measured by the Euclidean distance.

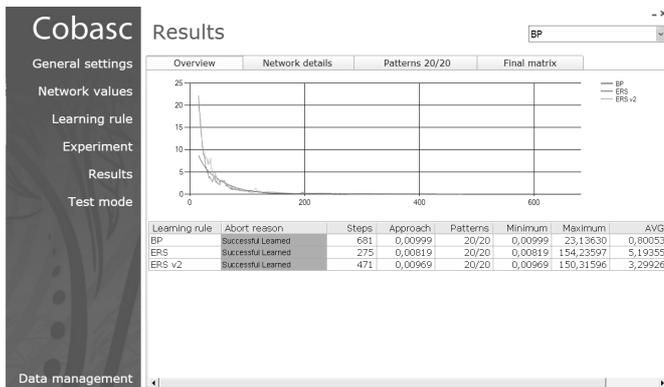


Figure 4: Results of the learning success of 20 Patterns

The task of the network was to learn 20 different patterns, i.e. to associate each input pattern with a specific output pattern. The three curves show the different learning successes by applying one of the three learning rules mentioned above.

If the user chooses the option for self-organized learning the shell offers two different types of self-organized learning networks, namely the Kohonen Feature Map (KFM) and the Self-Enforcing Network (SEN). The first network is a standard one, known for more than two decades and is particularly used in this shell for the classifying and systematizing of data. The SEN has been recently developed by us and serves the

same purposes. In both cases the user has in a first step to construct a so-called semantical matrix: It contains the values of the affiliation of attributes with respect to certain objects. For example, if data should be classified according to cars and their attributes the attribute “expensive” would be associated with the object “Ferrari” with a value of 0.9 or 1.0, if the according scale is from 0 up to 1.0; the affiliation degree in the case of “VW Golf” would be approximately 0.4. The dimensions and size of the matrix are generated by the number of the objects on the one hand and that of the attributes on the other.

Then the user again has to choose a stopping criterion: Either he predetermines again a certain number of learning steps or the program should stop if the network has reached an attractor, i.e. the weight values do not change any more. In the last step the user has to select an activation function. Such a selection is only possible in the case of SEN, where the user must choose between the tanh function, the “mean linear” function, and the “logarithmic linear” function; the last two were developed by us for the SEN. The KFM always operates with the same activation function, the sigmoid one, and the same learning function – the so-called “winner takes all” rule; the SEN also uses just one learning rule, namely the newly developed SER – Self-Enforcing Rule. In Fig. 5 shows an example for SEN:

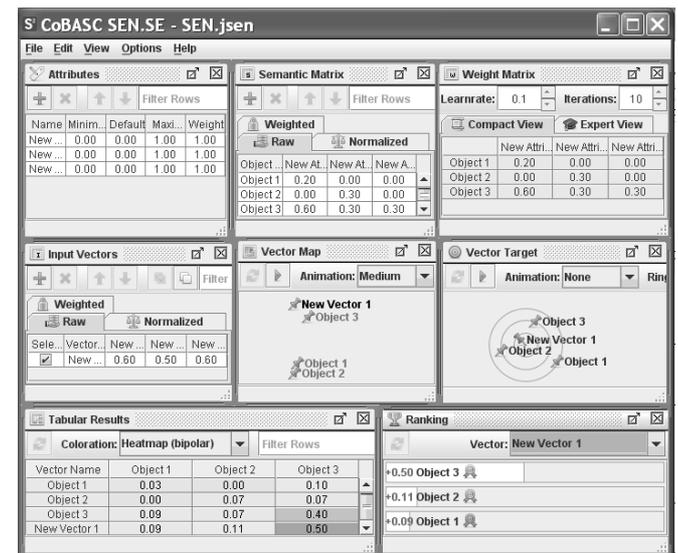


Figure 5: An opening page of SEN

Finally the user can in both cases see the ordering results on a visualization plane. We shall give an example of the operations of a SEN in the next section.

2.5 The Fuzzy Expert Shell

This shell allows the user constructing his own expert systems, combined with several enlargements of fuzzy logic operators. Expert systems consist basically of logical IF – THEN rules that can be used for diagnostic purposes, analyzing technical systems or steering systems. The addition of fuzzy operators allows the expert system to operate in a “more or less” fashion: A condition of a IF – THEN rule must not necessarily be formulated with a strict value but can be formulated within a certain range or interval respectively. If for example a medical diagnosis expert system has to decide which degrees of fever need certain treatments it can be pro-

grammed the way “IF the temperature lies between 37 and 38.5 degrees (Celsius of course) THEN the patient should be regularly observed but no medicine is still necessary” (Fig. 6).

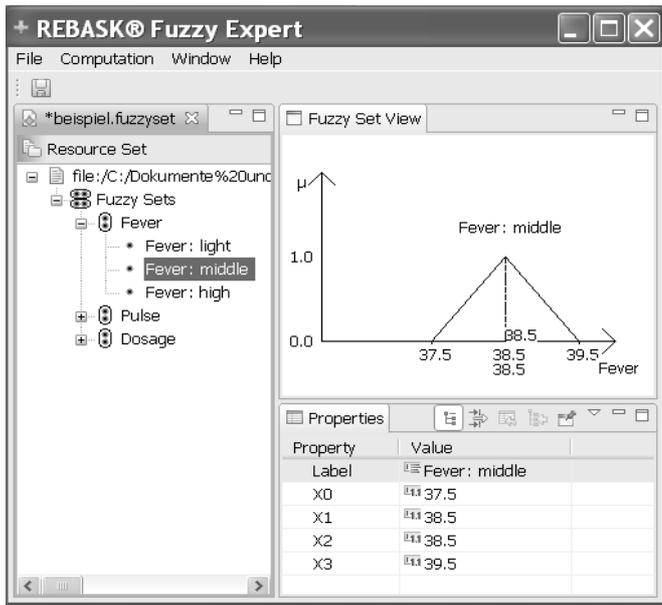


Figure 6: Example of Fuzzy Sets

Fig. 6 shows some Fuzzy sets constructed for the diagnosis system.

The Fuzzy Expert Shell allows the construction of rules in the same way as in the CA shell with different logical combinations in the condition part (Fig. 7). The user can decide between different fuzzy operators which ones are best suited for his problem.

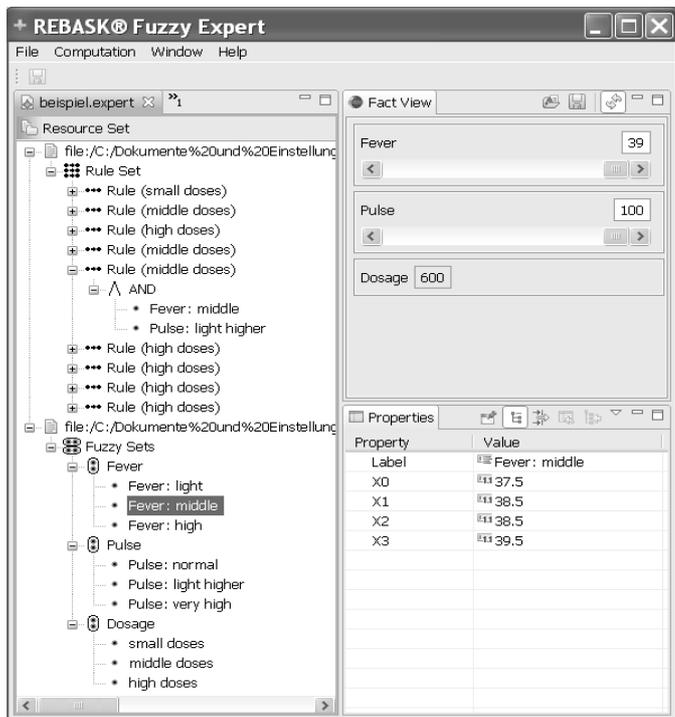


Figure 8: Example for Fuzzy rules and a result for the medication in one case

An example of a fuzzy expert system applied to the simulation of the Delphi method can be found in Pohl and Klüver 2012.

3. TWO EXAMPLES

3.1 A CA for reminder departments

On the basis of the COBASC CA-shell a M.Sc. student of us in business administration developed this model, supervised by us. At present we use an extended version of this model in the reminder department on commission of the *Deutsche Post*, one of the biggest logistic firms in the world market. The basics of the model are:

The cells represent either employees of the reminder department; because we operate with the factual numbers there are just as many employee cells on the grid as factual employees in the department. Or the cells represent customers that must be reminded because they pay their bills only in a dilatory way. Accordingly the customer cells are in one of four different sub states: In the beginning the customers are in the state “having still time for paying”. Afterwards before a first reminder they are in the first reminder level. After the first reminder the cells are transformed into the second level, if they have not paid after a certain time, which is measured in numbers of CA runs. If the customers have paid these cells are removed from the grid and are substituted by cells in the first level. If the customers in the second level do not pay within a certain time they are transformed into the third level; if they pay within this time they are also removed and substituted by cells in the first stage. If the customers in the third level do not pay they are transformed into the law department and hence also removed from the grid. The same is the case with customers who did pay; they are substituted again by cells in the first stage. The model, hence, simulates a permanent flow of customers who appear as problems for the reminder department and afterwards vanish because they have paid or because they are handed over to the law department. The employee cells always remain on the grid.

Contacting the dilatory customers is simulated by letting the employee cells move, i.e. letting them go into the Moore neighborhood of a customer in one reminder level. This rule represents the fact that employees in the reminder department need a certain time to deal with the according customers. The model is of course factually significantly more complicated but this basic information is enough to understand the model. Goal of the simulation with different parameters is a decision support how many employees the reminder department needs in relation to the costs of the employees and their success with respect to the customers. The first results perfectly satisfied the responsible managers of the *Deutsche Post*; Fig. 9 shows one stage of the CA:

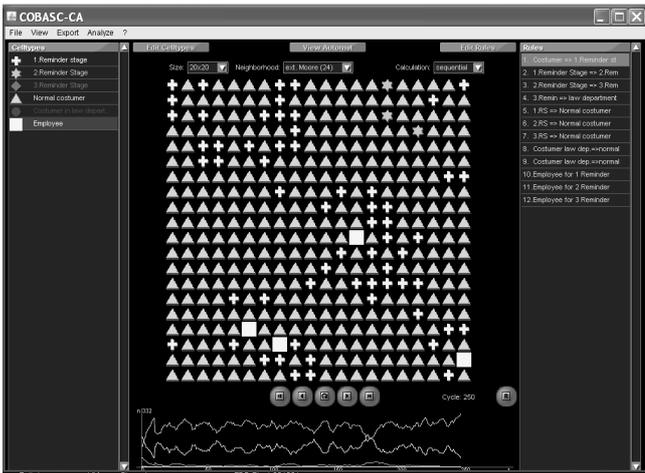


Figure 9: A stage of the reminder CA; most cells are “normal” customers, several are in the first and just a few in the second reminder stage.

3.2 Selecting a model of procedure for project planning with a SEN

There are many models of procedure for project planning that can be applied depending on the size of the project, the available time, the number of project members, the cost limit, and many other factors. Two business administration students of us selected for their M.Sc. and B.Sc. thesis a suited model of procedures for factual projects in the firms where they already worked as programmers and project managers.

As the first step when using a SEN is the construction of the according semantical matrix they selected 14 standard models of procedure that belong to the mostly used ones. Afterwards they listed 63 attributes that characterized the models of procedure “more or less”; the 14 * 63 matrix hence consists of values between 0 and 1. 0 means that the according attribute does belong to a model not at all; 1 means of course that the attribute fully belongs to the according model.

The next step was the characterization of the project for which a suited model of procedure should be selected. This characterization is represented as a vector; in the visualization this vector is placed into the center of the visualization plane, and the different models of procedure are placed in the beginning at the periphery. After the runs of the networks, i.e. after it has reached an attractor, the visualization algorithm compares the final activation states of the vectors that represent the different models with the project vector (Fig. 10).

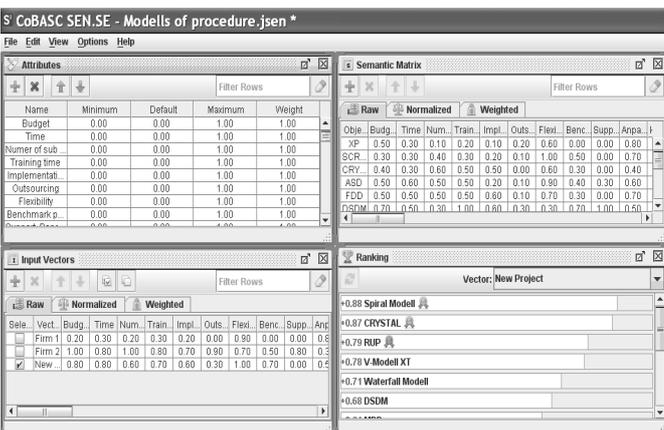


Figure 10: Inputs and a ranking of the result of a SEN

As result of this comparison the models are “drawn” by the visualization algorithm to the center. The model nearest to the center is the best suited one (Fig. 11). We name this visualization “Vector Target”.

SEN has another visualization algorithm, namely the so-called “Vector Map”-visualisation (Fig. 12). In this case the objects are additionally “clustered” in respect to their similarity among themselves although they also are drawn to a center vector.

In both visualisations it is shown that the best suited model of procedure for the new project is the Spiral Modell.

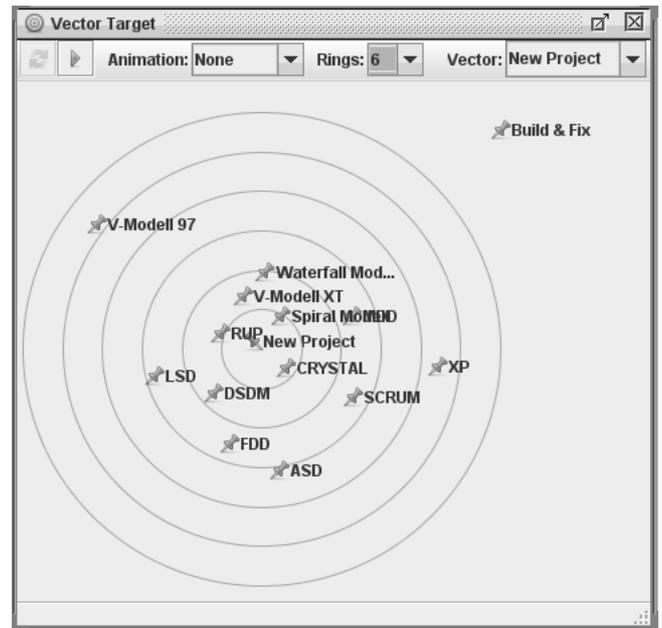


Figure 11: The “Vector Target” Visualisation of SEN

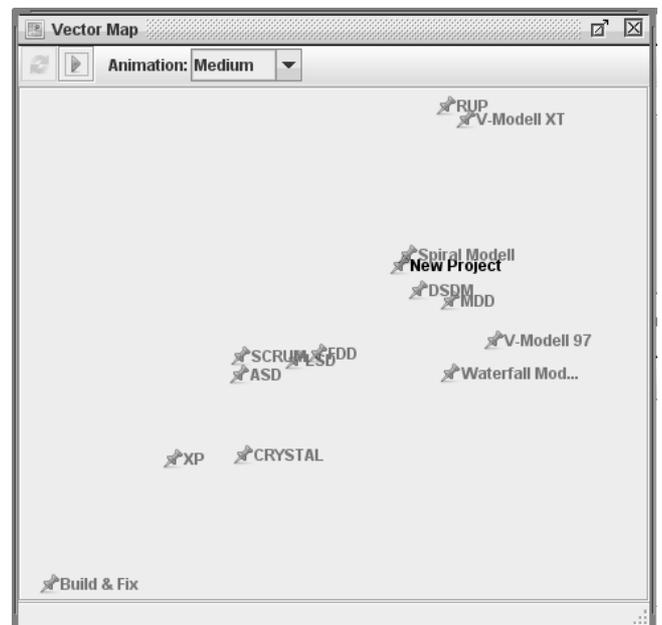


Figure 12: The “Vector Map” Visualisation of SEN

The responsible managers in both firms were very satisfied with the results of the SEN that confirmed their own considerations. The SEN, however, was significantly faster.

4. CONCLUSIONS

These examples give already an impression how many and diverse the possibilities of using Soft Computing techniques and according tools are. Numerous practical experiences with these tools encourage us to develop them further. In particular, these shells open new ways of teaching: The students learn the techniques of Soft Computing “by doing”, namely by constructing their own models. In addition, frequently we heard from students of, e.g., business administration and/or communication science that by transforming their specific problems into the logic of a suited Soft Computing shell they were forced to analyze their problems in a precise and formal way. Only then they felt that they now really understood their problems. These shells, hence, not only enable practitioners to deal with their problems in a new way but are very useful for new forms of teaching and learning too; several PhD thesis where one of the shells were used show their fruitfulness for research too. In addition, the shells have been frequently practically used in different firms and always to the satisfaction of the responsible managers. The two examples we have shown are only a small selection.

The chief innovative aspects of our whole tool can be summarized as follows: a) Only our tool contains all the different techniques of Soft Computing; all other tools known to us only offer the usage of one or few techniques. b) In contrast to other tools a user of our shells only needs to understand the basic logic of the technique he wishes to use and how he should transform his problem into the logic of the according technique. That is in particular very useful for users who are no computer scientists. c) Our tool offers totally new algorithms, namely the new self-organized learning neural network SEN, new learning rules for supervised learning neural networks, two new evolutionary algorithms, the RGA and the Lamarck algorithm, and finally the possibility of consistency checks when constructing a CA model. In these aspects our tool is quite unique.

Most important in the future work is to implement for each technique an interface to enable connections to data bases.

REFERENCES

- Esser, J. and Schreckenberg, M. 1997. Microscopic Simulation of Urban Traffic Based on Cellular Automata. *International Journal of Modern Physics C8*, 1025 – 1036
- Goonatilake, S. and Khabbal, S. (eds.) 1995. *Intelligent Hybrid Systems*. London: John Wiley
- Holland, J. 1998. *Emergence. From Chaos to Ordering*. Reading (MA): Addison Wesley
- Klüver, C., Klüver, J., and Schmidt, J. 2012. Modellierung komplexer Prozesse durch naturanaloge Verfahren. *Soft Computing und verwandte Techniken*. 2nd edition. Wiesbaden: Springer Vieweg
- Pohl, R., and Klüver, C. 2012. Simulation of the Delphi Method with a Fuzzy Expert System. In Klumpp, M. (Ed.). *Proceedings of the 26th European Simulation and Modelling Conference - ESM' 2012*. Essen, Germany
- Stoica-Klüver, C. and Klüver, J. 2007. Simulation of Traffic Regulation and cognitive Development. *Complex Systems* 17, 47 – 64

WEB REFERENCES AS EXAMPLES FOR DIFFERENT TOOLS

Boolean Networks

<http://www.softpedia.com/get/Network-Tools/Misc-Networking-Tools/Boolean-Network-Modeller.shtml>
<http://bioschool.iitd.ernet.in/NeuroDNet/boolean.php?in=8&t=0.001&f=9028&j=1>
<http://www.brothersoft.com/boolean-network-modeller-181276.html>

Cellular Automata

http://www001.upp.so-net.ne.jp/suzudo/index_e.html
<http://www.jcasim.de>

Evolutionary Algorithms

<http://sourceforge.net/directory/development/algorithms/genetic-algorithms/os:mac/freshness:recently-updated/>
<http://www.geatbx.com/docu/>

Fuzzy-Logik

<http://uhu.es/antonio.barragan/category/temas/fuzzy-logic-tools>
<http://www2.imse-cnm.csic.es/Xfuzzy/>
<http://www.kdnuggets.com/fdsearch/search.pl?Match=1&Realm=Recent&Terms=fuzzy+logic>

Neural Networks

<http://www.softpedia.com/get/Science-CAD/Sinapse-Neural-Networking-Tool.shtml>
<http://www.cnet.com/topic-software/neural-network.html>
<http://www.dicis.ugto.mx/profesores/sledesma/documentos/>

BIOGRAPHIES

CHRISTINA KLÜVER (Dr. phil., habil.) studied educational science and computer science. She obtained the Ph.D. in Communication Science and the Venia legendi in Computer Science. She is lecturer in computer science at the University of Duisburg-Essen, Germany, in the faculty of Economics and Business Administration. As a member of the research group COBASC (Computer Based Analysis of Social Complexity) her research interests are in applying models of nature analogous programming techniques like neural nets, cellular automata, and evolutionary algorithms to social and economical problems.

JÜRGEN KLÜVER (Prof. Dr. phil.) studied Philosophy and Mathematics at the Universities of Kiel and Hamburg. He is head of the research group COBASC (Computer Based Analysis of Social Complexity). His main field of research is the development of theories on social and cognitive dynamics, based on the construction and investigation of according computer models, and the analysis of formal models suited for the simulation of social, cognitive, and economical processes

A NEW APPROACH TO 3D SIMULATION TECHNOLOGY AS ENABLING TECHNOLOGY FOR eROBOTICS

Jürgen Rossmann, Michael Schluse, Christian Schlette, Ralf Waspe
Institute for Man-Machine Interaction, RWTH Aachen University, Ahornstrasse 55, D-52074 Aachen, Germany
E-mail: {rossmann, schluse, schlette, waspe}@mmi.rwth-aachen.de

KEYWORDS

eRobotics; 3D Simulation; Virtual Testbed; Simulation-based Control; Mobile, Space and Industrial Robotics

ABSTRACT

The use of simulation technology is well known in the field of engineering. It is very common for engineers e.g. to use block oriented simulation systems to develop controllers or to use FEM analysis tools to test and verify component layouts. By contrast, so far the use of 3D simulation technology based on kinematics, rigid body or sensor simulation seems to be limited to only a few specific application areas like robotics. Yet, 3D simulation technology has the potential to evolve from a support tool to a core engineering technology acting as a new focal point in engineering processes. Here, it is the goal of *eRobotics* to introduce a continuous and systematic computer support into the entire lifecycle of complex systems, where the “computer support” is provided by 3D simulation technology. This requires the development of new architectures for 3D simulation frameworks, setting new benchmarks in performance, flexibility, modularity, integration capabilities and realism. The frameworks also have to close the gap between virtual and real worlds, allowing for smooth bidirectional transitions between them. New approaches of “Virtual Testbeds” and “Simulation-based Control” provide the conceptual foundation in this process. This contribution introduces the basic concepts of eRobotics, summarizes new requirements on 3D simulation technology and introduces a new reference implementation that meets the formulated requirements.

INTRODUCTION

The research field of eRobotics is currently an active domain of interest for scientists working in the area of “eSystem engineering”. The **aim of eRobotics** is to provide a comprehensive software environment for the development of complex technical systems. Starting with user requirements analysis and system design, support for the development and selection of appropriate hardware, programming, system and process simulation, control design and implementation, and encompassing the validation of developed overall systems, eRobotics provides a continuous and systematic computer support during the entire life cycle of complex systems (see Figure 1). In this way, the ever increasing complexity of current computer-aided technical solutions will be kept manageable and know-how from completed work is electronically preserved and made available for further applications.

To provide the necessary degree of “computer support” mentioned above, eRobotics makes extensive use of 3D simulation technology. 3D simulations are used right from the beginning of the development process to test first system design studies in the concept phase. During system development, fully functioning interactive virtual prototypes allow for an efficient and goal directed development, test and verification both on component, as well on system level – at any point of time. Besides, 3D simulation technology not only allows to visualize, simulate, test and experience the virtual prototype by providing so called “Virtual Testbed” facilities (Rossmann et al. 2011a) as shown in Figure 2, but can also be used as a development framework to implement both control and supervisor algorithms (like motor controllers, robot programs, image processing algorithms) using concepts of “Simulation-based Control” (Rossmann et al. 2012a).

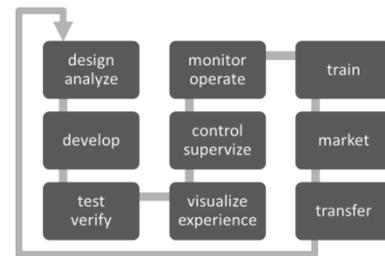


Figure 1: eRobotics supports the entire life cycle of complex systems by an extensive use of 3D simulation technology.

Furthermore, 3D simulation can be used to realize state of the art user interfaces to intuitively monitor and interactively operate the virtual, as well the physical system using “Projective Virtual Reality” methods (Freund et al. 1999), to provide training environments, support marketing activities and transfer the development results to other application fields.

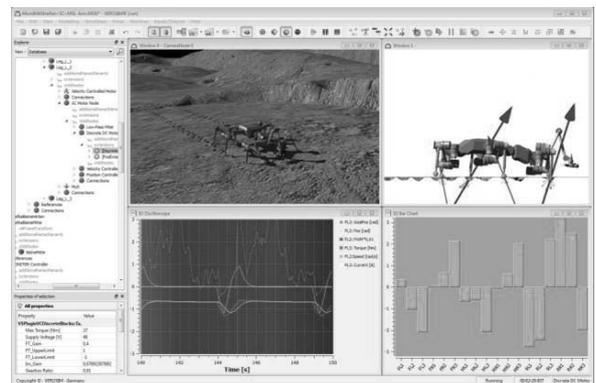


Figure 2: A typical Virtual Testbed environment, here used for the development of walking robots (Yoo et al. 2010) (robot: © DFKI Bremen)

It is important to mention that the 3D simulation-driven development process does not have to be fully sequential, as typical development processes like the waterfall model in software development or the technology readiness level oriented approach in space engineering. Because of the fact that all development is based on the same 3D simulation and the same 3D model (which therefore also integrates controllers, supervisors, user interfaces, test environments, etc.) also agile development processes become possible, thus lowering development risks and costs, while at the same time increasing the quality and robustness of the results.

To use the eRobotics concepts in the different development stages as well as in different application areas, new concepts for the underlying 3D simulation technology were necessary. Thus, we developed (together with different partners from research and industry) a new approach to 3D simulation, resulting in a novel architecture for 3D simulation systems and a new comprehensive implementation of a 3D simulation system showing the feasibility of the presented approach. The result has been used and verified in a wide range of applications ranging from industry over space to environment, as illustrated in

Figure 3: The eRobotics concepts have so far been used in more than 30 applications in different application areas

In the context of eRobotics, 3D simulation technology becomes a central factor for the realization and implementation of new technical systems - not only for supporting the development process, but as a central focal point of all simulation-based engineering processes in their lifecycles. This distinguishes our approach greatly from the current use of simulation technology in systems development: Generalized simulation systems such as block oriented or FEM simulations as well as application specific simulation systems such as robot simulators are used only at certain points of the development process – to develop controllers or to verify development results mainly on component level. Here, 3D simulation technology has the potential to integrate these existing simulations and carry them to a new level.

The contribution is organized as followed. We start with an introduction of the key concepts of the eRobotics approach. The concepts lead to new requirements of 3D simulation technology which are then compared to the state of the art in the field of simulation technology. In order to address the requirements, we introduce a new architecture for 3D simulation systems as well as the key concepts and modules of our implementation. The contribution closes with an overview of selected applications and a conclusion summarizing the current status as well as future work.

THE eROBOTICS APPROACH

When introducing the eRobotics approach, one has to distinguish applications from processes, methods and tools. On application level, eRobotics addresses a variety of different **application areas**. eRobotics technologies can be used to develop e.g. new remote sensing algorithms for forest inventory, new production lines in industry or new mobile robot technologies for space applications. The use of eRobotics is not restricted to certain **development processes**. eRobotics supports the various process models listed above, but also the V

model or modern agile development approaches. This is possible due to the fact that eRobotics provides various **engineering methods** (see Figure 4) which can be used in the different stages of a system life cycle. The methods all rely on 3D simulation technology and comprise first of all the methods necessary for 3D visualization, 3D animation, 3D simulation and “Virtual Reality”. “Projective Virtual Reality” methods link the “Virtual Reality” system to the real world to intuitively monitor and interactively command physical systems.



Figure 4: Classical and new 3D simulation based engineering methods provided by eRobotics

New concepts of “Semantic World Modeling” (Rossmann et al. 2009a) provide the methods necessary to set up models of components, systems, environments, controllers, supervisors, etc. The models can be built fully automatically, manually or semi automatically, using interactive modeling environments. Methods for system analysis and design support the developer even in the first development phases. The integration of data processing algorithms like image processing or generic controllers or supervisors leads to a comprehensive virtual testing facility, the “Virtual Testbed”, allowing for detailed (interactive or preprogrammed) tests at system level. “Simulation-based Control” concepts bridge the gap between simulation and reality by attaching the 3D simulation system to physical systems, allowing the virtual data processing algorithms to not only work on virtual devices, but also in real world scenarios in the context of “Rapid Control Prototyping” (Abel and Bolwig 2006).

On the tool level, all methods are implemented using 3D simulation technology and are integrated in one single comprehensive **3D simulation framework**. This framework itself is purely abstract, so that it not only can act as the basis for 3D simulation, but also for virtually any other type of simulation, such as block oriented simulations, discrete event simulations or FEM analysis methods. In addition, it can be linked to other simulation systems for co-simulation or hard-/software in the loop simulations.

REQUIREMENTS FOR 3D SIMULATION TECHNOLOGY

For realizing eRobotics concepts, the major prerequisite on the tool level is the use of one single but comprehensive and integrated 3D simulation framework which is able to implement all the methods and support all the processes outlined

above. The advantage of such an integrated framework is to minimize conversion tasks and the ability to be able to simulate, at the same time, all simulated components within one single but comprehensive “Virtual Testbed”. This leads to various requirements of the underlying 3D simulation framework:

Overall Flexibility: The simulation system must support a broad range of applications and usage scenarios (see Figure 3). It must be able to be used as an engineering tool on the desktop, as an interactive and immersive “Virtual Reality” system and as a tool for realizing control algorithms on real-time capable systems. Hence, it must separate simulation algorithms from user interface implementation.

Performance: To allow hands-on interaction the simulation must perform in real-time. The simulation must also be able to perform in a hard real-time mode to enable hardware-in-the-loop (HIL) scenarios. It is therefore necessary to implement the simulation system in a “hardware friendly” programming language like C++.

Freely Configurable Database: New usage scenarios require new data structures. Therefore, the data model must be adaptable to new simulation models, even at run time. Thus, a meta data system and a reflection API (like the one available in Java) are necessary. Such a flexible database can then be used in all kinds of data storage and data manipulation scenarios, not only 3D simulation, but any other kind of simulation. This way, all methods can use the same model which contains (on an equal level) geometric information, as well as e.g. sensor configurations or controller programs.

Distributed and Parallel Simulation: In order to separate user interaction and (hard) real-time simulation and to allow the parallel and distributed execution of the simulation, it is necessary to distribute the simulation state across computer nodes. The mechanisms for an efficient, recursion-free simulation state transfer have to be an integral part of the simulation database.

Modular Simulation Model & Adjustable Level Of Detail: Since the level of detail of the model will increase during the development process, the simulation models must be adaptable and interchangeable. Furthermore, some tests may require a more detailed simulation model, but no real-time performance. So it must be possible to record the simulation state changes and play them back later in real-time.

Realism: For many applications, such as games, a computationally fast and visually plausible model is sufficient. However, for the use in engineering processes the simulation must either be physically correct or the error must at least be quantifiable. Of course, the necessary degree of realism in the different aspects (visualization, physics, sensors, etc.) depends on the concrete use case and must be adjustable.

Calibrated Simulation Algorithms: In order to obtain valid and reliable results, the simulation algorithms have to be calibrated against real systems.

Flexible and Standardized Interfaces: To couple and interchange various simulation components, internal interfaces of

the simulation components need to be standardized. To use external components, such as special simulation models (co-simulation), control software (software-in-the-loop) or hardware (hardware-in-the-loop) external interfaces specifications must be adhered to. By mapping the external interfaces to the internal interfaces, it is then possible to easily interchange data between the 3D simulation and external co- and subsystems.

Integrating Data Processing Algorithms: Besides the simulation of its physical behavior, the complete reproduction of a system needs to incorporate control algorithms. Only then is it possible to simulate complex interactions, e.g. the influence of actuator control commands on sensor data.

Seamless Transition from Simulation to Reality: The use of block oriented data models and subsequent code generation for controller implementation is standard in “Rapid Control Prototyping”. This well-established workflow should also be available in the 3D simulation system, such that data processing algorithms developed with and integrated into the system can be used on the real hardware.

Cross Platform Support: The core of the simulation system should be platform independent, in order to be executable on a variety of desktop OS (Windows, Linux) and mobile devices with small changes of the graphical user interface. It should also be possible to run the simulation without any graphical user interface on embedded devices, using e.g. QNX.

STATE OF THE ART

Taking a look at the state of the art of simulation technology reveals various approaches to simulation technology. Discrete event simulation systems (Banks 2010), block oriented simulation approaches like the Matlab/Simulink framework (www.mathworks.de/products/simulink/) or the Modelica modeling language (Fritzson 2003) as well as various FEM-based simulation tools (e.g. www.comsol.com) are probably the most well-known ones. But even if focusing on (quasi continuous) 3D simulation technology, various approaches can be found, starting with game engines (e.g. www.unrealengine.com) over frameworks for the development of mobile robots (e.g. www.ros.org/wiki/gazebo) or generic mechatronic systems (e.g. www.mathworks.de/products/simmechanics/). Still missing is a holistic and encompassing approach which enables and encourages the synergetic use of simulation methods on a single database throughout the entire lifecycles of technical system. This is crucial to eRobotics, but most approaches focus on dedicated application areas, dedicated disciplines (electronics, mechanics, electronics, thermodynamics, etc.), or are restricted to the development of single components.

A NEW SIMULATION SYSTEM ARCHITECTURE

To overcome these limitations and to fulfill the requirements listed above, we developed a new architecture for simulation systems (which is not restricted to 3D).

Micro-Kernel Architecture: The key idea is the introduction of a micro kernel, the “Versatile Simulation Database” (VSD, Figure 5). Basically, the VSD is an object-oriented real-time

database holding a description of the underlying simulation model. Fully implemented in C++, it provides the central building blocks for data management, meta information, communication, persistence and user interaction. The VSD is called “active” because it is not simply a static data container, but also contains the algorithms and interfaces to manipulate the data. Furthermore, the VSD incorporates an intelligent messaging system that informs interested listeners of data creation, change and deletion events. In addition, the VSD provides essential functionalities for parallel and distributed simulation. Depending on the application, the performance of the simulation or the controller can thus be enhanced by being parallelized on multi-core processors or the distribution in a network of computers. The mechanisms for distribution allow for separation of real-time processes from graphical user interfaces and monitoring tools (see Figure 13). All simulation functionality of the framework is achieved by creating specialized plugins which build upon and interact with the VSD core.



Figure 5: The micro kernel architecture.

Model Representation: The tiered data model consists not only of the simulation model itself, but also incorporates a meta model layer. The meta model is essential for the flexibility as well as the developer and end user friendliness of the database and the simulation system. The design shown in Figure 6 is inspired by the Object Management Group (OMG) meta model hierarchy (Kurtev et al. 2004).

The uppermost layer (labeled M2) is the meta information system, the basis for persistence, user interface, parallel and distributed simulation, scripting and communication. It mainly consists of meta types, meta instances, meta properties and meta methods. In addition to “build-in” classes, it is also possible to generate meta instances with the corresponding meta properties and meta methods during runtime (e.g. for object oriented scripting or new data models). Such “run time meta instances” are treated in exactly the same way as the build in meta instances without any performance overheads in the data management.

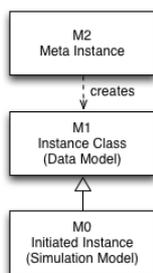


Figure 6: The meta model hierarchy of the VSD

The middle layer (labeled M1) describes the data model of the simulation. In order to be able to retain semantic information and integrate data and algorithms into one single database, the

VSD data model is an object oriented graph database (Gyssens et al. 1994), consisting of nodes and node extensions. A simplified class hierarchy of the VSD core is shown in Figure 7. All nodes in the graph database, the database itself and even the simulation environment are derived from a single base class called “Instance”. This base class provides mechanisms for inter-instance communication, as well as access to the meta information system which allows introspection of class hierarchy, properties and methods.

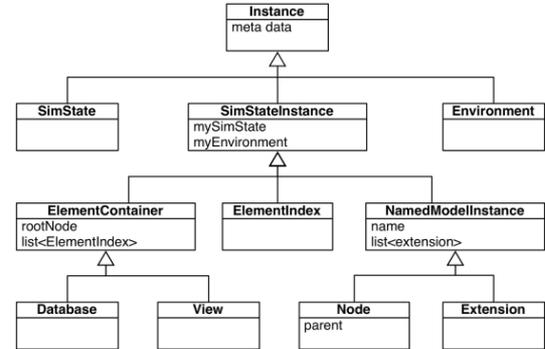


Figure 7: The core database class hierarchy

Following this approach, the database is able to integrate standard geometric models as well as block-oriented simulation models using input/output connections (see Figure 16), the intermediate representation of scripting languages (Rossmann et al. 2012b) or entirely different types of information like forest inventory data (Rossmann et al. 2009b).

System Integration: The functionality of the micro kernel is extended by various plugins implementing simulation or data processing algorithms, interfaces to hard- or software systems, user interfaces, etc. (see Figure 5). Using the VSD, the plugins can communicate with the database as well as establish directed communications between themselves. One crucial point is the combination of and the communication between different simulation algorithms. This can be challenging in complex scenarios which incorporate different application domains and require a mutual interaction between the different domains for realistic simulation results. An example is the interaction of a mobile robot on a sandy surface (interaction of rigid body dynamics, terra mechanics and robot control, Rossmann et al. 2010b). Here, gluing techniques are used to let the domain specific simulation aspects interact.

KEY MODULES FOR 3D SIMULATION

The basic simulation system architecture is now the basis for our implementation of a novel comprehensive 3D simulation system, extending the VSD kernel in various directions to fulfill the requirements introduced above.

Data Storage: The metadata approach to the simulation database enables the simulation system to mirror a large variety of different simulation models. Various interfaces to different data sources have been implemented to enable a seamless data interchange: Ranging from classical 3D data formats such as VRML, 3ds, STEP or IGES over different GIS data formats like SHP or GML (ISO 19136) up to application specific data formats like ForestGML for forest inventory data or SEDRIS (www.sedris.org) for military simulations. In addition to the classical file-based data storage, the simulation system can connect to (object-oriented) databases for online data synchronization (Hoppen et al. 2012a).

Visualization: A close-to-reality rendering of the current state of the geometric part of the model is important, because it enables the user (be it an engineering professional or an end user) to understand the systems behavior with the help of visual clues. The framework therefore provides a render engine optimized to handle large models which can be highly dynamic concerning state (e.g. poses) or structural (e.g. streaming) changes. It also supports various so called visualization metaphors to render aspects which are not visible to the user in the real world (like the laser beams in Figure 15).

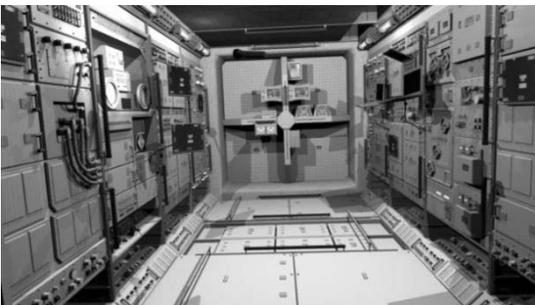


Figure 8: Real-time rendering: The virtual Columbus module

(Projective) Virtual Reality: “Virtual Reality” techniques can be used to further enhance the degree of immersion. Multi-screen projection environments, data-helmets or even simple stereo capable monitors can be used to let the user dive into the virtual world. Distributed simulation techniques allow for the distribution of the rendering load for many render outputs over different computers. Devices like data gloves, tracking devices, joysticks, space mice, Kinect, etc. let the user interact with the virtual world. For “Projective Virtual Reality”, methods to detect and interpret user actions are available which “project” them for a direct control of physical automation devices available (Freund et al. 2001). Haptic feedback devices like Stewart platforms let the user feel what happens in the real world.

Kinematics: The kinematics framework (Schlette 2012) allows for an efficient modeling of arbitrary kinematic trees (e.g. the “Virtual Human” or robots, see Figure 9). Such kinematic structures can incorporate an arbitrary number of joints and span over multiple devices. They are the basis for the definition of pose lists forming movement paths and trajectories

– in joint as well as in Cartesian space. Different forward and backward transformations allow for a stable transition between them. Interpolation algorithms move the modeled structures along the paths (Rossmann et al. 2010a).

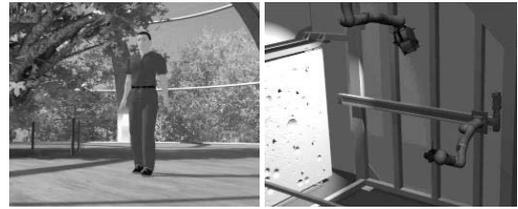


Figure 9: Examples for kinematic trees or kinematic chains

Rigid Body Simulation: As a central part of the framework, rigid body simulation techniques simulate the dynamic behavior of physical systems (Jung 2011). They even allow for a fast and stable simulation of the physical behavior of several rigid bodies connected by joints interacting with their environment. Besides collision detection the adherence to constraints is a major element of the rigid body simulation (see Figure 10). Our implementation is based on the conservation of impulse and torque and uses a maximum coordinate approach using Lagrange factors (Stewart and Trinkle 2000).

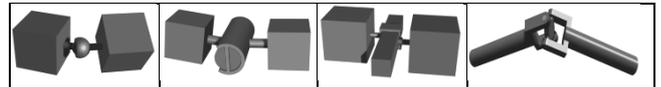


Figure 10: Basic joint variants in rigid body simulation (left to right: Spherical, pivotal, linear and cardan joint)

Process Simulation: The rigid body simulation can be extended by integrating specialized process simulation algorithms that would be beyond the scope of the standard algorithms, e.g. terra mechanics (Rossmann et al. 2010b).

Sensor/Actuator Framework: An integral part of nearly any technical system are sensors and actuators. To allow for an easy integration of different sensors and actuators as well as an intuitive visualization of actuator inputs and sensor outputs we developed a sensor/actuator framework (Emde et al. 2011, Yoo et al. 2010). This provides not only the necessary simulation algorithms, but also interfaces to the corresponding physical sensors for a seamless transition between virtual and real world operation modes. Record and playback functionality, error models and predefined data processing blocks allow for replay of real world sensor data as well as testing sensors and actuators under different conditions.

Identification and Calibration: The simulation algorithms need to be calibrated such that they produce data comparable to data obtained in real world scenarios. The calibration is achieved by comparing data obtained from a real mock-up to results obtained from a simulation of the very same mock-up. The simulation algorithms are then tweaked until they successfully mimic the real results. Examples for the calibration of simulated optical sensors like cameras, laser scanners or PMD devices are given in (Rossmann et al. 2013a). Furthermore we calibrated algorithms in terra mechanics as well as different engine models (Rossmann et al. 2010b). The calibration infrastructure can also be used to identify model properties (like the parameters of cameras or rigid joints) using standardized identification processes.



Figure 11: Calibration of a camera simulation under different lighting conditions (Rossmann et al. 2012c)

Data Processing: As mentioned above, the simulation database provides generic simulation methods which are used for (quasi continuous) 3D simulation, but are not restricted to this type of simulation. It is easy to add all necessary data processing algorithms like image or sensor data processing, actuator controllers or high level control programs as a “natural” part of the simulation model to simulate the overall behavior of the developed system on systems level. Examples for this, such as the integration of localization, navigation, robot or satellite control algorithms are given in the applications section. Other examples are the integration of remote sensing (Rossmann et al. 2009b) or action planning techniques. All data processing components interface with other model components or other algorithms using the standard communication means provided by the VSD (see Figure 16 for a block diagram visualization).

State Oriented Modeling: While the various algorithms mentioned above provide a sustainable basis for simulation and data processing, it is up to the user to combine the algorithms to an application via modeling the entire system. The biggest part of this work can be done by adding connections between algorithms and model components. But in most applications, a small part of application specific logic is needed to e.g. connect a user interface (like a joystick) to the simulated components, generate sound at different simulation states or detect and interpret the users actions. Normally, various kinds of procedural scripting languages close this gap. Compared to this, our approach to scripting is fairly different. Because of the fact that most of the time such application specific logic has to supervise simulation states or react to simulation changes (events) we use a newly developed object-oriented Petri net variant. This State Oriented Modeling approach combines the advantages of classical Petri nets with object-oriented programming paradigms. State Oriented Modeling is also used as a powerful programming language for complex controller algorithms or as an intermediate language for textual (Rossmann et al. 2011b) or graphical (Rossmann et al. 2008a) robot programming languages. outlines a simple example for this.

Controllers: Besides the possibility to add application specific framework extensions or to use State Oriented Modeling to write so called “Controller Extensions” which move objects, handle input and output values of “Controller Objects”, etc. the 3D simulation framework provides several standardized controllers for robot manipulators (currently supporting the KRL robot language), Programmable Logic Controllers (PLC), etc. as well as the user interfaces to program and test the controller programs. Figure 13 shows on its left side a typical user interface setup.

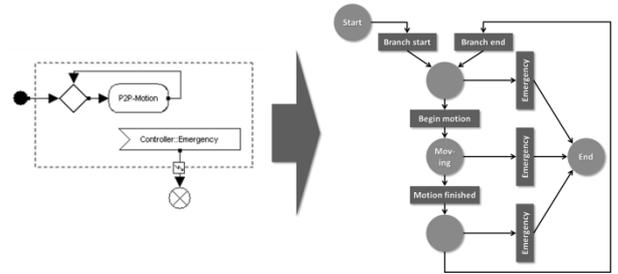


Figure 12: Translating a simple activity diagram for robot programming into a Petri net (Rossmann et al. 2008a)

Co-Simulation/Hard-/Software-in-the-Loop: In addition to this, existing hard- or software components can be integrated in the simulation flow. This enables a co-simulation to add other simulation algorithms to the overall simulation. Software-in-the-Loop scenarios can be used to e.g. integrate application specific controllers or data processing algorithms. Hardware-in-the-Loop scenarios let physical controllers operate on virtual controller objects. On the other hand the 3D simulation framework can be attached to real devices, so that simulation algorithms can be used to drive real hardware in Simulation-based Control scenarios.

Parallel and Distributed Simulation: The entire 3D simulation system is designed to fully support parallel and distributed simulation techniques. The instances setting up the simulation database (called SimStateInstances, see Figure 7) can be replicated in copies of the simulation state (called SimStates) that then can be configured using one single computer (parallel simulation) or a computer network (distributed simulation). Methods for bidirectional synchronization of state changes allow to keep the several SimStates in sync.

Scheduling: The scheduling algorithms support different timing modes (as fast as possible, real-time), as well as quasi continuous and discrete event based scheduling, allowing for an easy transition between these worlds or even using them in hybrid simulation scenarios. It is also possible for the scheduler to distribute the simulation tasks over several SimStates using multi-core or multi-computer environments.

(Graphical) User Interface: In desktop environments, all the components introduced above are accessible via a comprehensive graphical user interface (see Figure 13, left). In real-time setups, on platforms without support for 3D graphics, or for applications for which a 3D rendering is dispensable or not available (for example for controller implementation), a textual user interface is available (see Figure 13, middle). In addition to this, the VR methods summarized above can be used for the configuration of VR based user interfaces for example using projection environments (see Figure 13, right). It is important to mention that all these user interfaces can be used simultaneously, because the underlying simulation systems can link together and exchange state changes using the distributed simulation functionalities.

Operating Systems: The framework currently supports Windows, Linux and QNX Neutrino. On QNX, hard real-time simulations are possible, enabling the data processing algo-

rhythms in the simulation model to connect to and control/supervise real world hardware. Here, only a text-based user interface is available on the target hardware.



Figure 13: Graphical, textual or VR based user interface of the same simulation framework

SELECTED APPLICATIONS

Our 3D simulation framework introduced above has been used to realize a large variety of different eRobotics applications so far (see also Figure 3). Currently, eRobotics applications mainly focus on three application areas (see Figure 14), environment (e.g. forest inventory or forest machines), industry (e.g. industrial automation) and space (e.g. space robots), for which it provides a common development approach. In this chapter we shortly outline the results of three applications, one for each application area.



Figure 14: Major application areas for eRobotics

Localization of Forest Machines: In most application areas it is crucial to know the current position, orientation and movement direction of mobile systems as a prerequisite to apply various assistance and optimization algorithms like navigation or autonomy concepts. Most of the time, GPS is used for this. But in forest environments the GPS accuracy is often off by 10 to 50 m. We therefore developed the VisualGPS approach which compares a global tree map (pre-calculated using remote sensing data) to a local tree map using local sensors like laser scanners. Both maps are processed using Semantic World Modeling techniques. The position estimation is the result of a pattern matching process using particle filters. Here, 3D simulation technology has been used to test and verify the newly developed localization algorithms in an interactive forest machine simulator consisting of a fully operable virtual forest machine equipped with simulated sensors like GPS, compass, laser scanners and stereo cameras operating in a close-to-reality forest environment (see Figure 15, left). In addition to this, the same simulation has been used to design and implement the localization algorithm itself. The simulation database, able to store and update the world model, and simulation components like collision detection or input/output networks, were used to implement the algorithms and to interface them to virtual or real sensors or onboard computer hardware (see Figure 16). In addition, the 3D simulation was the basis for the implementation of the user interface (see Figure 15, right).



Figure 15: A Virtual Testbed to develop localization and navigation methods for forest machines and the resulting user interface (Rossmann et al. 2011c)

Following this approach, after approx. two years of development, the source selection switch in Figure 16 has been switched from virtual to real sensors. It then took about two days before the real system was fully operational.

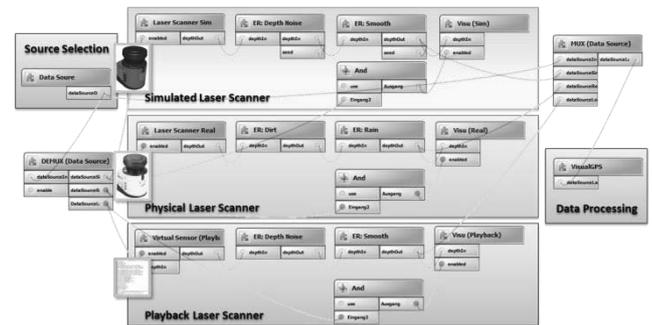


Figure 16: Block oriented view on a simulation model integrating simulated laser scanners as well as interfaces to physical laser scanners, error model, source selection and data processing blocks (Emde et al. 2011)

Program and Control Robot Manipulators: The second example focuses not only on the programming of robot manipulators but also on their control. The programming task has been well investigated during the last years. Here, 3D simulation technology is used to derive movement paths and to program and test the application logic. This has been carried out based on our 3D simulation framework (see Figure 13, left), following the Native Language Programming approach, enabling the user to program a robot using the manufacturer-specific robot programming language. The robot programs are compiled into the State Oriented Modeling language which then provides the basis for this simulation. For a close-to-reality simulation of the robots movements, the kinematics framework is used. Normally, at the end of the programming work, the robot program is downloaded to the physical robot controller. Here, the eRobotics approach offers a new alternative – to apply the 3D simulation technology to be the controller itself. This allows for the realization of intelligent control structures for complex multi-robot environments like the testbed shown in Figure 11, offering 3D user interfaces, automatic action planning components, a multi-robot coordination layer or online collision avoidance (see Figure 17). To implement the controller, the simulated robots are simply replaced by interfaces to the real ones. The resulting simulation model is then “simulated” using a stripped down version of the same 3D simulation system running under a real-time operating system like QNX Neutrino for a coordinated control of four separate devices (Rossmann et al 2012a).

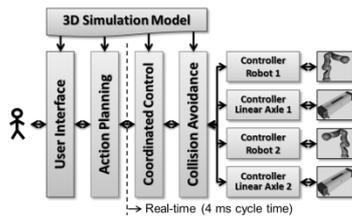


Figure 17: Simplified structure of an intelligent multi-robot control system implemented using 3D simulation technology

Development of Space Robots: eRobotics has its roots in space robotics, be it mobile robots, landers for planetary exploration, robot manipulators on satellites or the International Space Station. Here, the eRobotics framework is used to its full extend, starting with design studies over virtual testing environments for prototype testing and validation at systems level and ending with the development of intuitive and interactive user interfaces. One impressive example is the simulation of robot manipulators on satellites in space (see Figure 18, right). Here, the Virtual Testbed comprises the dynamic simulation of the robot (Kaigom et al. 2012) and the satellite, the simulation of the corresponding control algorithms (like robot programs or satellite attitude control algorithms) commanding virtual actuators and the simulation of various sensors.



Figure 18: Developing space robots: image processing for landers, sensors for exploration rovers and manipulators for debris removal satellites

CONCLUSION

In this contribution, we presented a new approach to 3D simulation technology as an enabling technology for eRobotics. It has the potential to promote 3D simulation-based engineering to be the central focal point in the development of complex systems and not only a support tool. It integrates various disciplines like mechanics, electronics, information technology and can be applied to different application areas. In this context, Virtual Testbeds allow for integrated tests at component or system level or at selectable levels of details. Simulation-based Control bridges the gap between simulation and real world operation allowing for a smooth transition between the two worlds. The approach is ambitious, but has already proven its feasibility in various applications. Our future work will concentrate on further extensions of the algorithmic basis (e.g. rigid body, sensor, actuator simulation) and on aspects of system integration such as co-simulation for strengthened reusability and connectivity to other, specialized simulators. Last but not least, we will focus on an optimized workflow to make eRobotics the concept of choice in diverse robotics and machine development processes.

REFERENCES

Freund, E.; Rossmann, J. 1999. "Projective Virtual Reality: Bridging the Gap between Virtual Reality and Robotics", IEEE Transactions on Robotics and Automation, Vol. 15, No. 3

- Rossmann, J.; Schluse, M. 2011. "Virtual Robotic Testbeds: A foundation for e-Robotics in Space, in Industry – and in the woods", DeSE, Dubai
- Rossmann, J.; Schluse, M.; Schlette, C.; Waspe, R. 2012. "Control by 3D Simulation – A New eRobotics Approach to Control Design in Automation". ICIRA, Montreal, Canada
- Yoo, Y.-H., Jung, T., Römmermann, M.; Rast, M.; Kirchner, F.; Rossmann, J. 2010. "Developing a Virtual Environment for Extraterrestrial Legged Robots with Focus on Lunar Crater Exploration", i-SAIRAS, Sapporo
- Rossmann, J.; Schluse, M.; Hoppen, M.; Waspe, R. 2009. "Integrating semantic world modeling, 3D-simulation, virtual reality and remote sensing techniques for a new class of interactive GIS-based simulation systems" International Conference on Geoinformatics, 2009, Fairfax, USA
- Abel, D.; Bollig, A. 2006. "Rapid Control Prototyping." Springer
- Banks, J. 2010. "Discrete-Event System Simulation", Prentice-Hall
- Fritzon, P. 2003. "Principles of object-oriented modeling and simulation with modelica 2.1", Wiley
- Rossmann, J.; Schluse, M.; Waspe, R. 2012. "Integrating object oriented petri nets into the active graph database of a real time simulation system". WSC, Berlin
- Rossmann, J.; Schluse, M.; Bücken, A.; Krahwinkler, P.; Hoppen, M. 2009. "Cost-efficient semi-automatic forest inventory integrating large scale remote sensing technologies with goal-oriented manual quality assurance processes". IUFRO Division 4 Conference, Quebec City, Canada
- Hoppen, M.; Schluse, M.; Rossmann, J.; Weitzig, B. 2012. "Database-Driven Distributed 3D Simulation." WSC, Berlin
- Freund, E.; Schluse, M.; Rossmann, J. 2001. "State Oriented Modeling as Enabling Technology for Projective Virtual Reality." IROS, Hawaii
- Schlette, C. 2012. „Multi-Agentensysteme zur Simulation, Analyse und Steuerung von anthropomorphen Kinematiken“, RWTH Aachen University, Dissertation
- Rossmann, J.; Eilers, K.; Schlette, C.; Schluse, M. 2010. "A Uniform Framework to Program, Animate and Control Objects, Kinematics and Articulated Mechanisms in a Comprehensive Simulation System". ISR/Robotics, Munich, Germany
- Jung, Thomas J. 2011. „Methoden der Mehrkörperdynamiksimulation als Grundlage realitätsnaher Virtueller Welten“, RWTH Aachen University, Dissertation
- Stewart, D.; Trinkle, J.C. 2000. "An Implicit Time-Stepping Scheme for Rigid Body Dynamics with Coulomb Friction", ICRA
- Rossmann, J.; Eilers, K. 2011. "Translating Robot Programming Language Flow Control into Petri Nets". ETFA 2011, Toulouse
- Rossmann, J.; Schluse, M.; Jung, T. 2008. „Introducing intuitive and versatile multi modal graphical programming means to enhance virtual environments“, IDETC/CIE, New York
- Emde, M.; Rossmann, J.; Sondermann, B.; Hempe, N. 2011. "Advanced Sensor Simulation In Virtual Testbeds: A Cost-Efficient Way To Develop And Verify Space Applications.", AIAA SPACE, Long Beach, California
- Rossmann, J.; Schlette, C.; Emde, M.; Sondermann, B. 2011. "Advanced Self-Localization and Navigation for Mobile Robots in Extraterrestrial Environments.", Computer Technology and Application, Vol. 2, Number 5
- Rossmann, J.; Steil, T., Springer, M.; 2012. "Validating the Camera and Light Simulation of a Virtual Space Robotics Testbed by Means of Physical Mockup Data", i-SAIRAS, Turin, Italy
- Rossmann, J.; Jung, T.; Rast, M. 2010. "Developing virtual testbeds for tasks in research and engineering." WINVR, Ames, USA
- Kurtev, I. and van den Berg, K. 2004. "Mistral: A language for model transformations in the MOF meta-modeling architecture." MDFAFA, Linköping, Sweden
- Gyssens, M.; Paredaens, J.; van den Bussche, J.; van Gucht, D. 1994. "A graph-oriented object database model. IEEE Transactions on Knowledge and Data Engineering 6
- Kaigom, E., Rossmann, J. 2012. "Simulation of electrically-driven robot manipulators." ICMA, Chengdu, China

ESTIMATING PROJECT RISKS' PROBABILITY WITH LIMITED STATISTICAL DATA

Michael Douloumpekis

Vrassidas Leopoulos

Elena Rokou

School of Mechanical Engineering
National Technical University of Athens
Heroon Polytechniou 9, 15780, Athens
Greece

Konstantinos Kirytopoulos
School of Natural and Built Environments
University of South Australia
City East Campus, Adelaide
Australia

E-mail: erokou@mail.ntua.gr

KEYWORDS

Risk Management, Monte Carlo Simulation, Expert judgments.

ABSTRACT

The main objective of project risk management is to “increase the probability and impact of positive events, and decrease the probability and impact of negative events in the project” (Project Management Institute, 2012).

In order to assess a risk's probability of occurrence, statistical data, expert judgment and/or modeling predictions can be used (Siu and Kelly, 1998). In this paper, expert judgment is modeled as a probability distribution function and is combined with statistical data, concerning similar projects carried out in the past. The Bayes theorem is used to combine the expert's opinion with the statistical data and get a new probability distribution function. The standard deviation of this new probability distribution function, describes the uncertainty of the assessment.

The idea is to provide a method that covers the case that we don't have a satisfactory number of statistical data but expert opinions can be gathered and used to fill the gap. The whole process is supported by Monte Carlo simulation techniques. Furthermore, the method balances the effect of expert's opinions in such a way that the more statistical data we have the less weight is given to the expert's opinion. A simple test case, taken from the construction field, is used to showcase the method.

INTRODUCTION

Project risk management uses expert judgment and statistical data, as well as other sources of data, to define a wide variety of project's features like project's cost and duration.

Project risk management often involves the analysis of low-probability events for which few data are available. As pointed out by (Elmaghraby, 2005a), use of single values for these parameters, or point estimations, could be extremely misleading. A commonly used approach to handle uncertainty is to represent the expert's opinion as a probability distribution function. Much attention should be given when computing probability distributions, as expert judgments are subjective opinions of the people that give the judgments and thus can easily lead to erroneous results (Siu and Kelly, 1998). On the

other hand, acquisition of statistical data may be expensive or impossible due to lack of similar projects previously implemented as in the case of innovative projects.

In practice, Bayesian parameter estimation techniques are used to combine a variety of information types like expert judgments with statistical data, during the estimation process. This way the disadvantages of classical statistical methods, like the need of large statistical samples to give accurate results, are overcome. Bayesian techniques are well-fitted for project risk management as they are based on the concept of subjective probability where probability is defined as a subjective measure of event likelihood, as proposed by current theories on decision-making under uncertainty. Finally, from a practical point of view another advantage of using Bayesian techniques propagation of uncertainties through complex models is that it is relatively simple (Siu and Kelly, 1998).

In this paper a combination of the expert judgment with statistical data, concerning similar projects carried out in the past, is proposed. The main objective of this paper is the probability risk assessment in cases that the statistical sample is limited and cannot lead to safe conclusions. The proposed method uses the expert judgments to make up for the small statistical sample. The idea is to get a better quality of posterior probability distribution using both statistical data, even if there are very limited, and expert's judgment to get a better probability risk assessment.

The rest of the paper is organized as follows: in section 2 a brief literature review of the methods used to compute risk probabilities is presented, in section 3 the proposed approach is in depth described, in section 4 a case study is used to showcase the method's implementation. Finally, Section 5 presents conclusions and directions for future research.

LITERATURE REVIEW

There have been a lot of studies around the appropriateness of Bayesian statistics for the probabilistic analysis of rare events (Apostolakis and Mosleh, 1979, Martz and Bryson, 1984, Mosleh and Apostolakis, 1986, Apostolakis, 1990, Levin et al., 1992, Bowers, 1994, Sanders and Ritzman, 1995, Tschang and Dowlatabadi, 1995), leading to the safe conclusion that Bayesian statistics are the appropriate framework within which expert opinions can be combined with experimental results and statistical observations to produce quantitative measures of the risks.

Usually, as in the case presented by (Tschang and Dowlatabadi, 1995) statistical data are combined with some prior probability distributions, representing the beliefs for input parameters in order to generate a new probability distribution. The use of Bayesian parameter estimation techniques, when a probability risk assessment is required, is in depth analyzed by Siu and Kelly (1998) and the corresponding advantages are stressed out, like the fact that the Bayesian techniques have the ability to incorporate a wide variety of information types and the effectiveness and efficiency of using probability distributions to reflect the uncertainty about the parameters needed to be assessed.

Having selected the Bayesian techniques as the vehicle for combining statistical data with expert's judgments, the next step is to gather the actual data to be used. The conduction of the quantitative risk analysis using appropriate data is of extreme importance as the use of inappropriate data may result in misleading conclusions. (Bowers, 1994) notes that data may be extracted from experience relative to the project in hand and statistical data, derived from similar projects carried out in the past. Thus, great attention should be given when selecting the statistical data to be used, which must refer to projects carried out in a similar environment and project phase. For example data taken before and during the projects executions should not be mixed up.

Expert judgment is extremely useful when the project's environment alters, because of human's ability to quickly research and assimilate these changes (Remus et al., 1995) or occasions that there is a lack of sufficient qualitative statistical data (Levin et al., 1992).

Attention should be given to the fact that the subjectivity of expert judgment, in combination with the analyst's prejudice concerning the results of analysis, can endanger the assessment's quality (Levin et al. 1992).

Furthermore, recent work in the field shifts the focus on how to enhance the results obtained by risk analysis by combining traditional techniques to more sophisticated decision support techniques and knowledge extraction methods (Elmaghraby, 2005b, Ngai and Wat, 2005, Perminova et al., 2008, Kutsch and Hall, 2010, Mojtahedi et al., 2010, Lee and Cho, 2012, Sousa and Einstein, 2012).

PROPOSING AN ALTERNATIVE APPROACH FOR ESTIMATING THE RISK PROBABILITY

In this study the goal is to enhance forecast accuracy by combining judgmental forecasts with those generated by statistical models as (Sanders and Ritzman, 1995) suggest. Our study differs from prior research efforts in this area along two important dimensions. On one hand it is how to assess risks in cases where statistical data are very difficult to gather, thus giving a distorted view of the realistic probabilities of risk occurrence due to unsatisfactory size of the sample. The second dimension concerns balancing the expert judgments by employing past experience computed as probability of occurrence of the events being examined in such a way that the bigger the statistical sample is the less the subjective opinion of experts is utilized.

Our prior belief for the probability of a risk's occurrence is provided by expert judgment. There are many ways proposed in related studies (Apostolakis and Mosleh, 1979, Martz and Bryson, 1984, Mosleh and Apostolakis, 1986, Levin et al., 1992, Sanders and Ritzman, 1995, Siu and Kelly, 1998) to transform expert judgment in probability distributions. Herein

we relate expert judgment with a range of values: $[\pi - \pi * \lambda\%, \pi + \pi * \kappa\%]$, where $\kappa\%$, $\lambda\%$ indicate the uncertainty of expert judgment, against the point estimation π . In other words $\lambda\%$ and $\kappa\%$ represent the expected error in the estimation of the probability π that a specific risk will occur during the project execution. This range of values can be transformed in a beta distribution function computing beta probability's distribution moments, via the equations (1) and (2):

$$mean = \frac{a}{a+b} \quad (1)$$

$$and \sigma = \sqrt{\frac{ab}{(a+b)^2(a+b+1)}} \quad (2)$$

We assume that $\sigma \cong \frac{max-min}{6}$, where $max = \pi + \pi * \kappa\%$, $min = \pi - \pi * \lambda\%$, and a, b are the parameter's of the beta probability's distribution.

Consequently, we can estimate σ as

$$\sigma = \sqrt{\frac{ab}{(a+b)^2(a+b+1)}} \cong \frac{\pi(\kappa\%+\lambda\%)}{6} \quad (3)$$

It is also accepted (Richardson, 2010) that the point estimation, π , equals to mode

$$\pi = mode = \frac{a-1}{a+b-2} \quad (4)$$

In the herein proposed approach we use mode instead of mean based on the assumption that mean does not differ much from the mode, as the used values of a and b are relatively large. For example, if $\pi = 0,30$, $\kappa\% = 20\%$ and $-\lambda\% = -20\%$, σ is just 0,0200. Solving the equations as follows results in computing the values of a and b. Hence,

$$mean = 0,3000 = \frac{a}{a+b} \Rightarrow b = \frac{7}{3}a$$

$$\sigma^2 = \frac{ab}{(a+b)^2(a+b+1)} = 0,0004 \Rightarrow a = 157,2 \text{ and } b = 366,8$$

Computing the mode, would give $mode = 0,2992$ that is very close to $mean = 0,3000$. Values a and b are even larger in cases of smaller expert judgment's uncertainty, thus there is even less loss of accuracy when using mode instead of mean. In order to have even greater accuracy this estimation could be updated numerically through a repetitive method.

In the proposed method the beta probability distribution function is chosen to represent the expert judgments. Its parameters - a and b - can be easily combined with the binomial probability distribution function's parameters - x and n, where n is the size of the available statistical data sample that is used to examine the occurrence- or not- of the risk being analyzed and x the number of risk's occurrences. The statistical data sample should be very carefully formed because without competent data all the analysis techniques used to assess the project risks will lead to worthless results (Bowers, 1994). Statistical data should be extracted from similar projects carried out in the past, independently from each other, under similar conditions.

The proposed process is shown in Figure 1. Starting from an initial set of historical data, of relatively small size, we initially eliminate those projects that are dependent to one another and then carefully check the remaining data so as to keep only those projects' data about risks that refer to the same execution stage and to similar circumstances. The remaining data will compose the statistical sample. The occurrences of each risk

being analyzed, x , and the size, n , of the statistical sample being used, will be calculated.

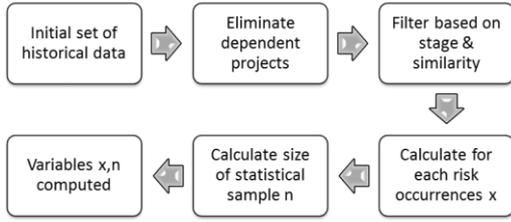


Figure 1: Selection of Statistical Data

The number of risk's occurrences, x , in the n independent past projects contained in the given set of data are the parameters of the binomial distribution function, as

$$P(x) = \binom{n}{x} p^x (1-p)^{n-x}, x = 0,1,2, \dots, n \quad (5)$$

The parameters x , n will be combined with the beta distribution function's parameters a , b to provide a posterior distribution function for the risk's occurrence probability (Figure 2). Parameters a and b are calculated using equations (1) and (2).

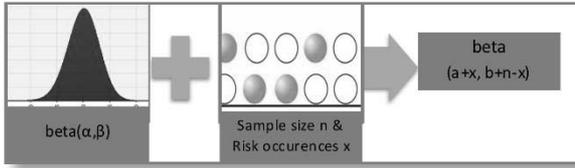


Figure 2: Exporting the Posterior Distribution

More specifically, for the binomial probability:

$$f(x|p) = P(x) \quad (6)$$

While, when the expert judgment is expressed as a beta distribution we have:

$$f(p) = \text{beta}(\alpha, \beta) \Rightarrow f(p) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1} (1-p)^{\beta-1} \quad (7)$$

Therefore, by the Bayes theorem, it is implied that

$$f(p|x) = \frac{f(x|p)f(p)}{f(x)} = \frac{\binom{n}{x} p^x (1-p)^{n-x} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1} (1-p)^{\beta-1}}{\alpha p^{a+x-1} (1-p)^{\beta+n-x-1}} \Rightarrow f(p|x) \sim \text{beta}(a+x, \beta+n-x) \quad (8)$$

Finally, we get the posterior probability distribution that represents an updated probability risk assessment of the analyzed risk, characterized by reduced uncertainty.

$$P|x \sim \text{beta}(a+x, \beta+n-x) \quad (9)$$

The proposed process (Figure 3) starts with the formation of the statistical sample based on historical data taken from similar projects that were in the same execution phase, then the occurrences of the risks being assessed are calculated and the parameters expressing the uncertainty of the expert's judgments are entered. Finally, the beta distribution function representing the experts judgments is formed and combined with the statistical data using $\text{beta}(x+a, b+n-x)$.

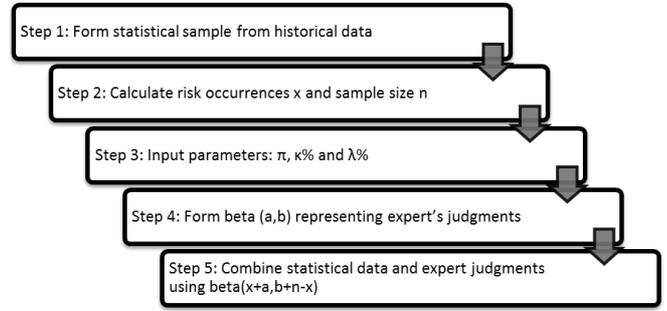


Figure 3: Probability Risk Assessment

It should be noted here that, the above formulation gives more weight to the statistical data rather than to the expert judgment as the sample gets bigger. This is a desired reaction, as expert judgments are used to get better results in cases that large statistical samples are not available and not easy to get, but in cases where satisfactory sizes of statistical samples exist the experts judgments should not cause great changes to the final risk assessment.

TEST CASE

The applicability of the proposed method has been tested in a company activating in office development. The company is part of a major group operating, for over fifty years, in the sector of construction industry. Within this period the group has executed a multitude of worth trying projects in several countries. The group has achieved to be one of the most productive and effective International Building and Civil Engineering Contractors, operating efficiently in several countries, mainly in the Middle East, Asia, Africa and Europe. The present study concerns the Greek market where the group begun operating in the early seventies.

The aim of the development of the proposed method was the assessment of the identified risks by combining a small statistical sample to the expert's judgments on the probability of occurrence of each risk in order to get more realistic results and thus take better decisions on how to face each identified risk.

The process described in section 3, is used to assess the risks (listed in Table 1) that were identified in a previous stage of the risk management process. More specifically, the risk assessment is based on a relatively small set of statistical data from this company's previous projects combined to expert judgments acquired by interviewing the most experienced project managers employed in the company.

Table 1: Identified Risks

# risk	Description of risk
1	Archaeologists' intervention in the building's design standards
2	Finding of ancient items
3	Earthquake
4	Non- final project's acceptance by the client
5	Failure to find sub-contractors in the area nearby the project
6	Low subcontractors' productivity
7	Changes of executive project members
8	Accident or sabotage
9	Differences between project planning and implementation due to the architect's fault

10	Changes of constructed parts due to company's fault
11	Changes of constructed parts due to subcontractors' fault
12	Project's complexity
13	Sue for infringement of hours' siesta

For each identified risk, five experts were asked to give the probability of occurrence based on their experience from similar past projects and taking into consideration the specifics of the under examination project. The results of this step are shown in Table 2.

Table 2: Probability of Risks' Occurrence (expert judgment) and Statistical Data

# risk	Probability of occurrence	x	n
1	30,00%	0	20
2	70,00%	1	20
3	30,00%	0	20
4	10,00%	0	20
5	30,00%	2	20
6	80,00%	0	20
7	10,00%	1	20
8	5,00%	0	20
9	70,00%	5	20
10	70,00%	1	20
11	80,00%	0	20
12	80,00%	0	20
13	80,00%	1	20

On the other hand, having an initial sample of 30 similar projects, a statistical sample of 20 projects (n parameter in equation (9)) was formed based on the commonalities to the in progress project and the occurrences of each risk are calculated, giving the results depicted in Table 2 (columns "x" and "n").

Then the expert judgment was combined with the statistical data to give a more accurate estimation for the risks' probability of occurrence. For this, parameters a, b of the beta probability distributions are computed based on (1) and (2) equations and then the statistical data were combined to the expert's judgments using equation (9) where the input parameters κ , λ were set to $\kappa = +15\%$, $-\lambda = -10\%$. Finally, Monte Carlo simulation was used to calculate the risks by generating inputs randomly from the beta probability distribution over the domain and then aggregate the results, as shown in Figure 4.

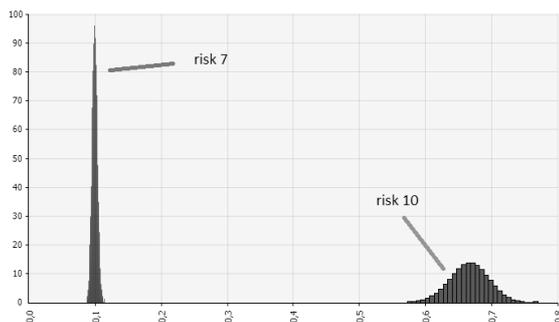


Figure 4: Combination of Expert Judgment with Statistical Data for Two Risks

To see the behaviour of two risks' estimation, when expert judgment's probability of risk's occurrence is close and away from the relative frequency, given by the x/n ratio, we take an insight on risks 7 and 10, as shown in Table 3.

Table 3 Probability of Occurrence and x/n Ratio

# risk	Prob. of occurrence	x/n
7	10,00%	1/20=0.05
10	70,00%	1/20=0.05

What we can observe is the increase of standard deviation when expert's judgment varies from statistical data much. Also, for risk 7, it seems that the pdf's mean is too sensitive and tends to equal expert's opinion of risk occurrence, while for risk 10, mean moves only slightly, as shown in Figure 5.

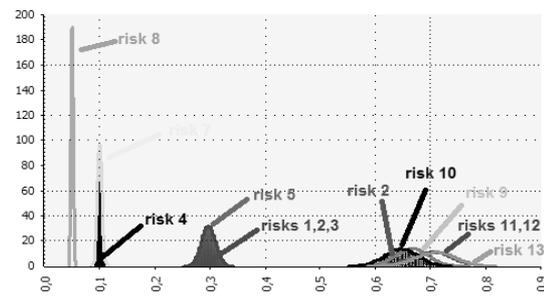


Figure 5: Combination of Expert Judgment with Statistical Data for all Project's Risks

Of course, standard deviation will reduce more and pdfs' mean movement will be slighter when the number n, of similar projects tested, be increased.

CONCLUSIONS

In this paper, it has been stated that the lack of qualitative data can be addressed by using expert judgments. The influence of expert judgments in the probability risk assessment is reduced as the amount of statistical data is increased. The risk assessment is calculated by combining statistical data and expert judgment using beta distribution and posterior probability through Bayesian techniques.

The rate, with which the expert judgment is being corrected, is higher when expert expresses a larger uncertainty about the estimation he makes. In those situations, that the expert knows little about the risk's probability of occurrence he is about to assess, the parameters describing this uncertainty (κ , λ) downsize the effect of the judgment on the final result.

If the statistical data differentiate much the prior probability distribution from the posterior probability distribution, statistical data should be filtered to attenuate the effect. On the other hand when statistical data are sufficient to lead to a safe conclusion then the expert judgment even if it is used will not cause differentiation on the final result.

However, when there is a small sample of statistical data, the analyst should be aware of bias included (Siu and Kelly, 1998) even when using the proposed approach, that will attenuate the effects of small statistical samples but cannot replace the accuracy of a large and accurate statistical sample.

Concluding, the proposed method enhances the risk assessment process in cases that the provided statistical sample is limited in size by combining it with expert judgments using a beta distribution. Preliminary results of the application of the method are very promising but still a large-scale application of the method and then comparison of the given results with the other methods proposed in the literature will provide valuable information both on the applicability of the method and its efficiency.

It would be interesting to combine risk's probability of occurrence, with their impact -monetary or time- to shape a risk response plan.

ACKNOWLEDGEMENTS

This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: THALES. Investing in knowledge society through the European Social Fund.

REFERENCES

- Apostolakis, G. 1990. The concept of probability in safety assessments of technological systems. *Science*, 250, 1359-1364.
- Apostolakis, G. & Mosleh, A. 1979. EXPERT OPINION AND STATISTICAL EVIDENCE: AN APPLICATION TO REACTOR CORE MELT FREQUENCY. *Nucl Sci Eng*, 70, 135-149.
- Bowers, J. A. 1994. Data for project risk analyses. *International Journal of Project Management*, 12, 9-16.
- Elmaghraby, S., E. 2005a. On the fallacy of averages in project risk management. *European Journal of Operational Research*, 165, 307-313.
- Elmaghraby, S. E. 2005b. On the fallacy of averages in project risk management. *European Journal of Operational Research*, 165, 307-313.
- Kutsch, E. & Hall, M. 2010. Deliberate ignorance in project risk management. *International journal of project management*, 28, 245-255.
- Lee, K. C. & Cho, H. 2012. Integration of General Bayesian Network and ubiquitous decision support to provide context prediction capability. *Expert Systems with Applications*, 39, 6116-6121.
- Levin, R. I., Rubin, D. S., Stinson, J. P. & Gardner, E. S. 1992. *Quantitative approaches to management*, New York, McGraw-Hill Book Company.
- Martz, H. F. & Bryson, M. C. 1984. A Statistical Model for Combining Biased Expert Opinions. *Reliability, IEEE Transactions on*, R-33, 227-232.

- Mojtahedi, S. M. H., Mousavi, S. M. & Makui, A. 2010. Project risk identification and assessment simultaneously using multi-attribute group decision making technique. *Safety Science*, 48, 499-507.
- Mosleh, A. & Apostolakis, G. 1986. The Assessment of Probability Distributions from Expert Opinions with an Application to Seismic Fragility Curves. *Risk Analysis*, 6, 447-461.
- Ngai, E. W. T. & Wat, F. K. T. 2005. Fuzzy decision support system for risk analysis in e-commerce development. *Decision Support Systems*, 40, 235-255.
- Perminova, O., Gustafsson, M. & Wikström, K. 2008. Defining uncertainty in projects – a new perspective. *International journal of project management*, 26, 73-79.
- Project Management Institute 2012. A guide to the project management body of knowledge (PMBOK® Guide). Newtown Square, Pa.: Project Management Institute, Inc.
- Remus, W., O'connor, M. & Griggs, K. 1995. Does reliable information improve the accuracy of judgmental forecasts? *International Journal of Forecasting*, 11, 285-293.
- Richardson, T. 2010. Lecture notes distributed in the Introduction to Mathematical Statistics Washington: Statistics Department of University of Washington.
- Sanders, N. R. & Ritzman, L. P. 1995. Bringing judgment into combination forecasts. *Journal of Operations Management*, 13, 311-321.
- Siu, N. O. & Kelly, D. L. 1998. Bayesian parameter estimation in probabilistic risk assessment. *Reliability Engineering and System Safety*, 62, 89-116.
- Sousa, R. L. & Einstein, H. H. 2012. Risk analysis during tunnel construction using Bayesian Networks: Porto Metro case study. *Tunnelling and Underground Space Technology*, 27, 86-100.
- Tschang, F. T. & Dowlatabadi, H. 1995. A Bayesian technique for refining the uncertainty in global energy model forecasts. *International Journal of Forecasting*, 11, 43-61.

AUTHOR BIOGRAPHY

VRASSIDAS LEOPOULOS is a Mechanical and Electrical Engineer specialized in Industrial Engineering, serving at the Sector of Industrial Management and Operational Research, School of Mechanical Engineering, National Technical University of Athens (NTUA) as Associate Professor. He holds a PhD of the university PARIS IX Dauphine. He specialises in Quality Management Systems, Project Management, Industrial Facilities Planning and Industrial Information Technology. He is responsible for the research activities of the Metrotechnic Laboratory of the School of Mechanical Engineers. His main research interests are quality management, risk management, project management and information systems for the industrial sector.

COMMERCIAL TOOLS

MULTI-METHOD MODELING

Andrei Borshchev
The AnyLogic Company
49 Nepokorenykh ave
St.Petersburg, 195220
Russia
E-mail: andrei@anylogic.com

KEYWORDS

Combined, Discrete, Continuous, System Dynamics, Agent Based, Marketing, Healthcare

ABSTRACT

Frequently, the problem cannot completely conform to one of the three existing modeling paradigms (discrete event, system dynamics, or agent based modeling). Thinking in terms of a single-method modeling language, the modeler inevitably either starts using workarounds (unnatural and cumbersome constructs), or just leaves part of the problem outside the scope of the model (treats it as exogenous). If our goal is to capture business, economic, and social systems in their interaction, this becomes a serious limitation. In this paper we offer an overview of most used multi-method (or multi-paradigm) model architectures, discuss the technical aspects of linking different methods within one model, and consider examples of multi-method models. The modeling language of AnyLogic is used throughout the paper..

INTRODUCTION

The three modeling methods, or paradigms, that exist today, are essentially the three different viewpoints the modeler can take when mapping the real world system to its image in the world of models.

- The **system dynamics** (SD) paradigm suggests to abstract away from individual objects, think in terms of aggregates (stocks, flows), and the feedback loops.
- The **discrete event modeling** (DE) adopts a process-oriented approach: the dynamics of the system are represented as a sequence of operations performed over entities.
- In an **agent based model** the modeler describes the system from the point of view of individual objects that may interact with each other and with the environment.

Depending on the simulation project goals, the available data, and the nature of the system being modeled, different problems may call for different methods. Also, sometimes it is not clear at the beginning of the project which abstraction level and which method should be used. The modeler may start with, say, a highly abstract system dynamics model and switch later on to a more detailed discrete event model. Or, if the system is heterogeneous, the different components may be best described by using different methods. For example in

the model of a supply chain that delivers goods to a consumer market the market may be described in system dynamics terms, the retailers, distributors, and producers may be modeled as agents, and the operations inside those supply chain components – as process flowcharts.

Frequently, the problem cannot completely conform to one modeling paradigm. A "single-method- minded" modeler inevitably either starts using workarounds (unnatural and cumbersome language constructs), or just leaves part of the problem outside the scope of the model. If our goal is to capture business, economic, and social systems in their interaction, this becomes a serious limitation. In this paper we offer an overview of most used multi-method model architectures, discuss the technical aspects of linking different methods within one model, and consider two examples of multi-method models:

- Consumer market and supply chain
- Epidemic and clinic.

MULTI-METHOD MODEL ARCHITECTURES

The number of possible multi-method model architectures is infinite, and many are used in practice. Popular examples are shown in the Figure 1. In this section we briefly discuss the problems where these architectures may be useful.

Agents in an SD environment. Think of a demographic model of a city. People work, go to school, own or rent homes, have families, and so on. Different neighborhoods have different levels of comfort, including infrastructure and ecology, cost of housing, and jobs. People may choose whether to stay or move to a different part of the city, or move out of the city altogether. People are modeled as agents. The dynamics of the city neighborhoods may be modeled in system dynamics way, for example, the home prices and the overall attractiveness of the neighborhood may depend on crowding, and so on. In such a model agents' decisions depend on the values of the system dynamics variables, and agents, in turn, affect other variables.

The same architecture is used to model the interaction of public policies (SD) with people (agents). Examples: a government effort to reduce the number of insurgents in the society; policies related to drug users or alcoholics.

Agents interacting with a process model. Think of a business where the service system is one of the essential components. It may be a call center, a set of offices, a Web server, or an IT infrastructure. As the client base grows, the system load increases. Clients who have different profiles and histories use the system in different ways, and their

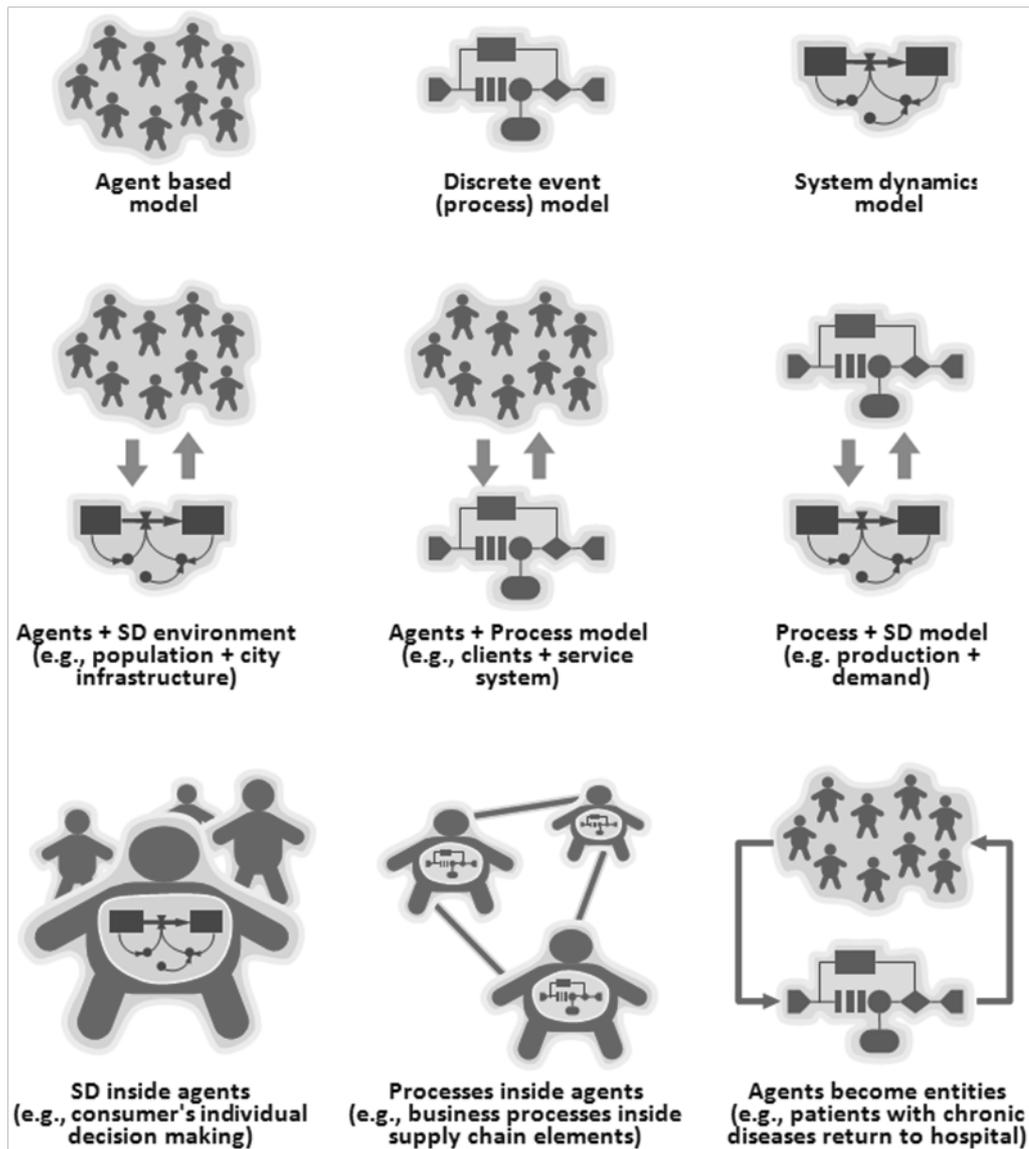


Figure 1: Popular Multi-Method Model Architectures

future behavior depends on the response. For example, low-quality service may lead to repeated requests, and, as a result, frustrated clients may stop being clients. The service system is naturally modeled in a discrete event style as a process flowchart where requests are the entities and operators, tellers, specialists, and servers are the resources. The clients who interact with the system are the agents who have individual usage patterns.

Note that in such a model the agents can be created directly from the company CRM database and acquire the properties of the real clients. This also applies to the modeling of the company's HR dynamics. You can create an agent for every real employee of the company and place them in the SD environment that describes the company's integral characteristics (the first architecture type).

A process model linked to a system dynamics model. The SD aspect can be used to model the change in the external conditions for an established and ongoing process: demand variation, raw material pricing, skill level, productivity, and other properties of the people who are part of the process.

The same architecture may be used to model manufacturing processes where part of the process is best described by continuous time equations – for example, tanks and pipes, or a large number of small pieces that are better modeled as quantities rather than as individual entities. Typically, however, the rates (time derivatives of stocks) in such systems are piecewise constants, so simulation can be done analytically, without invoking numerical methods.

System dynamics inside agents. Think of a consumer market model where consumers are modeled individually as agents, and the dynamics of consumer decision making is modeled using the system dynamics approach. Stocks may represent the consumer perception of products, individual awareness, knowledge, experience, and so on. Communication between the consumers is modeled as discrete events of information exchange.

A larger-scale example is interaction of organizations (agents) whose internal dynamics are modeled as stock and flow diagrams.

Processes inside agents. This is widely used in supply chain modeling. Manufacturing and business processes, as well as

the internal logistics of suppliers, producers, distributors and retailers are modeled using process flowcharts. Each element of the supply chain is at the same time an agent. Experience, memory, supplier choice, emerging network structures, orders and shipments are modeled at the agent level.

Agents temporarily act as entities in a process. Consider patients with chronic diseases who periodically need to receive treatment in a hospital (sometimes planned, sometimes because of acute phases). During treatment, the patients are modeled as entities in the process. After discharge from the hospital, they do not disappear from the model, but continue to exist as agents with their diseases continuing to progress until they are admitted to the hospital again. The event of admission and the type of treatment needed depend on the agent's condition. The treatment type and timeliness affect the future disease dynamics.

There are models where each entity is at the same time an agent exhibiting individual dynamics that continue while the entity is in the process, but are outside the process logic – for example, the sudden deterioration of a patient in a hospital

The Choice of the Model Architecture and Methods

The choice of the model architecture, the abstraction level(s), and the method(s) always depends on the problem you are solving. Among other things, this choice should be governed by the criterion of naturalness. Compact, minimalistic, clean, beautiful, easy to understand and explain – if the internal texture of your model is like that, then your choice was most probably right. The design patterns and model examples further in this paper are given in the AnyLogic modeling language (see, for example, Borshchev and Filippov 2004). This is a multi-method object-oriented language designed on the basis of "no workarounds" principle. This language allows you to create model architectures of arbitrary type and complexity, including all previously mentioned.

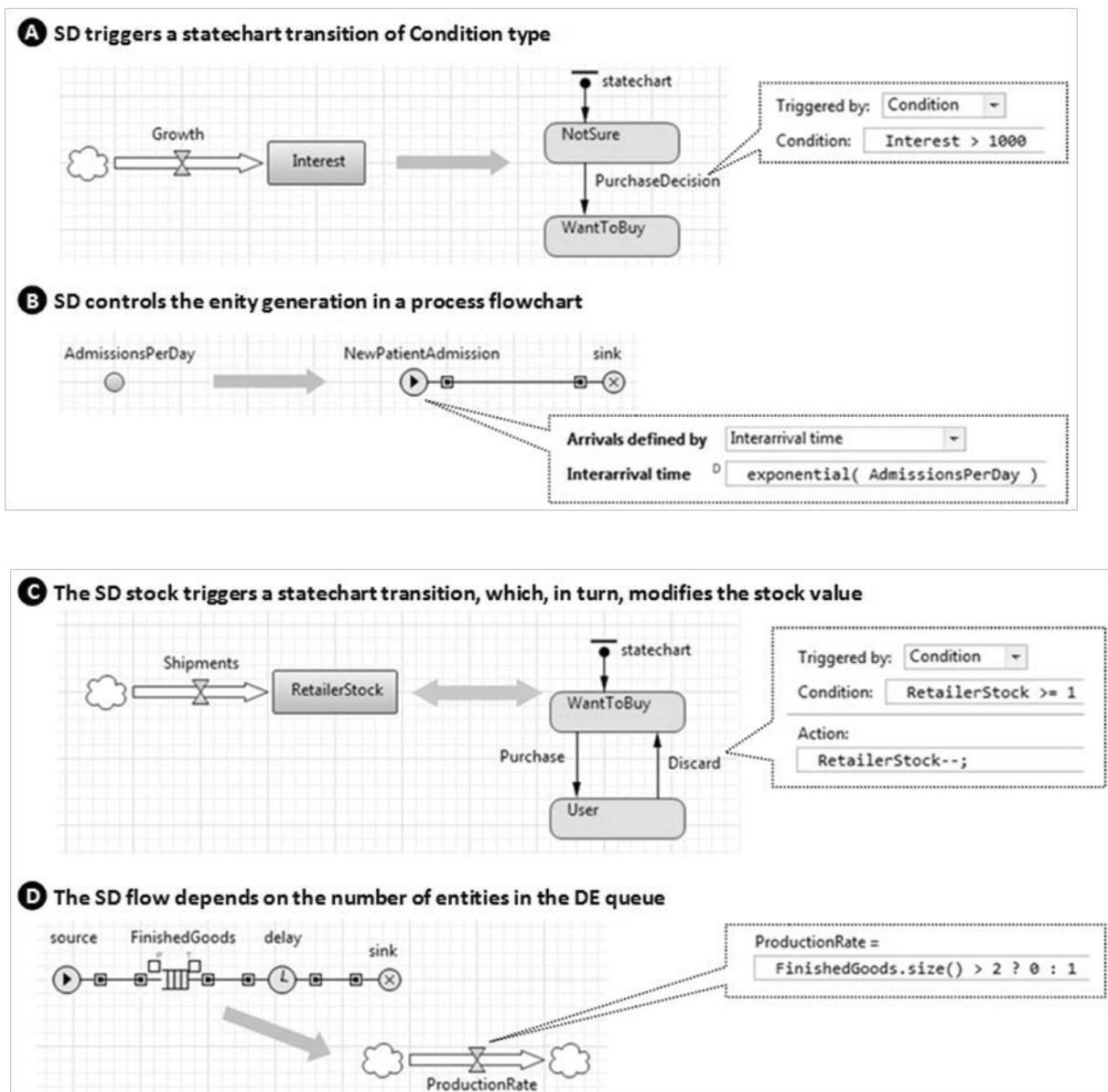


Figure 2: Interaction Between System Dynamics and Discrete Components of the Model

TECHNICAL ASPECT OF COMBINING DIFFERENT MODELING METHODS

In this section we will consider the techniques of linking different modeling methods together. Important feature of the modeling language we are using is that all model elements of all methods, be they SD variables, statechart states, entities, process blocks, exist in the "same namespace": any element is accessible from any other element by name (and, sometimes, "path" – the prefix describing the location of the element in the model hierarchy). The following examples are all taken from real projects and purged of all unnecessary details. This set, of course, does not cover everything, but it does give a good overview of how you can build interfaces between different methods.

System Dynamics Impacts Discrete Elements

The system dynamics model is a set of continuously changing variables. All other elements in the model work in discrete time (where any changes are associated with events). SD itself does not generate any events, so it cannot actively make an impact on agents, process flowcharts, or other discrete time constructs. The only way for the SD part of the model to impact a discrete element is to have that element watch on a condition over SD variables, or to use SD variables when making a decision. The Figure 2 shows some possible constructs.

In the Figure 2A a change of an SD variable triggers a statechart transition. Events (low-level constructs that allow scheduling a one-time or recurrent action) and statechart transitions are frequent elements of agent behavior. Among other types of triggers, both can be triggered by a condition – a Boolean expression. If the model contains dynamic

variables, all conditions of events and statechart transitions are evaluated at each integration step, which ensures that the event or transition will occur exactly when the (continuously changing) condition becomes true. In the figure the statechart is waiting for the *Interest* stock to rise higher than a given threshold value. In the Figure 2B the flowchart source block *NewPatientAdmissions* generates new entities at the rate defined by the dynamic variable *AdmissionsPerDay*, which may be a part of a stock and flow diagram.

Discrete Elements Impact System Dynamics

Consider the Figure 2C. The SD stock triggers a statechart transition, which, in turn, modifies the stock value. Here, the interface between the SD and the statechart is implemented in the pair condition/action. In the state *WantToBuy*, the statechart tests if there are products in the retailer stock, and if there are, buys one and changes the state to *User*.

You are free to change the values of the system dynamic stocks from outside the system dynamics part of the model at any time. This does not conflict with the differential equation integration: the solver will just start at the new value. However, trying to change the value of a flow or auxiliary variable that has an equation associated with it, is not correct: the assigned value will be immediately overridden by the equation, so the assignment will have no effect.

DE objects can be referenced in a SD formula. In the Figure 2D, the flow *ProductionRate* switches between 0 and 1, depending on whether the finished products inventory (the number of entities in the queue *FinishedGoods* returned by the function *size()*) is greater than 2 or not. Again, one can close the loop by letting the SD part control the production process.

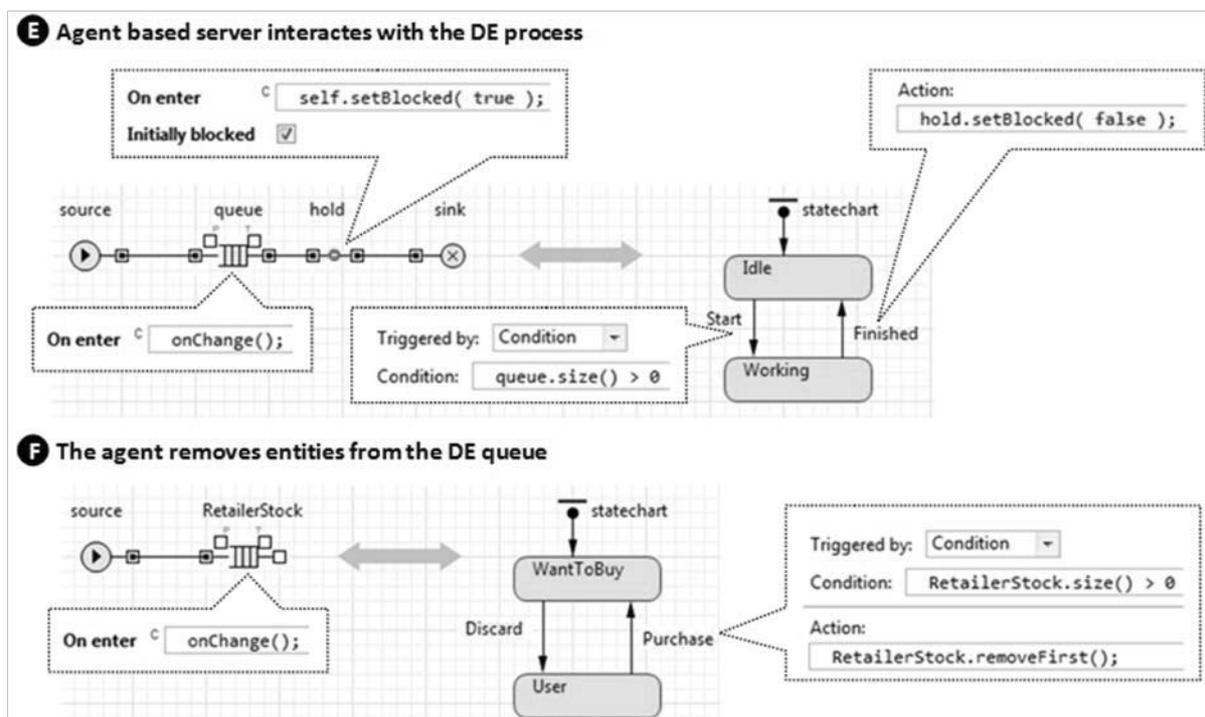


Figure 3: Interaction of Agent Based and Discrete Event Components of the Model

Interaction of Agent Based and Discrete Event Components of the Model

In the Figure 3E a server in the DE process model is implemented as an agent. Imagine complex equipment, such as a robot or a system of bridge cranes. The behavior of such objects is often best modeled "in agent based style" by using events and statecharts. If the equipment is a part of the manufacturing process being modeled, you need to build an interface between the process and the agent representing the equipment.

In this example, the statechart is a simplified model of a piece of equipment. When the statechart comes to the state *Idle*, it checks if there are entities in the queue. If yes, it proceeds to the *Working* state and, when the work is finished, unblocks the *hold* object, letting the entity exit the queue. The *hold* object is set up to block itself again after the entity passes through.

The next entity will arrive when the equipment is in the *Idle* state. To notify the statechart, we call the function `onChange()` upon each entity arrival (see the *On enter* action of the queue). This is necessary because, unlike in the models with continuously changing SD elements, in the

models built of purely discrete elements events and transitions triggered by a condition, do not monitor the condition continuously.

In the Figure 3F the agent removes entities from the DE queue. Here, the supply chain is modeled using discrete event constructs; in particular, its end element, the retailer stock, is a Queue object. The consumers are outside the discrete event part; they are modeled as agents. Whenever a consumer comes to the state *WantToBuy*, it checks the *RetailerStock* and, if it is not empty, removes one product unit. Again, as this is a purely discrete model, we need to ensure that the consumers who are waiting for the product are notified about its arrival – that's why the code `onChange()` is placed in the *On enter* action of the *RetailerStock* queue.

In this simplified version, there is only one consumer whose statechart is located "on the same canvas" as the supply chain flowchart. In the full version there would be multiple agents-consumers and, instead of calling just `onChange()`, the retailer stock would notify every consumer..

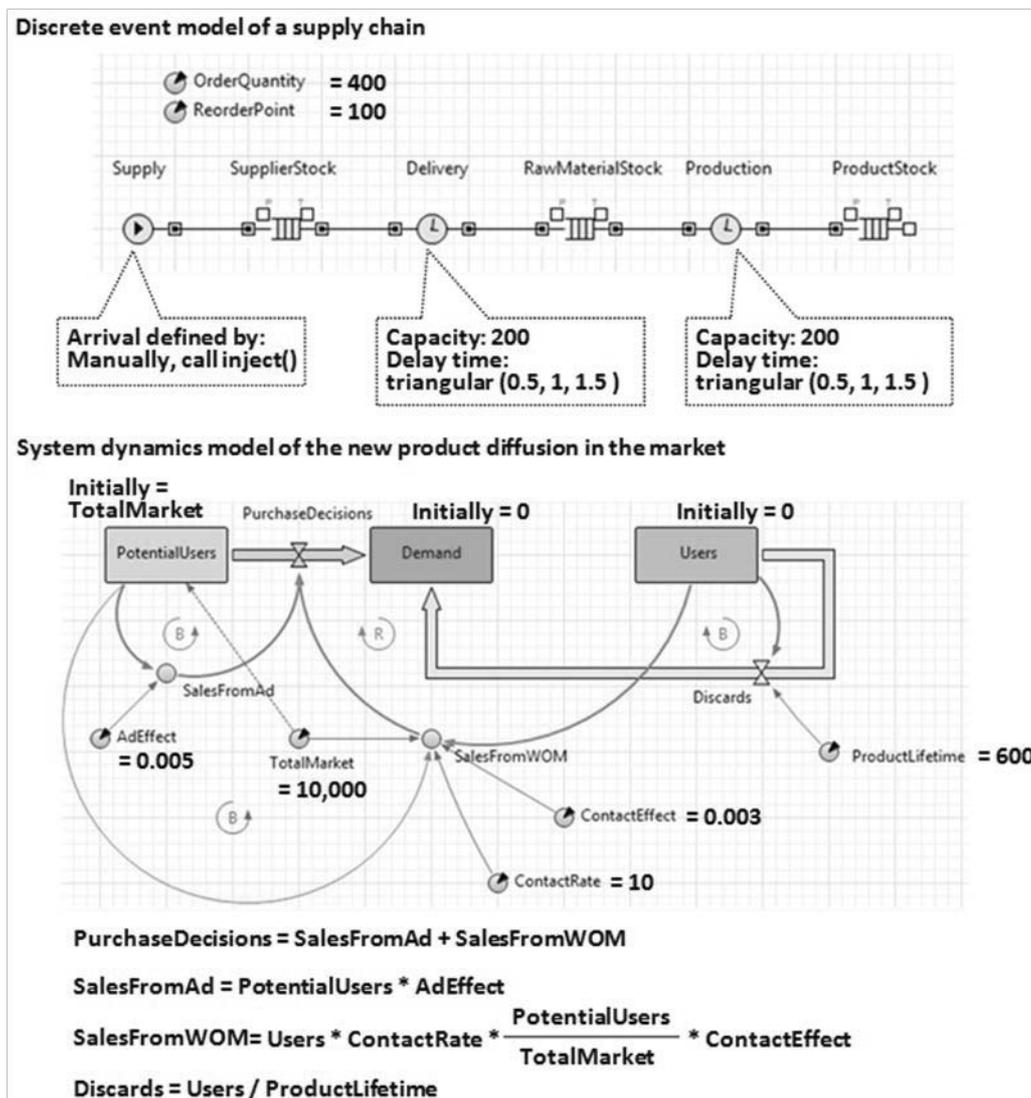


Figure 4: DE Model of the Supply Chain and SD Model of the Market (so far, not linked)

EXAMPLES OF MULTI-METHOD MODELS

Consumer Market and Supply Chain

We will model the supply chain and sales of a new product in a consumer market in the absence of competition. The supply chain will include delivery of the raw product to the production facility, production, and stock of the finished products. The QR inventory policy will be used. Consumers are initially unaware of the product; advertizing and word of mouth will drive the purchase decisions. The product has a limited lifetime, and 100% of users will be willing to buy a new product to replace the old one. The full version of the model is available at RunTheModel.com.

We will use discrete event methodology to model the supply chain, and system dynamics methodology, namely, a slightly modified Bass diffusion model (Bass 1969), to model the market. We will link the two models through the purchase events.

The supply chain flowchart (top of the Figure 4) includes three stocks: the supplier stock of raw material, the stock of raw material at the production site, and the stock of finished products at the same location. Delivery and production are modeled by the two Delay objects with limited capacity. The *Supply* block of the flowchart is not generating any entities unless explicitly asked to do so (the inventory policy is not yet present at this stage). To load the supply chain with some initial product quantity we will add this Startup code:

```
Supply.inject(OrderQuantity);
```

If we run this model, at the beginning of the simulation, four hundred items of the product are produced and accumulate in the *ProductStock*.

The market is modeled by a system dynamics stock and flow diagram as shown in the bottom of the same Figure 4. The SD part is located on the same canvas where the flowchart was created earlier. The difference of this market model from the classical Bass diffusion model with discards (Sterman 2000) is that the users, or adopters, stock of the classical model is split into two: the *Demand* stock and the actual *Users* stock. The adoption rate in this model is called *PurchaseDecisions*. It brings *PotentialUsers* not directly into the *Users* stock, but into the intermediate stock *Demand*, where they wait for the product to be available. The actual event of sale, i.e., "meeting" of the product and the customer who wants to buy it, will be modeled outside the system dynamics paradigm. If we run this part of the model alone, the potential clients will gradually make their purchase decisions (triggered by advertizing), building up the *Demand* stock.

How do we link the supply chain and the market? We want to achieve the following:

- If there is at least one product item in stock and there is at least one client who wants to buy it, the product item should be removed from the *ProductStock* queue, the value of *Demand* should be decremented, and the value of *Users* should be

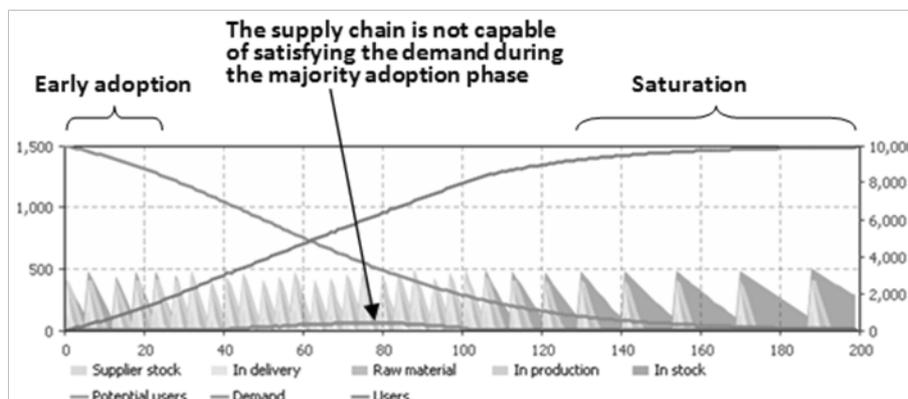
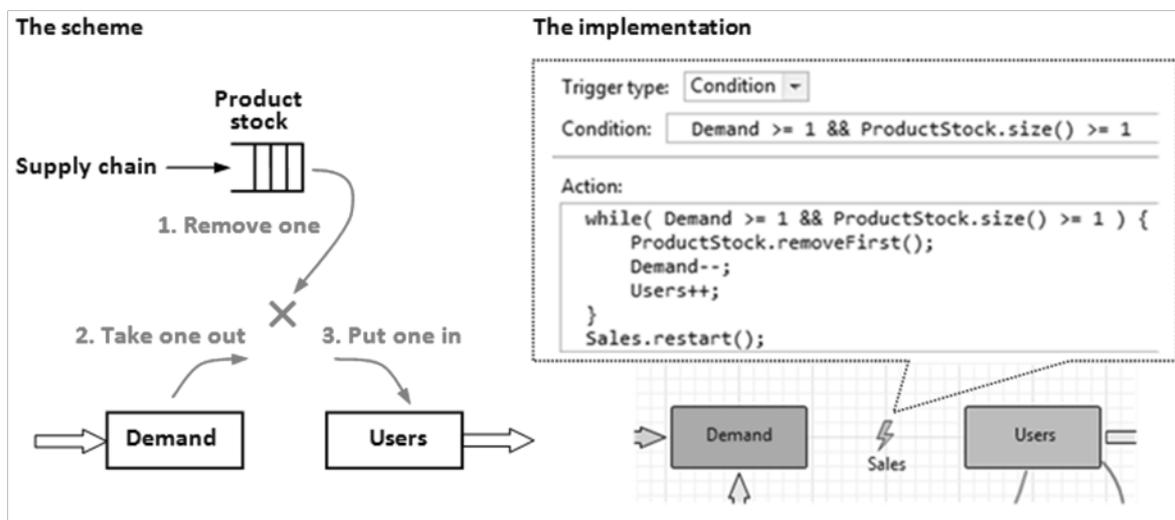


Figure 5: The Supply Chain Model Linked to the Market Model

incremented, see the Figure 5 on the top left.

Therefore, we have a condition and an action that should be executed when the condition is true. The construct that does exactly that is the condition-triggered event. The implementation of the scheme shown in same figure on the right. The condition-triggered events work this way: in the presence of continuous dynamics in the model the condition of the event is evaluated at each numeric micro-step. Once the condition evaluates to true, the event's action is executed. In the Action code of the event we put the sale into a while loop, because a possibility exists that two or more product items may become available simultaneously, or the *Demand* stock may grow by more than one unit per numeric step. Therefore, more than one sale can potentially be executed per event occurrence. By default, the condition event disables itself after execution. As we want it to continue monitoring the condition, we explicitly call `restart()` at the end of the event's action.

With the *Sales* event in place, the sales start to happen, and the 400 items produced in the beginning of the simulation disappear in about a week. The *Users* stock increases up to almost 400; it then slowly starts to decrease according to our limited lifetime assumption. And, since we have not implemented the inventory policy yet, no new items are produced. This is the last missing piece of the model. We will include the inventory policy in the same *Sales* event; the inventory level will be checked after each sale. The following piece of code is added to the Action of the *Sales* event:

```
//apply inventory policy
int inventory = //calculate inventory
    ProductStock.size() + //in stock
    Production.size() + //in production
    RawMaterialStock.size() + //raw inventory
    Delivery.size() + //raw being delivered
    SupplierStock.size(); //supplier's stock
if( inventory < ReorderPoint ) //QR pol.
    Supply.inject( OrderQuantity );
```

Now, the supply chain starts to work as planned, see the bottom of the Figure 5. During the early adoption phase the supply chain performs adequately, but as the majority of the market starts buying, the supply chain cannot keep up with the market. In the middle of the new product adoption (days 40-100), even though the supply chain works at its maximum throughput, still the number of waiting clients remains high. As the market becomes saturated, the sales rate reduces to the replacement purchases rate, which equals the *Discard* rate in the completely saturated market, namely $TotalMarket / ProductLifetime = 16.7$ sales per day. The supply chain handles that easily.

An interesting exercise would be to make the supply chain adaptive. One can try to minimize the order backlog and at the same time minimize the inventory by adding the feedback from the market model to the supply chain model.

Epidemic and Clinic

We will create a simple agent based epidemic model and link it to a simple discrete event clinic model. When a patient discovers symptoms, he will ask for treatment in the clinic, which has limited capacity. We will explore how the capacity of the clinic affects the disease dynamics. This model was

suggested in 2012 by Scott Hebert, a consultant at AnyLogic North America. The full version of the model is available at RunTheModel.com.

We will model patients as agents (individual objects); their initial number is 2,000. The patients are connected by a distance-based network: two patients are connected if they live at most 30 miles away one from another.

The behavior of a patient will be modeled by a statechart (as shown in the Figure 6 at the top) structured according to the classical SEIR (Susceptible Exposed Infectious Recovered) compartmental epidemiology model. The patient is initially in the *Susceptible* state, where he can be infected. Disease transmission is modeled by the message "Infection" sent from one patient to another. Having received such a message, the patient transitions to the state *Exposed*, where he is already infectious, but does not have symptoms. After a random incubation period, the patient discovers symptoms and proceeds to the *Infected* state. We distinguish between the *Exposed* and *Infected* states because the contact behavior of the patient is different before and after the patient discovers symptoms: the contact rate in the *Infected* state is significantly lower. The internal transitions in both states model contacts. We model only those contacts that result in disease transmission; therefore, we multiply the base contact rate by *Infectivity*, which, in our case, is 7%.

There are two possible exits from the *Infected* state. The patient can be treated in a clinic (and then, he is guaranteed to recover), or the illness may progress naturally without intervention. In the latter case, the patient can still recover with a high probability, or die. If the patient dies, it deletes himself from the model, see the Action of the *Dead* state. The completion of treatment is modeled by the message "Treated" sent to the agent. In the absence of the clinic model, this message is, of course, never received.

The recovered patient acquires a temporary immunity to the disease. We reflect this in the model by having the state *Recovered*, where the patient does not react to the message "Infection" that may possibly arrive. At the end of the immunity period the transition *ImmunityLost* takes the patient back to the *Susceptible* state. We need to create the initial entry of the infection into the population. This can be done at the top level of the model, by, for example, sending the message "Infection" to a few randomly chosen people in the beginning of the simulation.

If you run the epidemic model at this stage, you will see the dynamics like shown at the bottom of the Figure 6. The epidemic does not end after the first wave, because the immunity period is not long enough.

The next step is to add the clinic and to let the patients be treated there. Our clinic will be modeled via a very simple discrete event model: the Queue for the patients waiting to be treated and the Delay modeling the actual treatment. We will put the process flowchart (see the bottom right of the Figure 9) at the top level of the model. Unlike in classical discrete event models, however, the entities in this process are not generated by a Source object, but are injected by the agents via an Enter object. The communication scheme between the patients-agents and the clinic process is shown in the Figure 8 on the left.

Once the patient discovers symptoms, he creates an entity – let's call it "treatment request" – and injects the entity into the

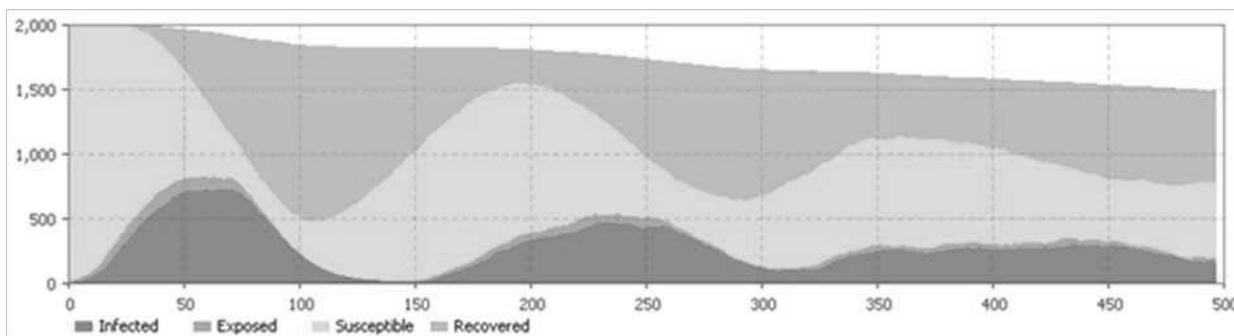
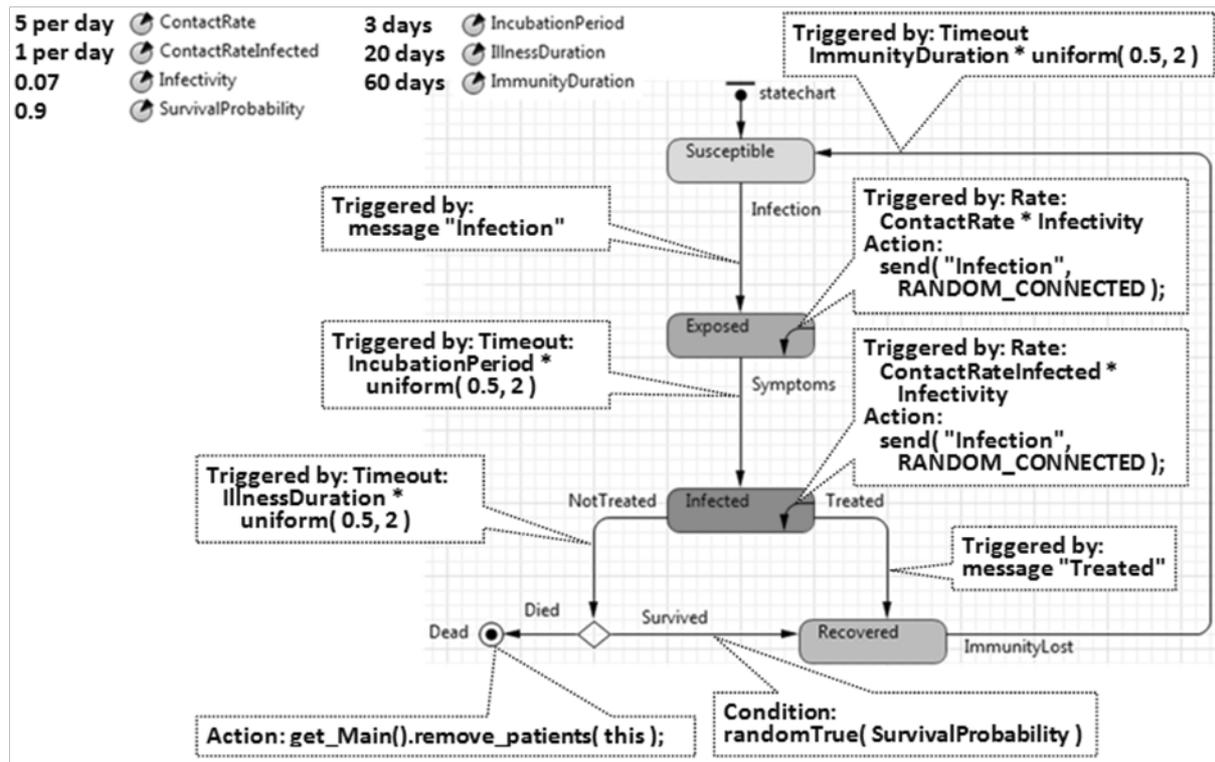


Figure 6: Patient Behavior (an SEIR Statechart) and the Output of the Agent Based Epidemic Model

clinic process. Once the treatment is completed, the entity notifies the patient by sending him a message "Treated" that causes the patient to transition to the *Recovered* state. If, however, the patient recovers or dies before the treatment is completed, he will discard his treatment request by removing it from the process, regardless of its stage.

With the clinic added, the model shows different range of dynamics. The oscillations are still possible, but a possibility also exists that the epidemic will end after the first wave, see the Figure 10. One may experiment with different clinic capacities to figure out the number of beds needed in order to treat everybody on time and prevent the further waves of the epidemic.

DISCUSSION

When developing a discrete event model of a supply chain, IT infrastructure, or a contact center, the modeler would typically ask the client to provide the arrival rates of the

orders, transactions, or phone calls. Having received them in the form of constant values, periodical patterns, or trends, he would treat the rates as variables exogenous to the model. In reality, however, those rates are outputs of other dynamic systems, such as a market, a user base. Moreover, that other system can, in turn, be affected by the system being modeled. For example, the supply chain cycle time, which depends on the order rate, can affect the satisfaction level of the clients, which impacts repeated orders and, through the word of mouth, new orders from other customers. The choice of the model boundary therefore is very important.

The only methodology that explicitly talks about the problem of model boundary is system dynamics (Sterman 2000). However, the system dynamics modeling language is limited by its high level of abstraction, and many problems cannot be addressed with the necessary accuracy. With multi-method modeling you can choose the best-fitting method and language for each component of your model and combine those components while staying on one platform.

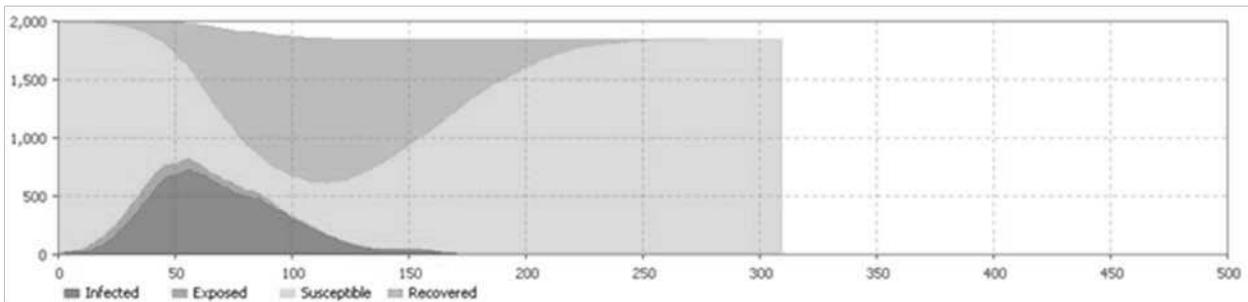
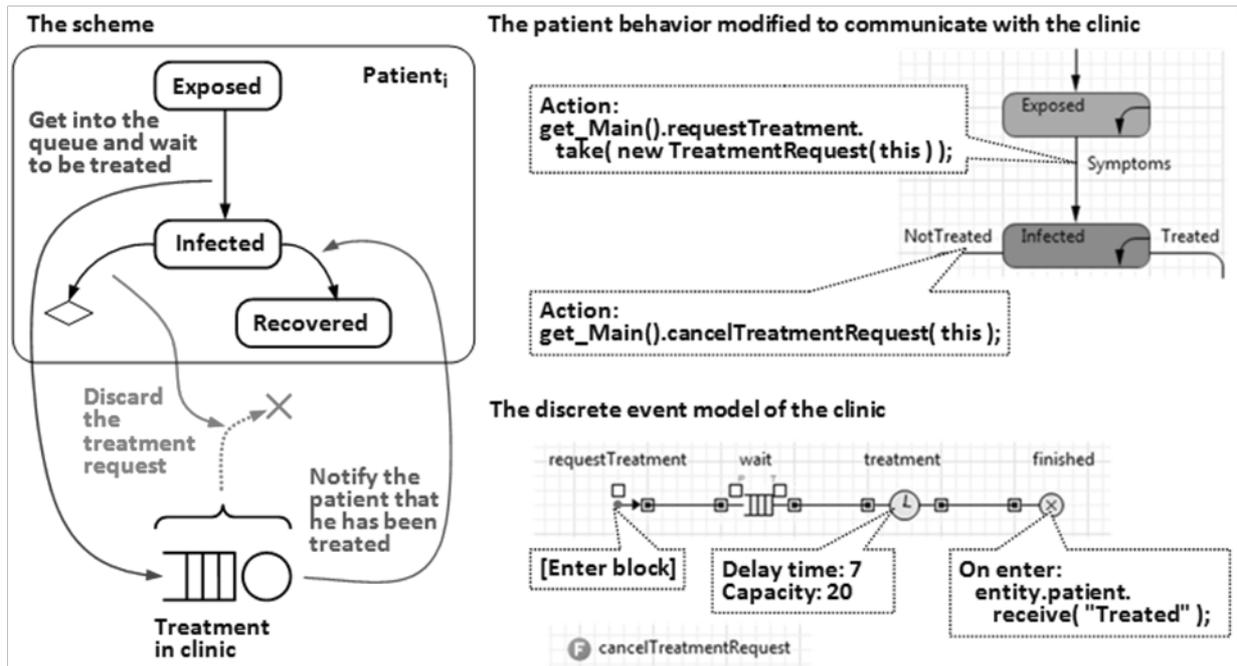


Figure 7: The Discrete Event Model of the Clinic Linked to the Agent Based Epidemic Model

REFERENCES

- Bass, F. 1969. A new product growth model for consumer durables. *Management Science* 15 (5): p215–227.
- Borshchev, A, and A. Filippov. 2004. From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools. In *Proceedings of the 22nd International Conference of the System Dynamics Society*, Oxford, England.
- Sterman, J. 2000. *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Irwin/McGraw-Hill.

WEB REFERENCES

- en.wikipedia.org/wiki/Compartmental_models_in_epidemiology

AUTHOR BIOGRAPHY

ANDREI BORSHCHEV is CEO and co-founder of The AnyLogic Company (former name XJ Technologies). He received his Ph.D. in Computer Science and Complex Systems Modeling from St.Petersburg Polytechnic University. Since 1998 Andrei has been leading the development of AnyLogic software product. His interests include applied agent based modeling, multi-method simulation modeling, hybrid discrete/continuous simulation, design of languages for dynamic simulation. Andrei has conducted numerous lectures, seminars, and training sessions on simulation modeling and AnyLogic. He is a member of the System Dynamics Society, and a constant participant and speaker at International System Dynamics Conference, Winter Simulation Conference, and INFORMS Annual Meeting.

HUMAN-AND-HARDWARE-IN-THE-LOOP SIMULATOR FOR STUDYING VEHICLE LONGITUDINAL DYNAMICS

Ionuț STOICA¹
Marius BĂȚĂUȘ¹
Valerian CROITORESCU¹

University POLITEHNICA of Bucharest
¹Automotive Engineering Department,
313 Splaiul Independentei st., 6th Sector,
060042, Room JC004, Bucharest, ROMANIA

E-mail: valerian.croitorescu@gmail.com

KEYWORDS

Powertrain, Comfort, Simulation, H²IL, Vehicle Dynamics

ABSTRACT

Vehicles' continuous improvement consists in both fuel efficiency and driving behavior, being one of the current objectives of automotive manufacturers. Taking into account the market demands and specifications, the powertrain architectures became more complex very fast.

Numerical simulation is extensively used for multiple evaluation and calibration processes of different architectures. Usually, the simulations are done "offline", without feedback from the real system. "Online" simulations have one feedback degree from components of the real system (hardware-in-the-loop - HIL) and cannot ensure the human feedback on the vehicle comfort or driving pleasure. Therefore, it is difficult to compare various transmissions types even though some objective criteria are defined for a given maneuver or type of transmissions and it is more challenging when each market's particularities must be considered.

This paper describes the development of a human-and-hardware-in-the-loop (H²IL) simulator. One of the main objectives of the simulator is to allow regular drivers to assess the comfort and driving pleasure for different engine/transmission/vehicle configurations. The simulator consists in a virtual electric vehicle, having its own control. The simulator is a test platform and a tool for further developments and optimizations. The simulator allows reproducing different powertrain architectures and different driving profiles.

INTRODUCTION

To meet the requirements with regard to lowering harmful emissions and fuel economy, the automotive manufactures have to enhance current powertrain technologies and develop some new ones. This can be seen in the multitude of transmissions types that are currently available as vehicles' equipments: manual transmissions (MT), automated mechanical transmissions (AMT), automatic transmissions (AT), dual clutch transmissions (DCT), continuously variable transmissions (CVT). Any powertrain architecture that consists of different engines and transmissions induces its own characteristic dynamic behavior to the vehicle. Furthermore, the vehicles dynamic behavior is also altered

by different hybridization architectures or mechanical systems used to improve the comfort such as torque gap filler devices (TGF). Modern control software also offers an increasing number of driving modes and degrees of comfort. Numerical simulation is used on a large scale to study the specific problems generated by such a high number of configurations. Although, there are many simulation environments available, all of them being very flexible and allowing accurate modeling. They cannot offer real feedback on the vehicle comfort and driving pleasure when they are used in "offline" simulation. The HIL (hardware-in-the-loop) simulations offer a degree of feedback from individual components of the real system (Schuette and Waeltermann 2005) while the modern H²IL (human-and-hardware-in-the-loop) systems provide a limited feedback of the human operator (Morse 2012), (Schreiber et al. 2009).

Despite becoming more precise and more complex, the H²IL systems offer data packages from simulated vehicles, with regard to comfort and performances, following specific driving conditions. Current H²IL systems can be used on studying a large variety of parameters: the dynamic and the energetic performances, the passive and the active safety features, the in-car entertainment systems etc.

However, there are some experiments in bringing the H²IL in real-life road traffic. The "InDrive Simulator" project from Ingenieurgesellschaft Auto und Verkehr (IAV) was developed as a testing platform for future cars and can be used before the first prototype is even built (Jaensch 2008). It is a simulator capable of being driven in real-life road traffic. It can be virtually used by anybody that is interested in, after appropriate training. The driver will control a virtual vehicle, not the real vehicle from the simulator, by using the acceleration pedal, the brake pedal and the gear selector lever. The virtual components are simulated as mathematical models on the computer and they are calculated in real time on the basis of the virtual vehicle's operating state and driver instructions. The results will then provide the input values for controlling the vehicle's longitudinal dynamics. There is also possible to be used for computing further target variables.

As the first prototype of this simulator is based on a standard vehicle, equipped with an ordinary combustion engine, it cannot be used for reproducing vibrations, jerks or all-electric, virtually vibration-free driving (Gerson et al. 2011).

The powertrain H²IL simulator development is based on an electric vehicle (EV) that can be used for studying the

comfort for different maneuvers (launch, gearshift, tip-in, tip-out).

This project's main advantage is that it limits the gap between the early design stage and prototype testing stage. There is no need to finalize the target hardware at this intermediary stage. The mathematical models and a real vehicle (not necessarily the target vehicle, but as similar as possible) represent the required components.

This paper aims to present the simulator structure and to demonstrate the capacity to use it for comfort studies.

SIMULATOR STRUCTURE

The simulator is developed as a complex tool. It is developed based on two commercial simulation software products: LMS.Imagine.Lab.AMESim and Simulink/Matlab. The powertrain models are implemented using the 1D multi-domain simulation platform AMESim. This platform is particularly suited for powertrain applications (Hayat et al. 2003; Bataus et al. 2010).

AMESim has a number of efficient analysis tools (e.g. Linear analysis, Activity index, State count) that can be used for model simplification and parameter tuning in support of real-time simulation (Alirand et al. 2005; Bataus et al. 2011). The AMESim RT (real-time) option enables the export of a model to a real-time environment such as dSPACE or xPC for using it into HIL simulation.

The control is implemented in Simulink and linked using the AMESim to Simulink interface. The AMESim-Matlab scripting interface can be used to set parameters, run simulations and retrieve results on AMESim systems using Matlab scripts. The real time interface, as long as both environments are used to achieve the simulation solver's together at the same time, is used as the simulation methodology. It allows individual components and systems to be simulated by different simulation tools running simultaneously and exchanging information in a collaborative manner. Each individual component can be also studied, taking into account the control, due to the advance real-time simulation interface (Croitorescu 2012).

The tool is developed in three modules in order to benefit from the advantages of all the simulation types (figure 1):

- A. "Offline" simulator;
- B. HIL simulator;
- C. H²IL simulator.

The "offline" simulator is essential for the validation of the proposed H²IL configuration and the development of the real time simulation model. Furthermore, it can be used to check different control solutions and to tune the control before it is employed.

The HIL simulator is used to check the real time capabilities of the models, to validate the sensor/actuator configuration and to tune the real control directly on the ECU (electronic control unit).

The H²IL simulator is employed to assess the comfort and driving pleasure for different engine/transmission/vehicle configurations.

Every module has two layers:

1. The acceleration control layer;
2. The virtual powertrain layer.

The acceleration control layer is responsible to generate the commands to the electric machine control unit in order to follow an imposed acceleration.

The virtual powertrain layer is responsible in generating target acceleration, based on the commands from the virtual or from the real driver.

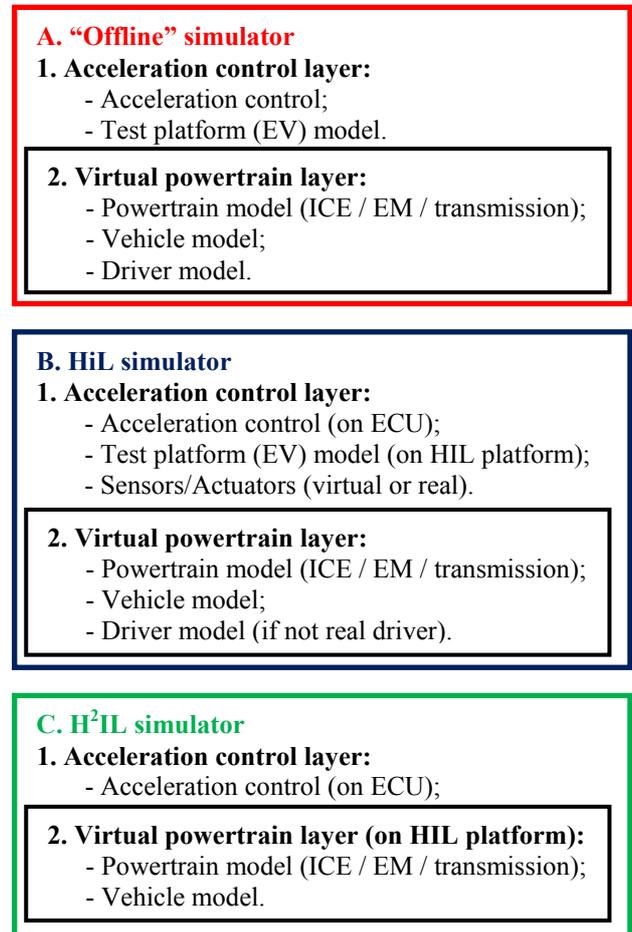


Figure 1: Tool structure

Matlab scripts are developed in order to simplify some operation such as vehicle cornering stability.

In (Bataus et al. 2010) and (Bataus et al. 2011) the possibility to develop high-fidelity powertrain models that are able to run in real time is shown. The models can be very complex including, for example, the clutch's hydraulic control system. Therefore, by using the given recommendations and methods, it is possible to develop the virtual powertrain layer.

This paper emphasis is represented by the acceleration control layer. The vehicle is an electrical one, having front driven wheels, integrated as C segment car. Even with a simple control, it is possible to simulate some standard maneuvers.

PLANT MODEL

The drive torque provided by the electric motor is 20kW. It is amplified with a fixed gear ratio being then distributed to the front wheels through the differential and the drive shafts. The vehicle behavior during the selection of the first three gears of the gearbox can be simulated using the chosen

architecture for the simplified model by static computation (figure 2) (Stoica et al. 2013).

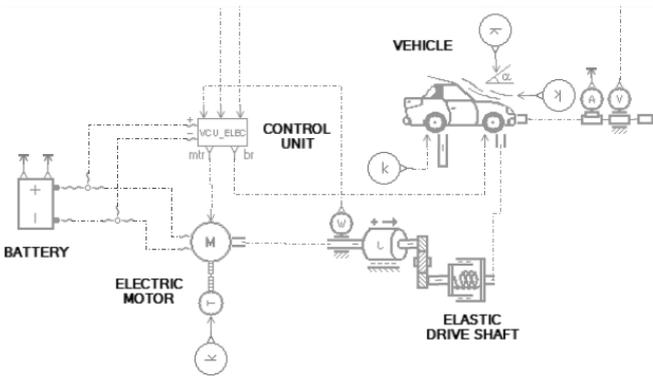


Figure 2: Simplified powertrain model

This simple model does not include the vehicle's suspension. The elastic drive shafts have been simulated using a rotary spring damper block with an equivalent stiffness. A new complex powertrain model is built after deciding the test platforms details (figure 3). The complex model contains:

- Electric drive (motor, battery and control);
- Electric motor stator (3D body – 6 DOF);
- Electric motor elastic mounts;
- Transmission;
- 2D planar vehicle body (3 DOF);
- Vehicle suspension;
- Wheels.

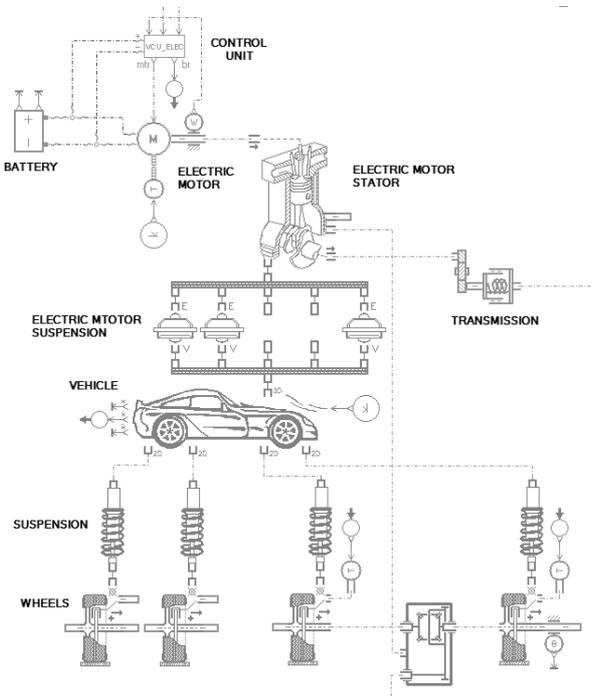


Figure 3: Complex powertrain model

Both the vehicle and the motor are suspended on elastic suspension. The stabilization process is necessary in order to obtain the equilibrium position (the electric motor on the mounts and the vehicle body on the suspension). A dedicated Matlab script was developed in order to make this process automatically. All models used for the components

(referred as submodels) have to be compatible with the real time simulation requirements. This will ensure an easy conversion of the “offline” model to the real time one.

CONTROL IMPLEMENTATION

The vehicle acceleration control is on the top layer of the simulator. It is responsible to generate the commands to the electric machine control unit in order to impose a certain acceleration profile. This profile can be predetermined or computed in real-time using a virtual powertrain.

A control algorithm with continuous states is used. It will be converted to a discrete one when it will be fully implemented on the dedicated ECU (electronic control unit). The first tests are done using an algorithm based on a proportional-integral-derivative (PID) controller that uses the target velocity as input (figure 4).

The back-calculation anti-windup method is used to prevent integration wind-up in the PID controller when the actuators are saturated (figure 5).

This method uses a feedback loop to discharge the PID Controller's internal integrator when the controller hits specified saturation limits and enters nonlinear operation.

The PID controller output is divided using positive (0 to +1) and negative (-1 to 0) saturation blocks in order to generate the acceleration and braking commands. These signals are controlling a standard control unit model for electric vehicles.

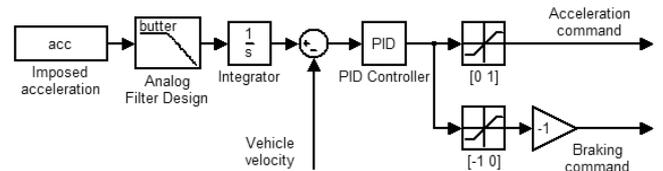


Figure 4: Control implementation

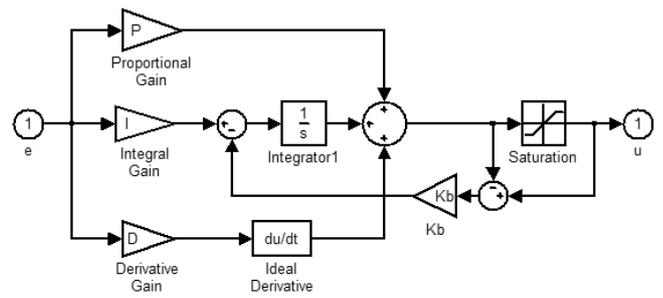


Figure 5: Under-mask view of the PID Controller block with back-calculation

The vehicle's velocity is obtained by integrating the vehicle's longitudinal acceleration measured using high sampling frequency. The measured signal is first filtered using a 3rd order Butterworth low-pass filter with a cutoff frequency of 10Hz.

RESULTS

Running for the first time the simulation has been made for a simple cycle with a high acceleration. The cycle consists of a dynamic launch and two gearshifts (figure 6). The acceleration has been measured with a 100 Hz sampling frequency on the front wheels.

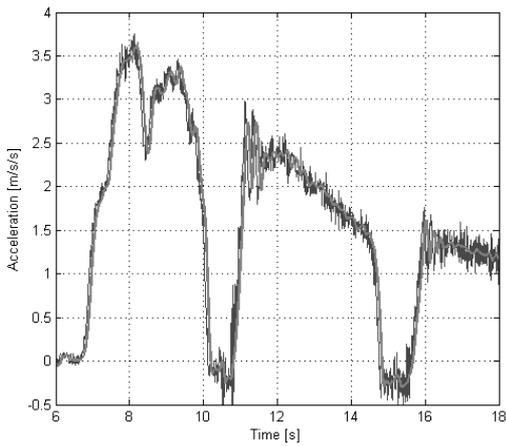


Figure 6: Imposed acceleration profile: measured at 100Hz sampling frequency (blue) and filtered (red)

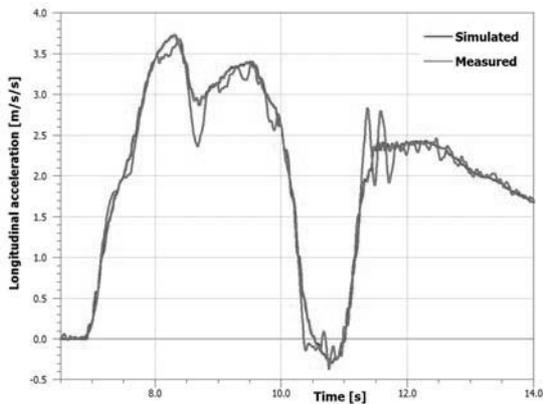


Figure 7: Imposed vehicle acceleration (delayed with 0.2s) and simulated vehicle acceleration

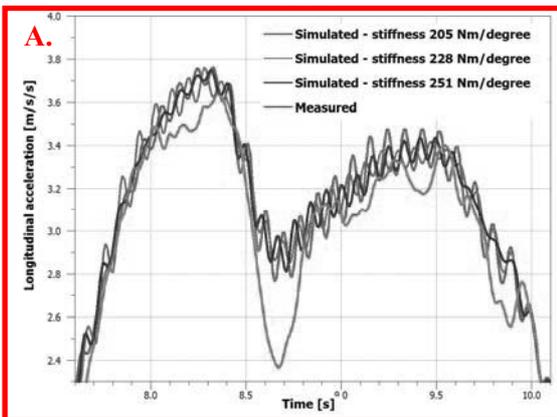
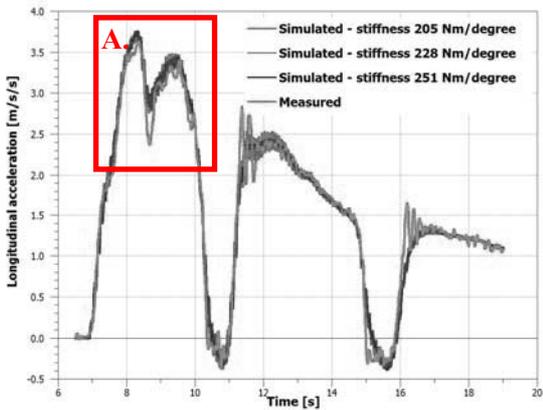


Figure 8: Influence of the stiffness variation on longitudinal acceleration

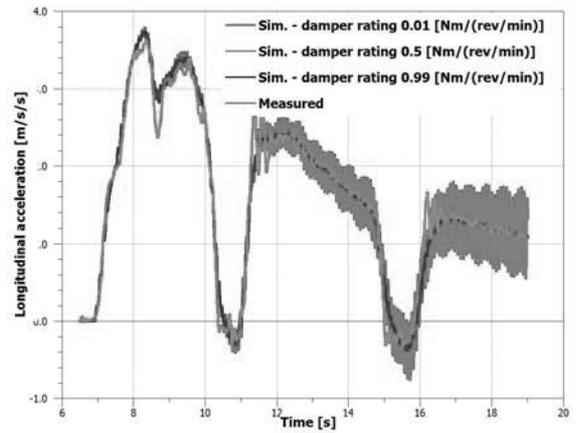


Figure 9: Influence of the damper rating variation on longitudinal acceleration

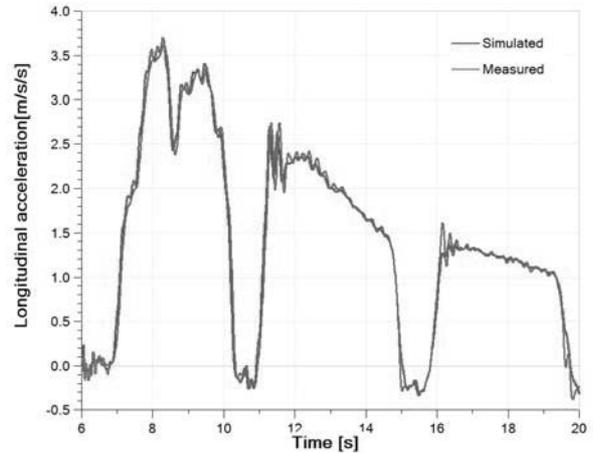


Figure 10: Imposed vehicle acceleration and simulated vehicle acceleration using the complex electric vehicle model

First simulation results show that the control is near the optimal values (figure 7). Selecting the first three gears of the gearbox can be simulated using an electric vehicle by following an imposed cycle or acceleration pattern within acceptable tolerances.

The next step is to investigate the influence of the drive shafts' stiffness on the controller performances. This is done by running simulations with steps variation of the stiffness. The medium and the extreme values were considered (figure 8). The best results are obtained using the highest value for the drive shafts' stiffness.

The influence of the transmission damping is also studied (figure 9). A damper rating of above 0.5 Nm / (rev/min) is necessary to be used for the drive shafts in order to assure good correlation with the measured data.

The details needed for the electric vehicle design were taken into consideration. In addition, the complex model is parameterized and the simulation model is being able to run (figure 10). The results represent a better tuning of the control parameters.

CONCLUSIONS

A tool structure that benefits from the advantages of all the simulation types is proposed for the simulator.

It has been shown using simulation that the proposed configuration for the H²IL simulator is feasible and it can be

used for studying the vehicles` dynamic maneuvers (launch, gear change etc.).

In order to have a good correlation between the imposed acceleration and the real one, a higher stiffness value has to be used.

A damper rating higher than 0.5 Nm/(rev/min) is also necessary to ensure the stability of the control in the higher gears. It is estimated that for such a high damping rating it is necessary to use a dedicated torsion damper.

In the future work, the B module will be developed by converting the existing model in a real time one. More complex types of controllers will also be tested.

ACKNOWLEDGEMENT

This project was supported by POSDRU/107/1.5/S/76903 contract number 817/01.03.2011.

REFERENCES

- Alirand, M.; Jansson, A.; Debs, W. 2005. "A First Idea of Continuity in Model Simplifications for Hydraulic Circuit – An Automatic Gearbox Real-Time Application", *2005 Scandinavian International Conference on Fluid Power*, June 1-3, Linkoping, Sweden.
- Bataus, M.; Maciac, A.; Oprean, M.; Vasiliu, N. 2010. "Real Time Simulation of Complex Automatic Transmission Models", In *Proceedings of 2010 Virtual Powertrain Creation*, ATZ Live, Munich, Germany.
- Bataus, M.; Vasiliu, Daniela; Vasiliu, N. 2011. "Real-Time Simulation of Automated Mechanical Transmissions", In *Proceedings of 27th European Simulation and Modelling Conference 2011*, Guimarães, Portugal. pp. 168.
- Croitorescu, V., "Modern Drives Using Unconventional Energy Storage Devices – Hybrid Electric Vehicles", PhD Thesis, Bucharest, Romania, 2012
- Eller, B. 2011. "Contribution a l'Etude Amont de l'Agrement de Conduite de Systemes de Propulsion de Nouvelle Generation", doctorate thesis, Ecole Centrale de Nantes, 2011.
- Gerson, S. et al. 2011, "InDrive Simulator. Innovative Tool for Simulating and Designing Complex Drive Structures in Real Operation", *8th Symposium Hybrid- and Electric Vehicles, Braunschweig*, 15 and 16 February 2011
- Hayat, O.; Lebrun, M.; Domingues, E. 2003. "Powertrain drivability evaluation: Analysis and Simplification of Dynamic Models", SAE paper no. 2003-01-1328, *SAE 2003 World Congress & Exhibition*, Detroit, MI, USA
- Jaensch, D. 2008, "Driving a Car That Does Not Even Exist Yet", *Automotion*, Edition 02, April 2008, Berlin, Germany, pp 4.
- Morse, P. 2012. "Right on Cue. Driver interactions are key to human-and-hardware-in-the-loop (H²IL) simulation", *Automotive Testing Technology International*, September 2012, Stuttgart, Germany, pp 28.
- Schreiber, U., Todtermuschke, K., Back, O. 2009., "Experiencing and definition of the real shifting confort at the virtual synchroniser in the virtual powertrain", *8th CTI Symposium, Innovative Automotive Transmissions*, Berlin, 2009.
- Schuette, H. and Waeltermann, P. 2005. "Hardware-in-the-Loop Testing of Vehicle Dynamics Controllers – A Technical Survey", SAE paper no. 2005-01-1660, *SAE 2005 World Congress & Exhibition*, Detroit, USA.
- Stoica, I.; Bataus, M.; Negrus, E. 2013. "Development of a Human-and-Hardware-in-the-Loop Simulator for the Study of Vehicle Longitudinal Dynamics", *The International Conference on Innovation and Collaboration in Engineering Research*, June 2013, Bucharest
- LMS Imagine 2010. "AMESim. Powertrain Library, Rev. 10", Roanne.

BIOGRAPHIES

IONUȚ STOICA graduated the „Politehnica” University of Bucharest in 2009, after attending the courses of the Faculty of Transports, department of Automotive Engineering. He also completed a master program earning a master’s degree in study of advanced drivetrains for hybrid and electric vehicles. Since 2006, he is the President and one of the founding members of the official Ford Club in Romania, a club with more than 100,000 members. Currently he is a Ph.D. student at the “Politehnica” University while being employed as a depollution specialist at Renault Tehnologie Roumanie.

MARIUS BĂȚĂUȘ has graduated the University Politehnica of Bucharest, Faculty of Transports, Automotive Engineering (2000) and has earned a Ph.D. degree in Power Engineering in 2011. He began in 2000 as teaching assistant in Automotive Engineering Department of UPB and became assistant professor in 2008. From 2006 to 2008 he was working as project engineer at Imagine (France) as part of the Marie Curie project “Virtual Powertrain”. He is member of the editorial board of the SIAR journal “Ingenieria Autovehiculelor”.

VALERIAN CROITORESCU earned his Automotive Engineer Degree in 2007, as valedictorian. He prepared the diploma project in France, inside Ecole Nationale D’Ingenieurs De Tarbes. He also attended two master of sciences programs, at University POLITEHNICA of Bucharest: `Efficiency and Security in Automotive Engineering` and `Environmental Management`. His academic records include numerous awards and certifications. In September 2007 Valerian joined the academic staff of Automotive Engineering Department. In 2008, he founded “autojobs.ro”, a specialized web portal in automotive industry having the goal to help and provide job seekers and employers the opportunity to be connected, followed by “aerojobs.ro” in 2012, aviation dedicated jobs web portal. In 2012, he earned a Ph.D. Degree at University POLITEHNICA of Bucharest, concerning Hybrid Electric Vehicles Development. A part of his Ph.D. thesis has been accomplished during a research stages at LMS International.

SIMULATION WITH ARENA OF TWO-MACHINE FLOWSHOP SCHEDULING USING MODIFIED JOHNSON RULE

Fawaz Abdulmalek
Kuwait University
College of Engineering & Petroleum
Industrial and Management Systems Engineering
P.O. Box 5969 Safat 13060 Kuwait
E-mail: fawaz.abdulmalek@ku.edu.kw

KEYWORDS: Simulation, ARENA, Johnson Rule, Flowshop Scheduling

ABSTRACT

In this research, we address the two-stage assembly scheduling problem where there are m machines at the first stage and an assembly machine at the second stage. The machines at the first and second stages are subject to random failures. The literature review reveals that this problem with machine breakdowns has not been addressed. Therefore, we consider the problem where machines are subject to unsystematic breakdowns. We use three jobs ranking rules, Johnson rule, and two modified Johnson rule, namely Johnson Largest rule and Johnson smallest rule. These rules will be used to determine jobs order sequence into the flowshop. Next we develop a simulation model with ARENA to model a two-machine flowshop where both machines are subject to random failures. The goal is to find which of the job ranking rules will provide the smallest makespan. Two simulation experiments were considered in this paper. The result of two simulation experiments showed that Johnson Largest ranking rule provided minimum makespan Johnson and Johnson smallest rules.

1. INTRODUCTION

In a two-stage assembly flowshop scheduling problem, there are n jobs where each job has $m+1$ operations and there are $m+1$ different machines to perform each of these operations. Each machine can process only one job at a time. For each job, the first m operations are conducted at the first stage in parallel and a final operation in the second stage. Each of m operations at the first stage is performed by a different machine and the last operation at the second stage may start only after all m operations at the first stage are completed. The two-stage assembly scheduling problem has several applications in industry. Potts et al. (1995) described an application in personal computer manufacturing where central processing units, hard disks, monitors, keyboards, and etc. are manufactured at the first stage, and all the required components are assembled to customer specification at a packaging station (the second stage). Lee et al. (1993) described another application in a fire

engine assembly plant. The body and chassis of fire engines are produced in parallel, in two different departments. When the body and chassis are completed and the engine has been delivered (purchased from outside), they are fed to an assembly line where the fire engine is assembled.

Another practical application of this problem is possibly in the area of distributed database systems. In recent years, there has been a rapid trend toward the distribution of computer systems over multiple sites that are interconnected via a communication network, Elmira and Navathe (1999). It is common with current technology to develop forms or reports that require tens of embedded queries that retrieve information from different sites on the networks and assemble them in one final report, Ceri and Pelagatti (1984). For this scheduling problem, it may be possible to look at the problem from a higher level of abstraction. The details about database and multimedia servers that are typically addressed at the server level, e.g., memory caching, disk scheduling etc., is not considered. This is an on-line problem where requests keep on arriving. However, a static version of the problem can be assumed where there are a fixed number of requests for a given period of time. This assumption is not restrictive since the requests are collected until the system becomes available from the previous batch of requests. Once it becomes available, the batch of accumulated requests is considered for processing next. Hence, this can be considered as a static system within a window of time that is equivalent in duration to the time taken to process the previously collected batch of requests.

The two-stage assembly flowshop scheduling problem was introduced independently by Lee et al. (1993) and Potts et al. (1995). Lee et al. (1993) considered the problem with $m=2$ while Potts et al. (1995) considered the problem with an arbitrary m . Both studies addressed the problem with respect to makespan minimization and both proved that the problem with this objective function is NP-hard in the strong sense for $m=2$. Lee et al. (1993) discussed a few polynomially solvable cases and presented a branch and bound algorithm. Moreover, they proposed three heuristics and analyzed their error bounds. Potts et al. (1995) showed that the search for an

optimal solution may be restricted to permutation schedules. They also showed that any arbitrary permutation schedule has a worst-case ratio bound of two, and presented a heuristic with a worst-case ratio bound of $2-1/m$. Hariri and Potts (1997) also addressed the same problem, developed a lower bound and established several dominance relations. They also presented a branch and bound algorithm incorporating the lower bound and dominance relations. Another branch and bound algorithm was proposed by Haouari and Daouas (1999). Sun et al. (2003) also considered the same problem with the same makespan objective function and proposed heuristics to solve the problem. Koulamas and Kyparisis (2001) generalized the two-stage problem to a three-stage assembly scheduling problem. They proposed several heuristics and analyzed the worst-case ratio bounds of the proposed heuristics for the makespan problem.

Tozkapan et al. (2003) considered the two-stage assembly scheduling problem but with the total weighted flowtime performance measure. They showed that permutation schedules are dominant for the problem with this performance measure. They developed a lower bound and a dominance relation, and utilized the bound and dominance relation in a branch and bound algorithm. They also proposed two heuristics to find an upper bound for their branch and bound algorithm. They indicated by computational analysis that problems with up to 20 jobs and $m=10$ can be solved in a reasonable time with their proposed branch and bound algorithm. They suggested developing efficient heuristics for large sized problems.

In all of the previous research on the two-stage assembly flowshop scheduling problem, it was assumed that machines are not subject to breakdowns. In this research, stage one consist of machine 1 and machine 2 where both machines are subject to random breakdowns and the second stage has the assembly. First we discuss three jobs ranking rules which will be used to determine jobs order sequence into the flowshop. Next we develop a simulation model with ARENA to model a two-machine flowshop where both machines are subject to random breakdowns. Last we discuss the results obtained from the simulation model. The goal is find which of the job ranking rules will provide the minimum makespan.

2. JOB RANKING RULES

Here we explain the three job ranking rules that are used in this research. These rules will used to determine the order sequence of jobs.

2.1 Johnson Rule

Under the traditional Johnson rule the objective is to rank jobs in particular sequence in order to minimize the makespan. Under Johnson rule all jobs will be listed along with their processing times on both machines. The job with the smallest processing time will be selected first. If the shortest time lies on the first machine the job is scheduled first. If the shortest time lies on the second

machine the job is scheduled at the end of the sequence. If there are ties among the processing times of the jobs, then ties will be broken arbitrary. The process is continued in that manner until are jobs are scheduled . Jobs will be processed in the flowshop according to the final sequence of Johnson ranking. To be able to apply the Johnson rule, it must be mentioned here that the average processing time for machine 1 and machine 2 is calculated and this new calculated time will be called (M1) in stage 1. M1 will be compared with the time of assembly which is called (M2) in stage 2 according to Johnson rule to determine the job sequence.

2.2 Johnson Largest Rule

This rule ranks the job processing sequence according to Johnson rule with some modification. Here the time that will be selected for a given job in the first stage will be called M1 (recall first stage consist of machine 1 and machine 2). M1 will be compared with M2 which is the time for assembly in the second stage for the same Job. The following explains the job sequence under Johnson Largest Rule:

1. choose the smallest processing time in M2
2. if $M2 < \text{largest}(\text{machine 1}, \text{machine 2}) = M1$, then schedule the job last, otherwise schedule the job first
3. select the next smallest processing time in M2 and go to step (2)
4. go to step (3) until all jobs have been scheduled

2.3 Johnson Smallest Rule

This rule ranks the job processing sequence according to Johnson rule with some modification. Here the time that will be selected for a given job in the first stage will be called M1 (recall first stage consist of machine 1 and machine 2). M1 will be compared with M2 which is the time for assembly in the second stage for the same Job. The following explains the job sequence under Johnson Largest Rule:

1. choose the smallest processing time in M2
2. if $M2 < \text{smallest}(\text{machine 1}, \text{machine 2}) = M1$, then schedule the job last, otherwise schedule the job first
3. select in the next smallest processing time in M2 and go to step (2)
4. go to step (3) until all jobs have been scheduled

3. SIMULATION MODEL

The two-machine flow shop simulation model is based on System Modeling Corporation's ARENA 5 package. In the simulation, each job is represented as a single entity. A batch of jobs will arrive into the flowshop. A

part of each job must be processed first on machine 1 and the other part on machine 2 (this could be done in parallel). Once both parts of the job are finished on both machines then it proceed to the assembly area to be assembled and then it leaves the system. Both machine 1, machine 2, and assembly are modeled as resources. Machine 1 and machine 2 are all subject to random breakdown which will be explained in more detail in the next section.

In order to validate the simulation model, we performed a sample case study with three jobs with constant processing times on both machines and the assembly station. These processing times have been predetermined. Makespan is the objective function. Simulation results and exact numerical results were both identical and the model was validated.

4. DISCUSSION OF RESULTS

For the flowshop simulation model it was decided that five different scenarios will be used to model machines breakdowns. Under every scenario each machine will have a randomly distributed uptime (time between machine failure) and a randomly distributed downtime (time to repair the failure). Scenarios are extracted from Allahverdi and Tarari (1997). Table 1 below describes each scenario.

Table 1: Description of Scenarios

Scenario	Machine1 Uptime (Minutes)	Machine1 Downtime (Minutes)	Machine2 Uptime (Minutes)	Machine2 Downtime (Minutes)
1	Uniform (17,32)	Uniform (4,8)	Uniform (25,35)	Uniform (1,5)
2	Exponential (8)	Exponential (2)	Exponential (10)	Exponential (1)
3	Exponential (7)	Uniform (3,7)	Exponential (9)	Uniform (1,4)
4	Uniform (5,15)	Uniform (8,17)	Uniform (12,25)	Uniform (2,5)
5	Exponential (5)	Uniform (1,4)	Exponential (2)	Uniform (6,12)

A total of two different experiments will be used in the simulation model. The two experiments are:

Experiment 1: The flowshop will consist of two machines and the assembly station while each run will stop when 60 jobs are processed.

Experiment 2: The flowshop will consist of two machines and the assembly station while each run will stop when 80 jobs are processed.

A total of 30 replications (runs) will be used in the simulation for each of the five scenarios in each experiment. In each run the processing times for each job in machine 1, machine 2 and the assembly will be generated from a discrete uniform distribution between 1 and 100 minutes. For every run a separate random number generator will be used. Once the processing times are generated for each run, three job ranking rules

namely, Johnson, Johnson Largest and Johnson smallest will be used to order jobs. According to that order, jobs will start to be processed in the flowshop. To clarify, in run j (replication j) for any given scenario and for the same random number generated processing times, the same simulation model will be ran at three separate times according to each of the three specific job ranking rule. The makespan for every job ranking rule (i) for each replication will be collected. Then the error percentage of each rule compared to smallest (best) makespan found for every run is calculated. For 30 replication a 95% confidence interval is calculated for the error for each rule. The error is calculated using the following formula:

$$\text{Error}_{ij} = \{[\text{Makespan}_{ij} - \text{Makespan (best)}_{ij}] / \text{Makespan (best)}_{ij}\} * 100$$

Where $i = 1,2,3$ and $j = 1,2,3,\dots,30$

The results of experiment 1 are presented in table 2. Examination of the results shows that among the three rules Johnson largest outperform Johnson smallest and Johnson in terms of the smallest makespan for the first three scenarios where as for scenario 4 and scenario 5 S Johnson largest and Johnson give just about the same results. Figure 1 shows the results of one selected scenario (scenario 3) which shows Johnson largest performing well compared to the other rules.

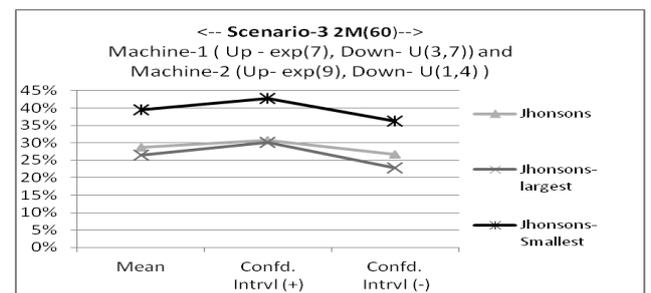


Figure 1: Results for the Third Scenario for Experiment 1

The results of experiment 2 are presented in table 3. Examination of the results also shows Johnson largest outperform Johnson smallest and Johnson in terms of the smallest makespan in all scenarios. Figure 2 shows the results of one selected scenario (scenario 3) which clearly shows Johnson largest performing well, compared to the other rules.

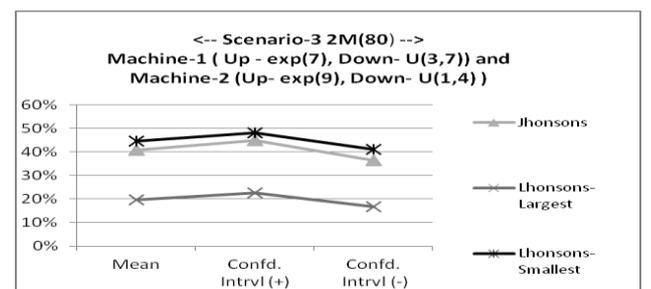


Figure 2: Results for the Third Scenario for Experiment 2

Table 2. 95% Confidence Interval for Scenarios Rule Combination for Experiment 1

Scenario 1. M1 (Up - U(17,32), Down- U(4,8)) and M2 (Up-U(25,35), Down-U(1,5))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	29.090%	31.148%	27.031%
Johnson-largest	25.227%	28.187%	22.268%
Johnson-Smallest	39.679%	42.892%	36.466%
Scenario 2. M1 (Up - exp(8), Down- exp(2)) and M2 (Up-exp(10), Down- exp(1))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	28.756%	30.884%	26.628%
Johnson-largest	24.786%	27.811%	21.760%
Johnson-Smallest	38.906%	42.221%	35.591%
Scenario 3. M1 (Up - exp(7), Down- U(3,7)) and M2 (Up-exp(9), Down- U(1,4))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	28.765%	30.748%	26.783%
Johnson-largest	26.451%	30.157%	22.744%
Johnson-Smallest	39.482%	42.651%	36.313%
Scenario 4. M1 (Up - U(5,15), Down- U(8,17)) and M2(Up-U(12,25), Down- U(2,5))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	28.185%	30.184%	26.187%
Johnson-largest	27.566%	30.507%	24.626%
Johnson-Smallest	40.390%	43.443%	37.337%
Scenario 5. M1 (Up - exp(5), Down- U(1,4)) and M2 (Up-exp(2), Down- U(6,12))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	27.546%	29.520%	25.572%
Johnson-largest	26.854%	29.198%	24.509%
Johnson-Smallest	39.605%	42.366%	36.844%

Table 3: 95% Confidence Interval for Scenarios Rule Combination for Experiment 2

Scenario 1. M1 (Up - U(17,32), Down- U(4,8)) and M2 (Up-U(25,35), Down-U(1,5))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	40.568%	44.894%	36.241%
Johnson -Largest	18.830%	21.710%	15.950%
Johnson-Smallest	43.921%	47.498%	40.344%
Scenario 2. M1 (Up - exp(8), Down- exp(2)) and M2 (Up-exp(10), Down- exp(1))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	40.275%	44.639%	35.911%
Johnson-Largest	18.590%	21.493%	15.687%
Johnson-Smallest	43.502%	47.259%	39.746%
Scenario 3. M1 (Up - exp(7), Down- U(3,7)) and M2 (Up- exp(9), Down- U(1,4))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	40.618%	44.901%	36.336%
Johnson-Largest	19.611%	22.534%	16.688%
Johnson-Smallest	44.622%	48.132%	41.112%
Scenario 4.M1 (Up - U(5,15) Down- U(8,17)) and M2 (Up-U(12,25), Down- U(2,5))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	40.561%	44.783%	36.339%
Johnson-Largest	22.107%	25.216%	18.998%
Johnson-Smallest	46.555%	50.017%	43.092%
Scenario 5.M1 (Up - exp(5), Down- U(1,4)) and M2 (Up- exp(2), Down- U(6,12))			
	Mean	Confidence level (+)	Confidence level (-)
Johnson	40.139%	43.879%	36.399%
Johnson-Largest	22.893%	25.799%	19.988%
Johnson-Smallest	46.834%	50.141%	43.526%

5. CONCLUSION

The problem considered here is the scheduling of n jobs on two machines where the two machines are subject to random failures. The performance measure used is the makespan. Two simulation experiments using ARENA software were considered in this paper. In experiment one, there are a total of 60 jobs. The result of this experiment showed that Johnson largest ranking rule performs better than Johnson smallest and Johnson for the first three scenarios where as in scenario 4 and scenario 5 Johnson largest and Johnson gave about the same results. In experiment two, there are a total of 80 jobs. The result of this experiment showed that Johnson largest ranking rule performs better than Johnson smallest and Johnson for all of the five of the scenarios considered. Future work would be looking at flowshop scheduling problem with n jobs and n machines for job processing times generated from distributions other than uniform probability distribution. In addition, one would examine different uptime and downtime probability distributions.

REFERENCES

- Al-Anzi FS, Allahverdi A. 2001. "The relation between three-tired client-server internet database and two-machine flowshop". *International Journal of Parallel and Distributed Systems and Networks*; 4: 94-101.
- Al-Fawzan MA, Haouari M. 2005. "A bi-objective model for robust resource-constrained project scheduling". *International Journal of Production Economics*; 96: 175-187.
- Al-Turki U, Fedjki C, Andijani A. 2001. "Tabu search for a class of single-machine scheduling problems". *Computers & Operations Research*; 28: 1223-1230.
- Allahverdi A, Al-Anzi FS. 2002. "Using two-machine flowshop with maximum lateness objective to model multimedia data objects scheduling problem for WWW applications". *Computers and Operations Research*; 29: 971-994.
- Allahverdi A, Gupta JND, Aldowaisan T. 1999. "A review of scheduling research involving setup considerations". *OMEGA The International Journal of Management Sciences*; 27: 219-239.
- Allahverdi A, Ng CT, Cheng TCE, Kovalyov MY. 2008. "A survey of scheduling problems with setup times or costs". *European Journal of Operational Research*; 187: 985-1032.
- Ceri S, Pelagatti G. 1984. *Distributed Databases: Principles and Systems*. New York: McGraw-Hill.
- Elmasri R, Navathe B. 1999. *Fundamentals of Database Systems*, 3rd edition. New York: Addison-Wesley.
- Haouari M, Daouas T. 1999. "Optimal scheduling of the 3-machine assembly-type flow shop". *RAIRO Recherche Operationnelle* ; 33: 439-445.
- Hariri AMA, Potts CN. 1997. "A branch and bound algorithm for the two-stage assembly scheduling problem". *European Journal of Operational Research*; 103: 547-556.
- Koulamas C, Kyparisis GJ. 2001. "The three-stage assembly flowshop scheduling problem". *Computers and Operations Research*; 28: 687-704.
- Lee CY, Cheng TCE, Lin BMT. 1993. "Minimizing the makespan in the 3-machine assembly-type flowshop scheduling problem". *Management Science*; 39: 616-625.
- Liaw CF. 2003. "An efficient tabu search approach for the two-machine preemptive open shop scheduling problem". *Computers & Operations Research*; 30: 2081-2095.
- Low, C. 2005. "Simulated annealing heuristic for flow shop scheduling problems with unrelated parallel machines". *Computers & Operations Research*; 32: 2013-2025.
- Mika M, Waligóra G, Weglarz J. 2005. "Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models". *European Journal of Operational Research*; 164: 639-668.
- Pan CH, Chen JS. 1997. "Scheduling alternative operations in two-machine flow-shops". *Journal of the Operational Research Society*; 48: 533-540.
- Potts CN, Sevast'janov SV, Strusevich VA, Van Wassenhove LN, Zwaneveld CM. 1995. "The two-stage assembly scheduling problem: complexity and approximation". *Operations Research*; 43: 346-355.
- Ruiz R, Maroto C. 2005. "A comprehensive review and evaluation of permutation flowshop heuristics". *European Journal of Operational Research*; 165: 479-494.
- Sadegheih, A. 2006. "Scheduling problem using genetic algorithm, simulated annealing and the effects of parameter values on GA performance". *Applied Mathematical Modeling*; 30: 147-154.
- Sridhar J, Rajendran C. 1993. "Scheduling in a cellular manufacturing system: a simulated annealing approach". *International Journal of Production Research*; 31: 2927-2945.
- Sun X, Morizawa K, Nagasawa H. 2003. "Powerful heuristics to minimize makespan in fixed, 3-machine, assembly-type flowshop scheduling". *European Journal of Operational Research*; 146: 498-516.
- Tozkapan A, Kirca O, Chung CS. 2003. "A branch and bound algorithm to minimize the total weighted flowtime for the two-stage assembly scheduling problem". *Computers and Operations Research*; 30: 309-320.
- Wang MY, Sethi SP, Van De Velde SL. 1997. "Minimizing makespan in a class of reentrant shops". *Operations Research*; 45: 702-712.
- Allahverdi A, Tatari F. 1997. "Stochastic Machine Dominance in Flowshops". *Computers and Industrial Engineering*; 34: 735-741.

BIOGRAPHY

Fawaz Abdulmalek is an assistant professor in the Industrial and Management Systems Engineering department at the Kuwait University. He received his B.S. degree in Mechanical Engineering from the University of South Carolina, and his M.S. and Ph.D. degree in Industrial Engineering from the University of Pittsburgh. His areas of interest are Simulation, Production Planning, Supply Chain Management, and Lean Manufacturing.

SIMULATION LANGUAGES FACILITIES FOR INNOVATION

Puhalschi Radu, Feher Szilard, Vasiliu Daniela, Vasiliu Nicolae
University POLITEHNICA of Bucharest, Fluid Power Laboratory
313, Splaiul Independentei, Sector 6
RO 060042 Bucharest, Romania
E-mail: vasiliu@fluid-power.pub.ro

Irimia Petru Cristinel
LMS Romania SRL
Bdul. Garii nr. 13A
RO 500203 Brasov, Romania
E-mail: cristi.irimia@lmsintl.com

KEYWORDS

Numerical simulation tools, RTS, HIL, fluid power systems

ABSTRACT

The paper presents a comparative analysis of three of the main numerical simulation tools available on the market today: Simulink from MathWorks, AMESim from LMS International and LabView from National Instruments. The authors are intimately familiar with all of them, having used extensively numerical simulation in both the teaching process and industrial design projects. The paper provides some objective opinions that can help design engineers to choose the best tool for different kinds of design, test, and optimization tasks.

INTRODUCTION

Today, simulation is a vital part of the design cycle of every new product, in almost any field, from aerospace to intelligent toys. In engineering, where systems become more complex, the safety, reliability and performance requirements are ever increasing. Simulation can be integrated in almost all stages of the design cycle of a new product and it presents a number of hard features to deny advantages compared to traditional techniques. The main advantages of using numerical simulations in industrial innovation are increased reliability, speed and cost-effectiveness.

Reliability: The accuracy of a simulation's result depends mostly on the used mathematical model, but once the model has been properly developed and validated, it can be relied on to produce reproducible results across multiple runs, and respond in a predictable way to changes in simulation conditions. Measurement errors, human errors and errors due to conditions that cannot be controlled which might influence the results of field trials and prototyping are eliminated.

Speed: Simulation helps increase the speed at which a product or solution is developed, as result of flexibility. Altering a simulation model to test something new is a relatively quick task, and even modifying a Hardware-in-the-Loop test bench takes only a fraction of the time needed to build and/or modify a full-size prototype of the whole system.

Cost-effectiveness: The cost of the hardware resources and simulation software licenses is often only a tiny fraction of the potential costs of physical prototyping and/or extensive field trials. In addition, further savings are facilitated by

the fact that reusing parts of older simulation models is much easier than reusing parts of older prototypes. The increasing popularity of simulation has also been greatly supported by advances in the information technology field. Complex system simulations (especially in real time) requires large amounts of computing power, which until recently was only available to large research and industrial entities due to prohibitive costs.

In order to respond to increased market demand, both hardware and software producers have developed various solutions to answer to different customer needs. Below are presented a few, with which the authors have had extensive work experience.

SIMULINK

Developed by MathWorks, Simulink, first released in 1994 together with MATLAB 4 for WINDOWS 3.1, is a data flow graphical programming language tool for modeling, simulating and analyzing multi-domain dynamic systems. Its primary interface is a graphical block diagramming tool and a customizable set of block libraries. It offers tight integration with the rest of the MATLAB environment and can either drive MATLAB or be scripted from it. Simulink is widely used in control theory and digital signal processing for multi-domain simulation and Model-Based Design (Vasiliu D. 1994).

The defining characteristic of SIMULINK is that everything is done directly with the equations that form the mathematical model of the simulated component. Even pre-built blocks, which are introduced with *Simscape* have their explicit mathematical model fully accessible by the user. This has several advantages. First, it allows a very strong control over pre-built components, unmatched by no other simulation software the authors have worked with. You can alter, add, remove, or simply reuse in other models part or the entire mathematical model of any of the pre-built blocks. Secondly, the fact that a mathematical model expressed in explicit equations, transfer functions or state-space representation is universal (operates similarly regardless of the engineering or science field the modeled system belongs to) means SIMULINK is equally applicable and easy to use in all fields. Figure 1 presents the Simulink network based on the Karnopp friction model of a dry clutch co non-linear electro-hydraulic valve (Bataus 2011).

Additionally, Simulink offers a very strong control over the parameters used to solve the differential equations in the mathematical models. Solver algorithm, step size, precision, maximum number of iterations to reach solution

convergence etc. can all be set to best fit the user's needs regarding speed, accuracy or any other special requirements (for example real-time simulation requires fixed step solvers, certain types of equations offer better precision with certain solver algorithms etc.).

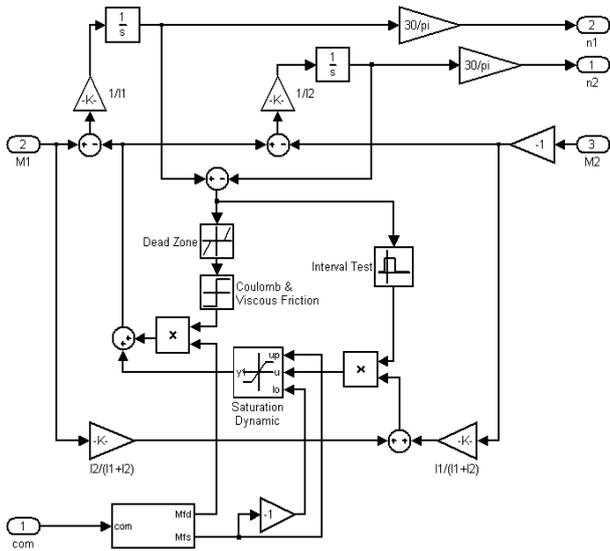


Figure 1: Simulink network for a dry clutch with Karnopp friction model (Băț ăuș , 2011)

Regarding real-time and Hardware-in-the-loop simulation, SIMULINK offers both an extremely accessible and an extremely powerful approach. SIMULINK Coder (known as Real-Time Workshop until MATLAB R2011b) is an extension that can generate C code and/or an executable file from any SIMULINK model, and this code or executable can run on a variety of machines and operating system. One of the most effective systems was developed by dSPACE Company for Rapid Controller Prototyping (Dragne 2011) in the field of automotive Engineering (Figure 2).

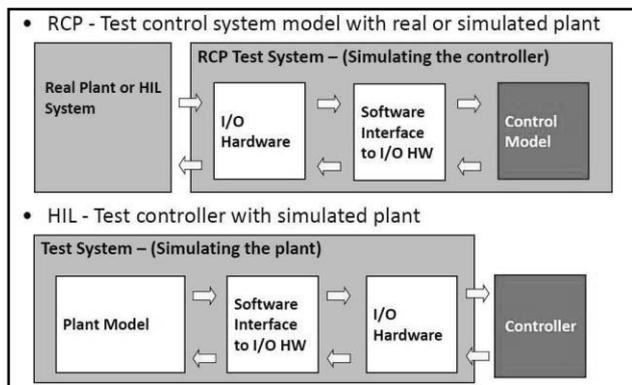


Figure 2: SIMULINK Real Time Interface for Rapid Controller Prototyping (dSPACE 2011)

Another big advantage of Simulink is MATLAB integration. The Simulink model exists inside the MATLAB workspace, and so the user has access to all the

extensive tools MATLAB offers, like stability analysis, interpolation, Bode and Nyquist plots, calculation of equivalent transfer functions, transfers between state-space and transfer function models etc. This makes SIMULINK and ideal control system design tool, due to its extensive fine-tuning tools (including a built-in PID tuner using a proprietary algorithm). A strong point of SIMULINK (and MATLAB), especially useful in the academic environment, is an excellent online support. The MathWorks website offers a large amount of material and free examples.

AMESim

The AMESim software has been developed by Imagine S.A., which was acquired in June 2007 by LMS International, which itself has been acquired in November 2012 by SIEMENS AG. Imagine S.A. was created in 1987 by Professor Michel Lebrun, as a spin-off from the University Claude Bernard, to control complex dynamic systems coupling hydraulic servo-actuators with finite-elements mechanical structures. The first engineering project involved the deck elevation of the sinking Ekofisk North Sea petroleum platforms. In the early 1990's, the association with Professor C.W. Richards (University of Bath), led to the first commercial release of AMESim in 1995, which was dedicated to fluid control systems. Any AMESim model is built around a network of pre-generated blocks and connections (Figure 3), available in various libraries.

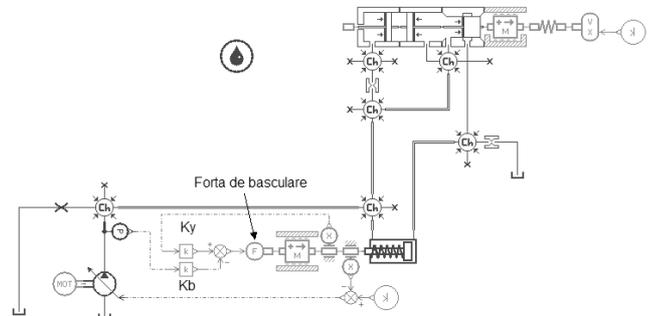


Figure 3: AMESim model of a PARKER swash plate electro hydraulic servo pump (Negoita 2011)

The blocks and connections can have various mathematical models associated to them, depending on model complexity and user needs. Unlike Simulink, it is not possible to directly access to mathematical model behind the AMESim blocks. AMESim libraries are written in C language and they also support Modelica; so, an external tool is needed to open and edit the structure of the blocks. Still, good programming knowledge is needed to make changes in AMESim models, which puts the operation outside the expertise of most users of this software. AMESim's approach to modeling allows great flexibility as parts of a system can be modeled as synthetic or complex as needed, using one of the three available levels of detail.

As an integrated conceptual modeling and simulation tool for engineering, with numerous applications in automotive, aerospace and other fields, AMESim offers numerous built-in tuning and data analysis tool. Batch runs, equivalent transfer functions, stability analysis etc. can all be performed directly from the program (Muraru 2000, Dragne 2010, Popescu 2011, Călinoiu 2012, Mitroi 2013). AMESim does not offer an integrated real-time or Hardware-in-the-Loop solution, but like Simulink the models are compatible with most dedicated solutions available on the market: dSPACE, LabVIEW RT, and OPAL RT.

LabVIEW

First of all, an important distinction needs to be made: LabVIEW (short for Laboratory Virtual Instrument Engineering Workbench), developed by National Instruments is not a simulation software, but rather a graphical programming language. The work on the initial LabVIEW release began in April 1983, with the first version released in October 1986 (for Macintosh only).

As a programming language, LabVIEW has a much wider application field than just numerical simulation. For example, figure 4 presents the remote control interface of a laboratory test bench for identification of the static characteristic of a servo valve, developed by the authors. This stand (Figures 5 and 6) can be considered the perfect example of what can be achieved using the integrated software and hardware tools offered by National Instruments.

From a hardware point of view, the assembly consists of the electro-hydraulic part of the stand (the servovalve, oil tank, pump, accumulators etc. together with all the necessary transducers), a CompactRIO high performance reconfigurable industrial controller running LabVIEW RT operating system the DAQ boards of which are connected to the servo valve, and a PXI industrial computer running Microsoft Windows connected to the CompactRIO via an Ethernet Network.

Software-wise, the National Instruments PXI acted as a host computer, being the main interface between the laboratory stand and the Internet, through which the students could access any laboratory test bench and perform experiments remotely.

The PXI houses the authentication service (each student is registered with a username and password in the database), the interface, and the database where the scheduling information (only one student can perform the experiment at a given time, so it needs to be scheduled in advance) and the experiment results.

The compactRIO acts as a control unit for the stand. One LabVIEW program, running on the FPGA chip effectively performs the experiment, by providing the command signals from the servo valve (according to the values received from the PXI program), reading the transducers values and converting them to the appropriate units, while a second LabVIEW program running on the x86 processor of the CompactRIO handles the transfer of the data over the Ethernet network to the PXI computer.

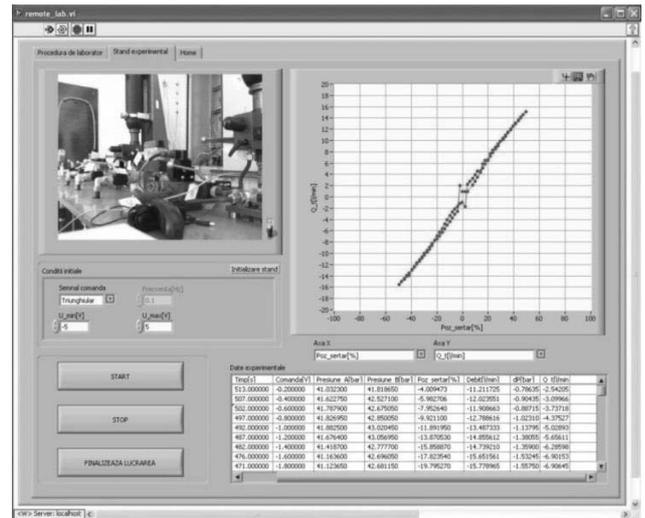


Figure 4: Remote control interface of the servo valves performances remote laboratory test bench in the HIL laboratory of U.P.B. (Ion Guta 2012)

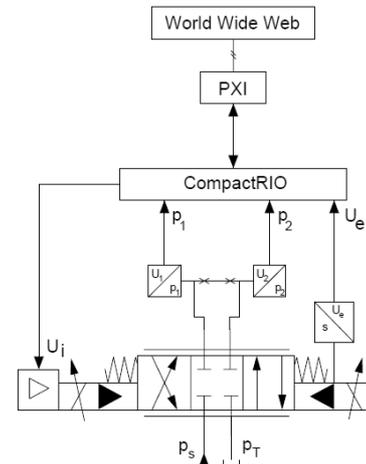


Figure 5: Principle schematic of the remote laboratory test bench for electro hydraulic servo valves



Figure 6: Partial view of the servo valves remote laboratory test bench (Vasiliu N. 2011)

This original solution has been chosen, due to the speed and reliability it offers.

The programming language used in LabVIEW, also referred to as G, is a dataflow programming language. Execution is determined by the structure of a graphical block diagram (the LV - source code) on which the programmer connects different function-nodes by drawing wires. These "wires" propagate variables and any node can execute as soon as all its input data become available. Since this might be the case for multiple nodes simultaneously, G is inherently capable of parallel execution. Multi-processing and multi-threading hardware is automatically exploited by the built-in scheduler, which multiplexes multiple OS threads over the nodes ready for executions.

This means that for the purpose of numerical simulation, LabVIEW can be used in the exact same manner as Simulink, by building the mathematical model of the studied system, either by using explicit equations or transfer functions. *The biggest downside of LabVIEW in this field is the complete lack of pre-built blocks.* Everything needs to be built from scratch, which is a time consuming task.

However, as a programming language, LabVIEW offers complete control on all the parameters of the numerical integration process, even allowing setting them individually per instance of integration, an unique feature among simulation languages.

Moreover, LabVIEW is capable of importing both Simulink as well as AMESim models (Ion Guta 2011, Cioranu 2012, Puhalschi 2013), and also run Matlab code directly inside a LabVIEW program (Figures 7 and 8). Much like MathWorks, National Instruments offers extensive free online support in the shape of documentation, examples, and interactive webinars.

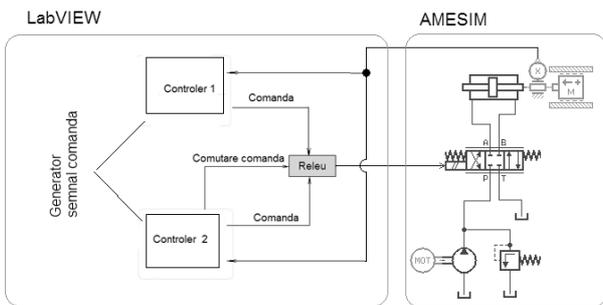


Figure 7: Co-simulation LabVIEW-AMESim for a redundant electro hydraulic servomechanism in the HIL laboratory of U.P.B. (Cioranu, Vasiliu N. 2012)

The main strong point of LabVIEW is in the field of real-time and Hardware-in-the-Loop simulation. The program offers dedicated modules for both real-time implementation of models (under Windows or deployed on dedicated real-time simulation platforms directly from LabVIEW) and data acquisition (vital part of any Hardware-in-the-Loop model).

Apart from that, National Instruments offers its own line of dedicated real-time simulation hardware (PXI and CompactRIO families) as well as software (the LabVIEW

RT operating system), all fully compatible with LabVIEW. The development of a real-time and/or Hardware-in-the-Loop simulation model can be done 100% within LabVIEW, (Vasiliu C. 2011) without the need of any third party solutions (Figure 9).

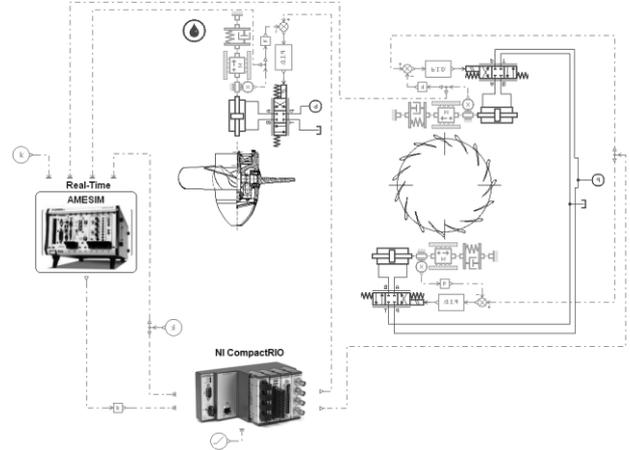


Figure 8: Co-simulation with LabVIEW and AMESim for the speed governor of a hydraulic turbine in the HIL laboratory of U.P.B. (Ion Guta 2011)

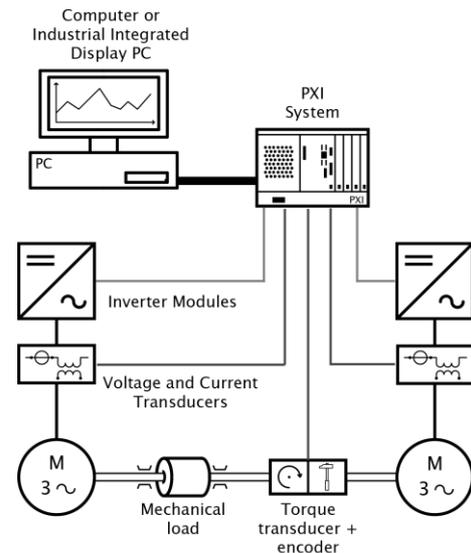


Figure 9: Using LabVIEW for HIL of electric powertrain in the HIL laboratory of U.P.B. (Vasiliu C. 2011)

CONCLUSIONS

The authors believe the simulation software market has currently reached maturity. This belief is also reinforced by the fact that recent releases of all major simulation software products have mainly contained small incremental improvements and very few (if at all), new, revolutionary features, which is a sign that customer requirements have largely been met.

Regarding the question "which simulation software is best", the answer that this paper is arguing for is 'it depends on the application'.

Simulink is the best both for users that desire a quick, accessible simulation and control tool as well as for users that have very high performance and accuracy requirements provided they have the required time and resources to develop and validate or acquire elaborate mathematical models for their system.

AMESim is the best tool for conceptual modeling and testing new solutions. Even complex models can be built reasonably fast (despite that, it has no issues with regard to accuracy) and all the needed data analysis can be performed directly in the software itself, for quick, accurate results and conclusions.

LabVIEW is the best tool for users whose needs extend beyond simple numerical simulation and control. Integrated software and hardware solutions offer the right fit for almost any circumstance that can be encountered in engineering. The price is that, due to LabVIEW's inherent structure as a programming language (as opposed to a dedicated simulation software), developing a solution is by far the most laborious and time consuming.

REFERENCES

- Bataus, M.V. 2011. "Hydraulic Control Systems for Automotive Mechanical Transmissions". Ph.D. Thesis, University POLITEHNICA of Bucharest.
- Calinoiu, C., Negoita, G. C., Vasiliu, D., Vasiliu, N. 2011. "Simulation as a Tool for Tuning Hydropower Speed Governors". ISC'2011, Venice, Italy.
- Cioranu, C.N. 2012. "Researches on the Redundant Hydraulic Control Systems." Ph.D. Thesis, University POLITEHNICA of Bucharest.
- Dragne, F.D. 2010. "Researches on numerical Modeling and Simulation of Automotive Hydraulic Systems with Application on an ABS/ESP Braking System." Ph.D. Thesis, University POLITEHNICA of Bucharest.
- Lebrun, M., Vasiliu D., Vasiliu, N. 2009. "Numerical simulation of the Fluid Control Systems by AMESim". Studies in Informatics and Control with Emphasis on Useful Applications of Advanced Technology, Volume 18, Issue 2, p.111-118.
- Mitroi, M.A.I. 2013. "Researches on the Modeling and Real Time Simulation of the Electrohydraulic Control Systems." Ph.D. Thesis, University POLITEHNICA of Bucharest.
- Muraru, V., Vasiliu, N., Vasiliu, D. 2000. "Structural Analysis of the Proportional Solenoids for Control Systems using ANSYS Software". World ANSYS Conference, Pittsburgh, U.S.A.
- Negoita, G.C. 2011. "Researches on the Dynamics of the Hydrostatic Transmission". Ph.D. Thesis, University POLITEHNICA of Bucharest.
- Popescu, T.C. et al. 2011. "Numerical Simulation - a Design Tool for Electro Hydraulic Servo Systems", in *"Numerical Simulations, Applications, Examples and Theory"*, INTECH PRESS, Zieglergasse 14, 1070 Vienna, Austria.
- Vasiliu D., Vasiliu G.C., Vasiliu N. 2012. "Using Automation Studio for the Design of the Fluid Control Systems". POLITEHNICA PRESS, ISBN 978-606-515-397-4.
- Vasiliu N., Vasiliu G.C., Vasiliu D., Irimia P.C. 2012. "Modern Developments of the Hydraulic Transmissions in Automotive Technology". ROMANIAN ACADEMY OF TECHNICAL SCIENCES and AGIR Press House, ISSN 2066-6586.
- Vasiliu C. 2011. "Real Time Simulation of the Electric Powertrains". Ph.D. Thesis, University POLITEHNICA of Bucharest.
- Vasiliu D. 1997. "Modeling, Simulation and Experimental Identification of the Servopumps and Servomotors of the hydrostatic Transmissions". Ph.D. Thesis, University POLITEHNICA of Bucharest.
- ***Imagine. 2001. "Numerical Challenges posed by Modeling Hydraulic Systems". Technical Bulletin 114.
- ***LMS International. 2012. "Advanced Modeling and Simulation Environment", Release 12, Leuven.

WEB REFERENCES

- <http://www.famicttech.com/>
<http://www.lmsintl.com/>
<http://www.mathworks.com/products/simulink/>
www.acslX.com
<http://www.nhancetech.com/>
<http://www.ni.com/matrixx/>
<http://www.dspace.com/>

BIOGRAPHIES

PUHALSCHI RADU graduated in Applied Computer Science from University POLITEHNICA of Bucharest in 2009, and a MD in Advanced Hydraulic and Pneumatic Systems in 2011. After a stage of web designer at HP Germany, he began a PhD research work on Real-Time Simulation of hydraulic systems at the Fluid Power Laboratory from the Power Engineering Department of the University POLITEHNICA of Bucharest.

FEHER SZILARD graduated in Mechanical Engineering from University POLITEHNICA of Bucharest in 2009, and a MD in Advanced Hydraulic and Pneumatic Systems in 2011. The same year he began a PhD research work on mobile electro hydraulic servo systems at the Fluid Power Laboratory from the Power Engineering Department of the University POLITEHNICA of Bucharest. He serves currently as technical advisor for the company HIDRAULICA SRL from Braşov.

DANIELA VASILIU graduated in mechanical engineering in 1981 and prepared the Ph.D. thesis in the field of the numerical simulation of the hydrostatic transmissions. She is currently professor in the Department of Hydraulics, Hydraulic Machines and Environmental Engineering, head of Fluid Power Laboratory of the University POLITEHNICA of Bucharest with works in the field of modeling, simulation, and experimental identification of the electro hydraulic control systems.

PETRU-CRISTINEL IRIMIA received the M.Sc. degree in Mechanical Engineering, Automotive Section, from the University Transilvania of Brasov in 1991. He developed a long career in R&D, having a deep education in finite element analysis with high level LMS software. He is serving currently as general manager of LMS Test Division - CAE Division - Engineering Services from Romania.

AUTHOR LISTING

AUTHOR LISTING

Abdulmalek F.....	61	Leopoulos V.....	39
Anthonis J.....	12	Puhalschi R.	66
Băţăuş M.....	56	Rokou E.....	39
Borshchev A.....	47	Rossmann J.....	31
Croitorescu V.	12/56	Schlette C.....	31
Douloumpakis M.	39	Schluse M.....	31
Feher S.	66	Stähl I.	5
Irimia P. C.	66	Stoica I.....	56
Kirytopoulos K.....	39	Vasiliiu D.	66
Klüver C.	23	Vasiliiu N.	66
Klüver J.	23	Waspe R.....	31