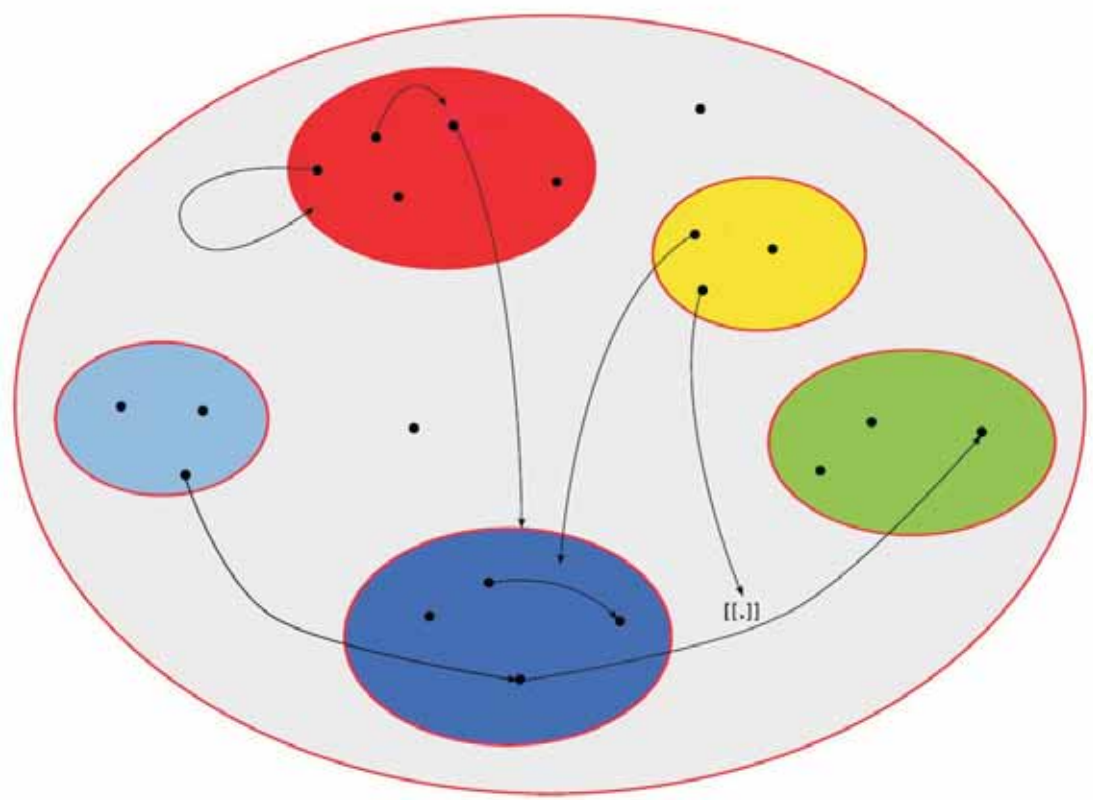
SNE SIMULATION NEWS EUROPE



Volume 18, Number 3-4 December 2008, ISSN 0929-2268

Journal on Developments and Trends in Modelling and Simulation

Membership Journal for Simulation Societies in EUROSIM

ARGESIM

Dear Readers,

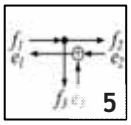
With this SNE double issue, SNE 18/3-4, our journal SNE is on publication schedule again, after delays in publishing of regular issues, caused by some changes in strategy and changes within the EUROSIM simulation societies. At present we are working on the last changes in publication strategy, driven by a new web page with new functionalities (layout see picture). This new webpage not only will document SNE and its contents, but also it will organise the SNE administration in submission, reviewing, publishing, and distributing contributions, and it will allow to evaluate the ARGESIM benchmarks. Authors may submit contributions of any kind via this system, followed by an appropriate reviewing. Also reports from societies will be managed by this system, so that new SNE issues can be compiled in time on basis of reviewed and refined contributions. We will report in detail about this new system in the first issue of 2009, SNE 19/1.

The Technical Notes, Short Notes and Benchmarks Solutions in this issue on the one side set an emphasis on hybrid modelling and simulation by contributions on hybrid bond graphs (I. Roychoudhury, G. Biswas), on comparison of hybrid features in simulator (N. Popper et al.), on multi-paradigm language engineering (H. Vangheluwe), and by hybrid approaches to benchmark solutions. On the other side, discrete modelling and simulation plays a major role, by contributions on LAN modelling and monitoring (M. Nazri et al), on discrete event modelling in facility management (D. Wiegand et al.), and by two solutions to the ARGESIM benchmark 'Restaurant Business Dynamics'. Two contributions deal with methodological aspects: functionality of a specific validation tool (P. Kemper), and an approach to the precise modelling of essential terms (J. R. Müller). Furthermore we are proud to publish an interesting contribution by the H. Fuss, a doyen among Petri net modellers, who strictly states 'True Computer Simulation of Real Parallel Processes is Not Possible - A Proof by the Five Dining Philosophers – and Their Children'. And last but not least, this issue also continues the new series of Educational Notes (started in SNE 18/1) with a contribution on a MATLAB-supported E-learning web system (G. Zauner, F. Judex). Furthermore, this issue presents the first complete newly-arranged News Section: after the Data & Quick Info, which summarizes the relevant basic information of all EUROSIM societies, SNE publishes actual reports from EUROSIM societies, which have sent in recent information. We hope, readers enjoy the novelties and the content, and we thank all contributors, members of the editorial boards, and people of our ARGESIM staff for co-operation in producing this SNE issue.

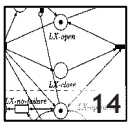
Felix Breitenecker, editor-in-chief, felix.breitenecker@tuwien.ac.at

SNE 18/3-4 in Three Minutes

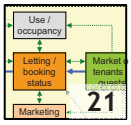
Impressum, Table of Contents: page 4



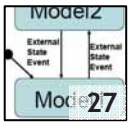
Efficient Method for Simulating Complex Systems with Switching Behaviors Using Hybrid Bond Graphs introduces an efficient method for analysing hybrid systems



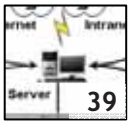
Clearification of Dependability Concepts – An Approach to the Precise Modeling of Essential Terms focuses on system theory and Petri nets to identify system properties.



Event based simulations:Enabling Improved Lifecycle and Risk Management of Facilities introduces DEV as tool for facility management

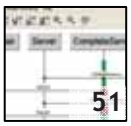


Extended and Structural Features of Simulators – A Comparative Study evaluates modern simulation systems w. r. t. features classified on basis of ARGESIM Benchmarks

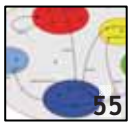


Prototype of Network Management System for Real Time Architecture Development provides a network analysis approach by queuing theory concepts

A PHP/MATLAB-Based E-Learning System for Education in Engineering Mathematics and in Modeling and Simulation . introduces an E-learning system.

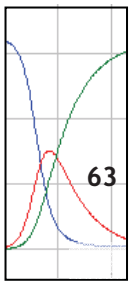


Traviando - a Trace Analyzer to Debug Simulation Models presents a tool for validation



Multi-Paradigm Language Engineering and Equation-Based Object-Oriented Languages gives an introduction to multi-paradigm modelling and simulation

True Computer Simulation of Real Parallel Processes is Not Possible discusses problems of concurrency on basis of Petri nets



ARGESIM Benchmarks section:

EUROSIM short info gives a short overview over EUROSIM and its national societies, as well as contact addresses

Reports section publishes reports from SLOSIM, SIMS, LSS, CROSSIM, and ASIM, and a 10-years-activity report of ASIM group ‚Simulation in Environmental Systems'

Table of Contents

SNE Contact. SNE - Editors /ARGESIM
c/o Inst. f. Analysis and Scientific Computation
Vienna University of Technology
Wiedner Hauptstrasse 8-10, 1040 Vienna, AUSTRIA
Tel + 43 - 1- 58801-10115 or 11455, Fax – 42098
sne@argesim.org; www.argesim.org

SNE Simulation News Europe ISSN 1015-8685 (0929-2268).

Scope: Technical Notes and Short Notes on developments in modelling and simulation in various areas /application and theory) and on bechmarks for modelling and simulation, membership information for EUROSIM and Simulation Societies.

Editor-in-Chief: Felix Breitenecker, Inst. f. Analysis and Scientific Computing, Vienna University of Technology, Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria; Felix.Breitenecker@tuwien.ac.at

Layout: Markus Wallerberger, markus.wallerberger@gmx.at

Administration, Web: Anna Mathe, anna.mathe@tuwien.ac.at

Printed by: Grafisches Zentrum, TU Vienna, Wiedner Hauptstrasse 8-10, 1040, Vienna, Austria

Publisher: ARGESIM/ASIM; ARGESIM, c/o Inst. for Scientific Computation, TU Vienna, Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria, and ASIM (German Simulation Society), c/o Wohlfartstr. 21b, 80939 Munchen;

TN

TECHNICAL NOTES

An Efficient Method for Simulating Complex Systems with Switching Behaviors Using Hybrid Bond Graphs

Indranil Roychoudhury, Gautam Biswas, Xenofon Koutsoukos, Vanderbilt University, USA
Matthew J. Daigle, University of Califonia, USA

Accurate and efficient modeling and simulation approaches are essential for design, analysis, diagnosis, and prognosis of complex embedded systems. This paper presents an efficient simulation scheme for systems with mixed continuous and discrete behaviors. We model hybrid systems using hybrid bond graphs (HBGs), a multi-domain physics-based modeling language that incorporates local switching functions that enable the reconfiguration of energy flow paths. We exploit the inherent causal structure in HBGs to derive hybrid simulation models as reconfigurable block diagram (BD) structures. Considerable computational savings are achieved during simulation by identifying fixed causal assignments when the simulation model is derived. Fixed causal assignments reduce the number of possible computational structures across all mode changes, and this leads to an overall reduction in the complexity of the BD simulation models and in their reconfiguration procedures. This approach has been implemented as a software tool called the MOdeling and Transformation of HBGs for Simulation (MOTHS) tool suite. Simulation models of an electrical power distribution system that includes a fast switching inverter system are derived using the MOTHS tool suite, and experimental studies on this system demonstrate the effectiveness of our approach.

SNE 18/3-4, December 2008

1 Introduction

Accurate and efficient modeling and simulation approaches are essential for design, analysis, diagnosis, and prognosis of complex embedded systems. To address these needs, we have developed component-oriented modeling techniques based on hybrid bond graphs [1], and a model-integrated design methodology for efficient simulation that facilitates diagnosis and prognosis experiments [2, 3]. Building accurate and efficient simulation models for hybrid, nonlinear systems is not trivial, especially since the simulation must deal with the computation of nonlinear behavior and system reconfigurations that produce discrete behavior changes. For systems where reconfigurations occur at high frequencies, such as modern electrical power distribution systems with electronic switching and control [4], it is especially important to maintain accuracy in the generated behaviors without sacrificing simulation efficiency.

The bond graph (BG) modeling language allows for multidomain, topological, lumped-parameter modeling of physical process, such as electrical power systems, by capturing their energy exchange mechanisms [5]. The nodes of a bond graph include primitive energy storage (*C* and *I*), energy dissipation (*R*), energy transformation (*TF* and *GY*), and energy source elements (*Se* and *Sf*). The *n*-port *I* (or *C*)-fields allow for extended energy storage models, where each *I* (or *C*)-field is defined by an $n \times n$ -matrix. The connecting edges, called *bonds*, define energy pathways between elements. Each bond has two associated variables: effort, *e*, and flow, *f*, and their product represents the rate of energy transfer. In the electrical domain, effort denotes voltage and flow denotes current. Each BG element relates the effort and flow variables on the bonds connected to it. For example, since the voltage drop across a resistor is the product of its resistance and the current flowing through it, the constituent equation relating the effort and flow at a resistor is $e = Rf$, where R is the resistance. BG components are connected to one another using two idealized connection elements, the 0-and 1-junctions. For a 0-junction, the efforts of all incident bonds are equal, and the sum of flows is zero, while for a 1-junction, the flows of all incident bonds are equal, and the sum of efforts is zero. Therefore, in the electrical domain, the 0- and 1-junctions represent parallel and series connections, respectively.

In describing the input-output behavior of a component, the independent variable at each bond must be determined. *Causality* expresses the computational dependencies between the effort and flow variables in

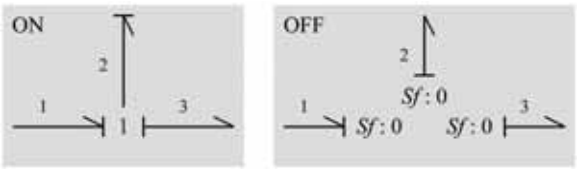


Figure 1. Semantics of a switching 1-junction.

the BG components. For example, the causality at a resistor, R, determines whether the direction of computation is $e = Rf$, or $f = e/R$. Similarly, at a 1-junction, causality denotes which bond's flow defines the flow values of all other incident bonds. Visually, causality is represented by a causal stroke at the end of the bond where the effort is imposed. Consequently, a flow is imposed on the other end of the bond. For example, in figure 1, bond 2 imposes flow on the 1-junction, when on.

The continuous BG representation has been extended to model hybrid systems by several researchers [6–9]. Our approach, hybrid bond graphs (HBGs), introduces discrete mode changes through idealized *switching junctions* that can turn *on* or *off*, thereby reconfiguring the energy flow paths in the model [10]. In the electrical domain, for example, a switching junction represents an electric switch that can connect or disconnect different circuit components. A two state (*on* and *off*) finite state machine implements the junction *control specification* (CSPEC). Transitions between states may be functions of system variables and/or system inputs. When a switching junction is on, it behaves like a conventional junction. When off, all bonds incident on a 1-junction (or 0-junction) are deactivated by enforcing 0 flows (or efforts) on all bonds incident on that junction (see Fig. 1). The system mode at any given time is determined by composing the modes of the individual switching junctions. The system configuration in each mode implies the causal structure among the system variables.

In our work, we adopt the block diagram (BD) formalism as the computation model for the HBGs because the input-output formulation of each block in a BD can be determined using the causality information captured by HBGs. BDs are also advantageous because: (*i*) the BD formalism is a widely used computational scheme, and (*ii*) BD models preserve the component structure of the model, which facilitates introduction of faults into components for simulation-based diagnosis and prognosis studies. When a junction switches on/off, it gains/loses its determining bond (see Fig. 1), thereby resulting in a change in the causality assignment, and hence the computational structure, at that junction. Moreover, the causality of adjacent elements may also need to be reassigned as a result of this change. Since mode changes result in causality reassignment in HBGs, we represent hybrid systems using *reconfigurable* BD models [2, 11]. These reconfigurable BD models include switching elements that enable the online reconfiguration of the BD components to account for different causality assignments in different system modes. Every time a mode change occurs, causalities are incrementally reassigned from the previous mode, and the effort and flow links are rerouted by the switching elements to ensure that the computational model matches the new causality assignment [2]. This approach, as is, produces acceptable results when mode changes are infrequent. However, for fast switching systems with frequent mode changes, e.g., electrical power conversion and distribution systems, invoking the procedure for reassigning causality at *every* mode change may produce unnecessary computational overhead, which leads to significant increases in the simulation execution times. Also, the BD models may include extraneous switching elements and signal connections, which account for causality assignments that never occur during the simulation. Hence, to improve the simulation efficiency, we identify bonds whose causal assignments are fixed across mode changes in the HBG model, and use this information to restrict possible reconfigurations that can occur in the simulation model. As a result, the generated BD models are space efficient because the number of switches needed for each BD component, as well as the number of possible signal connections are reduced. We also confine the propagation of the effects of mode changes to only the required parts of the simulation model, which are typically small in number, thereby reducing the computational effort required to execute mode changes during simulation. We demonstrate the effectiveness of our approach by applying it to a power conversion and distribution testbed developed for diagnosis and prognosis studies at NASA Ames Research Center [3].

2 Motivating example: AC inverter

The Advanced Diagnostics and Prognostics Testbed (ADAPT) models aircraft and spacecraft power distribution systems [3]. It includes batteries for power storage, inverters for DC to AC power conversion, a power distribution network made up of a number of

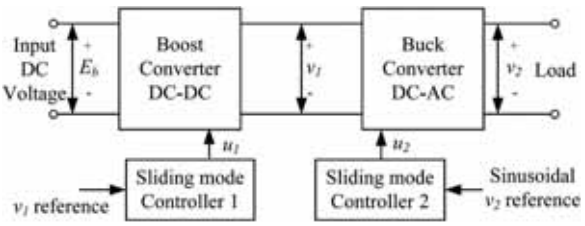


Figure 2. Block diagram of a boost-buck AC inverter.

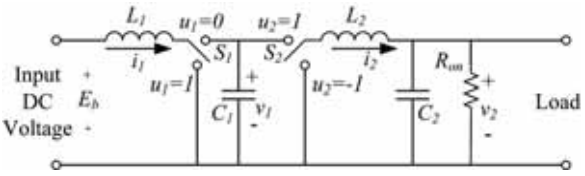


Figure 3. Circuit model of a boost-buck AC inverter.

circuit breakers and relays, and a variety of DC and AC loads. In this paper, we focus on the AC subsystem, and develop the fast-switching inverter model to motivate our approach. The inverter, a two-stage boost-buck DC-AC converter [4], consists of a cascade connection between a boost DC-DC converter with a full-bridge buck DC-AC converter to achieve a transformerless DC-AC step-up conversion (see figure 2). The boost converter boosts the input DC voltage to a higher value (190 V, in our case), and the buck converter stage generates the sinusoidal AC voltage. The fast switching in the boost and buck converters are governed by two sliding mode controllers, one for each stage of the inverter.

The equivalent circuit model of the boost-buck DC-AC inverter is shown in Fig. 3, where S_1 is a conventional power switch, and S_2 corresponds to a full bridge switch. The control signals for S_1 and S_2 are u_1 and u_2, respectively. The differential equations for the system can be found in [4], and the model parameters are listed in Tab. 1. The internal resistance R_{on} accounts for the current that the inverter draws from the battery when it is disconnected from its loads.

The HBG model of the inverter, derived from its circuit model, is shown in Fig. 4. Switch S_1 is represented by the synchronous switching junctions j_1 and j_2, i.e., they share the same CSPEC function. Switch S_2, is represented by the switching junctions j_3 - j_6, with j_3 and j_5 having the same CSPEC as junctions j_4 and j_6, respectively. The switching conditions for junctions j_3 and j_4 are logical negations of those for junctions j_5 and j_6.

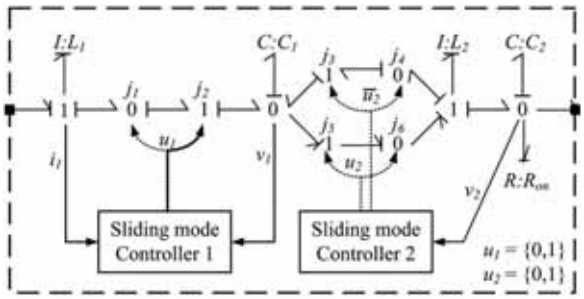


Figure 4. Inverter HBG component model.

Parameter	
Inductance (H)	$L_1 = 0.0022,\ L_2 = 0.075$
Capacities (F)	$C_1 = 0.0069,\ C_2 = 6\times10^{-6}$
Resistances (Ω)	$R_{on} = 489.49$
Sliding mode (controller 2)	$\alpha = 0.8,\ \beta = 4.3649,$ $\delta = 111.375,\ K = 829.3347$
Sliding mode (controller 1)	$a_1 = 15.915,$ $a_2 = 0.0048$
Reference voltages (V)	$v_{1ref} = 190,$ $v_{2ref} = 120\sqrt{2}\sin(120\pi)$

Table 1. Inverter Model Parameter Values

The sliding mode controllers are also modeled using HBGs. They generate signals that switch the inverter junctions at kilohertz rates. One approach to simulating the HBGs would be to invoke the causal reassignment procedure at every mode change to compute the updated model configurations before the continuous simulation is resumed [2]. However, careful inspection of the inverter HBG model shows that the causality assignments at the switching junctions remain the same when the junctions are on. When the junctions are off, the causal changes do not propagate to adjacent junctions (see the next section for details). Therefore, the calls to the causal reassignment procedure at every mode change are not necessary, and should be avoided since they considerably slow down the execution of the simulation. If we can identify the causality assignments that do not change when reconfigurations occur, the number of calls to the causal reassignment procedure can be reduced, and the model reconfiguration task can be simplified. The knowledge of causality assignments that do not change when reconfigurations occur can also be used to make the simulation models more efficient by not modeling system configurations that will never occur during system operation.

3 Efficient simulation of hybrid bond graph models

Efficient simulation models for hybrid systems should meet two primary requirements: *(i)* avoid pre-enumeration of all system modes, especially for systems with a large number of modes, and *(ii)* minimize the amount of computations performed to handle mode changes. Reassignment of causality produces changes in the computational model. But, we can minimize the number of changes during reconfiguration by *(i)* recognizing causal assignments that are fixed across all modes, and *(ii)* not allowing configurations that we can predetermine will never occur. As a result, we reduce the search space for the causal propagations possible as a result of the mode switch and also simplify the simulation models.

3.1 Converting bond graphs to block diagrams

Figure 5 shows the possible causal assignments for all BG elements [5]. The *Sf*, *Se*, *C* and *I* elements each have a unique causal assignment on their incident bonds which remain the same in all modes of system operation. In our work, we assume that the energy storage elements (*C* and *I*) are in integral causality, i.e., the computational models for these elements are in integral form, e.g., $e = \frac{1}{C}\int f\,dt$. The corresponding differential form, i.e., $f = C \cdot de/dt$, may introduce computational problems during simulation, and also requires knowing a future value to compute the derivative at the current time point [5]. The n-port I- and C-fields also have unique causal assignment across all system modes, and are not shown in Fig. 5 as they are simple functional and structural extensions to the 1-port I and C elements, respectively. The R, TF, and GY elements allow two possible causal assignments each, and each assignment produces a different BD model. We capture the notion of causality assignment at a junction through the commonly used notion of the determining bond.

Definition 1 (*Determining Bond*)
The *determining bond* of a 0-(1-)junction is the bond that establishes the effort (flow) value of all other bonds incident on the junction.

Mapping a junction to its BD model is facilitated by its determining bond. Fig. 6 shows the BD expansion for nonswitching junction (ignoring the off configuration). At a 1junction, all other bonds' flow values are equal to the determining bond's flow value, and the effort value of the determining bond is the algebraic sum of the effort values of the other bonds connected to this 1-junction, taking into account the direction of these bonds. A nonswitching junction with *m* incident bonds can have *m* possible BD configurations.

A standard algorithm for assigning causality to a BG is the Sequential Causal Assignment Procedure (SCAP) [5]. The basic idea of SCAP is to start at elements having a unique causality, such as energy sources and energy storage elements, to constrain the possible options for determining bonds for the junctions these elements are connected to, which in turn may constrain the options for determining bonds for adjacent junctions. If there exists a *unique* option for a determining bond for a junction, the junction is assigned that determining bond, and the constraints imposed by this assignment are propagated along the BG to further restrict the possible options for determining bonds at other junctions. After the causal changes have been propagated from all energy sources and energy storage elements, if a junction still has multiple options for its determining bond, one of its bonds with unassigned causality is arbitrarily assigned as its determining bond, and the constraints imposed by this assignment are propagated along the BG. SCAP terminates when every junction is assigned a determining bond.

Once a BG is assigned causality, a well-defined procedure can be applied for converting the BG structure to a BD model [5]. Based on the assigned causality, each BG element is replaced by the computational structure (see Figs. 5). In the BD model, each bond is replaced by two signals, i.e., the effort and flow variables for the bond. Once all necessary blocks of the BD are instantiated, they are connected appropriately to complete the BD model.

3.2 Converting hybrid bond graphs to block diagrams

The BD generation procedure described above needs to be extended for HBGs so that they can handle causality changes that occur due to mode changes. Instead of rebuilding the entire BD model every time mode changes occur, we include switching elements in the individual BD components to reconfigure the computational model during simulation. For example, a switching junction withmincident bonds can switch between $m+1$ possible computational configurations, m corresponding to each incident bond being a determining bond, and one corresponding to the junction being off, in which each outgoing signal is set to zero. With this method, the physical connections

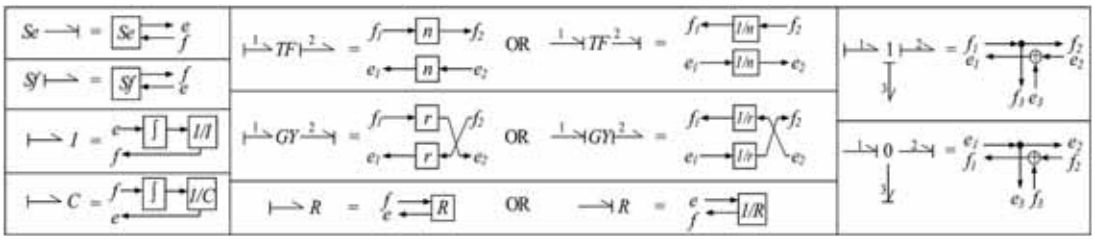


Figure 5. Computational structures for bond graph elements

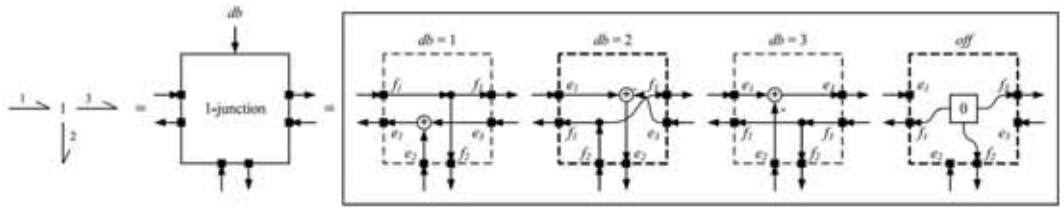


Figure 6. Block diagram expansion of a switching 1-junction (*db* denotes determining bond).

between blocks are fixed, but the interpretation of the signal on the connection (effort or flow) changes depending on the causal assignment to the bonds (see Fig. 6). In some cases, however, the causal assignment for a bond is invariant across *all* possible modes of system behavior, i.e., the causal assignment is *fixed*. For example, a *C* -element will always impose effort on a 1-junction through the connecting bond. In this case, the BD for the 1-junction does not need to include any switching mechanism to accommodate the possibility of this bond being its determining bond. If all bonds of a junction have fixed causal assignments, then its determining bond is invariant for all modes of the system. In this case, the BD for the junction does not need to include any switching mechanisms because it can assume a fixed structure. In previous work, we have termed a nonswitching junction with this property to have *fixed causality* [2]. Switching junctions, by definition, change causality when they turn off, but this change may not affect adjacent junctions. Therefore, we extend our previous definition of fixed causality to also include switching junctions.

Definition 2 (*Fixed Causality*)
A junction that does not switch is in *fixed* causality if, *for all* modes of system operation, its determining bond does not change. A switching junction is in *fixed* causality if, *for all* modes in which the junction is on, its determining bond does not change, and for all modes where it is off, the inactivation of its incident bonds does not affect the determining bond of any of its adjacent junctions.

Fixed causality of bonds and junctions can be identified efficiently using a SCAP-like algorithm before we construct the BD model from the HBG. In this algorithm, the causality assignment is first performed at junctions connected to sources and energy storage elements, because the bonds connecting them to these junctions have fixed causality. A 0- (or 1-) junction is in fixed causality if it is connected to a *Se* (or *Sf*) or a *C* (or *I*) element. Otherwise, a junction is in fixed causality if *(i)* its determining bond connects to a fixed causality junction (either directly, or through a *TF* or a *GY* element), or *(ii)* all incident bonds other than its determining bond are connected to fixed causality junctions. Once a junction is determined to be in fixed causality, we propagate this information to its adjacent junctions to check if they too are in fixed causality, or any of their bonds are in fixed causality.

Some additional analysis is required to determine whether a switching junction is in fixed causality. A switching junction is in fixed causality if: (*i*) whenever the junction is on, its determining bond is the same, and (*ii*) when the junction turns off the determining bond for its adjacent junctions should not change, i.e., none of its incident bonds can be a determining bond for its adjacent junctions. If two switching junctions are adjacent and share the same CSPEC, the knowledge that they switch together can help determine if causal changes will propagate when they switch. When we visit a junction for the first time, all its adjacent junctions may not have been checked for fixed causality yet. Hence, causality may need to be propagated from all adjacent junctions before it can be determined that the junction is in fixed causality.

If a junction has incident bonds with fixed causality, or if the junction itself is in fixed causality, the computational model of the junction block can be reduced by eliminating switching elements and signal connections which account for causality assignments that will never occur during simulation. Consider the 3-port switching 1-junction in Fig. 6. If the junction is not in fixed causality, its implementation can, in general, switch between four possible configurations as mode changes occur. However, if the junction is in fixed causality, the BD representation for this junction has only one valid on configuration, in addition to the off configuration. If the junction is not in fixed causality, for example, if its bond 1 is connected to a *Se*, its BD representation need not include a configuration with bond 1 as its determining bond. Switched junctions in fixed causality help minimize causality reassignment computations when mode changes occur. Therefore, when a 1-junction in fixed causality changes mode, we know exactly what the causality assignment at this junction is without having to call any external causality reassignment procedure, and can build this into the BD.

3.3 Efficiently reassigning causality in hybrid bond graphs

The naïve approach to causality reassignment in HBGs is to run SCAP on the new HBG configuration *every* time a mode change occurs, but this approach is likely to be inefficient for the following reasons: (*i*) usually, only a small part of the HBG needs to be reassigned causality, and (*ii*) sometimes, changes in causality do not propagate and the effect of a mode change only produces local changes in the computational model structure. Our causality reassignment method, called the *Hybrid Sequential Causal Assignment Procedure* (Hybrid SCAP) reassigns causality incrementally, starting from the junctions directly affected by the switching, and then propagating the changes only to those junctions whose causal assignments are affected by changes in the adjacent junctions using the causal assignment of the previous mode [2, 11]. At junctions where a unique choice for a new determining bond is not known, an arbitrary choice may be made. But, this choice may lead to an inconsistent assignment if the propagation reaches a fixed causality junction. An inconsistent assignment can also be made if the propagation reaches a junction whose determining bond has been unequivocally assigned for that particular system mode. Such junctions are considered to be in *forced causality*.

Definition 3 (*Forced Causality*)
A junction is in *forced causality* if it can be assigned only one possible determining bond in a given system mode.

We use the knowledge of junctions in fixed and forced causality to reduce the search space for the Hybrid SCAP algorithm by not propagating causal changes across fixed and forced causality junctions. As a result, expensive backtracking is avoided. The worst case computational complexity of our Hybrid SCAP approach is polynomial in the size of the HBG, as it is similar to the SCAP algorithm. The average case complexity of our approach, however, is better than that of SCAP, since in many cases, only small parts of the HBG change causality. Consider the inverter HBG model (Fig. 4), where all nonswitching junctions are in fixed causality. The incident energy storage elements specify a unique determining bond for these junctions. All switching junctions are also fixed. Consider switching junctions j_1 and j_2. Since they always change modes simultaneously (because they share the same CSPEC), when on, j_1 always imposes flow on its adjacent junction j_2 which is also on. When they are off, the causality assignment of other active junctions are not affected. The case is similar for pairs j_3 and j_4, and j_5 and j_6. Since all junctions are in fixed causality, the mode switchings in the inverter do not require reassignment of causality, because the changes never propagate. Therefore, Hybrid SCAP is not invoked, and minimal changes have to be made to the computational model when mode changes occur, thus considerably speeding up the inverter simulation, as we illustrate later.

3.4 Simulating the block diagrams

Our modeling and simulation approach defines the reconfigurable BDs and their reconfiguration procedures, and therefore is independent of the underlying fixed or variable-step solver being used by the simulation environment in which the simulation model is executed. Once the reconfigurable BD model is generated in the simulation environment, simulation starts with the BD structure corresponding to the current mode, and the simulation continues till a mode change occurs. If the mode change is attributed to the switching of junctions that are not in fixed causality, the simulation is paused, the Hybrid SCAP algorithm is invoked to reassign causality, and the BD is reconfigured accordingly before the simulation resumes. On the other hand, if a mode change is at-

tributed to a switching junction with fixed causality, the BD is reconfigured based on the junction mode, without invoking Hybrid SCAP.

4 Software implementation

We have implemented our methodology using a modelintegrated computing approach [12] as the MOdeling and Transformation of HBGs for Simulation (MOTHS) tool suite [13]. It consists of a modeling language for building graphical component-based HBG models [1], and a set of model translators, or *interpreters*, that convert these HBGs into block diagram-based simulation models for selected simulation tools. Initially, we have developed an interpreter that creates MATLAB Simulink models, details of which can be found in [2].

Simulation testbeds have a variety of applications. One of the primary objectives of our simulation approach is to facilitate prognosis and diagnosis experiments. Hence, we have developed schemes to inject faults in the simulation model. We model faults as off-nominal changes in system parameters (e.g., a resistance change representing a blockage in a pipe), called *parametric faults*, and off-nominal changes in system mode (e.g., a stuck valve), called *discrete faults*.

Our approach facilitates fault injection in Simulink through a graphical user interface that is constructed automatically during the interpretation process. The user can select the time of fault occurrence (we consider only persistent faults), the fault profile, and the fault magnitude (where applicable). Parametric faults are associated directly with HBG element parameter values (i.e., *Se*, *C*, *R*, etc.), and the available fault profiles include abrupt, incipient, and stuck-at faults. An abrupt fault is a fast change in a system parameter whose time constant is much smaller than that of the system dynamics, and is modeled as a constant bias term that gets added to the nominal parameter value at the time of fault occurrence. An incipient fault is a slow change in system parameters, and modeled as an linear drift term (with constant slope) that is added to the actual parameter value. A stuck-at fault sets the parameter value to a constant value at fault occurrence. Discrete faults cause the system mode to change, so they affect the switching junctions. Our simulation approach allows the injection of *stuck-on*, *stuck-off* faults, introduced by adding fault terms to the CSPEC transition guards. As a special case, sensor faults can typically be modeled as either parametric or discrete. In our approach, we treat them as parametric faults, allowing abrupt (bias), incipient (drift), and stuck-at faults (which can also model discrete faults such as sensor failure). Additionally, we allow the introduction of white Gaussian sensor noise, where the mean and variance can be specified. A change in variance can also be introduced as a sensor fault.

5 Case study

We demonstrate our modeling and simulation framework using the ADAPT system at NASA Ames Research Center [3]. In this paper, we focus on a subsystem of ADAPT shown in Fig. 7, i.e., a battery driving an inverter that is connected to two loads through two relays. One of these loads is a light bulb, while the other load is a large fan.

We build the system model using a component-based modeling approach [1, 3]. We use the HBG model of the inverter shown in Fig. 4 for our experiments. The HBG model of the lead-acid battery is based on an electrical equivalent circuit model, which captures the nonlinear battery behavior [11]. We omit a detailed description of the battery model due to space constraints. The resistive light bulb load consists of a single power port, a 1-junction, and a *R* element. The *R* element parameter value is a constant $234.6966\,\Omega$.

The AC fan is modeled as a single phase, fixed capacitor induction motor [14]. We represent the system using the standard $d-q$ model, described in [15]. We do not present the fan model equations due to space constraints. The model parameters are presented in Table 2. The HBG model of the fan, shown in Fig. 8, is adapted from that described in [16]. Note that the AC fan has two *I*-fields with parameters,

$$I_1 = \begin{bmatrix} L_{ss} & L_{ms} \\ L_{ms} & L'_{rr} \end{bmatrix}, \quad and \quad I_2 = \begin{bmatrix} L_{SS} & L_{ms} \\ L_{ms} & L'_{RR} \end{bmatrix} \qquad (1)$$

5.1 Experimental results

For the experimental results, we automatically derived a Matlab Simulink model of the subsystem shown in Fig. 7, from the HBG model of the system using the MOTHS tool suite. All experiments were performed on a 2.4 GHz Intel Pentium Core 2 Duo CPU desktop, with 2 GB RAM. The model was simulated using a fixed-step simulation with a sample period of $7.5\,\mu s$.

Fig. 9 shows the results of the simulation. We plot the voltages and currents at the output of the battery and

the inverter, as well as the rotational speed of the AC fan. The simulation model was run for 20 seconds of simulation time. In the first configuration, the light bulb is connected to the inverter from 2 − 5 seconds, followed by a second configuration where the AC fan is turned on between 7 − 15 seconds. An abrupt fault is injected in the light bulb resistance at 3 seconds to demonstrate the usefulness of the simulation approach for diagnosis applications. As we can see, the sliding mode controllers are robust to load changes, and generates true 120 V rms voltage for both the loads. However, the light bulb fault affects the inverter current, and, therefore, the battery current and voltage. The AC fan current shows a phase difference of 0.1346 rad. As can be seen in Fig. 9 when the AC fan is switched on, its speed of rotation increases until it reaches a steady state of about 78.5 rad/s. On turning off, the speed falls to zero.

Table 3 presents the result of an experiment to illustrate the efficiency gained by simplifying a reconfigurable BD model by identifying bonds with fixed causal assignments and junctions in fixed causality, and avoiding the need for causal reassignment for these modes. For this experiment, we assume that the AC fan is the only load and is on for the duration of the experiment. Each column in Table 3 reports the real time taken to simulate 1 second of simulation time for different HBG simulation runs. In all runs, all junctions of the HBG model are in fixed causality. In the first run, we call SCAP every time an inverter mode change occurs. Next, we repeat the previous run, using Hybrid SCAP. Finally, in the third run, we simulate the HBG without requiring any external calls to Hybrid SCAP, since all switching junctions are fixed. As can be seen from Table 3, our enhanced simulation approach, implemented in the third run, is 103.85 times faster than the first run, and 103.35 times faster than the second run. Our simulation approach also resulted in considerable improvements in the efficiency of simulation of a number of other configurations, especially for large systems like the VIRTUAL ADAPT simulation testbed [3]. Further increase in simulation efficiency can be obtained by running our simulation models in the Rapid Accelerator mode of Simulink.

6 Conclusions

In this paper, we have presented an approach for modeling and simulation of complex systems with switching behavior using hybrid bond graphs. The crux of this modeling and simulation framework, implemented as the MOTHS tool suite, is the identification of fixed causality of bonds, which not only avoids unnecessary invocations of the external Hybrid SCAP algorithm, thereby gaining increase in simulation efficiency, but also improves the efficiency of the Hybrid SCAP algorithm, as well as enables the simplification of the simulation models by removing parts that correspond to configurations that never occur during the simulation. In the future, we will extend our modeling approach and computational model generation schemes to systematically evaluate how our approach performs when applied to other real-world large hybrid systems. We also wish to develop interpreters to generate simulation models in other simulation software, such as Ptolemy [17].

Acknowledgements

This work was supported in part by the National Science Foundation under Grants CNS-0615214 and CNS-0347440; and NASA NRA NNX07AD12A.

References

[1] E.-J. Manders, G. Biswas, N. Mahadevan, G. Karsai. *Component-oriented modeling of hybrid dynamic systems using the Generic Modeling Environment*, in Proc of the 4th Workshop on Model-Based Development of Computer Based Systems. Potsdam, Germany: IEEE CS Press, Mar. 2006.

[2] Roychoudhury, M. Daigle, G. Biswas, X. Koutsoukos, P. J. Mosterman, *A method for efficient simulation of hybrid bond graphs*, in Proc of the International Conference of Bond Graph Modeling, San Diego, California, 2007, pp. 177 – 184.

[3] S. Poll, A. Patterson-Hine, J. Camisa, D. Nishikawa, L. Spirkovska, D. Garcia, D. Hall, C. Neukom, A. Sweet, S. Yentus, C. Lee, J. Ossenfort, I. Roychoudhury, M. Daigle, G. Biswas, X. Koutsoukos, R. Lutz, *Evaluation, selection, and application of model-based diagnosis tools and approaches*, in AIAA Infotech@Aerospace 2007 Conference and Exhibit, May 2007.

[4] D. Biel, F. Guinjoan, E. Fossas, J. Chavarria, *Sliding-mode control design of a boost-buck switching converter for ac signal generation*, IEEE Transaction on Circuits and Systems - I, vol. 51, no. 8, pp. 1539–1551, 2004.

[5] D. C. Karnopp, D. L. Margolis, R. C. Rosenberg, *Systems Dynamics: Modeling and Simulation of Mechatronic Systems*, 3rd ed. New York: JohnWiley & Sons, Inc., 2000.

[6] J. Buisson, H. Cormerais, P.-Y. Richard, *Analysis of the bond graph model of hybrid physical systems with*

ideal switches, Proc Instn Mech Engrs Vol 216 Part I: J Systems and Control Engineering, pp. 47–63, 2002.

[7] M. Magos, C. Valentin, B. Maschke. *Physical switching systems: From a network graph to a hybrid port hamiltonian formulation*, in Proc IFAC conf Analysis and Design of Hybrid Systems, Saint Malo, France, June 2003.

[8] J. van Dijk, *On the role of bond graph causality in modeling mechatronics systems*, Ph.D. dissertation, University of Twente, Enschede, The Netherlands, 1994.

[9] W. Borutzky, *Discontinuities in a bond graph framework*, Journal of the Franklin Institute, vol. 332, no. 2, pp. 141–154, 1995.

[10] P. J. Mosterman, G. Biswas, *A theory of discontinuities in physical system models*, Journal of the Franklin Institute, vol. 335B, no. 3, pp. 401–439, 1998.

[11] M. Daigle, I. Roychoudhury, G. Biswas, X. Koutsoukos, *Efficient simulation of component-based hybrid models represented as hybrid bond graphs*, in HSCC 2007, ser. LNCS, A. Bemporad, A. Bicchi, G. Butazzo, eds. Springer, 2007, vol. 4416, pp. 680–683.

[12] J. Sztipanovits and G. Karsai, *Model-integrated computing," Computer*, vol. 30, no. 4, pp. 110–111, Apr 1997.

[13] MOTHS, `http://macs.isis.vanderbilt.edu/software`.

[14] G. J. Thaler, M. L. Wilcox, Electric Machines: *Dynamics and Steady State*. John Wiley & Sons, Inc., 1966.

[15] P. C. Krause, Analysis of Electric Machinery. John Wiley & Sons, Inc., 1986.

[16] D. Karnopp, *Understanding induction motor state equations using bond graphs*, in Proceedings of the International Conference on Bond Graph Modeling and Simulation, vol. 35, no. 2, 2003, pp. 269 – 273.

[17] J. Buck, S. Ha, E. A. Lee, D. G. Messerschmitt, *Ptolemy: a framework for simulating and prototyping heterogeneous systems*, Readings in hardware/ software co-design, pp. 527–543, 2002.

Corresponding author: Indranil Roychoudhury,
Inst. Software Integrated Systems, Dept. of EECS
Vanderbilt University, Nashville, TN 37235, USA
indranil.roychoudhury@vanderbilt.edu

Received: September 17, 2008
Revised: October 20, 2008
Accepted: October 28, 2008

Clearification of Dependability Concepts – An Approach to the Precise Modeling of Essential Terms

Jörg Rudolf Müller, Eckehard Schnieder, Technical University Braunschweig, Germany

In the traditional scientific ”divide and conquer” method, systems are broken into distinct parts which are examined separately. By contrast, the system-theoretic approach focuses on systems as a whole. The underlying assumption is that systems can only be treated adequately in their entirety.

This paper focuses on systems and what one can know or recognise about them. So, system-theory as well as the theory of knowledge are addressed. In addition the correspondences between the non-formally but well distinguished concepts of system-theory and formal concepts of Petri net-theory are identified. On the basis of these correspondences some of the essential concepts of system-theory are defined in a formal way. In this context, special attention is paid to risk and safety. Unlike common definitions, they are treated as system-concepts and are defined in a corresponding way.

SNE 18/3-4, December 2008

Introduction

System-theory serves with its in fact not formally but well distinguished concepts as an instrument for the cognitive penetration of (circumstances of) systems as well as a platform for the interdisciplinary discourse. With the help of system theory it is possible to question and if so, to revise or to validate one's own individual perspective as well as to raise significantly the level of interdisciplinary discourses.

In this paper some of the essential concepts of system-theory are introduced informally (Section 1.1) and formally on the basis of petrinets (Section 1.2 and 1.3). The formal definitions are exemplified at a simplified level crossing example (Section 1.4). In Section 2 hazards, risk and functional safety are defined formally on a system theoretic basis. The essential results of this paper and the future work are outlined in Section 3.

1 Preliminaries

In this paper the position of constructivism is taken up. This position says that it is impossible to get an objective image of reality. Perception of real-world cutaways always base on individual knowledge, experience and perspective (see [10, 11]).

Through constructing an individual cognitive constitution-system (see [1, 2]) expectations and speculations concerning the *re*cognition arise. These expectations lead to a selective perception in a way, such that the constitution-system is manifested as the actual and objective image of reality (see [4, 10, 11]).

So, when talking about systems, one necessarily talks about systems as individual images of real-world systems. Therefore ”systems” in the following means individual representations of cutaways of the real world.

1.1 System-Theory

Ludwig von Bertalanffy introduced in 1949 system-theory as a response to certain limitations of classical physics with its divide and conquer method [9]. Here, problems are divided into parts which are examined separately. This principle of reductionism makes three important assumptions (see [6, 9]):

- The division into parts will not distort the phenomenon being studied.
- The components of the whole are the same when examined singly as when they are playing their part in the whole.
- The principles governing the assembling of the components into the whole are straightforward.

These assumptions are reasonable for many systems that may be described as exhibiting *organized simplicity* (see [6, 9]).

Systems, that are more complex but regular enough so that their behaviour can be described statistically may be labeled as *unorganized complexity* (see [6]).

The third type of systems is too organized for statistics but too complex for a complete analysis. This type is called *organized complexity* (see [6, 9]).

In this section a compilation of the central concepts of system-theory in natural language is introduced (see [8]).

Definition 1 (System)
A *system* is an effective combination of objects to a whole.

The constitutive properties of systems – *combination*, *whole* and *effective* are to be explained in detail:

- a system is a combination - therefore: a system consists of at least two ”objects” which we do not specify further here (remember that systems here are just cognitive representations of e.g. physical or mathematical objects). If we consider an object s belonging to a system S as a system, too, we call the system s a *subsystem* of S.
- the combination of the identified objects s of S establishes a whole. That means the combination itself is regarded as *one* object and no more several objects.
- the combination of objects to a whole is effective – that means it is a combination because of an aim or a purpose.

Especially the last point, the effectiveness, needs to be discussed in more detail: The fact, that we combine objects in an effective manner, implies firstly that these objects correlate in a special way (the objects are in some relation or they interact), secondly that different purposes lead to different combinations of (if so: the same) objects and so (identical) real-world extracts may lead to the perception of different systems. That means, if a whole is seen as a system, a purposeful order is established. Due to this purpose, the perception of systems occurs basically in a pragmatic way.

So, one can draw the consequence, that real systems do not exist a priori – a system can be seen as a brainchild.

Definition 2 (Open system)
Open systems react to (if so: interact with) their environment.

Due to a lack of knowledge about the environment, its effect on the system can not exactly be quantified and has to be (if so: on the basis of test series) estimated.

Definition 3 (System state)
The *state* of the system quantifies the useful inner values (of objects); in that way, it represents the system's (objects) informational and / or physical status. So, what we call a state is like the system itself, depending on the specific purpose of the system.

Definition 4 (System border and environment)
As a system is, as seen before, perceived due to pragmatic aspects, the *border of a system*, that means the exclusion of the system's environment from the system itself is done in a pragmatic manner, too. Therefore, the system's border, too, is dependable from purpose.

Definition 5 (System structure)
The objects of a system and the relations between them constitute the *structure of a system*. Thus, the structure of a system is (in accordance to the system's relations) dependable from a specific purpose. Additionally, open systems are in relation to their environment. These relations can only be – due to the lack of knowledge of the environment – approximated and are described with stochastic functions and nondeterminisms.

Definition 6 (System relations)
As mentioned in the context of ”effectiveness”, the objects belonging to a system establish specific relations. Thus, the objects-relating relations are mainly chosen in dependence of specific purposes and they influence the system's border.

Definition 7 (System behaviour)
The system's *behavior* is the set of all the performable and observable state sequences (state traces). If the behaviour of the objects belonging to a system and of the interactions between them is known, the behaviour of the system gets predictable. Due to the lack of knowledge in open systems according the environments influences, the behaviour-prediction gets non-deterministic or stochastic.

Definition 8 (System function)
The *function of a system* is the set of all transformations between system states, that fulfill the purpose under which the system is looked at. So, the system's function causes behaviour (see below), that suits the intended look. Behaviour contradictory to the systems purpose may evolve, due to reactions on influences from the system's environment.

Definition 9 (Emergence)
A property of a system S is called an *emergent* property, if it occurs only by or can be explained through interactions of objects s that belong to S. An isolated view on the $s \in S$ won't explain the very property.

Remark Emergent properties constitute hierarchical levels: An emergent property *ep* of level l_k is being created by the interaction of objects on level l_{k-1}. The property *ep* does not exist on the levels l_j $(j<k)$ – they are meaningless on these levels.

1.2 Petri-net Theory

Definition 10 (Petrinet)
A *petrinet* (pn) is a quadrupel $N=(P,T,F,m_0)$, with

P is a set of *places*
T is a set of *transitions*
$F \subseteq (P\times T)\cup(T\times P)$ is a *flow relation*
$m_0 : P \to \mathbb{N}$ is called a *marking* of N

In graphical representations places are drawn as circles, transitions as rectangulars, the elements of $f \in F$ as directed arcs. The marking $m(p)$ is usually drawn as black dots that are called *tokens*.

Definition 11 (Preset and postset)
Let N be a pn. For every $n \in P \cup T$ we call

$^\bullet n := \{n' \mid (n',n) \in F\}$ the preset of n, and
$n^\bullet := \{n' \mid (n',n) \in F\}$ the postset of n.

Definition 12 (Enabled, Firing, Follower Marking)
Let $N=(P,T,F,m_0)$ be a pn and m a marking of. The transition $t \in T$ is enabled under m (in symbols: $m[t\rangle$), if $\forall p \in {}^\bullet t : m(p) \geq 1$.

If t is enabled under m, t may fire and transform the marking m to the follower marking m' (in symbols: $m[t\rangle m'$) with

$$m'(p) = \begin{cases} m(p) & \text{if } p \in {}^\bullet t \cap t^\bullet \\ m(p)-1 & \text{if } p \in {}^\bullet t \setminus t^\bullet \\ m(p)+1 & \text{if } p \in t^\bullet \setminus {}^\bullet t \end{cases}$$

Definition 13 (Reachable Markings)
Let $N=(P,T,F,m_0)$ be a pn. If there is a sequence $\sigma = t_1,t_2,...,t_n$ with $m_0[t_1\rangle_N m_1[t_2\rangle_N ...[t_n\rangle_N m_n$, then marking m_n *is reachable* in N from the initial marking m_0 due to firing σ (in symbols: $m_0[\sigma\rangle_N m_n$). The set of all reachable markings (in symbols: $[m_0\rangle$ can be defined as follows:

$$[m_0\rangle_N := \{m \mid \exists \sigma \in T^\bullet : m_o[\sigma\rangle m$$

Definition 14 (Stochastic Petrinet)
A *stochastic petrinet* (spn) is a sextuple $N=(P,T,sT,F,\Pi,m_0)$ with

(P,T,F,m_0) is a pn,
$sT \subseteq T$ is the set of stochastic transitions
$\Pi : sT \to \text{Prob}$ is the probability function

with $\text{Prob} := \{\rho \mid 0 \leq \rho \leq 1\}$ is the set of probabilities. So, $\Pi(t)$ specifies the firing probability of transition $t \in sT$, with $({}^\bullet t)^\bullet = \{t\} \to \Pi(t) = 1$. Transitions in sT are drawn as grey boxes.

In general one has to distinguish the following transitions (see [12]):

- Causal transitions, i.e. transitions modeling timeless relations. They are usually drawn as thin bars.
- Deterministic timed transitions. These are drawn as thick bars.
- Stochastic timed transitions. If the firing time is exponentially distributed, such transitions are drawn as unfilled boxes.

Definition 15 (Nondeterminism)
Let $N=(P,T,sT,F,\Pi,m_0)$ be a spn. If $m \in [m_0\rangle_N$ and $m[t_1\rangle_n$, and $m[t_2\rangle_n$ with $t_1,t_2 \in T$ then the t_1 and t_2 are nondeterministic transitions.

1.3 The correspondences between system theory and petrinet theory

In this chapter the correspondences between some of the system-theoretic concepts introduced (in natural language) and the concepts presented (in a formal language) are outlined.

Remark With a *spn* the states of the objects of a system S and the relation between these states can be represented with markings of places and transitions of the net, respectively. PNs are not appropriate to model the static structure of systems – UML class diagrams are more adequate for this kind of representation.

Definition 16 (System border)
As stochastic and nondeterministic state changes in a system S refer to a lack of information within the system's state or structure, information from outside the system is needed. That means: In N the system border consists of the set of stochastic and nondeterministic transitions.

Predefinition In the following definitions let S be a system with the set of objects $O=\{o_1,...,o_n\}$ and $N=(P,T,sT,F,\Pi,m_0)$ be a spn.

Definition 17 (System state)

The set of possible states $\{s_{i_1}\{o_i\},...,s_{i_k}(o_i)\}$ of o_i is modeled in N with a set of places $\{p_{i_1},...,p_{i_k}\} \subseteq P$. If o_i is in state j, then $m(p_{i_j}) = 1$ and $m(p_{i_l}) = 0$, $\forall l \in \{1,...,k\}$ and $l \neq `j$.

The state of the system S is represented through the marking of $N, m(P)$.

Definition 18 (System relations)

Two objects o_i and o_j, with $i \neq j$, interfere, if there is a possible state change from $s_{j_m}(o_j)$ to $s_{j_n}(o_j)$ such that the state change from $s_{i_m}(o_i)$ to $s_{i_n}(o_i)$ is a necessary condition.

This relation of o_i and o_j can in N be modeled in the following way:

$$\exists t \in T : \{p_{j_m}, p_{i_m}) \subseteq {}^{\bullet}t \ \wedge \ \{p_{j_m}, p_{j_n}\} \subseteq t^{\bullet}$$

Definition 19 (System behaviour)

The behaviour of a system S, starting at an initial state s_0 is modeled through the set of reachable markings in N, starting at the initial marking $m_0 : [m_0\rangle_N$.

Definition 20 (System function)

The function of a system S is specified through these state changes that are responsible for the system's purpose.

On the highest system level and in open systems, these state changes lead to an interaction with the environment, on lower levels the behaviour of at least two objects (i.e. subsystems here) may interfere in a way that an emergent property is established.

Definition 21 (Emergence)

Let $N_1 = (P_1, T_1, sT_1, F_1, \Pi_1, m_1)$ and $N_2 = (P_2, T_2, sT_2, F_2, \Pi_2, m_2)$ be two spn, with $P_1 \cap P_2 = \varnothing$ and $T_1 \cap T_2 = \varnothing$. Let N_{12} be defined as $N_{12} := (P_1 \cup P_2, T_1 \cup T_2, F_1 \cup F_2, \Pi_1 \cup \Pi_2, (m_1, m_2))$ and let further be N_3 defined as follows $N_{12} := (P_1 \cup P_2, T_1 \cup T_2, F_3, \Pi_1 \cup \Pi_2, (m_1, m_2))$ with $F_1 \cup F_2 \subseteq F_3 \subseteq F_1 \cup F_2 \cup (P_1 \times T_2) \cup (P_2 \times T_1) \cup (T_1 \times P_2) \cup (T_2 \times P_1)$

Therefore, $[(m_1, m_2)\rangle_{N_3} \subseteq [(m_1, m_2)\rangle_{N_{12}}$, i.e. in comparison to the behaviour of N_{12} (through $[(m_1, m_2)\rangle_{N_{12}}$), the behaviour of N_3 is – at least in general – restricted (defined through the relations in the subset of $(P_1 \times T_2) \cup (P_2 \times T_1) \cup (T_1 \times P_2) \cup (T_2 \times P_1)$. If this restriction realizes a function on the N_3 system-level, then this function is an emergent property of N_3.

1.4 General Concepts of System-Theory – Example

In this section some of the system-theoretic concepts defined before are illustrated at a simplified example of a transportation system. From the viewpoint here, the purpose of a transportation system is to enable the safe transport of humans and goods. Against this background, the *spn* in Figure 1 is the model of a system, because

- the behaviour of two objects – an arbitrary train and a level-crossing – is modeled. In this example, we do not consider these objects as subsystems;
- the combination establishes a whole, here: a transportation system;
- this combination is effective, i.e. the train and the level-crossing account for a transport of humans and goods that is safe.

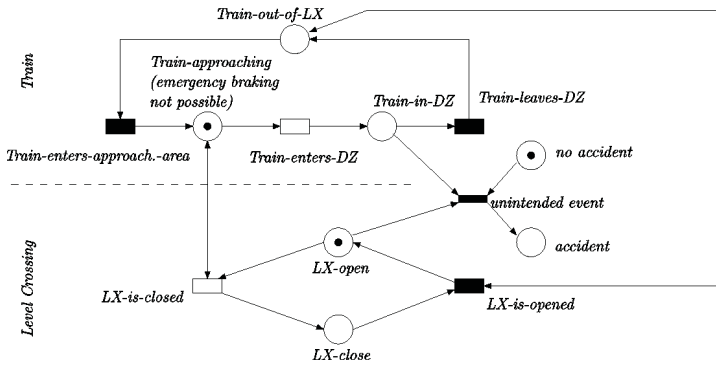


Figure 1. Petri-net model of a "transportation system"; environment implicitely modeled by stochastic transitions within the modules of the systems "Level Crossing" and "Train"

The border of the system consists of the stochastic transitions. For example, one does not know if the level crossing really closes when it should (it may be broken in some way). Stochastic transitions model uncertainties. The uncertainties concerning the behaviour of the level-crossing are rooted in the uncertainties about the reliability of the level crossing. So, one may include the reliability of the level-crossing (see Figure 2), but that does not help in the end.

The system's environment is everything that may directly or indirectly influence the system through the statistic transitions. For example weather conditions may influence the behaviour of the level-crossing.

As there are stochastic transitions in the model, the system itself is an open system.

The transitions model the relations of the systems. Besides the relations of different states of the same object, the relations between the status of the two objects are modeled.

The marking of the net represents the state of the system. The marking shown represents the initial state: The train is out of the level-crossing and the level crossing is open.

The behaviour of the system includes all reachable markings, starting from the initial one. Here, even the accident state is reachable. This state is reached, if the train is in danger zone and the level crossing is open.

The function of the system is to ensure the transport of humans and goods and to avoid accidents.

Here, safety is modeled as an emergent property, because only through the specific interaction between the two objects "level crossing" and "train", the *safe* transport can be assured. Please note, that "assure" does not mean "with a probability of a 100 %" due to possible environmental interferences (stochastic transitions "LX-is-closed" and "LX-is-opened").

2 The formal modelling of hazards, risk and safety

2.1 Hazards and Risk

According to IEC 61508-4 (see [5]) *risk* is the "combination of probability of occurrence of harm and the severity of that harm" and *harm* is a "physical injury or damage to the health of people either directly or indirectly as a result of damage to property or to the environment".

The precursors of harm are often defined as states and are called hazards. The following definition is taken from (see [6]):

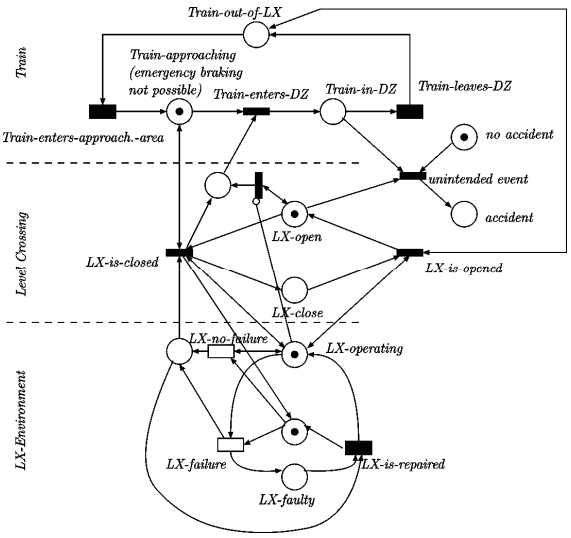


Figure 2. Petri-net model of a "transportation system"; environment explicitely modeled

Definition 22 (Hazards)
A *hazard* is a state or a set of conditions of a system (or an object) that, together with other conditions in the environment of the system (or object), will lead inevitably to an accident (loss event).

After that definition the pictured states in Fgure 1 and Figure 2 are hazardous states: This state will, under certain environmental conditions (LX-defective), lead inevitably to an accident state. In the state shown in figure 1, transition "Train-enters-DZ" fires with probability of 1% without previous firing of transition "LX-is-closed". So, one had to control, i.e. to prevent, this state – but there is nothing one can do to prevent this state in that model and at the same time maintain the availability of the train and of the level-crossing. The state reached is m(Train-in-DZ) = 1 = m(LX-open) and the only activated transition is "unintended event".

In Figure 2 the environment of the system is modelled explicitely: The only stochastic transitions of the model are in the modul that models the environment of the level-crossing. It can be further mentioned, that we adopt here the modelling technique described in the IEC 56/1266/CD (see [12]): The model can be devided into a process, a function and a dependability part – the corresponding submodels are "Train", "Level Crossing" and "LX-Environment" which models the dependability of the "Level Crossing".

The hazard state of the system in Figure 2 is

$$\text{hazard} = \{\text{Train-approaching}, \text{LX-open}\}$$

Note that the local states (markings) of all other places do not influence the hazard-property. The probability of that hazard is

$$P(\text{hazard}) = P(\text{Train-approaching}) \cdot P(\text{LX-open})$$

That means, only the system-inner states (i.e. states of the modules "Train" and "Level Crossing") are considered when calculating the probability of a hazard. By contrast, calculating the probability of the risk emerging from that hazard takes into account environmental influences – here, a possible failure of the level crossing:

$$P(\text{risk}) = P(\text{hazard}) \cdot P(\text{LX-faulty}) \quad \text{and}$$
$$\text{risk} = P(\text{risk}) \cdot \text{risk-severity}$$

To control that hazard, one may refine the dependability of the LX like in figure 3): The failing of the LX depends on two events:

- The supervisor of the LX must fail due to some external reasons (exponentially distributed transition "Sup-fail") and
- The LX must fail due to some external reasons as well (exponentially distributed transition "LX-failure").

That means that the marking of the net does not model a hazard any more: The failing of the supervisor *or* the level crossing does not necessarily lead to an accident. In contrast, the probability of the hazard states are

$$P(\text{hazard}_1) = P(\text{Train-approaching}) \cdot P(\text{LX-open}) \cdot P(\text{Sup-faulty})$$
$$P(\text{hazard}_2) = P(\text{Train-approaching}) \cdot P(\text{LX-open}) \cdot P(\text{LX-faulty})$$

Now, when calculating the probability of the risk that is emerging from these hazard one has to take into account two (here: independent) environmental conditions: the level crossing as well as the controller must be faulty:

$$P(\text{risk}) = 1 - \big((1 - P(\text{hazard}_1) \cdot P(\text{LX-faulty})\big) \cdot \big((1 - P(\text{hazard}_2) \cdot P(\text{Sup-faulty})\big)$$

and $\text{risk} = P(\text{risk}) \cdot \text{risk-severity}$.

2.2 Functional Safety

According to IEC 61508-4 (see [5]) *safety* is the "freedom of unacceptable risk" and *tolerable risk* is the "risk which is accepted in a given context based on the current values of the society". In Europe, often one of the following acceptable criteria is applied (see e.g. [3]):

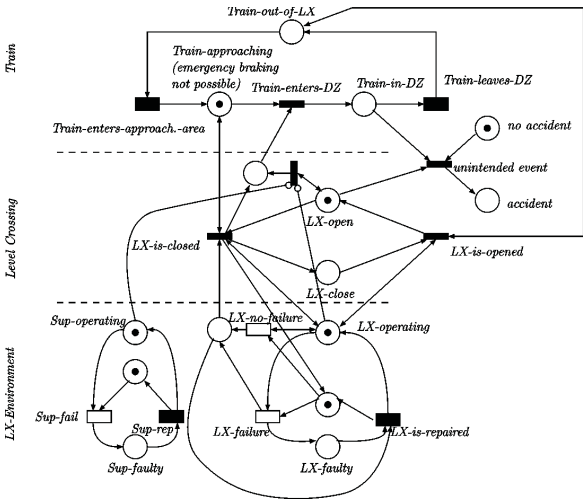


Figure 3. Petri-net model of a "transportation system"; environment explicitely modeled, supervisor failure taken into account

- As Low As Reasonably Procticable (ALARP) – often used in Great Britain.
- Globalement Au Moins Aussi bon (GAMAP) (in the meaning of "at least as good as") – often used in France.
- Minimum Endogenous Mortality (MEM) – often used in Germany.

In any case, a system can be classified as "safe", if the system's risk is below a certain threshold. In general, there are two ways to reduce the risk of a system:

- Through the functional interactions between at least two subsystems.
- inherent safety through design (without subsystem-interaction).

Therefore *functional safety* is defined here as:

Definition 23 (Functional safety)
Every emergent property of a system, that reduces the system's risk is an instance of functional safety.

Remark A non-functional way to increase safety in the transportation-system example is to tunnel the train.

3 Conclusion and future work

In this article system-theory has firstly been informally introduced. Essential here is the definition of systems as an *effective* combination, that means systems do not exist a priori but are brainchildren based on purposes of an observer. Secondly, after the intro-

duction in Petri-nets, some of the system-theoretic concepts have been formally defined. For example the system border as the set of stochastic and nondeterministic transitions. After illustrating these correspondences with an example, hazards, risk and functional safety have been expressed on the basis of system-theoretic concepts.

System-theory seems to be a very powerful tool on the one hand to overcome the traditional ”divide and conquer” method and on the other hand to foster defining concepts like ”system border”, ”emergence” or ”functional safety”. Unfortunately, system-theory today lacks having a consistent and formal basis. In this paper we have shown, that Petri-nets may be used to model at least the dynamic aspects of system-theory in a descriptive and formal way.

References

[1] Carnap, R. (1928). *Der logische Aufbau der Welt*. Weltkreis Verlag, Berlin-Schlachtensee (EA).

[2] Carnap, R. (1999). *Der logische Aufbau der Welt*. Weltkreis-Verlag.

[3] European Comission (1999). EN 50126: *Railway applications. The specification and demonstration of reliability, availability, maintainability and safety (RAMS)*. European Comission.

[4] Herz, H. (1894). *Gesammelte Werke Band III: Die Prinzipien der Mechanik in neuem Zusammenhange dargestellt*. H. von Helmholtz, Leipzig,Barth (EA).

[5] International Electrotechnical Commission (1998). *IEC 61508-4, Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 4: Definitions and abbreviations*. International Electrotechnical Commission.

[6] Leveson, N. G. (1995). *Safeware - System Safety and Computers*. Addison-Wesley.

[7] Quine, W. V. O. (1960). *Word and Object. Cambridge*, MIT Press (EA).

[8] Schnieder, Eckehard und Müller, J. (2006). *Petrinets as a Concept for Modeling Transport Automation Systems*. In I. Breitenecker, F.; Troch (Ed.),Proceedings of the 5th Vienna Symposium on Mathematical Modelling- Mathmod.

[9] von Bertalanffy, L. (1968, revised (1976)). *General Systems Theory: Foundations, Development, Applications*. Braziller (George) Inc., U.S.

[10] von Foerster, Heinz und von Glasersfeld, E. (2006). *Wie wir uns erfinden. Eine Autobiographie des radikalen Konstruktivismus*. Carl-Auer-Systeme Verlag, Heidelberg.

[11] von Glasersfeld, E. (2005). *Radikaler Konstruktivismus: Ideen, Ergebnisse, Probleme*. Suhrkamp.

[12] International Electrotechnical Commission (2008). IEC 56/1266/CD:2008, *Analysis techniques for dependability – Petri net modeling*.

Corresponding author: Jörg R. Müller
Eckehard Schnieder Institute for Traffic Safety and Automation Engineering, Braunschweig
Langer Kamp 8, 38106 Braunschweig, Germany
{mueller,schnieder}@iva.ing.tu-bs.de

Received: November 18, 2008
Accepted: November 29, 2008

SNE 18/3-4, December 2008

Event-based Simulations: Enabling Improved Lifecycle and Risk Management of Facilities

Dietmar Wiegand, University of Technology Vienna, dietmar.wiegand@tuwien.ac.at
Priska Mebes and Veronika Pichler, Swiss Federal Institute of Technology Zurich

The flexibility of a property and the ability of facilities management (FM) to react to changing user demands, demographic development, etc. are imperative for sustainable environmental, economic and social urban development. Due to missing methods and tools this "system behaviour" plays a minor role in economic appraisals of the stakeholders in Real Estate Business.

In the research and development project "discreteFD" at the Swiss Federal Institute of Technology Zurich (ETH Zurich) discrete event simulations were developed in cooperation with the Swiss industry partners SUVA and redKG and the German BASF AG, which map "system behaviour" of facilities and facilities management across the life cycle of a property. As a consequence, urban real estate development concepts can be optimized for sustainable use over a single day and the entire life cycle. As a consequence risk management and the optimisation of real estate development concepts advance to a new level.

The project team translated alternative model mappings of the relevant components and behavioural patterns of the property during the life-cycle into Architecture-Spikes (test implementations) in order to immediately examine the effectiveness of the new concepts and life-cycle models for practice and to provide the opportunity to find the optimal software architecture for the problem defined. Subsequently the models were translated into software and verification and validation were carried out to examine the plausibility of the abstraction of a real system.

Empirical tests of the new simulation models and the software tools proved that the complex dynamic system of Real Estate and Facilities Management can be evaluated more precisely via discrete event simulations, thus achieving an enormous optimization potential for Real Estate Development (RED) and Facilities Management.

Introduction

Due to the increasing pace in which both individual lifestyles and the general economic situations of companies change, properties that are inflexible, poorly interpretable, badly managed and/or have high operating costs no longer meet the changing needs of users or tenants. The results of this development can be observed in many cities. Properties are vacant, renovated for alternative use at high costs or demolished for redevelopment at an early stage of their life cycle. As a consequence, neither the targeted return on capital nor a sustainable use is achieved.

Research at the ETH Zürich showed that institutional investors as well as users increasingly develop buildings which are flexible in their use over time. Within in the framework of the research and development project "discreteFD" funded by the Kommission für Technologie und Innovation (KTI) the ETH Zürich developed in cooperation with its partners SUVA, redKG and BASF AG new concepts of real estate development where the layout of the building is developed simultaneously with the respective facilities management concepts. As a result these concepts ensure successful long-term investments. Even though improved risk management and the taping of the market for service-enhanced properties represent the current research objectives in real estate, so far adequate methods and tools to evaluate and estimate the risks of this new type of facilities as well as for the optimisation of building, use, financing and operation concepts are lacking i.e. to prepare facilities for a dynamic and complex future.

The objective of the project "discreteFD" was to understand current and future business models within real estate development, to find adequate abstractions for the application of simulations in lifecycle-oriented real estate development and to develop simulation tools for risk management and the optimisation of concepts.

1 Research and development approach

Research at the ETH Zürich regarding *Discrete Event Simulations* (DES) relates to facilities developments defined as the phase prior to the actual building plan-

ning and to projects where building, use, financing and operation is developed in an integrative way. It is based on the thesis that the adaptability of a building, its interpretation potential i.e. its fitness as well as the types of management within the operating phase and the property-related services are critical for sustainable property value and use. In the preliminary studies the concepts of facilities management are to be optimised regarding possible future events on which developer or operator have no bearing. This seems evident concerning owner-occupied properties (CREM) and public real estate (PREM), but also regarding letting and sales of properties to third parties the performance and adaptability of properties and services over time becomes more and more relevant, the more customers regard those aspects as significant and the legislator requires transparency e.g. in the form of a "building pass".

As "discreteFD" pursues business, simulation-related and information technology-related objectives methods and tools from business analysis and business modelling on the basis of software development were applied. Via the narrative forms of story telling from eXtreme Programming the business processes of facilities development, the requirements of the business field and the expectations of the end-user were collected and defined.

In addition business data of the project partners and their customers were evaluated empirically in order to find influencing factors, which decide success or failure of facilities development (FD). From this a Meta model for FD was deducted representing the basis for test implementations and simulation software. With various test implementations and Architecture-Spikes potential modelling types as well as technical solutions and software architecture were evaluated, yet again based on the methodology of eXtreme Programming. These test implementations perform single features (user stories) in order to answer the simulation questions e.g. the question of the optimal room mix of a hotel (single and double rooms) for an assumed demand.

For the translation of the simulation Meta model, in which the business processes were abstracted into software, transition diagrams and state charts were used; specifications of Unified Modelling Language (UML).

The validation of the simulation Meta model and tests of the software through practical cases represent the last step of the project and are continuing.

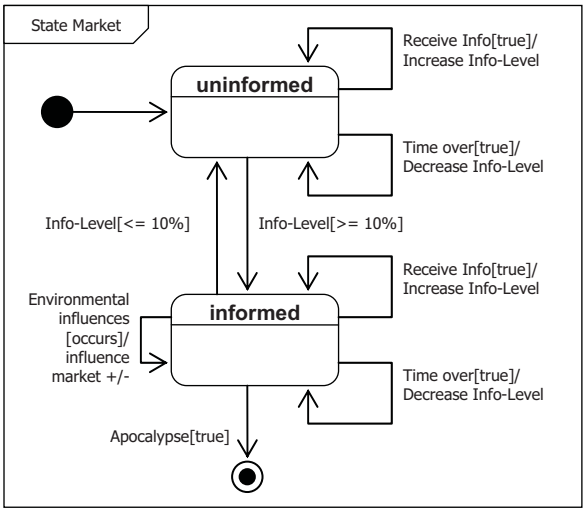


Figure 1. State chart of the marketing of facilities.

2 State of the art simulations and risk management in the construction sector and facilities management

In some areas of civil engineering sporadically dynamic simulations were applied. Chahrour and Franz [3] specifically applied simulations from Logistics and Business Engineering in construction site organisation in order to determine the utilisation of specific equipment and their waiting time. At the 16th International Conference on the Application of Computer Science and Mathematics in Architecture and Civil Engineering in Weimar in 2003 Eichenauer, B. (IBE Simulation Engineering GmbH) and Scherer, K. (Fraunhofer-IMS and inHaus-Zentrum Duisburg) made a presentation on "Modelling and Simulations of Intelligent Building Systems with attributed Petri-Nets." Literature research showed that whilst event simulations are applied in some areas of civil engineering, the application of simulations in real estate development represent a new phenomenon.

For an assessment of the profitability of an investment currently methods of dynamic investment analysis also known as Discounted Cash Flow or Net Present Value method are applied. With the help of spreadsheet analysis as for example Excel the future progression of the net present value as well as of the interest rate is projected. These methods are often referred to as dynamic [10][12][14], as through the inclusion of the interest payment for equity and dept capital the factor time. Strictly speaking these methods represent merely static simulations as they do not include a simulation clock and in the instance of conditions and event no changes in the system status

occur. To our knowledge dynamic event simulations in real estate development and FM are up to now not applied.

As real estate development represents the business with the highest risks involved in the sector, the appropriate handling of risks is of vital importance. In real estate development at the time of investment neither reliable information on the expected costs are available, nor on the prospective return. In addition real estate development is confronted with unpredictable events which are not within the influence radius of the developer. Consequently the higher risk involved, on the one hand generates extremely high returns, but on the other hand the cooperation may get into a financial disorder in the case of a poor development of the project. However, decisions on investments are generally still taken subjectively rather than analytically.

Up to today only two publications on risk management in real estate development are known [14, 17]. Real estate development as an academic discipline was first described in [2] at the European Business School. Isenhöfer expanded this approach in 1999 in his publication on strategic management of real estate development companies [6].

Risk management in the real estate industry in general, but also which risks are typically involved in real estate development, were dealt with by several authors (et al. [1, 4, 5, 9, 10, 13, 16, 18]). The options to manage those risks, though, were not examined or covered.

3 Research results

3.1 Analysis and evaluation of alternative modelling types and simulation tools

The first milestone of the project "discreteFD" was to examine whether the event based simulation is better suitable than other methods e.g. continuous simulations for mapping reality i.e. the lifecycles of facilities from a facilities development perspective. For that reason existent real estate development and properties in operation were analysed and test implementations of the mappings of the lifecycles of respective properties were developed and evaluated utilising various modelling methods and tools.

For the various test implementations different simulation methods and tools were used:

1. Monte Carlo Method (MC)
2. System Dynamics
3. Petri-Nets
4. Discrete Event Simulations (especially eM-Plant)
5. Simulation package SimPy and the programming langugage Python as Excel-AddIn

The Monte Carlo simulation represents one of the most used simulation method and got its name from the world-famous casino. The method is particularly appropriate for the analysis of static problems including known probabilities. The Monte Carlo simulation represents a static simulation and does not cover dynamic situational or status changes.

System Dynamics represents a systems theory, based on the paradigm of information feedback as the behaviour-determining structural component. System Dynamics roots in the findings of cybernetics and applies continuous simulations in order to examine behaviour of non-linear models over time. This theory does not map results.

Petri Nets represent a modelling type which is very neutral in terms of application. It is able to model and analyse dynamic system behaviour. An event based simulation may be applied with a time-related Petri Net model. Various simulation tools are based on Petri Net theory e.g. PACE, Umberto, etc. For practical use higher-level Petri Nets are required for which no constituent notion exists. Higher-level Petri Nets are complex to develop and analyse.

The simulation software eM-Plant represents an integrated simulation system. The advantage of eM-Plant is the complex support of simulation projects. The user concentrates on mapping the relevant system components and not on programming. The efficiency of modelling with eM-Plant is generally higher as with other simulation languages, but the high costs for purchase and operation of the software represent a disadvantage. In addition there is a risk that the results are misinterpreted by inexperienced users. The simulation software partially includes its own programming language which must be acquired for efficient and professional use.

Expert interviews showed that facilities development is always based on assumptions of events for which probabilities concerning their certainty to happen are assumed. The empirical analysis of lifecycles demonstrated that the events or conditions which are considered to be certain to happen generate system behaviour which contributes significantly to success or failure of a project: customer behaviour, behaviour of FM, etc. The modification of properties tends to be volatile. The exact process between events is of minor

interest i.e. discrete event based simulations, which jump over time from event to event and simulate discrete modification of the system status in relation to the event, are highly suitable to map the lifecycle of facilities appropriately.

In the project "discreteFD" not only the question which tools are the most appropriate for the simulation was raised, but also the question how the simulation tool developed is to be applied. The project partners e.g. the future users of the tool decided against a component simulator which would have enabled them to develop individual models, but decided in favour of a proprietary development of a parameter-based model where parameters are changeable.

3.2 Integration of system behaviour in economic appraisal

As a result of the findings outlined distribution probabilities and system behaviour of simulations of lifecycle-related real estate development were included in the development of "discreteFD". Discrete event simulations (DES) were developed to provide a tool for new forms of real estate development and risk management for the industry project partners. The simulations implemented primarily generic features which are applicable within a wide range of projects.

The performance of facilities over the lifecycle i.e. the profitability for investors, operators and users, but also the eco-efficiency depend on a multitude of factors. The project "discreteFD" modelled and simulated those factors in variable and flexible form including the factor time, which were defined as relevant on the basis of data analysis and expert interviews and which have not been mapped before.

The life or durability of properties and building components, which has an important impact on the performance of facilities in connection with time, is not included in the simulation. "discreteFD" simulates on the level of building concepts.

3.3 Conceptual Model for Discrete Event Simulations of Facilities Development

1. The questions which are to be answered by the simulation tools were defined by the project partners as follows:
2. Optimisation of the building layout e.g.: What is the best room mix for a hotel or how many conference rooms are necessary in a building?
3. Management optimisation in development e.g.: What is the best point in time to inform the market, what are the letting terms?
4. Optimisation of rooms and accommodation management of school buildings e.g.: What room mix in connection with which accommodation management is required for the most efficient use of the building during teaching hours?

On the basis of the questions defined a lifecycle of facilities was modelled from the perspective of facilities management and continuously evaluated by the project partners. Thus the system structure was developed and mapped in a conceptual model including the system processes, the individual components as well as their interrelations and impacts.

The conceptual model (see Figure 2) is divided into modules and classes respectively, which communicate via information (blue, bold lines) and cash flows (green, dotted lines). The facilities in this model are presented in four categories and are able to adopt various statuses within those categories which maps reality closely:

- Building LC: the status of a building over the lifecycle of a property (from planning to use)
- Letting/booking status
- Marketing of the facilities
- Use/occupancy level: balance of the actual activities happening in the building

The cash desk (i.e. the profitability analysis) is required in order to value real estate development. Here the relevant key figures such as net present value are evaluated. Through the inclusion of existing valuation models in Excel, the users of the simulation are able to change the calculation modes in Excel.

Changes in the status of one entity class may cause status changes in others. Occupancy may inform lifecycle that the occupancy level has reached a defined threshold and consequently the property is to be used alternatively. In the case of an insufficient occupancy level e.g. the type of use of a building can be changed. If there is no more demand for hotel rooms, part of the building can be converted to offices or apartments. In the case of a high occupancy rate, an extension of the building or an alternative use of other parts of the building can be considered.

The modular structure supports the flexibility of the model. The fact that the lifecycle of a building needed to be implemented only once facilitated the transformation into a flexible model.

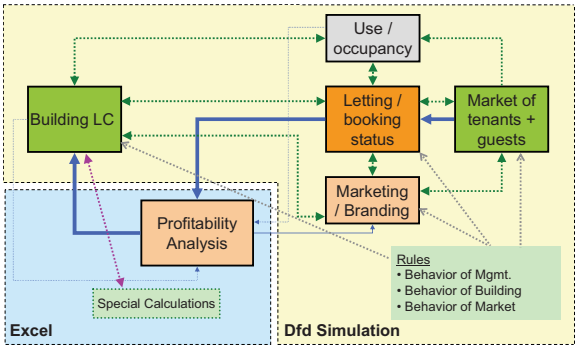


Figure 2. Conceptual facilities development simulation model

3.4 Implementation and testing

In order to answer the questions outlined flexible models were developed from component simulators (eM-Plant) and open source modules (SimPy/Python), which then were validated and tested through existing projects from the hotel, retirement and nursing home and schooling sector. For illustration purposes the empirical findings are demonstrate in the following on the example of school buildings.

The questions determined which were to be answered by the simulation concerning this sector are related to optimal room number, room quality and strategic accommodation management. Firstly rooms were distinguished into rooms for general use and rooms for use for specific subjects such as Biology, Arts, etc. The room demand was assumed as identified i.e. a specific number of teaching hours for a specific number of classes must be catered for with accommodation. The accommodation principles varied as follows:

1. Class room and subject specific class room principle i.e. Physics, Chemistry, Art, Music, Information Technology and Sport is taught in special rooms assigned for these purposes. Classes are exclusively assigned to a specific class room for general teaching.
2. Class room principles apart from the subjects Chemistry and Sports i.e. only Chemistry and Sports are taught in specially equipped class rooms, all other subjects are taught in an exclusively assigned class room.
3. Teachers' room principle i.e. teachers are assigned to specific room either via desk sharing or hoteling. Student/pupils change rooms when teachers vary within the schedule.
4. Specific class room and course room principle: corresponds to option 1. Class room principle is substituted by course room principle.
5. Course room principle for all subjects apart from Chemistry and Sports i.e. for all subjects apart from Chemistry and Sports teaching hours and classes can be assigned freely to rooms

At Bildungszentrum SeeCampus Niederlausitz an event based simulation was conducted for the forms 7 to 13. The results of the simulation show that when the class room and subject specific class room principle is no longer applied which usually represents general practice, 50% of the room capacity could be saved or made available other uses. Considering the fact that school buildings represent the largest segment within the public real estate portfolio, the economic and ecological potential of event based simulations in facilities management becomes evident.

The continuous development of a software simulating lifecycle-oriented facilities management with a significantly increased user-friendliness compared to currently available tools through focussed user guidance while entering relevant data represents the conclusion of the project "discreteFD".

4 Conclusion

Discrete event based simulation tests of flexible facilities in the retirement, educational and hotel sector showed that with the help of simulations optimised building layouts and management concept can be developed which due to the complexity and the dynamic of the business field could not be develop without these tools. The imputed operating costs per teaching unit at the SeeCampus Niederlausitz could

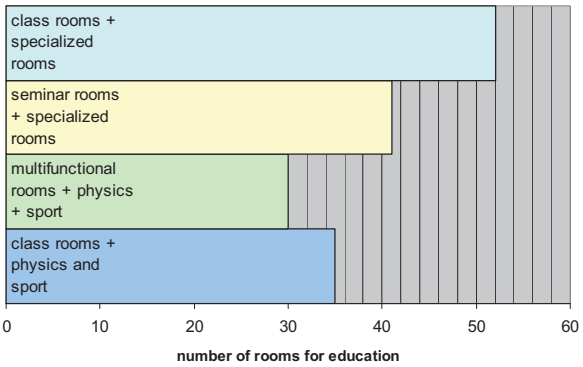


Figure 3. Accomondation management principles and accomondation need/demand illustrated by the example of a German school (Forms 7-13).

be reduced by 40% through the application of a simulation of the lifecycle costs and benefits as well as the resulting optimisation of the building and operation concept.

As outlined the optimisation of facilities development concepts and the evaluation of risks can improved significantly and designed more realistically with the methods and tools developed within the project "discreteFD". Even though the project "discreteFD" represent a generic approach to facilities development not all problems are solved as real estate developments and the questions raised in connection with development projects vary significantly especially in comparison to production and logistics for example. The expenses for modelling, development and implementation can be founded via the resulting increase in efficiency of facilities development projects. For the application of discrete event based simulations on a broader scale i.e. mass production, it is prerequisite though that real estate developer and investors are prepared to increase their investment during the planning phase in view of future benefits. The ETH Zürich plans to transfer and expand the outlined project more and more to questions of eco-efficiency as well as to provide tools supporting the efficient operation of facilities and the decision–making process during the various operating phases respectively.

References

[1] Brauer, K.-U. (2003), Grundlagen der Immobilienwirtschaft Recht - Steuern - Marketing - Finanzierung - Bestandsmanagement - Projektentwicklung. Gabler Verlag, Wiesbaden

[2] Bone-Winkel, S. (1994), Das strategische Management von offenen Immobilienfonds - unter besonderer Berücksichtigung der Projektentwicklung von Gewerbeimmobilien. Verlag Rudolf Müller, Köln

[3] Chahrour, R. und Franz, V. (2004), Computersimulation im Baubetrieb, Forschungsstand und innovative Einsatzmöglichkeiten, In Tagungsband 11. ASIM-Fachtagung Simulation in Produktion und Logistik, „Experiences from the Future", Fraunhofer IRB-Verlag, Stuttgart

[4] Diederichs, C. J. (2006), Immobilienmanagement im Lebenszyklus. Springer Verlag, Berlin

[5] Hellerforth, M. (2001), Der Weg zu erfolgreichen Immobilienprojekten durch Risikobegrenzung und Risikomanagement. RKW-Verlag, Eschborn

[6] Isenhöfer, B. (1999), Strategisches Management von Projektentwicklungsunternehmen. Rudolf Müller Verlag, Köln

[7] Kosturiak J. und Gregor M. (1995) Simulation von Produktionssystemen, Springer-Verlag Wien

[8] Law A. M. und Kelton W. D. (2000), Simulation Modelling and Analysis, 3. Auflage, McGraw-Hill, Boston

[9] Maier, K. (2004), Risikomanagement im Immobilien- und Finanzwesen ein Leitfaden für Theorie und Praxis. Verlag Fritz Knapp, Frankfurt am Main

[10] Ropeter, S.-E. (1998), Investitionsanalyse für Gewerbeimmobilien, Müller, Köln

[11] Sauerbier T. (1999), Theorie und Praxis von Simulationssystemen, Vieweg Verlag

[12] Schulte, K.-W. und Allendorf, G. J. (2005), Betriebswirtschaftliche Grundlagen, Oldenbourg Verlag, München

[13] Schulte, K.-W. (2005), Handbuch Immobilien-Investition. 2. Auflage, Müller, Köln

[14] Schelkle, H. P. (2005), Phasenorientierte Wirtschaftlichkeitsanalyse für die Projektentwicklung von Büroimmobilien, Dissertation, Institut für Baubetriebslehre der Universität Stuttgart

[15] Siegert H. J. (1991), Simulation zeitdiskreter Systeme, Oldenbourg Verlag

[16] Vogler, J. (1998), Handbuch Immobilien-Investition. Müller, Köln

[17] Wiedenmann, M. (2005), Risikomanagement bei der Immobilien-Projektentwicklung, Dissertation, Institut für Standentwicklung und Bauwirtschaft der Universität Leipzig

[18] Wüstefeld, H. (2000), Risiko und Rendite von Immobilieninvestments. Verlag Fritz Knapp, Frankfurt am Main

Corresponding author: Dietmar Wiegand,
Inst. für Städtebau, Landschaftsarchitektur
Vienna University of Technology
Gußhausstraße 30, 1040 Vienna, Austria
dietmar.wiegand@tuwien.ac.at

Received: August 30, 2008
Revised: September 11, 2008
Accepted: September 18, 2008

TN

Extended and Structural Features of Simulators – A Comparative Study

Nikolas Popper, DWH Simulation Services; Vienna, *niki.popper@drahtwarenhandlung.at*
Felix Breitenecker, Vienna University of Technology, *felix.breitenecker@tuwien.ac.at*

The paper first follows roots of the CSSL standard for simulation languages, from simple ODE modelling structures to discrete elements and state events in ODE modelling, from explicit state space to implicit state space and DAE modelling. Thereby *Extended Features* – e.g. sorting, DAE solving, event handling - are discussed and compared in classical simulators like MATLAB, ACSL and Dymola. In the following, the new developments in simulation systems are presented: object-oriented modelling, a-causal physical modelling, the new Modelica standard for ODE and DAE modelling, state chart and structural dynamic systems. Here, *Structural Features* – e.g. textual / graphical physical modelling, textual / graphical state chart modelling, Modelica compatibility, dynamic state space – are outlined and documented in examples. The last section reviews state-of-the-art simulators for availability of these *Extended Features* and *Structural Features* and summarises results in a comparing table

1 CSSL structure in continuous simulation

Simulation supported various developments in engineering and other areas, and simulation groups and societies were founded. One main effort of such groups was to standardise digital simulation programs and to work with a new basis: not any longer simulating the analog computer, but a self-standing structure for simulation systems. There were some unsuccessful attempts, but in 1968, the CSSL Standard became the milestone in the development: it unified the concepts and language structures of the available simulation programs, it defined a structure for the model, and it describes minimal features for a runtime environment. The CSSL standard suggests structures and features for a model frame and for an experimental frame. This distinction is based on Zeigler's concept of a strict separation of these two frames. Model frame and experimental frame are the user interfaces for the heart of the simulation system, for the simulator kernel or simulation engine. A translator maps the model description of the model frame into state space notation, which is used by the simulation engine solving the system governing ODEs. This basic structure of a simulator is illustrated in Figure 1, together with an extended structure for service of discrete elements.

Mathematical basis is for the simulation engine is the state space description

$$\dot{\vec{x}}(t) = \vec{f}(\vec{x}(t), \vec{u}(t), t, \vec{p}), \quad \vec{x}(t_0) = \vec{x}_0,$$

which is used by the ODE solvers of the simulation engine. Any kind of textual model formulation, of graphical blocks or structured mathematical description or host languages constructs must be transformed to an internal state equation of the structure given above, so that the vector of derivatives $\vec{f}(\vec{x}, \vec{u}, t, \vec{p})$ can be calculated for a certain time instant $\vec{f}_i = \vec{f}_i(\vec{x}(t_i), \vec{u}(t_i), t_i, \vec{p})$. This vector of derivates is fed into an ODE solver in order to calculate a state update $\vec{x}_{i+1} = \Phi(\vec{x}_i . \vec{f}_i, h)$, h stepsize (all controlled by the simulation engine). Essential is CSSL's concept of `SECTION`s or `REGION`s, giving a certain structure to the model description. The dynamic model description builds up the `DYNAMIC` or `DERIVATIVE` section of the model description.

As example, we consider the model description for a pendulum. The well-known equations (length *l*, mass *m*, and damping coefficient *d*) and initial values, parameter and static relations and dependencies are given by

$$\ddot{\varphi}(t) = -\frac{g}{l}\sin\varphi - \frac{d}{m}\dot{\varphi}, \quad \varphi_0 = \frac{\pi}{n}, \ \dot{\varphi}_0 = 0,$$
$$d = \sqrt{2D}, \ a = \frac{g}{l}, b = \frac{d}{m};$$

A structured model description, e.g. in ACSL (Listing 1), generates more efficient code: only the `DERIVATIVE` section is translated into the derivative vector function, while `INITIAL` and `TERMINAL` section are translated into functions called evaluated separately only once. It is task of the translator, to recognise the static elements, and to sort them separately from the dynamic equations, so that for the simulation engine dynamic equations (`derivative`), initial and parameter equations (`initial`), and terminal equations (terminal), are provided in separate modules.

With graphical window systems, graphical model descriptions became important. Consequently, simulation systems offered this kind of model description.

```
PROGRAM math_pendulum
! structured CSSL model
! model parameters
  CONSTANT m=1, l=1, d=0.3 ! kg, m, N*s/m
  CONSTANT g=9.81, pi=3.141592653; dphi0=0
  CONSTANT pintel=2
INITIAL ! calculation with parameters
  phi0 = pi/pintel; a = g/l; b = d/m
END ! of INITIAL
DERIVATIVE ! ODE model
  phi = integ ( dphi,        phi0)
  dphi = integ (-gdl*sin(phi)-ddm*dphi, dphi0)
END ! of DERIVATIVE
END ! of Program
```

Listing 1. ACSL Structured Textual Model Description.

However, in a graphical modelling system one disadvantage appears: the graphical structure consisting of directed dynamic signal flow allows almost no structure for dynamic calculations and static calculations. Calculation of static parameter equations are modelled by dynamic blocks – consequently evaluated at each evaluation call of the ODE solver.

2 Implicit models – differential-algebraic equations

For a long time the explicit state space description played the dominant role; additional constraints and implicit models had to be transformed 'manually'. From the 1990s on, the simulators started to take care on these very natural phenomena of implicit structures (implicit state space, DAE models):

$$F(\dot{\vec{y}}(t), \vec{y}(t), \vec{u}(t), t, \vec{p}) = \vec{0} \quad \vec{y}(t_0) = \vec{y}_0$$

The so-called extended state vector $\vec{y}(t)$ can be splitted into the differential state vectors $\vec{x}(t)$ and into the algebraic state vector $\vec{z}(t)$:

$$\dot{\vec{x}}(t) = \vec{f}(\vec{x}(t), \vec{z}(t), \vec{u}(t), t, \vec{p}) = 0, \quad x(t_0) = \vec{x}_0,$$
$$g(\vec{x}(t), \vec{z}(t), \vec{u}(t), t, \vec{p}) = 0$$

The above given DAEs can be solved by extended ODE solvers and by implicit DAE solvers. Three different approaches may be used:

1. *Nested Approach*, using classical ODE solver
   a. given x_n, solving first numerically $g(x_n, z_n) = 0 \Rightarrow z_n = z_n(x_n) = \hat{g}^{-1}(x_n)$, e. g. by modified Newton iteration, and
   b. applying ODE method, evolving $x_{n+1} = \Phi_E(x_n, z_n(x_n), t_n)$.
2. *Simultaneous Approach*, using an implicit DAE solver; given x_n, solving $g(x_{n+1}, z_{n+1}) = 0$ and $\Phi_I(x_{n+1}, x_n, z_{n+1}, t_{n+1}) = 0$ simultaneously.

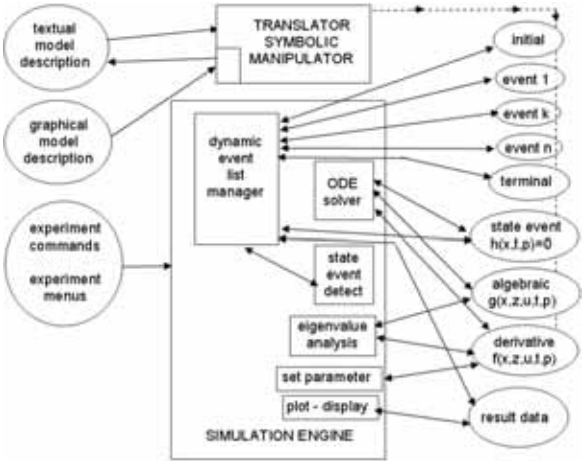


Figure 1. Extended Structure of a Simulation System due to Extensions of the CSSL Standard with Discrete Elements and with DAE Models

3. *Symbolic Approach*, determining in advance the explicit form solving $g(x,z) = 0 \Rightarrow z = z(x) = \hat{g}^{-1}(x)$ by symbolic computations e.g. within the model translator, and using classical ODE solvers.

The *Symbolic Approach* requires a symbolic inversion of the algebraic equations, which in many cases is not possible or not adequate. The *Nested Approach* – up to now most commonly used – requires a numerical inversion of the algebraic equations in each ODE - function evaluation. This approach can be very expensive and time-consuming, although classical ODE solvers can be used. The *Simultaneous Approach* requires an implicit ODE solver – usually an implicit stiff equation solver. Although also working with iterations, these solvers show much more efficiency and flexibility (DASSL, IDA-DASSL, LSODE, etc).

However, hidden is another problem: the 'DAE index' problem. Roughly speaking, a DAE model is of index *n*, if n differentiations of the DAE result in an ODE system (with an increased state space). The implicit ODE solvers for the *Simultaneous Approach* guarantee convergence only in case of DAE index $n = 1$. Models with higher DAE index must / should be transformed to models with DAE index $n = 1$. This transformation is based on not-unique symbolic differentiation / manipulation of the high index DAE system. Unfortunately, in case of mechanical systems modelling and in case of process technology modelling indeed DAE models with DAE index $n = 3$ may occur, so that index reduction may be necessary. Index reduction is a new challenge for the translator of simulators, and still point of discussion.

3 Discrete elements in continuous simulation

The CSSL standard also defines segments for discrete actions, first mainly used for modelling discrete control. So-called `DISCRETE` regions or sections manage the communication between discrete and continuous world and compute the discrete model parts. For incorporating discrete actions, the simulation engine must interrupt the ODE solver and handle the event. For generality, efficient implementations set up and handle event lists, representing the time instants of discrete actions and the calculations associated with the action, where in-between consecutive discrete actions the ODE solver is to be called.

In order to incorporate DAEs and discrete elements, the simulator's translator must now extract from the model description the dynamic differential equations (`derivative`), the dynamic algebraic equations (`algebraic`), and the events (`event i`) with static algebraic equations and event time, as given in Figure 1 (extended structure of a simulation language due to CSSL standard). In principle, initial and parameter equations (`initial`, `terminal`) are special cases of events at time $t = 0$ and terminal time.

Much more complicated are the so-called state events. Here, a discrete action takes place at a time instant, which is not known in advance, it is only known as a function of the states. As example, we consider the pendulum with constraints (*Constrained Pendulum*). If the pendulum is swinging, it may hit a pin positioned at angle φ_p with distance l_p from the point of suspension. In this case, the pendulum swings on with the position of the pin as the point of rotation. The shortened length is $l_s = l - l_p$ and the angular velocity $\dot{\varphi}$ is changed at position φ_p from $\dot{\varphi}$ to $\dot{\varphi} l / l_s$, etc. These discontinuous changes are state events, not known in advance.

For such state events, the classical state space description is extended by the so-called state event function $h(\vec{x}(t), \vec{u}(t), \vec{p})$, the zero of which determines the event:

$$\dot{\vec{x}}(t) = \vec{f}(\vec{x}(t), \vec{u}(t), \vec{p}, t), \quad h(\vec{x}(t), \vec{u}(t), \vec{p}, t) = 0$$

In this notation, the model for *Constrained Pendulum* is given by

$$\dot{\varphi}_1 = \varphi_2, \ \dot{\varphi}_2 = -\frac{g}{l}\sin\varphi_1 - \frac{d}{m}\varphi_2, \quad h(\varphi_1, \varphi_2) = \varphi_1 - \varphi_p = 0$$

The example involves two different events: change of length parameter (SE-P), and change of state (SE-S), i.e. angle velocity). Generally, state events (SE) can be classified in four types:

1. Parameters change discontinuously (SE-P),
2. Inputs change discontinuously (SE-I),
3. States change discontinuously (SE-S), and
4. State vector dimension changes (SE-D), including total change of model equations.

State events type 1 (SE-P) could also be formulated by means of `IF-THEN-ELSE` constructs and by switches in graphical model descriptions, without synchronisation with the ODE solver. The necessity of a state event formulation depends on the accuracy wanted. Big changes in parameters may cause problems for ODE solvers with stepsize control. State events of type 3 (SE-S) are essential state events. They must be located, transformed into a time event, and modelled in discrete model parts. State events of type 4 (SE-D) are also essential ones. In principle, they are associated with hybrid modelling: models following each other in consecutive order build up a sequence of dynamic processes. And consequently, the structure of the model itself is dynamic; these so-called structural dynamic systems are at present (2008) discussion of extensions to Modelica, see next chapters.

The handling of a state event requires four steps:

1. Detection of the event, usually by checking the change of the sign of *h(x)* within the solver step over $[t_i, t_{i+1}]$
2. Localisation of the event by a proper algorithm determining the time t^* when the event occurs and performing the last solver step over $[t_i, t^*]$
3. Service of the event: calculating/setting new parameters, inputs and states; switching to new equations
4. Restart of the ODE solver at time t^* with solver step over $[\, t^* = t_{i+1}, t_{i+2}]$

Figure 1 also shows the necessary extensions for incorporating state events. The simulator's translator must extract from the model description additionally the state event functions (`state event j`) with the associated event action – only one state event shown in the figure). In the simulator kernel, the static event management must be made dynamically: state events are dynamically handled and transformed to time events. In principle, the kernel of the simulation engine has become an event handler, managing a complex event list with feedbacks. It is to be noted, that different state events may influence each other, if they are close in time – in worst case, the event finders run in a deadlock.

```
PROGRAM constrained pendulum
CONSTANT m = 1.02, g = 9.81, d =0.2
CONSTANT lf=1, lp=0.7
DERIVATIVE dynamics
  ddphi = -g*sin(phi)/l - d*dphi/m
  dphi  = integ ( ddphi, dphi0)
  phi   = integ ( dphi, phi0)
  SCHEDULE hit   .XN. (phi-phip)
  SCHEDULE leave .XP. (phi-phip)
END ! of dynamics
DISCRETE hit
  l = ls; dphi = dphi*lf/ls
END ! of hit
DISCRETE leave
  l = lf; dphi = dphi*ls/lf
END ! of leave
END ! of constrained pendulum
```

Listing 2. *Constrained Pendulum*: Continuous Model with State Events (ACSL).

The *Constrained Pendulum* example involves a state event of type 1 (SE-P) and type 3 (SE-S). A classical ACSL model description works with two discrete sections `hit` and `leave`, representing the two different modes, both called from the dynamic equations in the derivative section (Listing 2).

In graphical model descriptions, we again are faced with the problem of calculations at discrete time instants. For the detection of the event, SIMULINK provides the `HIT CROSSING` block (in new Simulink version implicitly defined). This block starts state event detection (interpolation method) depending on the input, the state event function, and outputs a trigger signal, which may call a triggered subsystem servicing the event.

4 Extended features of simulators

Model sorting (MS) and time events /sampled data systems) are basic features of a simulator. Event description (ED), state event handling (SEH) and DAE support (DAE) with or without index reduction (IR) became desirable structural features of simulators, supported directly or indirectly.

Table 1 compares the availability of these features in MATLAB/Simulink System, in ACSL and in Dymola.

In this table, the availability of features is indicated by 'yes' and 'no'; a 'yes' in parenthesis '(yes)' means, that the feature is complex to use. MS - 'Model Sorting', is a standard feature of a simulator – but missing in MATLAB (in principle, MATLAB cannot be called a simulator). On the other hand, MATLAB's ODE solvers offer limited features for DAEs (systems with mass matrix) and an integration stop on event condition, so that SHE and DAE get a ('yes').

	MS - Model Sorting	ED -Event Description	SEH - State Event Handling	DAE - DAE Solver	IR - Index Reduction
MATLAB	no	no	(yes)	(yes)	no
Simulink	yes	(yes)	(yes)	(yes)	no
MATLAB/Simulink	yes	yes	yes	(yes)	no
ACSL	yes	yes	yes	yes	no
Dymola	yes	yes	yes	yes	yes

Table 1. Comparison of Simulators' Extended Features (Event Handling and DAE Modelling)

In Simulink, event descriptions are possible by means of triggered subsystems, so that ED gets a '(yes)' because of complexity. A combination of MATLAB and Simulink suggest putting the event description and handling at MATLAB level, so that ED and SEH get both a 'yes'. DAE solving is based on modified ODE solvers, using the nested approach (see before), so ED gets only a '(yes)' for MATLAB - Simulink.

ACSL is a classical simulator with sophisticated state event handling, and since version 10 (2001) DAEs can be modelled directly by the residuum construct, and they are solved by the DASSL algorithm (simultaneous approach), or by modified ODE solvers (nested approach) – so 'yes' for ED, SEH, and DAE. For DAE index $n = 1$, the DASSL algorithm guarantees convergence. ACSL does not perform index reduction (IR 'no').

Dymola is a modern simulator, implemented in C, and based on physical modelling. Model description may be given by implicit laws, symbolic manipulations extract a proper ODE or DAE state space system, with index reduction for high index DAE systems – all extended features are available. Dymola started a new area in modelling and simulation of continuous and hybrid systems.

5 From CSSL to Modelica and VHDL-AMS

In the 1990s, many attempts have been made to improve and to extend the CSSL structure, especially for the task of mathematical modelling. The basic problem was the state space description, which limited the construction of modular and flexible modelling libraries. Two developments helped to overcome this problem. On modelling level, the idea of physical modelling gave new input, and on implementation level, the object-oriented view helped to leave the constraints of input/output relations.

In physical modelling, a typical procedure for modelling is to cut a system into subsystems and to account for the behaviour at the interfaces. Balances of mass, energy and momentum and material equations model each subsystem. The complete model is obtained by combining the descriptions of the subsystems and the interfaces. This approach requires a modelling paradigm different to classical input/output modelling. A model is considered as a constraint between system variables, which leads naturally to DAE descriptions. The approach is very convenient for building reusable model libraries.

In September 1996, an international effort for common modelling language was initiated, bringing together expertise in object-oriented physical modelling (port based modelling) and defining a modern uniform modelling language – mainly driven by the developers of Dymola.

The new modelling language is called *Modelica*. Modelica is intended for modelling within many application domains such as electrical circuits, multibody systems, drive trains, hydraulics, thermodynamical systems, and chemical processes etc. It supports several modelling formalisms: ordinary differential equations, differential-algebraic equations, bond graphs, finite state automata, and Petri nets etc. Modelica is intended to serve as a standard format so that models arising in different domains can be exchanged between tools and users.

Modelica is a not a simulator, Modelica is a modelling language, supporting and generating mathematical models in physical domains. When the development of Modelica started, also a competitive development, the extension of VHDL towards VHDL-AMS was initiated.

Both modelling languages aimed for general-purpose use, but VHDL-AMS mainly addresses circuit design, and Modelica covers the broader area of physical modelling; modelling constructs such as Petri nets and finite automata could broaden the application area, as soon as suitable simulators can read the model definitions. Modelica offers a textual and graphical modelling concept, where the connections of physical blocks are bidirectional physical couplings, and not directed flow. The translator from Modelica into the target simulator must not only be able to sort equations, it must be able to process the implicit equations symbolically and to perform DAE index reduction (or a way around).

```
equation /*pendulum*/
  v = length*der(phi);
  vdot = der(v);
  mass*vdot/length + mass*g*sin(phi)
  +damping*v = 0;
algorithm
 if (phi<=phipin) then length:=ls; end if;
 if (phi>phipin) then length:=l1; end if;
```

Listing 3. Modelica Model for *Constrained Pendulum*

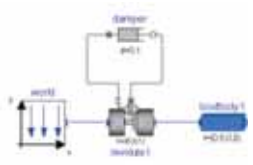
Up to now – similar to VHDL-AMS – some simulation systems understand Modelica (2008; generic – new simulator with Modelica modelling, extension - Modelica modelling interface for existing simulator):

- Dymola from Dynasim (generic),
- MathModelica - MathCore Engineering (generic)
- SimulationX from ISI (generic/extension)
- Scilab/Scicos (extension)
- MapleSim (extension, announced)
- Open Modelica - since 2004 the University of Lyngby develops an provides an open Modelica simulation environment (generic),
- Mosilab - Fraunhofer Gesellschaft develops a generic Modelica simulator, which supports dynamic variable structures (generic)
- Dymola / Modelica blocks in Simulink

As Modelica also incorporates graphical model elements, the user may choose between textual modelling, graphical modelling, and modelling using elements from an application library. Furthermore, graphical and textual modelling may be mixed in various kinds. The minimal modelling environment is a text editor; a comfortable modelling environment offers a graphical modelling editor.

The *Constrained Pendulum* example can be formulated in Modelica textually as a physical law for angular acceleration. The event with parameter change is put into an `algorithm` section, defining and scheduling the parameter event SE-P (Listing 3). Instead of angular velocity, tangential velocity is used as state variable, the second state event SE-S 'vanishes'.

Modelica allows combining textual and graphical modelling. For the *Constrained Pendulum* example, the basic physical dynamics could be modelled graphically with joint and mass elements, and the event of length change is described in an `algorithm` section, with variables interfacing to the predefined variables in the graphical model part (Figure 2).



```
algorithm
if (revolute1.phi
     <= phipin) then
    revolute1.length:=ls;
end if;
if (revolute1.phi
     < phipin) then
    revolute1.length:=ll;
end if;
```

Figure 2: Mixed Graphical and Textual Dymola Model for *Constrained Pendulum*

6 Modelling with statecharts

In the end of the 1990s, computer science initiated a new development for modelling discontinuous changes. The *Unified Modelling Language* (UML) is one of the most important standards for specification and design of object oriented systems. This standard was tuned for real time applications in the form of a new proposal, *UML Real-Time* (UML-RT).

In 1999, a simulation research group at the Technical University of St. Petersburg used this approach in combination with a hybrid state machine for the development of a hybrid simulator (*MVS*), from 2000 on available commercially as simulator *AnyLogic*. The modelling language of AnyLogic is an extension of UML-RT; the main building block is the *Active Object*. Active objects have internal structure and behaviour, and allow encapsulating of other objects to any desired depth. Relationships between active objects set up the hybrid model.

In an AnyLogic implementation for the *Constrained Pendulum* a primary active object *(Constrained Pendulum*) 'holds' the equations for the pendulum, together with a state chart (*main*) switching between short and long pendulum. The state chart nodes are empty; the arcs define the events (Figure 3). Internally, AnyLogic restarts at each hit the same pendulum model (trivial hybrid decomposition).

7 Hybrid and structural-dynamic systems

Continuous simulation and discrete simulation have different roots, but they are using the same method, the analysis in the time domain. During the last decades a broad variety of model frames (model descriptions) has been developed. In continuous and hybrid simulation, the explicit or implicit state space description is used as common denominator. This state space may be described textually, or by signal-oriented graphic blocks (e.g. SIMULINK), or by physically based block descriptions (Modelica, VHDL-AMS).

Hybrid systems often come together with a change of

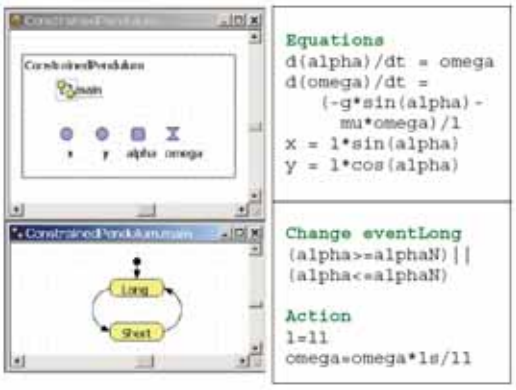


Figure 3. AnyLogic model for *Constrained Pendulum*, Simple Implementation

the dimension of the state space, then called *structural-dynamic systems*. The dynamic change of the state space is caused by a state event of type SE-D. In contrary to state events SE-P and SE-S, states and derivatives may change continuously and differentiable in case of structural change.

Maximal State Space for Structural-Dynamic Systems. Most implementations of physically based model descriptions support a big monolithic model description, derived from laws, ODEs, DAEs, state event functions and internal events. The state space is maximal and static, index reduction in combination with constraints keep a consistent state space. For instance, Dymola, OpenModelica, and VHDL-AMS follow this approach. This approach can be classified with respect to event implementation. The approach handles all events of any kind (SE-P, SE-S, and SE-D) within the ODE solver frame, also events which change the state space dimension (change of degree of freedoms) – consequently called *internal events*.

Using the classical state chart notation, *internal state events I-SE* caused by the model schedule the model itself, with usually different re-initialisations (depending on the event type I-SE-P, I-SES, I-SE-D; Figure 4). For instance, VHDL-AMS and Dymola follow this approach, handling also DAE models with index higher than 1; discrete model parts are only supported at event level. ACSL and MATLAB / Simulink generate also a maximal state space.

Hybrid Decomposition for Structural-Dynamic Systems. The hybrid decomposition approach makes use of *external events* (E-SE), which control the sequence and the serial coupling of one model or of more models. A convenient tool for switching between models is a state chart, driven by the *exter-*

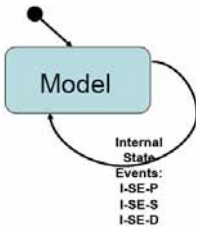


Figure 4. State Chart Control for *Internal Events* of one Model

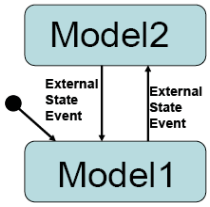


Figure 5. *State* Chart Control for *External Events* for two Models

nal events – which itself are generated by the models. Following e.g. the UML-RT notation, control for continuous models and for discrete actions can by modelled by state charts. Figure 5 shows the hybrid coupling of two models, which may be extended to an arbitrary number of models, with possible events E-SE-P, E-SE-S, and ESE-D. As special case, this technique may be also used for serial conditional 'execution' of one model – Figure 11 (only for SE-P and SE-S). This approach additionally allows not only dynamically changing state spaces, but also different model types, like ODEs, linear ODEs (to be analysed by linear theory), PDEs, etc. to be processed in serial or also in parallel (co-simulation).

The approach allows handling all events also outside the ODE solver frame. After an event, a very new model can be started. This procedure may make sense especially in case of events of type SE-D and SE-S. As consequence, consecutive models of different state spaces may be used.

The structure of a simulator supporting structural dynamic modelling and simulation must allow the use of independent models. Main extension of the simulator structure (Figure 1) is a translation of several DAE models from the independent models, and of external events from the connection model, controlling the model execution sequence in the highest level of the dynamic event list.

Both approaches have advantages and disadvantages. The classical Dymola approach generates a fast simulation, because of the monolithic program. However, the state space is static. Furthermore, Modelica centres on physical modelling. A hybrid approach handles separate model parts and must control the external events. A challenge for the future lies in the combination of both approaches. The main ideas are:

- Moderate hybrid decomposition
- External and internal events
- Efficient implementation of models and control

8 Structural features of simulators

While the *extended features* discussed before address the CSSL-standard, *structural features* characterise features for physical modelling and for structural dynamic systems. This section investigates the availability or of structural features in some simulators, and summarises the results in Table 10. The features may be classified as follows:

- Support of a-causal physical modelling (sometimes called port-based modelling) at textual (PM-T) or graphical level (PM-G),
- Modelica standard (MOD) for a-causal physical modelling ,
- Decomposition of structural dynamic systems with dynamic features (SD) – features for external events, and
- Support of state chart modelling or a of a similar construct, by means of textual (SC-T) or graphical (SC-G) constructs.

In principle, each combination of the above features is possible. By means of the maximal state space approach, each classic simulator can handle structural dynamic systems, but a-causal modelling may be supported or not, and state chart modelling may be available or not. Simulators with a-causal modelling may support hybrid decomposition or not, and state chart modelling may be available or not. Simulators with features for state chart modelling may support hybrid decomposition or not, and a-causal modelling may be offered or not. In general, interpreter-oriented simulators offer more structural flexibility, but modern software structures would allow also flexibility with precompiled models or with models compiled 'on the fly'.

In addition, of interest are also structural features as

- simulation-driven visualisation (with visualisation objects defined with the model objects; VIS),
- frequency domain analysis and linearization for steady state analysis (FA), and
- extended environment for complex experiments and data pre- and postprocessing (ENV).

8.1 MATLAB/Simulink/Stateflow/SimScape

The mainly interpretative systems MATLAB / Simulink offer different approaches. First, MATLAB itself allows any kind of static and dynamic decomposition (SD 'yes'), but MATLAB is not a simulator, because the model equations have to be provided in a sorted manner, to be called from an ODE solver (MS 'no').

Second, MATLAB allows hybrid decomposition at MATLAB level with Simulink models. There, from MATLAB different Simulink models are called conditionally, and in Simulink, a state event is determined by the hit-crossing block (terminating the simulation). Simulink is MATLAB's simulation module for block-oriented dynamic models (directed signal graphs), which can be combined with Stateflow, MATLAB's module for event-driven state changes described by state charts (SC-T and SC-G 'yes').

At Simulink level, Stateflow, Simulink's state chart modelling tool, may control different submodels. But Simulink can only work with a maximal state space and does not allow hybrid decomposition (SD 'no). Neither basic MATLAB nor basic Simulink support a-causal modelling. First MATLAB/Simulink modules for physical modelling (e.g. *Hydraulic Blockset* and others, 2004 - 2008) were precompiled to a classical state space (PM-T and PM-G 'no'), Modelica modelling is not supported (MOD 'no').

For DAEs, MATLAB and Simulink offer modified LSODE solvers (implicit solvers) for the nested DAE solving approach. In MATLAB any kind of simulation – driven visualisation can be programmed and used in MATLAB or Simulink or in both, but not based on the model definition blocks (VIS '(yes)'). From the beginning on, MATLAB and Simulink offered frequency analysis (FA 'yes'), and clearly, MATLAB is a very powerful environment for Simulink, Stateflow, for all other Toolboxes, and for MATLAB itself (ENV 'yes').

In 2008, Mathworks introduced a basic physical modelling language, Simscape, with a basic graphic component library – based on Simscape language. This modelling language is similar to Modelica, and it is not clear, whether the developers at Mathworks will keep Simscape compatible to Modelica, or whether they will set a competing standard. The Simscape translator is also capaple of index reduction (IR 'yes'). So combinations of MATLAB / Simulink / Stateflow with Simscape would offer any modelling possibilities (PM-T and PM-G 'yes', SD 'yes', etc.).

8.2 ACSL

ACSL – Advanced Continuous Simulation Language – has been developed since more than 25 years. ACSL' software structure is a direct mapping of the CSSL structure in Figure 1. The new developers (Aegis Technologies) concentrate on application-oriented simulation solutions, with models are tailor-made for the specific application. Last extensions were a change to C as basic language (instead of FORTRAN), and DAE features using the nested approach with classical solvers, or direct implicit DAE solving with DASSL Code (DAE 'yes', IR 'no'). From the beginning on, steady state calculation, linearization and frequency analysis was a standard feature of ACSL's simulator kernel (FA 'yes'). Since 2000, the environment has been enriched by modules for modelling and environment modules. The first module was a graphical modeller, which seems to make use of physical modelling, but in behind a classical state space is used – PM-T and PM-G 'no'. Furthermore, a simulation-driven visualisation system (third party) is offered (but hard to use) – VIS '(yes)'.

ACSLMath was intended to have same features as MATLAB; available is only a subset, but powerful enough for an extended environment (ENV 'yes'), which can be used for hybrid decomposition of a structural dynamic model in almost the same way than MATLAB does (SD 'yes'). Unfortunately the development of ACSLMath has been stopped. In general, there is no intention to make a-causal physical modelling available, also Modelica is not found in the developers' plans (PM-T, PM-G, and MOD 'no').

8.3 Dymola

Dymola, introduced by F. E. Cellier as a-causal modelling language, and developed to a simulator by H. Elmquist, can be called the mother of Modelica. Dymola is based on a-causal physical modelling and initiated Modelica; consequently, it fully supports Modelica these structural features (PM-T, PM-G, and MOD 'yes'). Together with the model objects, also graphical objects may be defined, so that simulation based pseudo-3D visualisation is available (VIS 'yes'). A key feature of Dymola is the very sophisticated index reduction by the modified Pantelides algorithm, so Dymola handles any DAE system, also with higher index, with bravura (DAE and IR 'yes'). For DAE solving, modified DASSL algorithms are used. In software structure, Dymola is similar to ACSL, using an extended CSSL structure as given in Figure 1 – with the modification that all discrete actions are put into one event module, where CASE - constructs distinguish between the different events (this structure is based on the first simulator engine Dymola used, the DS-Block System of DLR Oberpfaffenhofen).

Dymola comes with a graphical modelling and basic simulation environment, and provides a simple script language as extended environment; new releases offer

also optimisation, as built-in function of the simulator. Furthermore, based on Modelica's matrix functions some task of an environment can be performed – so ENV ('yes') – available, but complex/uncomfortable. Dymola offers also a Modelica – compatible state chart library, which allows to model complex conditions (internally translated into `IF-THEN-ELSE` or `WHEN` constructs - SC-T and SC-G '(yes)').

Up to now (2008) the Modelica definition says nothing about structural dynamic systems, and Dymola builds up a maximal state space. And up to now, Modelica does not directly define state charts, and in Dymola a state chart library in basic Modelica notation is available, but working only with internal events within the maximal state space (SD 'no'). For Modelica extension, a working group on hybrid systems has been implemented, in order to discuss and standardise hybrid constructs like state charts, and hybrid decompositions (independent submodels).

8.4 MathModelica

MathModelica, developed by MathCoreAB, was the second simulation system, which understood Modelica modelling. MathModelica is an integrated interactive development, from modelling via simulation to analysis and code integration. As the MathModelica translator is very similar to Dymola's model translator, clearly all related features are available, including index reduction and use of implicit solvers like DASSL (all DAE, IR, PM-T, PM-G and MOD 'yes').

MathModelica follows a software model different to CSSL standard. The user interface consists of a graphical model editor and notebooks. There, a *simulation center* controls and documents experiments in the time domain. Documentation, mathematical type setting, and symbolic formula manipulation are provided via Mathematica, as well as Mathematica acts as extended environment for MathModelica (ENV 'yes') – performing any kind of analysis and visualisation (FA and VIS 'yes'). By means of the Mathematica environment, also a hybrid decomposition of structural dynamic systems is possible, with the same technique like in MATLAB (SD – 'yes').

8.5 Mosilab

Since 2004, Fraunhofer Gesellschaft Dresden develops a generic simulator *Mosilab*, which also initiates an extension to Modelica: multiple models controlled by state automata, coupled in serial and in parallel. Furthermore, Mosilab puts emphasis on co-simulation and simulator coupling, whereby for interfacing the same constructs are used than for hybrid decomposition. Mosilab is a generic Modelica simulator, so all basic features are met (ED, SEH, DAE, PM-T, and PM-G 'yes', and MOD '(yes)' – because of subset implementation at present, 2008). For DAE solving, variants of IDA-DASSL solver are used.

Mosilab implements extended state chart modelling, which may be translated directly due to Modelica standard into equivalent IF – THEN constructs, or which can control different models and model executions (SC-T, SC-G, and SD 'yes'). At state chart level, state events of type SE-D control the switching between different models and service the events (E-SE-D). State events affecting a state variable (SE-S type) can be modelled at this external level (E-SE-S type), or also as classic internal event (I-SE-S). Mosilab translates each model separately, and generates a main simulation program out of state charts, controlling the call of the precompiled models and passing data between the models.

Mosilab is in developing, so it supports only a subset of Modelica, and index reduction has not been implemented yet, so that MOD gets a '(yes)' in parenthesis, and IR gets a '(no)'. Index reduction at present not available in Mosilab, but planned (IR '(no)') – has become topic of discussion: case studies show, that hybrid decomposition of structural dynamic systems results mainly in DAE systems of index $n = 1$, so that index reduction may be bypassed..

Mosilab allows very different approaches for modelling and simulation tasks. In a standard Modelica approach, the *Constrained Pendulum* is defined in the MOSILAB equation layer as implicit law; the state event, which appears every time when the rope of the pendulum hits or 'leaves' the pin, is modelled in an `algorithm section` with if (or when) – conditions.

But Mosilab's state chart construct allows also any kind of hybrid composition of models with different state spaces and of different type (from ODEs to PDEs, etc.). Listing 5 shows a Mosilab implementation of the *Constrained Pendulum* making use of two different pendulum models, controlled externally by a state chart. The `transitions` organise the switching between the pendulums (`remove`, `add`).

Mosilab's state chart construct allows also any kind of hybrid composition of models with different state spaces and of different type (ODEs, PDEs, etc.).

But Mosilab's state chart construct allows also any kind of hybrid composition of models with different state spaces and of different type (ODEs, PDEs, etc.).

```
event Boolean lengthen(start = false),
              shorten(start = false);
equation
   lengthen=(phi>phipin); shorten=(phi<=phipin);
equation /* pendulum*/
   v = l1*der(phi); vdot = der(v);
   mass*vdot/l1 + mass*g*sin(phi)+damping*v= 0;
statechart
state LengthSwitch extends State;
   State Short,Long,Initial(isInitial = true);
   transition Long -> Short event shorten
      action length := ls;
   end transition;
   transition Short -> Long event lengthen
      action length := l1;
   end transition; end LengthSwitch;
```

Listing 4. Mosilab Model for *Constrained Pendulum* – State Chart Model with Internal Events (I-SE-P)

Listing 5 shows a Mosilab implementation of the *Constrained Pendulum* making use of two different pendulum models, controlled externally by a state chart. The `transitions` organise the switching between the pendulums (`remove`, `add`).

Mosilab offers also strong support for simulator coupling (e.g. MATLAB) and time-synchronised coupling of external programs. This feature may be used for any kind of visualisation not based the model definition (VIS '(yes)'). For complex experiments, Mosilab allows to mix model frame and experimental frame and sets up a common extended environment (ENV 'yes'), where also frequency analysis can be implemented (FA '(no)').

8.6 Open Modelica

The goal of the *Open Modelica* project is to create a complete Modelica modelling, compilation and simulation environment based on free software distributed in binary and source code form. Open Modelica is a generic Modelica simulator, so all basic features are met (ED, SEH, DAE, PM-T, PM-G, IR and MOD 'yes'; for DAE solving, variants of DASSL solver are used). P. Fritzson, the initiator of Open Modelica puts emphasis on discrete events and hybrid modelling, so documentation comes with clear advice for use of IF – and WHEN – clauses in Modelica, and with state chart modules in DrModelica – so SC-T gets 'yes'. For graphical state chart modelling the experimental Modelica state chart library can be used – so SC-G '(yes)'. The notebook features allow interfaces and extensions of any kind, e.g. for data visualisation and frequency analysis – FA and VIS '(yes)'; they allow also for controlled executive of different models, so that hybrid decomposition of structural dynamic systems is possible – SD '(yes)'.

```
model Long
  equation
     mass*vdot/l1 + mass*g*sin(phi)+damping*v = 0;
end Long;
model Short
  equation
     mass*vdot/ls + mass*g*sin(phi)+damping*v = 0;
end Short;
event discrete Boolean lengthen(start=true),
                       shorten(start=false);
equation
   lengthen = (phi>phipin);
   shorten  = (phi<=phipin);
statechart
 state ChangePendulum extends State;
   State Short,Long,startState(isInitial=true);
   transition Long -> Short event shorten action
      disconnect ….; remove(L); add(K); connect …
   end transition;
   transition Short -> Long event lengthen action
      disconnect …; remove(K); add(L); connect …
   end transition;
 end ChangePendulum;
```

Listing 5. Mosilab Model for *Constrained Pendulum* – State Chart Switching between Different Pendulums Models by *External Events* (E-SE-P)

8.7 SimulationX

SimulationX is a new Modelica simulator developed by ITI simulation, Dresden. This almost generic Modelica simulator is based on ITI's simulation system ITI-SIM, where the generic IT-SIM modelling frame has been replaced by Modelica modelling. From the very beginning on, ITI-SIM concentrated on physical modelling, with a theoretical background from power graphs and bond graphs. Consequently, all features f related to physical modelling are available: (ED, SEH, DAE, PM-T, PM-G, and MOD 'yes'; index reduction is not really implemented – IR '(no)'. State chart constructs are not directly supported (SC-T 'no'), but due to Modelica compatibility the Modelica state chart library can be used (SC-G – '(yes)'. Frequency analysis is directly supported in the simulation environment (FA – 'yes'), as well as interfaces to other systems (ENV – '(yes)') and pseudo-3D visualisation (VIS – 'yes').

8.8 AnyLogic

AnyLogic – already discussed in a previous section) is based on hybrid automata (SC-T and SC-G - 'yes'). Consequently, hybrid decomposition and control by external events is possible (ED, SD 'yes'). AnyLogic can deal partly with implicit systems (only nested approach, DAE '(yes)'), but does not support a-causal modelling (PM-T, PM-G - 'no') and does not support Modelica (MOD - 'no'). Furthermore, new versions of AnyLogic concentrate more on discrete modelling

and modelling with System Dynamics, but without event detection (SEH '(no)'. AnyLogic offers many other modelling paradigms, as System Dynamics, Agent-based Simulation, and DEVS. AnyLogic is Java-based and provides simulation-driven visualisation and animation of model objects (VIS 'yes') and can also generate Java web applets. From software engineering view, AnyLogic is a programming environment for Java – so ENV – '(yes)'.

In AnyLogic, various implementations for the *Constrained Pendulum* are possible. A classical implementation is given in Figure 3, following classical textual ODE modelling, whereby a state chart is used for switching (I-SE-P, I-SE-S). A hybrid decomposed model may make use of two different models, each defined in substate - both part of a state chart switching between these submodels. But is is also possible to decompose model in 'parallel': In *Constrained Pendulum* example, the ODE for the angle may be put in the main model, together with transformation to Cartesian coordinates (Figure 6).

8.9 SCILAB / SCICOS

Scilab is a scientific software package for numerical computations with a powerful open computing environment for engineering and scientific applications. Scilab is open source software. Scilab is since 2003 in the responsibility of the Scilab Consortium. *Scicos* is a graphical dynamical system modeller and simulator toolbox included in Scilab. Scilab / Scicos is an open source alternative to MATLAB / Simulink, developed in France. So it has nearly the same features than MATLAB: no equation sorting– MS – 'no'!; DE, IR, PM-T, PM-G, MOD, SC-T, and SC-G – 'no'; SEH, DAE, and VIS – '(yes)', remarkably – SD, FA and ENV – 'yes'. Similarly, Scicos has extended features ED, SEH, and DAE – 'yes'.

The developers of Scicos started early with a kind of physical modelling (PM-T, PM-G – yes). They are working on extensions in two directions:

- extending the model description by full Modelica models (text/graphic) – so MOD and IR '(yes)'
- refining the `IF-THEN-ELSE` and `WHEN` clause introducing different classes of associated events, resulting 'state chart clauses' - so SC-T – 'yes'

Standalone Scicos has no features for frequency analysis, structural decomposition and extended environment (FA, SD, ENV – 'no'), but limited visualisation (VIS – '(yes)'); Scicos controlled by Scilab has all these features (VIS, FA, SD, ENV – 'yes').

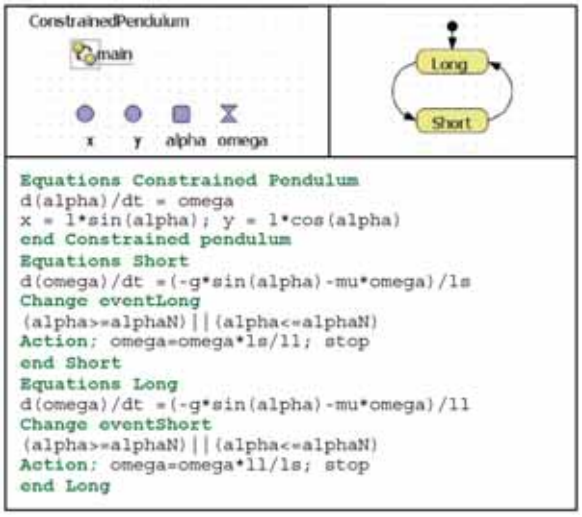


Figure 6. AnyLogic Model for *Constrained Pendulum*, Hybrid Model Decomposition with Two Models for Angular Velocity and Overall Model for Angle

8.10 Maple

Maple – developed by Maplesoft, Canada, is working on a toolbox *MapleSim*, which will understand Modelica models (PM-T, PM-G, and MOD – 'yes'). Maple acts as environment and provides sophisticated DAE solvers, as well as symbolic algorithms for index reduction (DAE, IR, ENV, VIS, FA – 'yes').

In development are constructs for events and event handling (ED–'yes)', SEH–'(no)'); textual state chart modelling has not been discussed yet, graphical state chat notation may com from the experimental Modelica state chart library (SC-T – 'no', SC-G – '(yes)').

8.11 Availability of features - comparison

Table 2 provides an availability comparison of the discussed features within the presented simulators, as given before in detail.

Clearly such comparison must be incomplete, and using simple 'yes' and 'no' might be too simple (although the items in parenthesis give a refinement) . Consequently, it should be a hint for further detailed feature comparison.

Main source for the evaluation of the simulator are solutions to the ARGESIM Benchmarks, published regularly in the journal SNE – Simulation News Europe.

References

As a really adequate reference list, with details on structures, features, and detailed developments and background would cover again 10 pages, alternatively the list is restricted to only few main sources.

[1] F. Breitenecker, I. Troch. 2004. 'Simulation Software – Development and Trends'. In *Modelling and Simulation of Dynamic Systems / Control Systems, Robotics, and Automation.* H. Unbehauen, I. Troch, and F. Breitenecker (Eds.). Encyclopedia of Life Support Systems (EOLSS), UNESCO, Eolss Publishers, Oxford ,UK, www.eolss.net.

[2] Cellier, F.E. (1991). *Continuous System Modeling*. Springer, New York.

[3] Fritzson, P. 2005. Principles of Object-Oriented Modeling and Simulation with Modelica. Wiley IEEE Press.

[4] Fritzson, P., F.E.Cellier, C. Nytsch-Geusen, D. Broman, and M. Cebulla, Eds. 2007. *EOOLT'2007 - Proc. 1st Intl. Workshop on Equation-based Object-oriented Languages and Tools*. TU Berlin Forschungsberichte, Vol. 2007-11.

[5] Nytsch-Geusen C., and P. Schwarz. 2005. 'MOSILAB: Development of a Modelica based generic simulation tool supporting model structural dynamics'. In *Proc. 4th Intern. Modelica Conference,* G. Schmitz (Ed.), Modelica Association modelica.org, 527 – 535.

[6] Strauss J. C. 1967. 'The SCi continuous system simulation language (CSSL)'. Simulation 9, SCS Publ. 281-303.

Corresponding author: Nikolas Popper,
dieDrahtwarenhandlung Simulation Services
Neustiftgasse 57-59, 1070 Vienna, Austria
niki.popper@drahtwarenhandlung.at

Accepted EOOLT 2007, June 2007
Received: August 20, 2007
Major revision received: September 13, 2008
Accepted: September 20, 2008

	MS - Model Sorting	ED -Event Description	SEH -State Event Handling	DAE - DAE Solver	IR - Index Reduction	PM-T - Physical Modelling -Text	PM-G - Physical Modelling -Graphics	VIS – 'Onlie' - Visualisation	MOD – Modelica Modelling	SC-T – State Chart – Modelling - Text	SC-G – State Chart Modelling - Graphics	SD – Structural Dynamic Systems	FA – Frequency Analysis	ENV – Extended Environment
MATLAB	no	no	(yes)	(yes)	no	no	no	(yes)	no	no	no	yes	yes	yes
Simulink	yes	(yes)	(yes)	(yes)	no	no	(no)	(yes)	no	no	no	no	yes	(yes)
MATLAB / Simulink	yes	yes	yes	(yes)	no	no	(no)	(yes)	no	no	no	yes	yes	yes
Simulink/Stateflow	yes	yes	yes	yes	no	no	(no)	(yes)	no	(yes)	yes	no	yes	(yes)
Simulink/Simscape	yes	yes	yes	yes	(yes)	yes	yes	(yes)	(no)	no	no	no	yes	(yes)
Simulink/Simscape/Stateflow	yes	yes	yes	yes	(yes)	yes	yes	(yes)	(no)	(yes)	yes	no	yes	(yes)
MATLAB/SL/SS/Stateflow	yes	yes	yes	yes	(yes)	yes	yes	(yes)	(no)	(yes)	yes	yes	yes	yes
ACSL	yes	yes	yes	yes	no	no	(no)	(yes)	no	no	no	no	yes	yes
Dymola	yes	yes	yes	yes	yes	yes	yes	yes	yes	(yes)	(yes)	no	(no)	(yes)
MathModelica	yes	yes	yes	yes	yes	yes	yes	(yes)	yes	(no)	(yes)	no	(no)	(no)
MathModelica/Mathematica	yes	yes	yes	yes	yes	yes	yes	yes	yes	(no)	(yes)	yes	yes	yes
Mosilab	yes	yes	yes	yes	(no)	yes	yes	(no)	(yes)	yes	yes	yes	no	(yes)
Open Modelica	yes	yes	yes	yes	yes	yes	(no)	(no)	yes	(no)	(yes)	no	no	no
SimulationX	yes	yes	yes	yes	yes	yes	yes	yes	yes	(no)	(yes)	no	yes	(yes)
AnyLogic	yes	yes	(yes)	(yes)	no	no	no	yes	no	yes	yes	yes	no	no
Model Vision	yes	yes	yes	yes	yes	yes	no	yes	no	yes	yes	yes	yes	no
Scilab	no	no	(yes)	(yes)	no	no	no	(yes)	no	no	no	yes	yes	yes
Scicos	yes	(yes)	yes	yes	(yes)	yes	yes	(yes)	(yes)	yes	(yes)	no	no	no
Scilab/ Scicos	yes	yes	yes	yes	(yes)	yes	yes	(yes)	(yes)	yes	(yes)	yes	yes	yes
MapleSim	yes	(yes)	(yes)	yes	yes	yes	yes	yes	yes	no	no	(yes)	(yes)	yes

Table 2. Availability of *Extended* and *Structural Features* in Simulators - DAEs, State Events, Modelica Notation, Structural Decomposition, and Related Features

TN

Prototype of Network Management System for Real Time Architecture Development

Mohd Nazri Ismail, Sera Syarmila, Univ. Kuala Lumpur, Malaysia, *{mnazrii, sera}@miit.unikl.edu.my*

We present a novel approach for the development of network management system in real time environment. This research investigates performance evaluation of network monitoring servers in Local Area Network (LAN) environment. We propose an enhanced equation to evaluate the performance of network traffic management via Queuing theory. To get accuracy results on the performance of network management prototype, we measure and capture traffic in and out from real network environment. We use network management tool to capture those traffic accessing the servers. As a result, this prototype of network management system for real time can provide a good approximation of the real traffic observed in the real network environment. Through real experiments, it shows that the network management system via prototype is capable of approximating the performance of the servers within a minimum error rate.

SNE 18/3-4, December 2008

Introduction

Prototype of network management system for real time was developed to monitor the network performance of the servers. Considerable research has been conducted to model and quantify the performance of network (e.g. [1, 2, 3]). Accurate measurements and analyses of network characteristics are essential for robust network performance and management. Evaluating the performance of servers usually involves constructing an appropriate model to predict the server performance via prototype development. The server performance is then analyzed using mathematical techniques. For example, several flow-level network traffic models have been proposed to describe/stimulate [4,5,6,10]. In contrast to other works in the literature (e.g., [7, 8, 9, 15, 16]), we developed prototype of network management system to measure the performance of servers. Our prototype can be used to monitor server performance in a live network environment. The significant of this study was to develop a prototype of network management system for real time to measure the performance of servers using Queuing theory. The beneficial and contribution of this model can monitor and manage the server performance. This prototype model is designed to:

1. manage the performance of servers;
2. monitor the uploading and downloading speed of network performance; and
3. assist network administrator to prepare, propose and plan the server activities more effective and systematic.

Moreover, in the future, the integration of data and communication services, almost every 'Internet Ready' device will be a communicable device [11]. Many factors may contribute to the congestion of network interface, such as a heavy load in the network that usually generates higher traffic. Thus, this research is critical to be conducted in order to predict and measure of servers performance.

1 Related Works

Multi-traffic in the network infrastructure has become more complex to observe and analyze [12], [13]. The main factors of network congestion are related to network design and bandwidth capacity [17]. Many tools have been developed, and only a few tools have successfully achieved a close estimation of network bandwidths. Therefore, retrieving and sending information from servers to Internet or Intranet in Higher Educational Institutes should be analyzed and evaluated via prototype of network management system. We have setup a real network environment to monitor and analyze of network traffic at University of Kuala Lumpur in Malaysia. This study posits several research questions: i) what is the traffic performance level of server for real time; and ii) Is the prototype of network management system for evaluating and measuring the server's performance effective?

2 Methodologies

Figure 1 shows the overall framework of the network management prototype. There are four performance techniques to validate the simulation model: i) graphical representation; ii) tracing; iii) parameter variability; and iv) predictive validation. In addition, there are two techniques to judging how good a model is with respect to the real network: i) prototype verification; and ii) prototype validation. Comparison

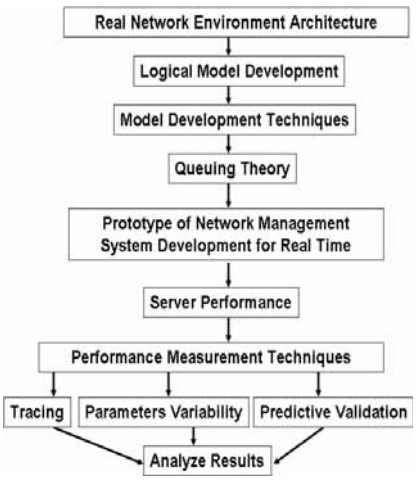


Figure 1. Prototype Development Methodology

with a real environment is the most reliable and preferred method to validate a prototype model (refer to Figure 2).

3 Proposed network monitoring development for server performance

Many different types of modeling and simulation applications are used in various disciplines such as acquisition, analysis, education, entertainment, research and training [14]. Network management prototype is divided as follows: i) to study physical environment of real client-server environment; ii) transform physical model of client-server network environment into logical model; and iii) develop and implement the client-server model.

3.1 Physical model of real network server environment

Before we start to develop prototype of client-server environment, we need to define the situation of server environment in real world. Figure 3 shows the client-

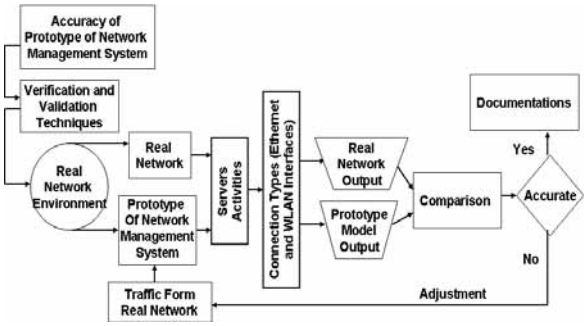


Figure 2. Prototype Verification and Validation Methodology

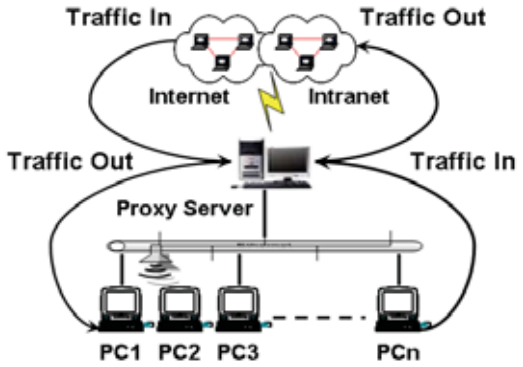


Figure 3. Client-Server Real Network Environment Experiment

server environment in real world. The physical model of network server environment is based on traffic in and traffic out at University of Kuala Lumpur.

3.2 Logical model of network server environment

Figure 4 depicts the open queuing network based on M/M/1 will use to develop logical model of client-server environment for server performance. The logical model is the phase where mathematical techniques are used to stimulate client-server environment. The logical model is the important area need to define which mathematical techniques should be used in development of client-server environment.

3.3 Development of network server environment model and architecture

This section describes a simple analytical queuing theory coding that captures the performance characteristics of server operations in real time. We use Visual Basic (VB) application to develop our real time prototype for network monitoring, refer to list-

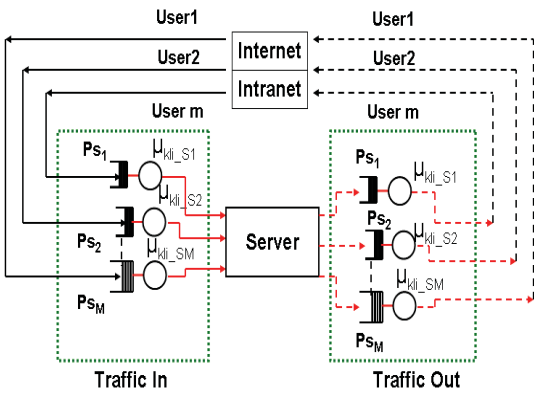


Figure 4. Logical model of Client-Server Network Environment

Model parameters	Meaning
$P(P_1, P_2, \ldots, P_m)$	Various services
μ	Size of packet services request by client

$$P_{S1} + P_{S2} + \ldots + P_{Sm} = \mu_{kl,S1} + \mu_{kl,S2} + \ldots + \mu_{kl,Sm}$$

$$\mu_{kl,S1} + \mu_{kl,S2} + \ldots + \mu_{kl,Sm} = BytesRcvd$$

$$\mu_{kl,S1} + \mu_{kl,S2} + \ldots + \mu_{kl,Sm} = BytesSent$$

Table 1. Notations for prototype development

ing 1. The link bandwidth is the rate at which bits can be inserted into the medium. Table 1 shows the parameters that have been used in the model development.

```
DoEvents
' Define Parameters:
Dim DS As Long, US As Long
DS = BytesRecv - LastRecvBytes
US = BytesSent - LastSentBytes

If DownloadSpdTop<DS Then DownloadSpdTop=DS
If UploadSpeedTop<US Then UploadSpeedTop=US
DoEvents

' Apply queuing theory
lblRecv.Caption = Format(BytesRecv/1024, …)
lblSent.Caption = Format(BytesSent/1024, …)

' Average of download and upload speed
DownloadSpeedAvg = (DownloadSpeedAverage+DS)/2
UploadSpeedAvg = (UploadSpeedAverage + US)/2

lblDownloadSpeedTop = "Top download speed: "
      & Format(DownloadSpeedTop/1024, …)
lblUploadSpeedTop = "Top upload speed: "
      & Format(UploadSpdTop/1024, …)
```

Listing 1. Coding of bytes received and sent for Real-time Basic using Visual Basic

Figure 5, show prototype of network monitoring architecture development, which is used to measure and analyze the server performance in real time basic. This prototype model can measure and analyze LAN and WLAN network card performance, see figure 5.

Figure 6 shows main menu of network monitoring interface. This server network monitoring is protected by the password and only authorized person can logon to this prototype model.

Figure 7 shows the sample of bytes received and sent interface. Network administrator can select either LAN or WLAN network interface card to monitor and measure the server performance. Figure 7 shows how the model has been formulated from real network environment to prototype model in real time measurement.

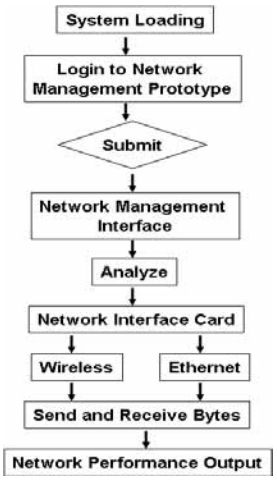


Figure 5. Prototype of networking monitoring architecture development in real-time.

Figure 6. Main Menu of Network Monitoring Interface in Real Time

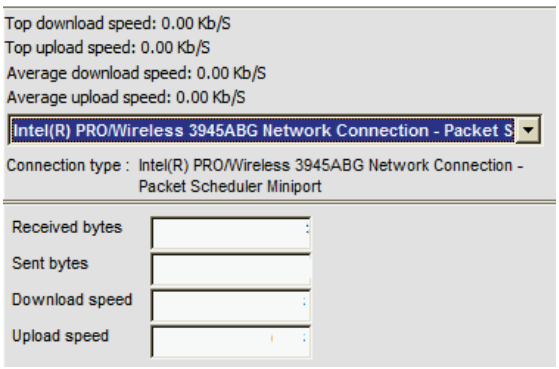


Figure 7. Bytes Receive and Send Interface Network Monitoring

4 Verification/validation of prototype model with real network experiment

Network management application is used to capture server traffic in and out in real network environment. Figure 8 (see box) shows the experimental setup of

real network used in our tests. By using varying number of clients and size of packet services, we are able to capture and measure network traffic.

The real network experiment is based on measurement and monitoring server network interface card performance. Low bandwidth link affects the size of packet services and number of clients' access to the network server. The experiment is using network management application (NMS) to measure traffic in and out into server platform (see Figure 9). Three sets of experiments were conducted with different scenarios based on bytes sent and received. The results from real network experiments will compare with prototype model. We captured the results that have been generated in prototype model (see Figure 10) to estimate our data that must be closely resemble to real network environment.

We conclude that based on our findings, the prototype is able to predict and estimate traffic in and out for real network environment (see Table 2).

Figure 11 shows relative error rates occurring between prototype modeland network management system (NMS) in real network environment. As a result, it confirms that our prototype model in real time is closely resembled to real environment with minimum error range.

5 Conclusions and future work

In this article, we have shown how prototype network model can be used to understand the performance of the server. The most apparent aspect is traffic in and out into the server platform. This prototype model, has demonstrated that it can measure accurately the performance of server in real time networks. Through real network experiments, the prototype model is verified and validated for providing accurate performance information for server activities. In network management, by monitoring and analyzing network traffic (bytes receive and send) network administrator can monitor the performance of the servers, thus to study whether network is normal, optimal or overloaded. Future work is to develop a prototype model to measure and monitor bandwidth management, ping utility and port scanning for real time network.

References

[1] Kawasaki, S., Niwa, et. al. 2006. *A Study on Formulation of the Ubiquitous Cloud Model*, Mobile Data Management, 2006. MDM 2006. 7th Int. Conference, pp:148-148.

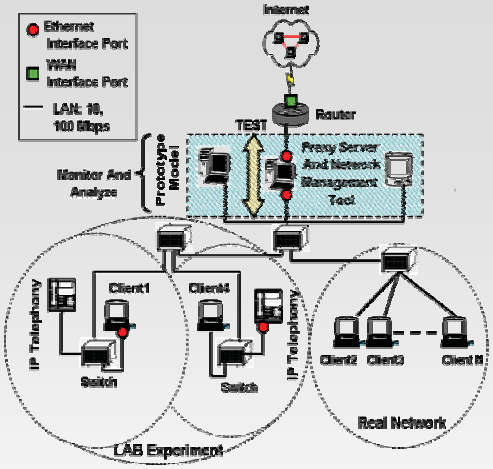


Figure 8. Experimental for Real Network

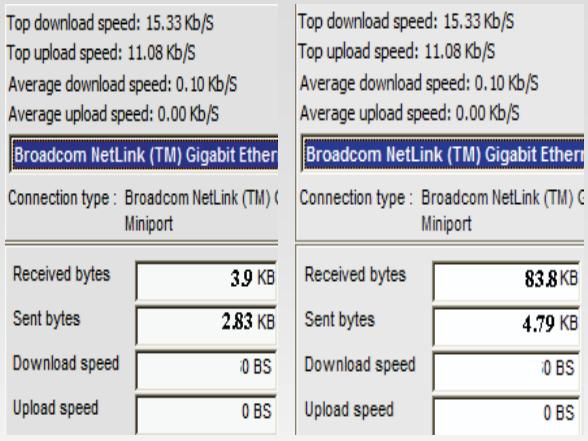


Figure 10. Prototype model Gigabit Ethernet interface results: Experiment 1 (left), experiment 2 (right)

Adapter Name	Adapter Description	Link Speed	State	Bytes Sent	Bytes Received
Local Area Connection	Broadcom NetLink (TM) Gigabit Ethernet - Pa...	100 Mbps	Operational	2,734	4,704
Wireless Network Connection	Intel(R) PRO/Wireless 3945ABG Network Co...	54 Mbps	Non Operational	0	0

Adapter Name	Adapter Description	Link Speed	State	Bytes Sent	Bytes Received
Local Area Connection	Broadcom NetLink (TM) Gigabit Ethernet - Pa...	100 Mbps	Operational	4,680	85,950
Wireless Network Connection	Intel(R) PRO/Wireless 3945ABG Network Co...	54 Mbps	Non Operational	0	0

Figure 9. Network management system results: Experiment 1 (top), experiment 2 (bottom).

Exp.	kBytes received		kBytes sent	
	Prototype	NMS tool	Prototype	NMS tool
1	3.900	4.704	2.830	2.734
2	7.312	8.804	4.412	4.334
3	83.800	86.950	4.790	4.680

Table 2. Comparison between prototype model and NMS tool

[2] Tsalgatidou, G. Athanasopoulos & et. al. 2006. *Developing scientific workflows from heterogeneous services*, ACM SIGMOD, Vol. 35 (2), pp: 22-28

[3] Q. Liu, S. Zhou, et. al,. 2006. *Cross-layer modeling of adaptive wireless links for QoS support in heterogeneous wired-wireless networks*, Wireless Networks, Vol. 12 (4), Kluwer Academic Publishers.

[4] S. Fredj, T. Bonald, A. Proutiere, G. Regnie, and J. Roberts. *Statistical bandwidth sharing: A study of congestion at flow level*. In Proc. of ACM SIGCOMM '01, San Diego, CA, August 2001.

[5] Barakat, P. Thiran, G. Iannaccone, C. Diot, P. Owezarski *A flow-based model for Internet backbone traffic*. Proc. of the 2nd ACM SIGCOMM Workshop on Internet measurement, p.p: 35–47, 2002, Marseille

[6] G. Jin, B.L. Tierney. 2003. *System capability effects on algorithms for network bandwidth measurement*. Proc. of the 3rd ACM SIGCOMM conference on Internet measurement, p.g 27-38.

[7] S.B. Fredj, T. Bonald, A. Proutiere, G. Regnie, J. Roberts. *Statistical Bandwidth Sharing: A Study of Congestion at Flow Level*, ACM SIGCOMM, August 2001.

[8] T. Bu, D. Towsley. *Fixed Point Approximation for TCP behavior in an AQM Network*, ACM SIGMETRICS, Jun. 2001.

[9] A.A. Kherani, A. Kumar, *Performance Analysis of TCP with Nonpersistent Sessions*, Workshop on Modeling of Flow and Congestion Control, INRIA, Ecole Normale Superieure, Paris, September 4-6, 2000.

[10] O. Balci. *Quality Assessment, Verification and Validation of Modeling and Simulation Applications*. Proc. 2004 Winter Simulation Conference. Simulation Conference, 2004. Proc. of the 2004 Winter Vol. 1, Issue , 5-8 Dec. 2004 pp:-129

[11] Qigang Zhao; Xuming Fang; Qunzhan Li; Zhengyou He. 2005. *WNN-based NGN traffic Prediction*. ISADS 2005. Proc.Autonomous Decentralized Systems: 230-234.

[12] Thai, R. Wan, A. Seneviratne, T. Rakotoarivelo.2003. *Integrated Personal Mobility Architecture: A Complete Personal Mobility Solution*. Kluwer Academic Publishers.

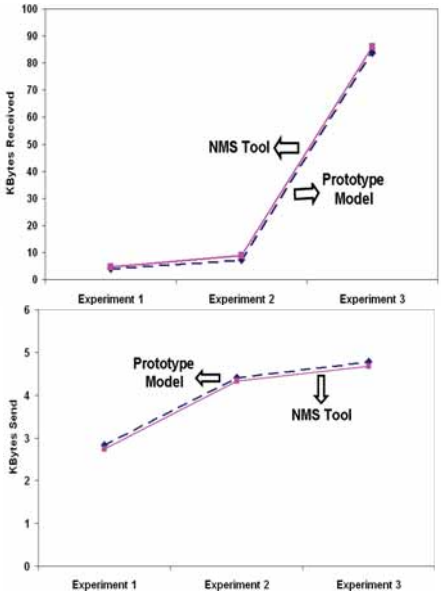


Figure 11. Comparison of real experiment and prototype model: kBytes received (top), Bytes sent (bottom)

[13] P. Podhradsky, 2004. *Migration scenarios and convergence processes towards NGN (present state and future trends)*. Electronics in Marine Proc. Elmar. 46th International Symposium, :39-46.

[14] J. Curtis, T. McGregor, *Review of bandwidth estimation techniques*. In Proc. New Zealand Computer Science Research Students' Conf., vol. 8, New Zealand, Apr. 2001.

[15] Jing Cong, Bernd E. Wolfinger. 2006. *A unified load generator based on formal load specification and load transformation*, Proc. of the 1st international conference on Performance evaluation methodolgies and tools, ACM International Conference Proc. Series, Pisa, Italy, Vol. 180 (53).

[16] J. Kontio, R.Conradi. *Evaluating the Performance of a Web Site via Queuing Theory* Software Quality-ECSQ 2002: 7th International Conference, Helsinki, Finland, June 9-13, 2002, pp. 63-72. Springer-Verlag Berlin Heideberg.

[17] David R.Gerhan, Stephen M. Mutula. 2005. *Bandwidth bottlenecks at the University of Botswana*, Published by Emerald Group, Vol. 23(1), pp 102-117.

Corresponding author: Mohd Nazri Ismail,
Faculty of MIIT
University of Kuala Lumpur, Malaysia,
mnazrii@miit.unikl.edu.my

Received: May 11, 2008
Revised: September 18, 2008
Accepted: October 25, 2008

EDUCATION NOTES

A PHP/MATLAB-Based E-Learning System for Education in Engineering Mathematics and in Modeling and Simulation

Günther Zauner, Nikolas Popper, die Drahtwarenhandlung Simulation Services, Vienna, Austria
Felix Breitenecker, Florian Judex, Vienna University of Technology, Austria

The goal of this work is to present an e-learning tool based on the MATLAB Webserver technology. We offer an adaptive PHP framework which can be used for interactive learning in lessons and in project practices, as well as for web presentations of computer algebra solutions made in MATLAB (e.g. nonlinear fit problems for medical data). All mathematical/numerical solutions of the tasks are done in MATLAB. Another basic of the concept is not only to show the students the solution of a problem via internet for several different functions or parameters, but also to offer them the source code. The students can download the code and test other features and learn programming of mathematical solutions with a computer numeric/algebra package. An explanation of the detailed structure of the PHP – framework is given.
The main focus of this paper lies on model attempts for physiological systems. An example of a simple infusion model is used for interactive learning and system testing.
In several parts the way from a poor data interpolation to a data model with exponential functions up to a solution with transfer functions and parameter optimization is described. The outlook concerns the expandability of the defined framework, and will also focus on the restrictions of the system and how to deal with them.

SNE 18/3-4, December 2008

General

At Vienna University of Technology a very similar problem like in other technical schools/universities in presenting lectures dealing with modeling and simulation basics, as well as summing up the necessary mathematical theory occurs: How to present the theory of modeling and simulation of special tasks based on examples, so that the students can follow it easily and thereby learn the most important basics?

As known from theory "learning by doing" is one of the best options. That is why the department for analysis and scientific computing designed the following structure for education. One of the main goals is to present dynamical models with a praxis interrelationship, which will be explained during the lessons, but should be also available via a web interface for advanced learning at home. Another principle is based on mathematics and computer numeric. For example it is much more comprehensible for students that the associative and distributive laws working with floating point numbers are hurt, when the conclusion is confirmed by interactive examples. In this case it is also important to show the programmed code in an easy readable language or with pseudo code.

This leads to the next benefit of the defined structure: the algorithmic part of the examples is all done in MATLAB and MATLAB/Simulink. This computer algebra/numeric package including the *symbolic math toolbox* is also used in the lessons "Einführung in das Programmieren für technische Mathematiker" and "Computermathematik". Therefore the main part of the visitors of the lessons, where the Webserver applications are included, are familiar with reading MATLAB code.

1 Background of MATLAB webserver

In general, as the examples are all realized in the MATLAB Release 2006a, the MATLAB Webserver application [1] needs an additional Webserver to act as server in client/server web architecture. In our case we use an Apache Webserver [2]. The interface used for input or input/output representation is defined by standard HTML (Hypertext Markup Language) [3] frames, which interact with MATLAB via a CGI – script.

In our case we decided to use PHP and interconnect the files directly with the Webserver. This has the following benefits:

- The system becomes more stable, because we have only two layers left instead of three.
- After defining the structure once, the whole system acts in modular concepts, which means that

we can adapt examples and add new ones, without any code writing in PHP or HTML.

To get a general reusable system we have to define a global concept, capable to support the most important features for education in mathematics, modeling and simulation. Thus, we will get some restrictions in graphical representation and/or textual representation, but on the other hand a well defined structure supports easy model implementation and illustrating special content of teaching, which is not that easy explained in the common way at a blackboard.

The main part of the work is to create the basic structure. We chose the following frame definition, as shown in Figure 1.

The upper left part of our input/output interface is the title of the chapter and a short explanation of the context, this part is dealing with.

In the same frame, beginning in the middle until the right side, there is the list of the examples corresponding to the chapter/block. This part has place for three columns with up to five rows providing the links to the specific examples. The lower part of the website, which is split into two frames, has an empty right side at the beginning of an example. On the left part a short description of the actual model is given. Under this part the definition and the settings of the selectable variables and parameters are placed.

2 Detailed structure

As already explained a general network is important to make an easy useable structure and to allow the fast growth of the system after a one-time detailed structural definition and implementation.

This is done in PHP with some special features. For fast transformation of the poor MATLAB m-file code into an MATLAB Webserver application we have to define some restrictions for the right side of the lower frame. This frame represents the output part of the system. As MATLAB allows different types of outputs and our system has restrictions in place and abilities for data representing (e.g. no rotation of 3D – graphics possible), we have to define strict rules for output creation and representation.

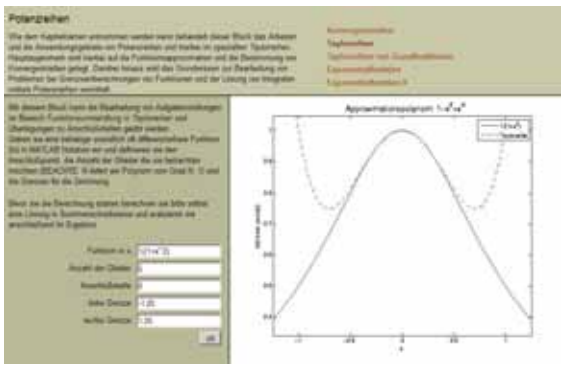


Figure 1. General framework of the user interface including the first example

2.1 The three different output structures

In many cases of data transformation and in modeling and simulation, the user or developer is only interested in a textual form of the output. As known MATLAB can easily organize textual output to the command window (e.g. commands `disp` or `print`) or to a file. For our structure we define the synonym `retstr` as standard return value. This can be set to a string beginning with `'TEXT'` and then filled with the return text, in which standard HTML commands, like `<br>` or `<i>`, can be implemented. The originated text is then transformed to standard HTML text in the output frame after executing the m-file.

The second sort of output is the classical plotting window. This is in our structure a so called 'IMAGE' – structure and therefore the `retstr` return value is set to this string. The implementation of such a structure needs a few extra code lines. An implementation of a graphical output can be done for example like this:

```
1 Pic = figure('visible','off');
      % in this part the plot is defined in the same way
      % as this is done in standard MATLAB notation
2 drawnow;
3 wsprintjpeg(Pic, instruct.mlimgfilename);
4 retstr = 'IMAGE';
```

The third output class is the so called 'ERROR' – class. To define the return value we first have to define what we consider to be an error. The first part where this class occurs is, when a MATLAB internal error arises. This can be for example because of wrong dimensions of input vectors or singularity of matrices. In these cases the original error message from MATLAB should be displayed via the web interface. The second sort of error message is the developer defined case. Such error messages are more or less equivalent to the work/usage of the `TEXT` construct and are in many cases used for programmer defined breaks before a MATLAB internal bug can occur.

2.2 Types of input variables

As everybody who is handling IO – interfaces for user applications knows, it is very hard to define a

structure in which the user has a broad field of testing potentials, but concurrent catching all parameter settings and structures which are not allowed because this takes a lot of programming effort. Therefore we define special data types in PHP and perform the basic checks for the range directly in the definition part of the class.

The following code fragment shows all types of variables. The detailed explanation can be found below.

```
1 <?php
2    $pageVars['ml_mfile']='example_ergo3';
3    $form = new Form();
4    $form->addField(new ComboField('var1',
         'paramter to optimize:', 1,
   array(1 =>'TI',2 =>'KI',3 =>'TB',4 =>'KB')));
5    $form->addField(new FloatField('var2',
         'starting value:', 0.1, 0.0001, 60));
6    $form->addField(new IntegerField('var3',
         'Number of steps:', 5, 1, 30));
7    $form->addField(new TextField(
         'var4,'Answer:', 'Optim.'));
8    $pageVars['form'] = &$form;
9 ?>
```

The second code line defines the coupled MATLAB file, which runs within the defined structure. The following lines define examples for the four different input types: `ComboField`, `FloatField`, `IntegerField` and `TextField`.

The `ComboField` is a classical combo box (see also Figure 2) as defined in several GUIs (General User Interface). In the example above the user can switch between four cases, which are than in MATLAB represented as numbers one to four.

The next part is the definition of a `FloatField`. The MATLAB name of the corresponding variable stands first and is the string `var2`. The next part separated by comma is the name the user will see in the interface. The third part is the default setting followed by the minimum value and the maximum, which can not be reached. The testing, if the defined value is valid, is executed by PHP, and in case it is not the input name is highlighted in red color and furthermore an error message occurs.

The definition of an `IntegerField` is done in a similar way as the `FloatField`. It is implemented because in many cases it does not make sense to define everything as floating point number. Moreover it improves the data filtering for the m-file.

The fourth example is a standard `TextField`. Compared to the `IntegerField` construction in this structure the last two parts of definition are missing. That is because a string does neither have a minimum value nor maximum value.

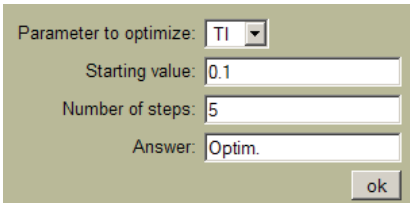


Figure 2. Graphical output corresponding to the code fragment on the left side of the page.

If we save this definition in the so called `initialize` text document, we get a system like depicted in fig. 2.

After explanation of the different data types the definition of the structure for the other components has to be done.

2.3 Input structure

We distinguish two different levels of our structure. This is done to ensure the reusability of the system and allow other staff, after a very short instruction period, to define new chapters and Webserver examples after defining the appropriate MATLAB code.

The main layer is the so called chapter level. The folder for one layer includes subfolders with the examples and four text files (There are several other files which are not important for the developer of new examples. These files include PHP code.). The first one, *description*, contains the text which is then represented in the top frame at the left side. This text can be defined in a standard editor using basic HTML commands.

The second, *headline*, is self-explanatory.

The third text file includes a few PHP commands in which the user can define a number for sorting the chapters. This part is optional and is only implemented for advanced system definition and extension for chapters in a higher hierarchical order.

The last file in the folder, *navigationLabel*, defines the chapter number. This block is also optional if we focus on only one chapter as a web application.

Now we have already defined the structure of an example chapter. One thing missing until now is the structure of the examples included in such a package. But as we will see the structure is quite similar and that is why the user does not have to learn many things before starting the implementation of examples.

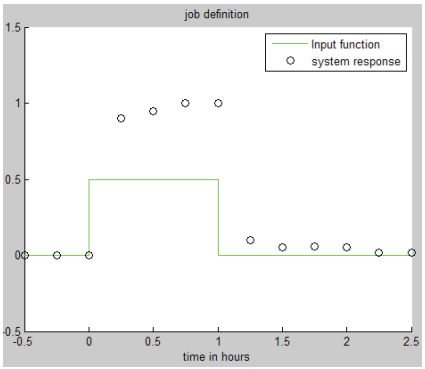


Figure 3. Model assumption

Again we have a text file called *description*, where the short explanation of the file is created in textual form or with a few HTML additions. The *initialize* part is explained in detail in Section 2.2.

The *navigationLabel* text file includes the name of the example. This name shows up as an entry in the link list in the upper frame of our example. Summing up this description we see, that after the user has written the *mfile*, only the input variables have to be set in PHP, all the rest is only writing text in an editor.

3 Application for transfer functions

After the definition of the whole framework – which is the main part of the work – we can go a step forward and show the structure and its benefits/restrictions within an application in the field of modeling and simulation in education.

In many cases it is easier for the students to understand system behavior of a class of problems using an example. Therefore a simple infusion model is chosen to explain the way from data measures up to a dynamic model structure.

3.1 Assignment of tasks

Backgrounds for the considerations in our model are physiological and metabolic processes

- which are observed over a time interval and for which measurements are available,
- which are influenced by factors from outside(e.g. medicine) and are reacting in a special way, and
- systems where the attitude can be focused on relatively isolated from the surrounding, this means that the reaction to an excitation is not depending on other physiological components.

In the center of interest is the time dependent coherence between input and model reaction. The model has to be able to

- describe the time dependent characteristics in a mathematical comprehensible form,
- solve the problem with good correlation between input and output,
- reflect the physiological and biological coherence qualitatively well,
- to define the individual quantitative reaction only by parameter finding,
- fit the measurements as good as possible, and
- predict the process reaction also with other conditions (e.g. changed excitation).

3.2 The model definition

The basic experimental structure of the model [4] is given by the following (virtual) assumptions:

An infusion with 500 ml of a substance over one hour is leading to an increase of the concentration of a well defined substance in the blood. Measurements *ci* showing the increment of the substance in the blood are available at all time points *ti* (every 15 minutes). Also important for the modeling point of view is what happens before the simulation starts. This is pointed out with two extra measurements half an hour and 15 minutes before the infusion starts. Figure 3 shows the basic system.

3.3 Data approach

The general question in the first iteration is, if there exist a mathematical model (formula) which can interpret the increase of concentration by a function which is at the measurement time close to the measured values. In the simplest case we are searching for an interpolation function $f(t)$ with

$$f(t_i) = c_i \quad \forall i \in \{1,...,n\} \tag{1}$$

One of the easiest ways to handle this is to make a polynomial interpolation. This is a classical data approach and therefore it is implemented as the first example in our MATLAB Webserver system. The modeling environment looks like the screenshot in Figure 4. The students have the possibility to test the model with different compensation polynoms and with the interpolation polynom of order twelve.

The students see the problems that occur when someone adds an extra measurement and that the interpolation is good in the data points, but outside it tends towards infinity. Also the problematic of the impossibility to handle other input functions and make a

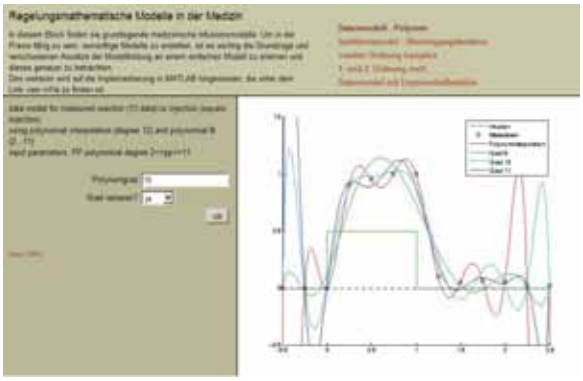


Figure 4. MATLAB Webserver example for a polynomial interpolation of the task defined in chapter 3.2

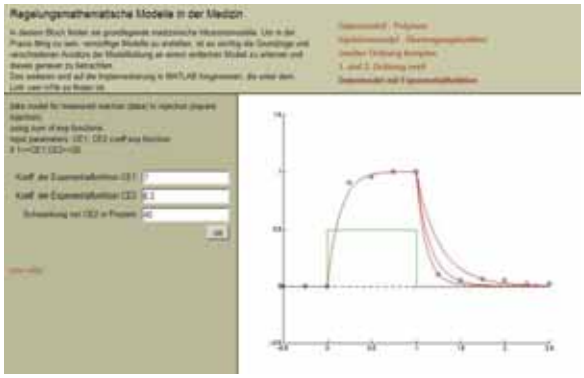


Figure 5. MATLAB Webserver implementation for the exponential model assumption

feasible model with the polynomial function can be shown. From modeling point of view this model is not appropriate because our system description acts with physiological impossible (negative) values.

The main problem of the function $f(t)$ is, that it does not depend on the input function. It tries to handle the output, whereby the measurements are the only used knowledge. The infusion is never taken into account.

To sum it up we can see that all of this interpolation methods (polynom interpolation, splines, …) just build a data model – the model does exclusively represent the measurements c_i.

3.4 Exponential function approach

The next step towards an universally valid model is established on basic considerations:

From physiological point of view we know, that the function should be relatively smooth over time. Furthermore we see coherence between the input function defined as a rectangular by the infusion input.

The coherence between infusion and the system reaction is described by exponential functions. The first time before the infusion starts to flow into the blood we assume the zero function as only valid solution. When the infusion starts, the output function tries to come up to a fixed level. In general this is explained by formular (2).

$$f(t) = a - b * e^{-c(t-t_0)} \quad a,b,c \in \Re \tag{2}$$

In the end of the infusion time the input function jumps to zero. Afterwards the measured data seems to follow this function. This can be modelled again with a negative exponential function.

Summing up these results, we get a function which describes the output behaviour of the physiological system with three general exponential functions, whereby the first part, the zero function, is trivial.

The implementation of this model assumption is shown in Figure 5. In this model the extra possibility of parameter variation for the third part can be done.

As can be seen, the model output fits the measurements *ci* very good. But what has to be taken into account is, that this model is no interpolation with exponential functions because the exponential functions used in the model act on different intervals.

A big disadvantage of the structure is that a change in the activation function (infusion) changes the whole reaction function. Nevertheless this model is not any more a simple data model because the input influences the output structure (defines when to switch between the different exponential functions).

3.5 Transfer function approach

As we want to use our MATLAB Webserver in education to teach the students to handle simple physiological examples, the solutions we give until now are not good enough. That is the reason why we make the next step towards a general system description.

Control theory leads to an appropriate model description between the input $u(t)$ and the output $x(t)$ in a structural-graphical, as well as in a mathematical-formulary way. The general structure is depicted in Figure 6.

To get a feeling for the work of a transfer function and how to handle a system in an adequate way with this control structure we implemented several examples on the MATLAB Webserver. The solutions for the first order activation function and a parameter combination for a second order system with real zero points of the denominator are shown in Figure 7.

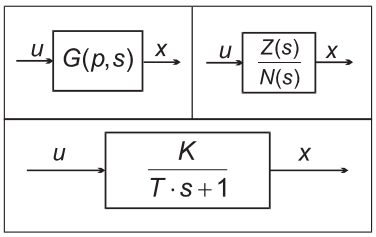


Figure 6. Structural-graphical representation of the relation; input $u(t)$/output $x(t)$, general model (top) and first order function as for the infusion model(down)

The PT1 – element is plotted as dashed red line, in the representation of the second order transfer function the turning point of the function is visualized as a blue star.

The students can test optimization algorithms and for all the examples they have the button *view mfile* in the lower left corner of the example definition and parameter frame. This leads to another benefit of our system. The people working with this interface learn about modelling and simulation and, furthermore, by using this simple example they can advance their knowledge about programming in the computer numeric/algebra package MATLAB, which is in a wide area more or less the standard.

The next class of functions to be focussed on are the complex second order transfer functions. In this context the characteristic parameters are damping and frequency. A composition of all three systems (first order, second order real and second order complex) for our infusion model is shown in Figure 8. The output functions are all coloured red, whereby the dashed line shows the PT2 – real model and the dotted one represents the first order transfer function.

For all of these transfer function models we see, that we are now (far) away from the poor data model. Our

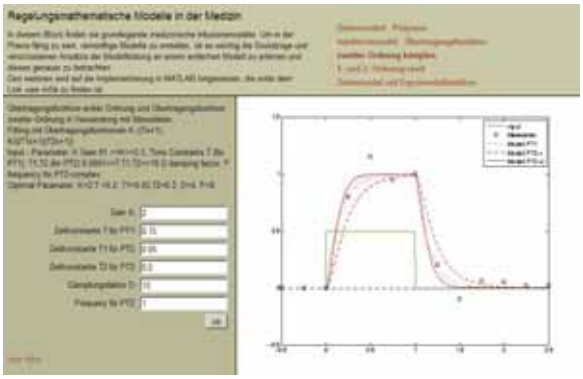


Figure 8. Three different transfer function approaches for the system from chapter 2.2

system can now handle the dynamic features of this model and the solution which is generated with this method is capable to react in an adequate way to input changes.

4 Fourier series application

Another often used application in techniques is the approximation of a periodic function by a sum of trigonometric functions. The theory behind is called Fourier series [5] and is a part of teaching in all technical branches of study at Vienna University of Technology. For this reason additional blocks with examples are implemented on the MATLAB Webserver.

One class of examples starts with a picture of a predefined example function. A sample is shown in Fig. 9.

The students first have to answer questions and then go on with the solution of the system. The focus hereby does not only lie on the mathematical solution of the task into an indefinite sum, but also to explain effects like Gibbs phenomena and to communicate the feeling how many elements are necessary to get an effective solution for further calculations.

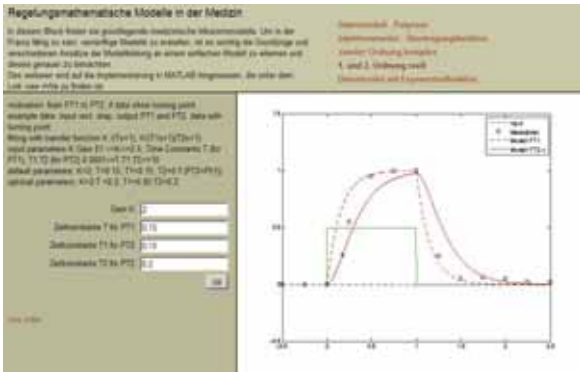


Figure 7. First order and second order with real zero points.

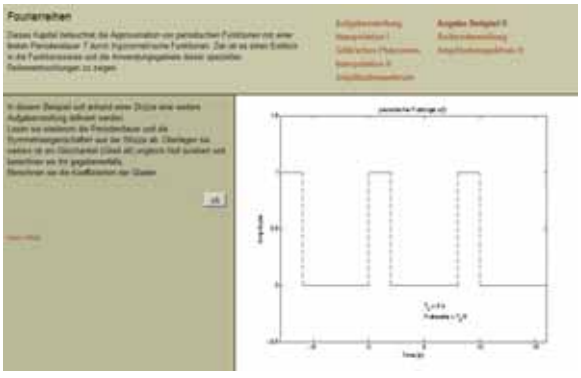


Figure 9. Setting of a standard task on our interface.

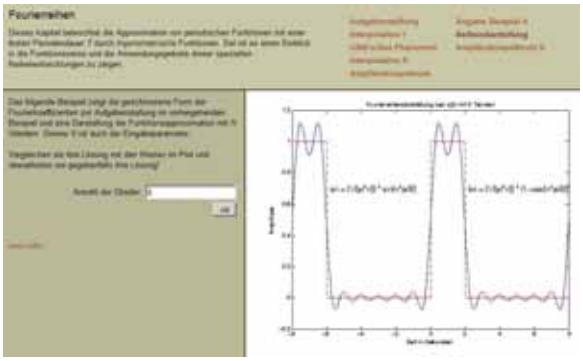


Figure 10. Solution with eight elements to the example defined like in Figure 9.

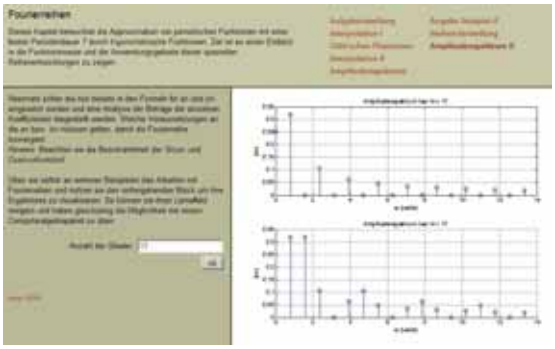


Figure 11. The upper plot shows the amplitudespectra of the a_n, which means the coefficients of the *cos* part. The lower plot depicts the coeff.of the *sin* elemts.

The approximation of the input function with a defined number of elements can be calculated. The solution for the function defined in Figure 9 is shown in Figure 10. Also the formulas for the coefficients of the *sinus* and *cosinus* function are returned.

The last questions discussed in the block are the frequency and absolute value spectra. This representation explains again the behavior of the absolute value for higher number of summing index.

For the discussed example the solution of the block is shown in Figure 11.

Within these two plots the students can see on of the necessary conditions for convergence of the sum of trigonometric functions: the coefficients have to be a null sequence. The coefficients of a convergent majorant series can be chosen to depict the knowledge graphical. Therefore the *symbolic math toolbox* is used again.

5 Outlook

As pointed out in sections 4 and 5 the MATLAB Webserver solution is compatible for use in modeling applications. Two main goals are of interest:

- Easy handling parameter/function-variation for given examples, and thus, supporting learning by doing, and
- the possibility to have a look at the source code in every step and make further work with it in the own MATLAB workspace.

The shown system is a tool which is not only used to present tasks in modeling and simulation, but also to impart the students in special fields of mathematics, where the graphical view combined with the formal solution helps a lot to understand the theory (e.g. Fourier Series, higher dimensional extreme value analysis, Taylor Polynom, Interpolation, …).

One of the most interesting next steps will be the implementation of a state flow simulation environment based on the MATLAB Webserver.

References

[1] http://www.mathworks.com/

[2] http://httpd.apache.org/

[3] M. Lubkowitz. Webseiten programmieren und gestalten – HTML, CSS, JavaScript, PHP, Perl, MySQL, SVG. Galileo Press, Bonn 2003, ISBN-10: 3898423131.

[4] F. Breitenecker, H .Ecker and I. Bausch-Gall. Simulation mit ACSL : eine Einführung in die Modellbildung, numerischen Methoden und Simulation . Braunschweig : Vieweg, 1993. - XI, 399 S

[5] H. Amann, J. Escher. Analysis II. Basel Bosten Berlin, Birkhäuser, 1999

Corresponding author: Günther Zauner,
dieDrahtwarenhandlung Simulation Services
Neustiftgasse 57-59, 1070 Vienna, Austria
guenther.zauner@drahtwarenhandlung.at

Received: July 15, 2008
Revised: September 19, 2008
Accepted: October 15, 2008

SNE 18/3-4, December 2008

SHORT NOTES

Traviando – A Trace Analyzer to Debug Simulation Models

Peter Kemper, College of William and Mary, *kemper@cs.wm.edu*
Carsten Tepper, Universität Dortmund, *carsten.tepper@udo.edu*

Verification and validation of simulation models are crucial steps to the success of a simulation project. Traces are a common and powerful mean to document the dynamic behavior of a model and are generated by most simulation engines. In this paper, we describe techniques and corresponding tool support that helps a modeller to gain insight in the dynamic behavior of complex simulation models based on trace analysis. We propose to visualize traces by message sequence charts. We use a common modal logic, namely the linear time logic (LTL), to identify states of interest and a pattern system to make specification of formulas more productive. The proposed techniques are implemented in Traviando, a stand alone tool with an open XML interface to import traces from various modeling environments, e.g., the APNN toolbox, the ProC/B toolset and Möbius.

Introduction

Traces that document the dynamic behavior of a system by a sequence of states and events are considered in many different settings. We focus on traces that are generated from a simulation model by discrete event simulation for verification, validation, debugging or animation purposes. Such a trace contains different pieces of information, information with respect to time and with respect to a causal order of events. While timing information is usually used for estimates of performance figures, the causal order and qualitative properties are of interest in the search for flaws in a verification and validation phase of a simulation project. Qualitative properties of dynamic behavior are usually described in a modal logic and among the many logics that are known, linear time logic (LTL) is a natural choice since a trace is a single and finite sequence of observations. Havelund et al. describe in [5] how to perform model checking for linear time logic (LTL) formulae on a trace by specially adjusted algorithms.

For practical applications, model checking has a number of challenges. First, a human computer interface is required that is easy to use and that helps a practitioner to be confident on the property he or she specified in a formal notation like LTL. For instance, a formula like $G((q \wedge \neg r \wedge Fr) \rightarrow (\neg pUr)$ with atomic propositions p, q, and r is not crystal clear to a non-expert. Dwyer et al. [3] recognized regularities in the use of logic formulas for model checking and organized those into a pattern system. Second, given a model and a formula, efficient algorithms are necessary to decide whether a model fulfills that formula or not. Much research has gone into that problem and impressive results have been obtained by so-called symbolic model checking techniques. However, in case of traces, model checking remains rather simple from a practical point of view since there is only a single and finite sequence that has to be analyzed. Third, once a result has been obtained, it must be represented or visualized in a way that maximizes the insight it may give to a modeller. In this paper, we contribute to those challenges in the following way. We describe a new tool that visualizes traces by a variant of MSCs. It provides a pattern system with a formula editor for LTL to describe properties of interest. For a given set of formulas and propositions, it evaluates which states fulfill which formulas and visualizes those properties in the MSC representation. In addition, we provide a number of visualization features that help a user to retain an overview on the dynamic activity [7], and to identify fragments of traces that are critical for common performance measures in process oriented simulation models [6]. The tool we present has an open XML interface for traces and has been applied with several modelling frameworks including the ProC/B toolset [1], the APNN toolbox [2] and the multi-paradigm multi-solution framework Möbius [4].

We are interested in the analysis of finite traces. Formally, we consider a trace as a sequence $\sigma = s_0 e_1 s_1 \ldots e_n s_n$ of states $s_0, \ldots, s_n \in S$ and events $e_0, \ldots, e_n \in S$ over some (finites or infinite) sets S, E for an arbitrary but fixed $n \in \mathbb{N}$. For elements of S,

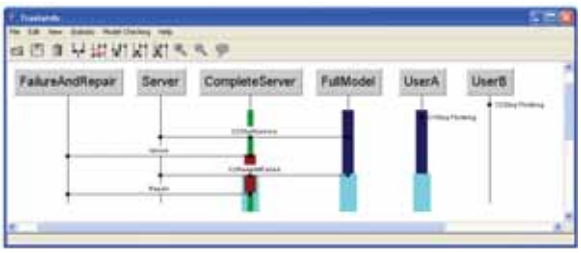


Figure 1: MSC of Example Trace

we assume an equivalence relation denoted by "=".

We start from a totally ordered sequence σ and represent it by an MSC. Fig. 1 shows an example trace of six processes with local and synchronized events, which was generated from a Möbius model for a server system with two customer classes A and B where the server is subject to failure and repair. The model is compositional and each submodel matches a process in the MSC. A process is visualized by a vertical line with its name on top. Local events are represented by small dots, synchronized events by horizontal lines, and a states $s = G(v)$ is visualized in an additional window for a selected event v. The thickness and colorings of the vertical lines (timelines) reflect results of model checking a particular formula. The vertical order of events (top-down) follows the order of events in a visualized sequence σ.

In the following, we focus on the qualitative analysis of a trace, for the visualization of a trace by an MSC we refer to [KT05b, KT05a]. Section 1 is devoted to LTL model checking of traces and a pattern system for the specification of formulas. In Section 2 we introduce our new tool for trace visualization, its architecture and main functionality.

1 LTL model checking

Given a sequence σ, it is interesting to identify what properties of the system are present or absent. Modal logics of various kinds have been developed to analyze dynamic systems. In our case, a single and finite sequence would be considered a trivial case in concurrency theory and a linear time logic like LTL seems a reasonable choice for our purpose.

We assume a set of atomic propositions AP, such that an $a \in AP : S \to \{tt, ff\}$ evaluates states $s \in S$ to true or false or in other terms $s \models a$ iff $a(s) = tt$. Since we allow for a relation = on S, AP has to be consistent with =, i.e., we require for $s, s' \in S$ that if $s = s'$ then $a(s) = a(s')$ for all $a \in AP$. This requirement is trivial for equality; it becomes relevant if some form of bisimulation is considered as "=".

LTL considers the behavior of a system as a set of sequences and has formulas of the form Af where f is a path formula. An LTL path formula can be of two kinds. If $p \in AP$, then p is a path formula. If f and g are path formulas, then $\neg f$, $f \vee g$, $f \wedge g$, Xf, Ff, Gf, fUg, and fRg are path formulas. Implication and equivalence follow from the boolean operators in the usual manner, $f \to g \equiv \neg f \vee g$ and $f \leftrightarrow g \equiv (f \to g) \wedge (g \to f)$.

We mildly adjust the semantics of LTL for the case of a single finite sequence that is usually a prefix of a potentially much longer or infinite sequence. Hence for operators that refer to future behavior, we can be optimistic or pessimistic which we formally address by an additional artificial pair $e_{n+1}s_{n+1}$ that is attached to σ and we assume that atomic propositions are defined for s_{n+1} as well. For a desired property $\Psi \in AP$, we say we are optimistic if we define $s_{n+1} \models \Psi$ and pessimistic if $s_{n+1} \not\models \Psi$. With that extension we define the semantics of LTL operators for a sequence $s_0,...,s_n$ and states s_i, $i = 0,...,n+1$:

1. $s_i \models \neg f$ iff $s_i \not\models f$
2. $s_i \models f \vee g$ iff $s_i \models f$ or $s_i \models g$
3. $s_i \models f \wedge g$ iff $s_i \models f$ and $s_i \models g$
4. $s_i \models Xf$ iff $s_{i+1} \models f$, $i \le n$
5. $s_i \models Ff$ iff there exists a $j, i \le j \le n+1$ such that $s_j \models f$
6. $s_i \models Gf$ iff for all $j, i \le j \le n+1$ holds $s_j \models f$
7. $s_i \models fUg$ iff there exists a $k, i \le k \le n+1$ such that $s_k \models g$ and for all $j, i \le j \le k$ holds $s_j \models f$
8. $s_i \models fRg$ iff for all $k, i \le k \le n+1$, for all $j, i \le j \le k$ holds $s_j \not\models f$ then $s_k \models g$.

Finally, we define:

1. $s_{n+1} \models Xf$ iff $s_{n+1} \models f$
2. $s_{n+1} \models Ff$ iff $s_{n+1} \models f$
3. $s_{n+1} \models Gf$ iff $s_{n+1} \models f$
4. $s_{n+1} \models fUg$ iff $s_{n+1} \models g$
5. $s_{n+1} \models fRg$ iff $s_{n+1} \models f \wedge g$

Since we consider the special case of a single and finite σ, an algorithmic treatment of LTL path formulas is straightforward. Atomic propositions are evaluated for individual states. X, F, G, U, R operators are evaluated backwards starting at state s_{n+1}. Note that by definition the evaluation at s_{n+1} for X, F, G, U, R is immediate. With known results at s_{n+1}, we can make use of the following four properties:

1. $s_i \models Ff$ iff $s_i \models f$ or $s_{i+1} \models Ff$
2. $s_i \models Gf$ iff $s_i \models f$ or $s_{i+1} \models Gf$
3. $s_i \models fUg$ iff $s_i \models g$ or ($s_i \models f$ and $s_{i+1} \models fUg$)
4. $s_i \models fRg$ iff $s_i \models f \wedge g$ or ($s_i \models g$ and)

From an implementation point of view, an evaluation can be performed with the help of two arrays in the length of all subformulas and by sweeping through σ in a backward manner.

Patterns

Dwyer et al. [3] derived a pattern system from a substantial empirical study on the use of modal logics in verification and validation of systems. They analyzed about 500 example specifications taken from at least 35 different application areas and experienced that very few kinds of formulas occur in practice. They observed that patterns *response*, *universality*, and *absence*, all with scope *global*, cover about 80% of the considered cases. This motivated us to integrate that taxonomy into an editor for LTL formulas. Fig. 2 shows the editor window in the pattern system of Traviando with 3 representations of formula f of pattern absence with scope between. Formula $G((q \wedge \neg r \wedge Fr) \rightarrow (\neg pUr)$ has atomic propositions p = *User_A*, q = *server_avail*, and r = *server_failure* that are formulated by equalities /inequalities on arithmetic expressions with state variables, e. g., *server_avail* is defined as *avail* = 1 where avail is a state variable of submodel *CompleteServer*. p is defined as *C1WaitsForServer* + *C1WaitsForUser* + *C1Thinking* = 1. So state variables like *C1Thinking* are used to make state information $s \in S$ accessible to a specification of properties in atomic propositions. The upper right window in Fig. 2 shows the formula in a graphical, tree-type representation that is used to create and refine formulas, to assign atomic propositions to the leafs of the structure, and to select colors for its graphical representation. The lower right window gives the common formal representation for users that are more familiar with that type of notation. Finally, the lower left window describes the formula in technical prose that is initially derived from predefined phrases associated with patterns and that can subsequently be worked on by a user with further comments. Note that those representations describe the same information, but should complement each other for human understanding. The tree-type representation is what is used as input for the model checking algorithms.

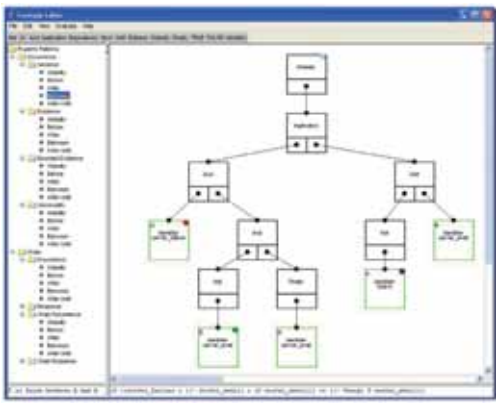


Figure 2: Graphical formula editor with pattern system

Visualization

Figure 1 presents the visualization of the formula $G((q \wedge \neg r \wedge Fr) \rightarrow (\neg pUr)$ as discussed before and as specified in the formula editor in Figure 2. We can assign a color to each atomic proposition (leafs of the formula tree) and subformulas and use those colors to highlight in the MSC representation where a formula or subformula holds. The model checking algorithm computes additional information for each node in the tree of a formula. That information includes the total number of states that fulfill a (sub)formula as well as the position of first and last occurrence of those states for ease of navigation. Buttons are provided that make the visualizer scroll to the first or last occurrence. This supports the intended usage of model checking: the modeler can specify properties of interest and the model checker guides the modeler to that particular fragment of the trace.

2 Tool

Traviando supports the visualization and analysis of traces of interacting processes. The visualization is based on MSCs and its particular purpose is to make information accessible to a human user, information that is otherwise hidden in a large trace file. Traviando imports sequences in an open XML format that consists of two parts, namely a prefix and the sequence of events that constitutes the trace. The prefix contains definitions for processes, events, their type and association with processes, state variables and more. The prefix helps to keep the sequence of events concise in its description and also allows for some preprocessing and consistency check for the trace based on the given structural information. The sequence of events in the second part of a trace can be enhanced in many ways by additional information, for instance by time stamps and by information on changes to state variables.

Traviando is implemented in Java and is able to work on traces of different kinds and from different sources. It operates on traces generated with the APNN toolbox [2]. Those traces result from Petri net models. Traviando operates also on traces generated with the ProC/B toolset [1] which follows the common process interaction approach for simulation modeling. ProC/B models are structured hierarchically based on function calls.

The most recent advance of Traviando is towards traces generated by the multi-formalism multi-solution framework Möbius [4]. Models are structured in a hierarchical manner such that a decomposition into processes according to the composition of atomic and composed models is a natural choice. Fig. 3 shows a Möbius model of a server with failure and repair for a performability study. We generated a trace from this model that serves as a running example for the visualization in Fig. 1 and the formula in Figure 2.

Traviando's new visualization features include LTL model checking of traces that is supported by a pattern system. Atomic propositions are built by arithmetic expressions on state variables and equalities and inequalities. As illustrated in Fig. 2, we combine three descriptions of a formula to make LTL model checking more accessible to a user. The upper right window shows the formula in a graphical, tree-type representation the lower left window describes the formula in technical prose, and the lower right window gives the common formal representation. Note that those representations describe the same information, but should complement each other for human understanding.

The graphical, tree-type representation with one node per logical operator and atomic propositions as leafs is used to derive formulas by refinement. Atomic propositions can be selected by mouse-clicks from a menu that is derived from a currently defined set of atomic propositions. Atomic propositions are defined in an additional editor window. Colors are associated with nodes in the formula editor to define the coloring and highlighting used for the visualization of the model checking results on a MSC, for instance the colors selected in Fig. 2 match those in Fig. 1. Novel trace reduction techniques, which are not presented here for lack of space, help to separate repetitive from progressing parts of a trace and can substantially cut down on the number of events considered for trace analysis.

More information on Traviando can be found at `http://www.cs.wm.edu/kemper/Traviando`.

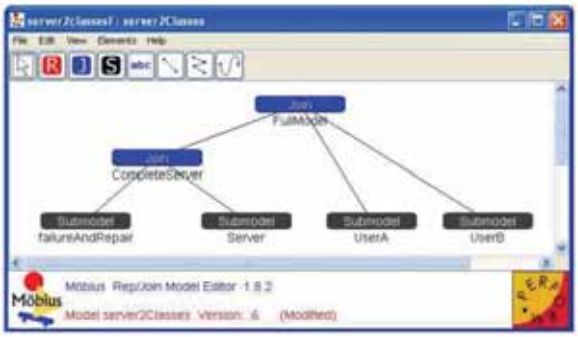


Figure 3: Composed Model

Acknowledgements

We would like to thank W.H. Sanders, Tod Courtney and Michael McQuinn for supporting us with an appropriate XML trace output of Möbius and several students who contributed to Traviando in various ways.

References

[1] F. Bause, et al. *The ProC/B toolset for the modelling and analysis of process chains*. In Proc. TOOLS, LNCS 2324, Springer, pages 51–70, 2002.

[2] Falko Bause, Peter Buchholz, and Peter Kemper. *A toolbox for functional and quantitative analysis of DEDS*. In Proc. TOOLS'98, LNCS 1469, Springer, 1998.

[3] Matthew B. Dwyer, et al. *Patterns in property specifications for finite-state verification*. In ICSE, pages 411–420, 1999.

[4] Daniel D. Deavours, et al. *The Möbius framework and its implementation*. IEEE Trans. Software Eng., 28(10):956–969, 2002.

[5] Klaus Havelund and Grigore Rosu. *Synthesizing monitors for safety properties*. In Proc. TACAS'02, LNCS 2280, pages 342–356, Springer, 2002.

[6] P. Kemper and C.Tepper. *Trace based analysis of process interaction models*. In Proc. of the 2005 Winter Simulation Conference, pages 427–436, 2005.

[7] P. Kemper and C. Tepper. *Visualizing the dynamic behavior of ProC/B models*. In Proc. Simulation und Visualisierung, pages 63–74, SCS, 005.

Corresponding author: Peter Kemper,
Department of Computer Science,
College of William and Mary,
Williamsburg, Virginia 23187-8795, USA
kemper@cs.wm.edu

Accepted: EOOLT 2008, June 2008
Received: October 5, 2008
Accepted: October 27, 2008

SNE 18/3-4, December 2008

Multi-Paradigm Language Engineering and Equation-Based Object-Oriented Languages

Hans Vangheluwe, McGill University, Montréal, Canada, *Hans.Vangheluwe@mcgill.ca*

Models are invariably used in Engineering (for design) and Science (for analysis) to precisely describe structure as well as behaviour of systems. Models may have components described in different formalisms, and may span different levels of abstraction. In addition, models are frequently transformed into domains/formalisms where certain questions can be easily answered. We introduce the term "multiparadigm modelling" to denote the interplay between multi-abstraction modelling, multi-formalism modelling and the modelling of model transformations.

The foundations of multi-paradigm modelling will be presented. It will be shown how all aspects of multi-paradigm modelling can be explicitly (meta-)modeled enabling the efficient synthesis of (possibly domain-specific) multi-paradigm (visual) modelling environments. We have implemented our ideas in the tool AToM (A Tool for Multi-formalism and Meta Modelling) [3].

Over the last decade, Equation-based Object-Oriented Languages (EOOLs) have proven to bring modelling closer to the problem domain, away from the details of numerical simulation of models. Thanks to Object-Oriented structuring and encapsulation constructs, meaningful exchange and re-use of models is greatly enhanced.

Different directions of future research, combining multiparadigm modelling concepts and techniques will be explored:

1. meta-modelling and model transformation for domain-specific modelling as a layer on top of EOOLs;
2. on the one hand, the use of Triple Graph Grammars (TGGs) to declaratively specify consistency relationships between different models (views). On the other hand, the use of EOOLs to complement Triple Graph Grammars (TGGs) in an attempt to come up with a fully "declarative" description of consistency between models to support co-evolution of models;
3. the use of graph transformation languages describing structural change to modularly "weave in" variable structure into non-dynamic-structure modelling languages.

1 Multi-paradigm modeling

In this section, the foundations of Multi-Paradigm Modelling (MPM) are presented starting from the notion of a *modelling language*. This leads quite naturally to the concept of *meta-modelling* as well as to the explicit modelling of *model transformations*.

Models are an *abstraction* of reality. The structure and behaviour of systems we wish to analyze or design can be represented by models. These models, at various *levels of abstraction*, are always described in some *formalism* or *modelling language*. To "model" modelling languages and ultimately synthesize (visual) modelling environments for those languages, we will break down a modelling language into its basic constituents [4]. The two main aspects of a model are its syntax (how it is represented) on the one hand and its semantics (what it means) on the other hand.

The syntax of modelling languages is traditionally partitioned into *concrete syntax* and *abstract syntax*. In textual languages for example, the concrete syntax is made up of sequences of *characters* taken from an *alphabet*. These characters are typically grouped into *words* or *tokens*. Certain sequences of words or *sentences* are considered valid (i.e., belong to the language). The (possibly infinite) *set* of all valid sentences is said to make up the language.

For practical reasons, models are often stripped of irrelevant concrete syntax information during syntax checking. This results in an "abstract" representation which captures the "essence" of the model. This is called the *abstract syntax*. Obviously, a single abstract syntax may be represented using multiple concrete syntaxes. In programming language compilers, abstract syntax of models (due to the nature of programs) is typically represented in *Abstract Syntax Trees* (ASTs). In the context of general modelling, where models are often graph-like, this representation can be generalized to *Abstract Syntax Graphs* (ASGs).

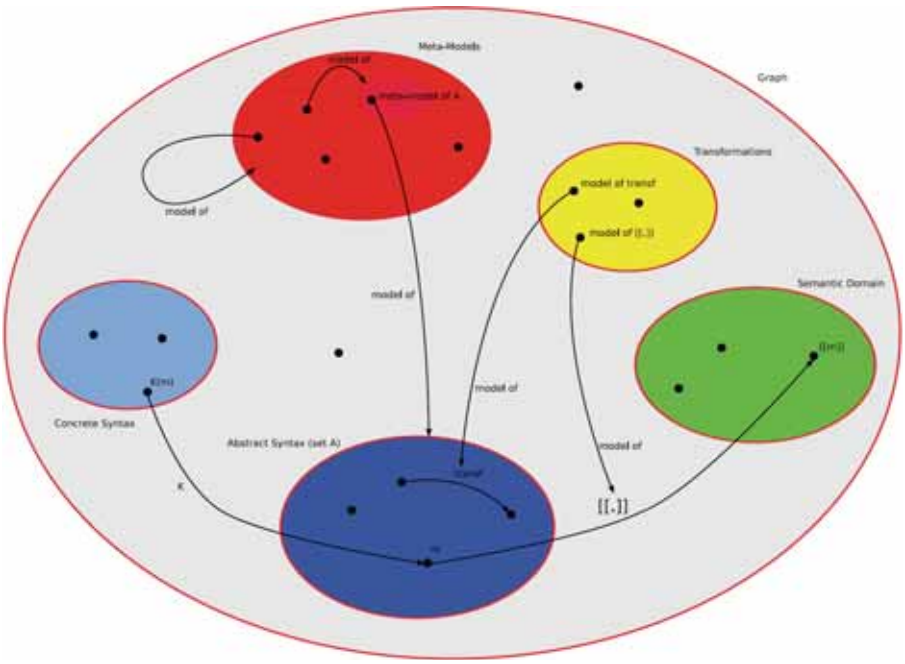


Figure 1. Modelling languages as sets

Once the syntactic correctness of a model has been established, its meaning must be specified. This meaning must be *unique* and *precise*. Meaning can be expressed by specifying a *semantic mapping function* which maps every model in a language onto an element in a *semantic domain*. For example, the meaning of a Causal Block Diagram (e.g., a Simulink diagram) can be specified by mapping onto an Ordinary Differential Equation. For practical reasons, semantic mapping is usually applied to the abstract rather than to the concrete syntax of amodel. Note that the semantic domain is a modelling language in its own right which needs to be properly modelled (and so on, recursively). In practice, the semantic mapping function maps abstract syntax onto abstract syntax.

To continue the introduction of meta-modelling and model transformation concepts, languages will explictly be represented as (possibly infinite) sets as shown in Figure 1. In the figure, insideness denotes the sub-set relationship. The dots represent model which are elements of the encompassing set(s).

As one can always, at some level of abstraction, represent a model as a graph structure, all models are shown as elements of the set of all graphs Graph. Though this restriction is not necessary, it is commonly used as it allows for the design, implementation and bootstrapping of (meta)modelling environments.As such, anymodelling language becomes a (possibly infinite) set of graphs. In the bottom centre of Figure 1 is the abstract syntax set A. It is a set of models stripped of their concrete syntax.

1.1 Meta-models

Meta-modelling is a heavily over-used term. Here, we will use it to denote the explicit description (in the form of a finite model in an appropriate meta-modelling language) of the *Abstract Syntax* set. Often, meta-modelling also covers a model of the concrete syntax. Semantics is however not covered. In the figure, the *Abstract Syntax* set is described by means of its *meta-model*. On the one hand, a meta-model can be used to *check* whether a general model (a graph) *belongs to* the *Abstract Syntax* set. On the other hand, one could, at least in principle, use a meta-model to *generate* all elements of the language.

1.2 Concrete syntax

A model in the Abstract Syntax set (see Figure 1) needs at least one concrete syntax. This implies that a concrete syntax mapping function κ is needed. κ maps an abstract syntax graph onto a concrete syntax model. Such a model could be textual (e.g., an element of the set of all Strings), or visual (e.g., an element of the set of all the 2D vector drawings). Note that the set of concrete models can be modelled in its own right.

1.3 Meaning

Finally, a model *m* in the Abstract Syntax set (see Figure 1) needs a unique and precise meaning. As previously discussed, this is achieved by providing a Semantic Domain and a semantic mapping function *M*. Rule-based Graph Transformation formalisms are often used to specify semantic mapping functions in particular and model transformations in general. Complex behaviour can be expressed very intuitively with a few graphical rules. Furthermore, Graph Grammar models can be analyzed and executed.

1.4 Formalism transformation

In an attempt to minimize accidental complexity [2], modellers often transform a model in one formalism to model into another formalism, retaining salient properties.

2 Domain-specific modelling

Domain- and formalism-specificmodelling have the potential to greatly improve productivity as they [5].

- match the user's mental model of the problem domain;
- maximally constrain the user (to the problem at hand, through the checking of domain constraints) making the language easier to learn and avoiding modelling errors "by construction";
- separate the domain-expert's work from analysis and transformation expert's work.
- are able to exploit features inherent to a specific domain or formalism. This will for example enable specific analysis techniques or the synthesis of efficient (simulation) code exploiting features of the specific domain.

The time required to construct domain/formalism-specific modelling and simulation environments can however be prohibitive. Thus, rather than using such specific environments, generic environments are typically used. Those are necessarily a compromise. The above language engineering techniques allow for rapid development of domain-specific (visual) modelling environments with little effort if mapping onto a semantic domain (such as an EOOL) is done.

3 Consistency/Co-evolution of model views

In the development of complex systems, multiple views on the system-to-be-built are often used. These views typically consist of models in different formalisms. Different views usually pertain to various partial aspects of the overall system. In a multi-view approach, individual views are (mostly) less complex than a single model describing all aspects of the system. As such, multi-view modelling, like modular, hierarchical modelling, simplifies model development. Most importantly, it becomes possible for individual experts on different aspects of a design to work in isolation on individual views without being encumbered with other aspects. These individual experts can work mostly *concurrently*, thereby considerably speeding up the development process. This realization was the core of Concurrent Engineering. This approach does however have a cost associated with it. As individual view models evolve, inconsistencies between different views are often introduced. Ensuring consistency between different views requires periodic concerted efforts from the model designers involved. In general, the detection of inconsistencies and recovering from them is a tedious, error-prone and manual process. Automated techniques can alleviate the problem. Here, we focus on a representative sub-set of the problem: consistency between geometric (Computer-Aided Design – CAD) models of a mechanical system, and the corresponding dynamics simulation models. We have selected two particular but representative modelling tools: SolidEdge for geometric modelling [8], and Modelica [1] for dynamics and control simulation

The core geometric entities are Assemblies. SolidEdge Assemblies are composed of other Assemblies, Parts and Relationships. Relationships describe mechanical constraints between geometric features of two distinct parts, and there can be many such relationships between parts.

On the dynamics side, to represent an equivalent structure in Modelica, we have a model which can be hierarchically composed of other models, bodies, relationships and geometric features. This last type of model element is introduced to have a counterpart to represent the geometric information which is intrinsic to a SolidEdge part.

Associations (correspondences) that must exist between SolidEdge and Modelica models are shown in a meta model triple in Figure 2. Note that this model is *declarative* as it does not specify how and what to modify to correct possible inconsistencies. Triple Graph Grammar theory [7] introduced by Sch¨ur provides a procedure for automatically deriving operational update transformations (in the form of triple graph rewrite rules) from the declarative meta-model [6]. If either the geometry or dynamics models change, the association model can be used to determine what has been added or deleted from either side.

On the other hand, the use of EOOLs to complement Triple Graph Grammars (TGGs) in an attempt to come up with a fully "declarative" description of consistency between models to support co-evolution of models.

4 Modelling of variable structure

Various formalisms have been devised to describe the discontinuous change of the structure of systems. The

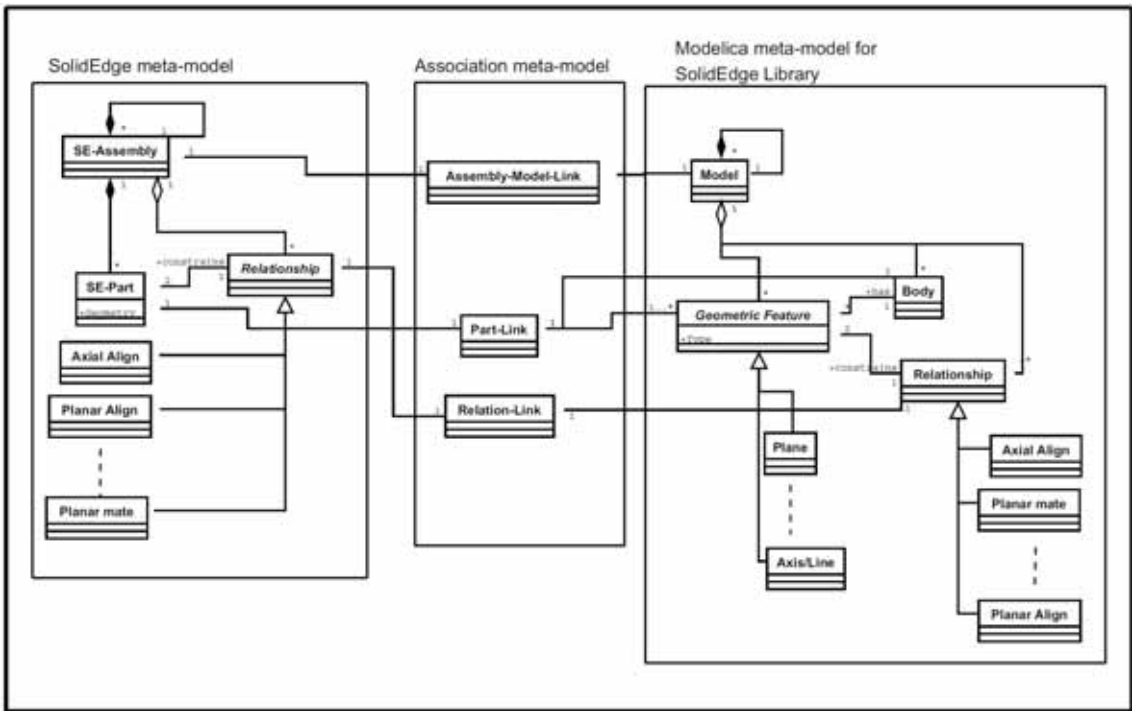


Figure 2. Relating SolidEdge and Modelica models.

rulebased description of graph transformations is ideal to elegantly describe structural change. A rule's left-hand-side describes the conditions under which a state-event occurs. In modelling languages for hybrid systems, crossing conditions on variable values are used to specify *when* a stateevent occurs. The handling of a state-event may introduce discontinuous changes in the value of variables. The rulebased approach adds detection of particular object configurations to the low-level variable-value conditions. A rule's right-hand-side describes the handling of the state-event. This may not only include variable value changes, but also creation/destruction of entities and their interconnections. A promising avenue for future research is the modular "weaving in" of rule-based variable structure description language constructs into non-dynamic-structure modelling languages such as EOOLs.

References

[1] Modelica Association. *A unified object-oriented language for physical systems modeling*. Modelica homepage: www.modelica.org, since 1997.

[2] F. P. Brooks. *No silver bullet: Essence and accidents of software engineering*. Computer, 20(4):10–19, 1987.

[3] J. de Lara, H. Vangheluwe. *AToM3: A tool for multi-formalism and meta-modelling*. In European Joint Conference on Theory And Practice of Software (ETAPS), Fundamental Approaches to Software Engineering (FASE), Lecture Notes in Computer Science 2306, pages 174 – 188. Springer, April 2002. Grenoble, France.

[4] D. Harel, B. Rumpe. *Modeling languages: Syntax, semantics and all that stuff, part i: The basic stuff*. Technical report, Jerusalem, Israel, 2000.

[5] S. Kelly, J.-P. Tolvanen. *Domain-Specific Modeling: Enabling Full Code Generation*. Wiley, 2008.

[6] A. Königs. *Model Transformation with Triple Graph Grammars*. In Model Transformations in Practice Satellite Workshop of MODELS 2005, Montego Bay, Jamaica, 2005.

[7] A. Schürr. *Specification of Graph Translators with Triple Graph Grammars*. In G. Tinhofer, editor, WG'94 20th Int. Workshop on Graph-Theoretic Concepts in Computer Science, volume 903 of Lecture Notes in Computer Science (LNCS), pages 151–163, Heidelberg, 1994. Springer Verlag.

[8] SolidEdge. www.solidedge.com.

Corresponding author: Hans Vangheluwe,
School of Computer Science
McGill University, Montréal, Canada,
Hans.Vangheluwe@mcgill.ca

Accepted EOOLT 2008, June 2008
Received: October 5, 2008
Accepted: October 27, 2008

True Computer Simulation of Real Parallel Processes is Not Possible

A Proof by the Five Dining Philosophers – and Their Children

Hans Fuss, Bonn

1 Notions

The term 'parallel' is used in many different ways, thus causing many misunderstandings. We know parallel lines from geometry, which do not intersect because they keep their distance constant forever. We might call the rails of a railway track parallel as well, though they are not straight, because they keep their distance constant as well; different tracks meet at railway junctions. The notion parallel can be extended even further, to the different tracks between junctions, though the geometrical distance is not constant (e. g. a bypass). They may offer two different, but equally good, 'parallel' ways of travelling to the passenger. In this paper, we are not primarily interested in the static parallelism of how things *are* (parallelism in space, as of lines or tracks), but in the dynamic parallelism, i.e. how things and systems *act*; parallelism in time, as of trains and passengers, where the synchronizations points are stations where passengers may change from on train to another.

We are in the lucky position that we can use ideas which have been developed in the Theory of Petri Nets. There it is shown that the relation 'simultaneous' can be decomposed into 'concurrent', which denotes that two events are causally *neither before nor after*, i.e. neither reason nor result of each other, which means independent of each other, and 'coincident', which denotes *at the same time at the same spot*.

The progression of two real processes is certainly not synchronized to that extent that all events happen stepwise at the same time at the same spot; this will be so only at their synchronization points. Hence, parallel real processes are of the type 'concurrent'.

Parallel programs behave in the same way as trains do on different, but parallel tracks: they may run on different processors, maybe at different speeds, until they get synchronized at certain points, where they communicate with each other for data transfer reasons.

In the concurrent branches of the processes, parallel subsystems are independent of each other, subsequently having a partial autonomy. Within this autonomy, they can act and decide deliberately, explicitly uncontrolled by the others.

Note: **concurrency is a similarity relation**, not an equivalence, which in particular means: concurrency is not transitive.

Another interesting similarity relation in this context is 'indistinguishable', because it is not transitive as well, in contrast to: 'equal', which is an equivalency relation.

The term 'true' shall indicate a simulation which is a fairly bi-directional mapping. One direction in modelling always is simple: (abstractions of) real *objects* are mapped into model objects, and the abstraction of the real structure is mapped into a model structure.

In most cases, it is not too cumbersome to find realistic abstractions of the *objects* and to make correct mappings of them between the real and the model system. It is harder to map the *structure* of the systems. And it is hardest to map what causes the changes in the system: the *causal relations* between objects, cases, and situations.

But the crucial part is to find a mapping that works both ways: from reality to the model – and back from the model to reality. The quality of the simulation depends on the degree of correctness of this mapping.

The term 'possible' in the headline is to be seen in conjunction with the term 'true' in the field of natural parallelism. We notice that if one is willing to make concessions in truth, reliability or correctness of the simulation results, the possibilities will increase.

We finally – depending on the level of detail – state impossibility because of the special property of concurrent systems: the course of affairs, its *process*, is in general not unique; it depends on the decision of its independent sub-systems. An observed process is, in general, just *one* of several possible processes.

It is, of course, possible to construct parallel program systems of all kinds, including parallel simulation programs, which are absolutely correct. But if we consider the most general cases, *genuine* parallel *real* processes, we run into trouble. A central question when simulating parallel systems is: how much of the

concurrency (parallelism) of the original system maps to a simulation program – and vice versa.

2 The Problem

On of the oldest and most well-known examples of independent parallel processes is that of E. W. Dijkstra's Five Dining Philosophers. This problem was also used for simulation program comparison in EuroSim's technical newsletter *Simulation News Europe*, and even twice it adorned the title page of that journal, e.g. in Nov. 1991.

We look at the problem in its simplest form: Five philosophers sit around a table, either thinking or eating. There shall be enough food on the table and in order to eat each philosopher shall need two forks (or better: chopsticks). But, between any two philosophers there is just on form, which they can pick up, use them, and then return to their original position. That means, a philosopher can eat only if both his neighbours don't.

There shall be no communication among them about the forks.

There have been hundreds of solutions to this problem in lectures and literature, which show how to control the situation, often under special restrictions (e.g. with simultaneous grip to and return of the forks), or with special aims, e.g. no starvation of the philosophers, keeping the distribution of time or food fair, under side conditions of coalitions, etc. – but do we want solutions?

3 The Arguments

For better understanding of all arguments, let us put ourselves into the position of one philosopher, say P_1, with P_2 as our left neighbour, and P_5 as our right one.

As long as we want to eat and the two forks are not available – we can't. While we eat, our neighbours can't. If there are both forks available, but we either don't take notice of them or we prefer thinking to eating, the forks stay on the table for possible use by us or by our neighbour(s). It is entirely at our own discretion, whether we continue thinking or start eating; and if we grab the forks, it is again entirely up to us how long we use them. Now let us change the role and assume that we are simulated philosopher, i.e. a computer program with the ability to be in the states either 'thinking' or 'eating', and connected to places of memory which represent the presence or absence of forks.

Consider the case 'eating'. In our previous role as the real P_1, it was up to our own discretion how long to eat. But now, as we are the simulated philosopher, somebody else, perhaps a random numer generator, tells us how long to stay in the state 'eating' (maybe longer than we would have liked in reality), or maybe he takes our forks away, even if we would like to eat still more. If the forks are available, and actually we would have preferred to continue thinking, we have no free will again: maybe he starts to force-feed us. Maybe he goes round and distributes forks (and hence: food) to $P_1, P_2, ...P_5, P_1$ etc. Or maybe he consults somebody else, maybe another random number generator, to decide how to distribute the forks.

There is nothing left for us (i.e. the simulated philosopher P_1) to decide anymore. Maybe we have been equipped with our individual random number generator, which tells us and the main program loop how to proceed. Everything is decided for us by so-called 'decision makers'. And the same happens to all our colleagues $P_2 ... P_5$.

4 Changing the Original

In the simulation, the course of the game is not determined any more by the 5 philosophers, but by a 6th party. The 5 philosophers, who are the acting subjects in reality, become passive objects in the simulated game, with the simulation program as the only acting body. Hence, the game has changed – from a 5-person-game to a 6-person-game, as it seems at first sight – but as a matter of fact, it has changed to a 1-person-game. The change is from conflicting and causal relations in the game to well-ordered and statistical relations in the simulated one – which is a most drastic change of the structure that one can imagine.

If we make a structural true reversal mapping of the simulated system back to reality, it would lead to a quite different system than to the 5 philosophers. The nearest original in reality might perhaps be a kindergarten of 5 philosopher's children, where the nurse assigns time for eating and playing philosophy, to each of the 5 children as *she* likes.

If two original systems, entirely different in the causal structure (here: the table of the five philosophers, and its kindergarten), lead to the same simulation – the here must be something wrong with the principles of simulation.

The most important feature of the original philosophers' game in reality, namely the concurrency among them, has gone lost by the traditional transformation into a simulation program; the game became a clockwork ruled by the advance of the instruction counter of the main loop or the random number generator.

The original task, to simulation a concurrent process, has been changed into a different, not so hard problem, into one with a different causal structure: a system without concurrency.

But we know from many other disciplines that the use of false assumptions will lead to uncertain, mostly unusable results.

The reason why the swap to a different original system is widely and in general not noticed, is because the results of simulated process could well be a true mapping of the original process – but not necessarily will be *all* results of it.

A very fascinating simulation as well would be a weather forecast, not according to meteorological rules, but along the lines of tarot. The predicted weather situation might *look like* a serious forecast, the predicted values may even become true in the forecast period – nevertheless, nobody could accept such a procedure as a reliable, 'true' method.

5 Conclusion

What do we expect from a 'true' simulation? It should describe the original system closely enough. In particular, we would like, to a certain extent of precision, to conclude from the simulation back to the original system (having a bisimulation): we expect when we alter parameters in the model system M yielding certain simulation results – that correspondingly the same alteration in the real system R would yield similar real results (similar to the simulated ones). Of course we would accept that reality and simulation drift apart the longer the simulation runs. But what in particular about the simulation of parallel – or more precisely: concurrent systems?

If we take a technical device, e.g. a robot, with all its motors working in parallel – we might get pretty close to our expectations. This is so, because the robot has been constructed the way that there is little slack of freedom left. All concurrency will be synchronized closely enough according to the will of the constructing engineer.

A mailing system (among computers) is different, and the Five Dining Philosophers claim that a simulation program does not represent their behaviour correctly.

The situation is even worse: in a concurrent system, it is not possible to predict *one* step. If we watch the game of the 5 philosophers from the initial state, we couldn't tell who will be the first person to start eating, and when.

But the simulation program tells, it pretends to know.

A possibly more correct simulation of the 5 philosophers problem would be, to use 5 different, independent processors and to connect them to the places 'forks', and let every processor run according to the same rules as the philosophers thought and acted. Then we see the essence of the mapping problem: how can we map correctly the way of how a philosopher come to the conclusion whether and when to think or to eat.

To express certain problems, one needs appropriate languages. There still is a necessity to develop further simulation languages, which can express concurrency.

Further reading

There is a long list of references (2 pages) in this author's contribution to:

Kampe/Zeitz: *Simulationstechnik*, Proc. 9. Sympos. ASIM Jahrestagung Oct. 1994 Stuttgart; pp. 355ff., Reihe *Fortschritte in der Simulationstechnik*, Band 9, Vieweg-Verlag, 1994

The underlying theory to this paper is C.A.Petri's net theory. The newsletter of the Petri Net community is a quaterly publication of the GI (Gesellschaft f. Informatik). Proceedings of the annual international conferences appear in LNCS, *Lecture Notes in Computer Science*, Springer-Verlag.

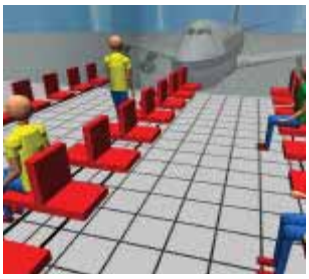
An electronic mailing net is run by K. Jensen, Aarhus University; a bibliography of Petri Net publications (with about 5000 entries, status 1995) is maintained there as well.

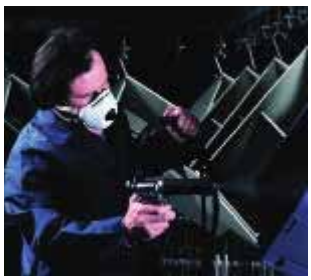
Corresponding author: Hans Fuss, Bonn
hans.fuss@t-online.de

Received: April 24, 2008
Accepted: August 10, 2008, 2008

Don’t Speculate... Simulate!

You are operating in a complex and unpredictable business environment where balancing capacity defines your performance and profits. You deal with a lot of uncertainty and have to rework your planning continuously. Then Enterprise Dynamics simulation and control software is the ultimate powerful decision and planning tool to balance your resources. Enterprise Dynamics gives you an accurate image of your business processes and insight into utilization and yield of your resources, effectiveness, and logistical performance.





Enterprise Dynamics®, built by the people that brought Taylor to the market, is today's leading simulation software. Our software combines the powerful Enterprise Dynamics® Engine and many ready-to-use building blocks grouped into Enterprise Dynamic Suites for specific lines of business. Enterprise Dynamics® meets the latest standards in dynamic engineering and can be integrated into your existing system.

Incontrol Enterprise Dynamics
· The Netherlands **+31 346.552500**
· Germany, **+49 203.3051.1170**
info@EnterpriseDynamics.com

ENTERPRISE DYNAMICS
Simulation Software®

Download your free evaluation version at:

www.EnterpriseDynamics.com

Comparing ODE Modelling and Electrical Network Modelling for ARGESIM Benchmark C3 'Class-E Amplifier' using SimulationX

Günther Zauner, Gemma Ferdinand Kaunang, Vienna University of Technology, Austria

Simulator: SimulationX from ITI is a modelling and simulation software based on the Modelica standard, for valuation of the interaction of components of multi domain technical systems. It is the universal CAE tool for modelling, simulation and analyzing of physical effects – with ready-to-use model libraries, like for example 3D multibody systems, power transmission, hydraulics, thermodynamics, electrics, electrical drives, magnetics as well as controls – postprocessing with powerful features included.

Model: The basic class-E power amplifier was introduced by N.O. Sokal and A.D. Sokal in their classic paper from 1975. It is a switching-mode amplifier that operates with zero voltage and zero slope across the switch at switch turn-off. The equations describing the circuit are the state-equations where inductor currents and capacitor voltages are chosen as system variables. By using the Kirchhoff voltage and current laws we get the following differential equations:

$$dx1/dt = (-x2 + VDC)/L1$$
$$dx2/dt = (x1 - x2/R(t) - x3)/C2$$
$$dx3/dt = (x2 - RL*x3 - x4)/L3$$
$$dx4/dt = x3/C4$$

The aim was to implement these equations into simulationX structure. Therefore two different ways have been chosen to model these equations. The first solution is realized by using a block diagram model. The differential equations above were built by block integrator, add/subtract and gain, also defining a function block as division for $x2/R(t)$ and defining a signal generator for the constant voltage VDC. The time dependent resistor is built by a type designer block using Modelica code. The graphical representation of the model is given by Figure 1. The code for the time dependent resistor is as follows:

```
1  algorithm
2     if (0<=t_red) and (t_red<TRF)
3       then Rt:=(5e-2)+k*t_red;
4     elseif (TRF<=t_red) and (t_red<(5e-6))
5       then Rt:=5e+6;
6     elseif (5e-6<=t_red) and (t_red<5e-6+TRF)
7       then Rt:=(5e+6)-k*(t_red-(5e-6));
8     elseif 5e-6+TRF<=t_red and t_red<10e-6
9       then Rt:=5e-2;
10    else
11      Rt:=-5;
12    end if;
13 equation
14    t_red = mod(time,10E-6); k =5e+6-5e-2/TRF;
```

Listing 1. Time Dependent Resistor algorithm

The second solution uses an electrical model representation. Designing the model by using basic resistor, inductor, capacitor, ground and constant voltage as VDC. The time dependent resistor was built by type designer using Modelica code. The model of the system is shown in Figure 2.

The code for the time dependent resistor looks as follows:

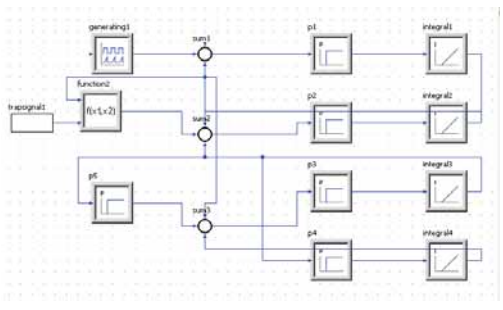


Figure 1. Model of the System as classical block implementation with SimulationX

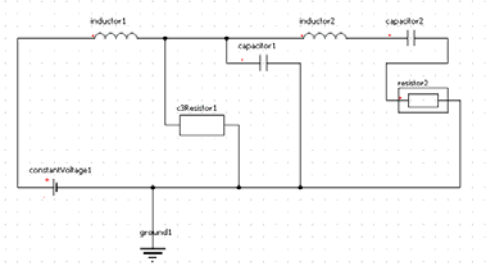


Figure 2. Electrical model implementation

```
1  equation
2     v = pin1.v - pin2.v; v = R*i;
3     pin1.i = i; pin2.i = -i; tred =
         mod(time, 10E-6); k=((5e+6)-(5e-2))/TRF;
4     if (0<=tred) and (tred<TRF)
5       then R = (5e-2) + k*tred;
6     elseif (TRF<=tred) and (tred<(5e-6)
7       then R = 5e+6;
8     elseif (5e-6<=tred) and (tred<5e-6+TRF)
9       then R = (5e+6) - k*(tred - (5e-6));
10    elseif (5e-6+TRF<=tred) and (tred<10e-6)
11      then R = 5e-2;
12      else R = 0;
13    end if;
```

Listing 1. Equations including the code for $R(t)$

Setting up the simulation parameters and simulation process are done in *simulation tab*. The results of simulation were shown in the model view section.

A-Task: The eigenvalues of the system are calculated for the two states when $R(t) = 5m\Omega$ (on) and when $R(t) = 5M\Omega$ (off), respectively, by simulating the whole system first and then going to the tab *analysis* (natural frequencies and mode shapes). In simulationX the eigenvalues are calculated automatically. The calculated eigenvalues are shown in Table 1 for all solutions.

B-Task: In task B the system is simulated by setting the BDF- method as integration solver, $1a$sec as minimal step size, $1p$sec as minimal output step size, $10n$sec as absolute tolerance, $0...100\mu$sec as simulation time interval and $1e-8$ as relative tolerance. Using the intial value zero for $x1, x2, x3$ and $x4$, the result for the variable current at time dependent resistor $IR(t)$ and output voltage VRL is depicted in Figure 3. It took 1.2528s to simulate task B for the first solution and 0.8461s for the second solution, using a standard home PC.

C-Task: The parameter of TRF is varied as follows: 1fsec, 10psec, 1nsec, 100nsec. The initial value for task C is equal to the final solution from task B. The time interval is $0\ldots9\mu$sec. As result, the phase plane curve $dx3/dt = VL3$ as a function of $x3 = IL3$ is shown in Figure 4. The four simulations are executed separately.

Eigenvalues $R(t)$ OFF	Eigenvalues $R(t)$ ON
-54708+1,0408E+5i	-1,11731E+9
-54708-1,0408E+5i	-625,78
-58228+5,3275E+5i	-1,1304E+5 +6,5835E+5i
-58228+5,3275E+5i	-1,1304E+5 -6,5835E+5i

Table 1. Eigenvalues of $R(t)$

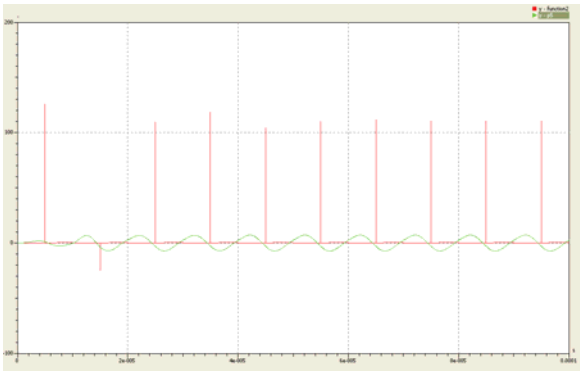


Figure 3. Time Curve $IR(t)$ and VRL.

Resumé: SimulationX is a simulation tool that offers all necessary features to model the system in a convenient way. Users can build their own model blocks based on Modelica language, a free common used standard for physical modelling.

Eigenvalue calculation is provided automatically for every simulation process. The postprocessing offers time curve plots as well as phase plots. Reinitialization of the startparameters of $\vec{x}$ with the final value from task b for task c was done manually.

Corresponding author: Günther Zauner
Vienna University of Technology
Department of Analysis and Scientific Computing
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
guenther.zauner@drahtwarenhandlung.at

Received: March 3, 2008
Revised: August 5, 2008
Revised: October 5, 2008
Accepted: October 18, 2008

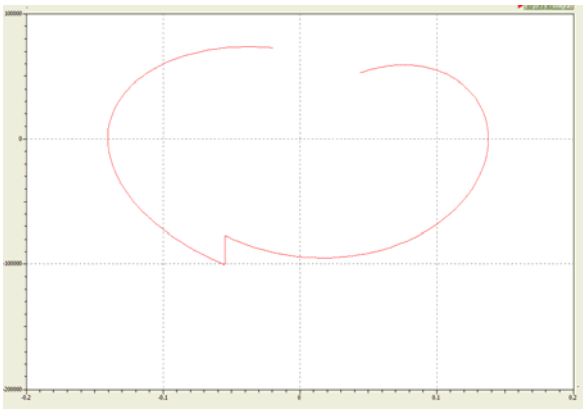


Figure 4. Phase plane curve $VL3$ function $IL3$

A Java-programmed Object-oriented Solution to ARGESIM Benchmark C16 'Restaurant Business Dynamics'

Michael Gyimesi, Max Arends, Gregor Pridun, Johnannes Zweng, Vienna Univ. of Technology,
mgyimesi@osiris.tuwien.ac.at

Simulator: We used Java as simulation environment for this comparison. Java is a powerful object-oriented programming language. Java's main domain nowadays is the programming of sophisticated net applications, but it does not come with dedicated support for simulation purposes. Our solution consists of two parts, which interact with each other. The core is the java simulation model, which does all the calculations. For output and data analysis a graphical frontend was developed as a JAVA Applet. Therefore our application can be run in any web browser, which supports Java.

Model: The implementation of the model uses a schedule with equidistant time steps, representing weeks in real life. At each time step the defined operations are performed. By repeatedly performing these weekly steps, we simulate the progress over time.

We identified and modelled the following entities in the problem domain:

- *Cities:* Five cities are placed initially at unchangeable predefined locations.
- *Persons:* Persons are randomly distributed around the cities at start-up and do not move. Every person holds a dynamic list of all restaurants in range.
- *Restaurants:* Restaurants are distributed initially in a grid pattern, and their number and locations change dynamically during the simulation run. Each restaurant offers a method "`eatAtThisWonderfulRestaurant()`" and holds internal variables `weekRevenue`, `weekProfit` and `totalProfit` for storing the according values.
- *Cells:* The space is organized in square cells, where each cell holds information about the number of persons and restaurants within the cell and offer a method for retrieving the ratio of people to restaurants within the cell (`getPeopleRestaurantRatio()`).
- *CellMatrix:* A CellMatrix object was designed for organizing the cell objects, and performs tasks like recalculating the restaurant density and calculating the best cell for the next restaurant based on the value *k*.
- *ModelParameters:* All variable input values of the model have been encapsulated in a parameters object. This way it's very easy to reset the model with different settings.

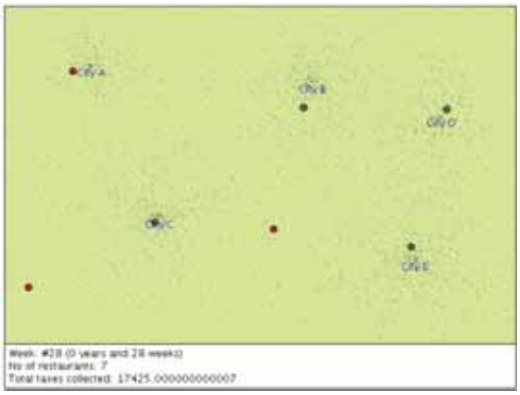


Figure 1: Visual representation oft he model

All these entities are represented as Java objects.

All the simulation calculations are performed within this model and are complete separated from the representation layer. This way it's possible to perform all the simulation steps without any visual output and this way we solved the tasks below.

The same model also serves as the core of the Java applet which visualizes the simulation runs. The applet triggers the calculation of the next step regularly and visualizes the values it gets from the model object after every step.

By using Java Applet technology the visualization is independent of operating systems and simply can be started over the web in every browser as long as Java is installed. Figure 1 displays the graphical representation of the model area with restaurants as red or green circles and people as dots on the plane.

A-Task: Simulation – Average Treatment Time: We wrote some small Java classes with a main-method which make use of our model and perform

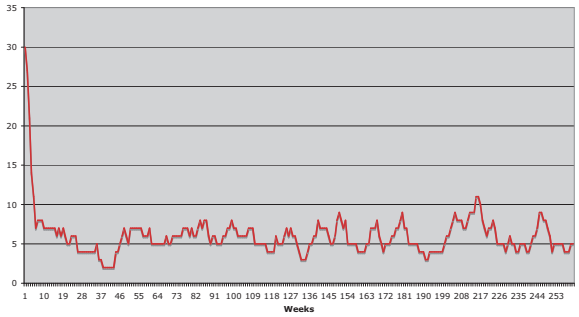


Figure 2: Number of Restaurants in a 5 year period

several simulation runs. Since Java doesn't have advanced features to analyze and produce visualisations (like diagrams) we plotted the results of the simulation runs into a CSV-File and worked with the data in a spreadsheet environment.

Figure 2 illustrates a typical development of the restaurant's numbers in a 5 year period.

While it is notable, that there is a significant drop in the total number of restaurants, no absolute stabilisation in the total number of restaurants after a certain time is observed.

We changed the output in order to get the average limit value of number of Restaurants after the 5th year. Calculation's average results in 6,34 restaurants after 5 years.

B-Task: Maximum tax income For this task we varied the parameter of the tax rate from 1 to 100 percent. For each tax rate we did five different calculation rates and averaged them in order to stabilise our results. This is due to the fact, that each simulation is different and has a different number of restaurants for each tax rate.

After the start-up the tax-return increases up to a maximum at 33%, after that the tax income drops very fast to a low level. Very high tax rates don't bring a lot of tax income.

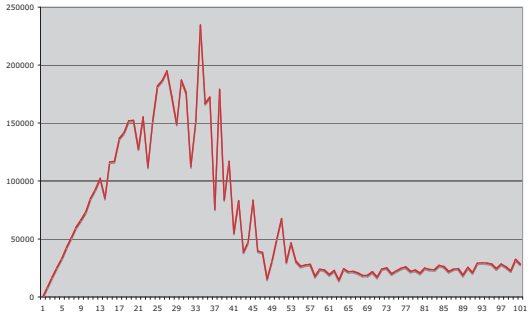


Figure 3: Maximization of Tax income

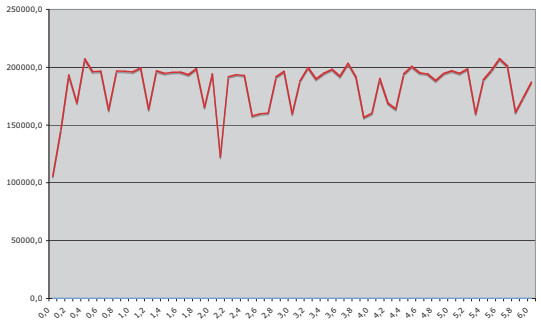


Figure 4: Restaurant Revenue Analysis

33% is a pretty good tax rate, since 1/3 of the profit has to be taxed. This still leaves a high amount of restaurants on the market and is not too high to bankrupt small restaurants.

From a government point of view it is obvious that high tax-rates make no sense, since the restaurant owners are unhappy and there are not many restaurants left. This would also lead to a high unemployment rate.

C-Task: Restaurant Revenue Analysis In order to show how the revenue is influenced by the density-parameter k, we calculated the revenue-value five times and took the average. The results can be seen in Figure 4.

No significant correlation between the density parameter k and the revenue of restaurants is displayed.

Resumé: This solution of the benchmark is based on a pure JAVA environment without any specific simulation engine. Despite the lack of a discrete event scheduler, the advance of time is realized by an equidistant discretization of the timeline. The object oriented programming paradigm of JAVA fits perfectly to the agent based problem allowing to develop every type of agents as a different class with its specific methods. The possibility to provide visualization very fast via JAVA applet technology gives an additional asset.

Corresponding author: Michael Gyimesi
Vienna University of Technology
Department of Analysis and Scientific Computing
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
mgyimesi@osiris.tuwien.ac.at

Received: June 29, 2008
Revised: September 10, 2008; October 12, 2008
Accepted: October 20, 2008

A Programmed Solution to ARGESIM Benchmark C16 'Restaurant Business Dynamics' using GNU R

Florian Judex, Vienna University of Technology, Austria, *florian.judex+E101@tuwien.ac.at*

Simulator: GNU R is an open source statistical software package, aimed to replace the popular S software. Due to a high level vector oriented scripting language akin to MathScript and the ability to handle huge amounts of data without slowing down, it can readily be adapted for Simulation Purposes. Like MATLAB, it distinguishes between the line oriented command shell, scripting, and windows for graphic output, as shown in Figure 1 below.

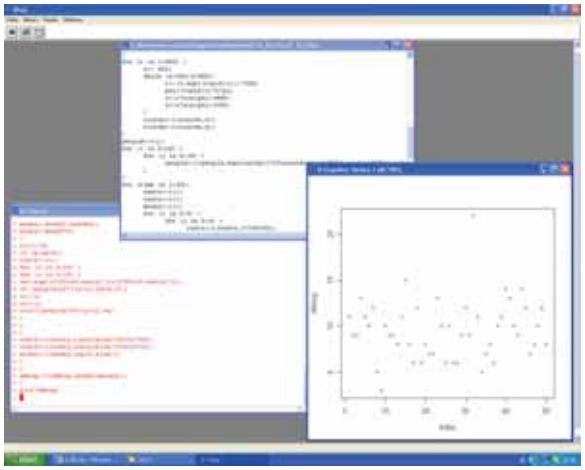


Figure 1. The Gnu R Interface.

Model: Although C16 deaks with a discrete model, there is no need for event handling, as all decision are only based on the current state of the system, and there are no time delays which would eventually result in the cancellation of events.

Therefore the activity scanning approach is feasible.

Customers and restaurants are modelled as list containing, adding time in case of the customers and money in case of the restaurants. For the placement of the customers, the inverse cumulative density function of the given triangular distribution, which is not implemented in R, was derived by hand. The random values where then simulated using this formula and a [0,1] continuous uniform probability distribution.

To simplify computation, the continuous uniform probability distribution on [0,8] of the intervals between consummations was changed to a discrete uniform probability distribution on [1,7], preserving the mean of 4 and slightly lowering the standard deviation from 2.3 to 2.

This allows an easy advancement of time, using nested loops for weeks and days, e.g.

```
for (week in (1:years*52))
    for (day in (1:7) {
        time <- time - 1;
    }}
```

Using the vector oriented notation in R, many computations or evaluations of logical constructs can be done simultaneously, further speeding up the simulation. For example, determining the indices of the persons using a restaurant on a certain day can be determined by the command

```
active <- (1:3000) [time==0]
    for (counter in active) {...}
```

which creates a vector using those elements of the vector (1,2, … 3000) satisfying the condition the time counter of corresponding person equals zero, and uses this list to loop through these entries. A similar construct can be used to determine which restaurant is visited.

On the other hand, entries can also be removed from vectors, by slightly changing the arguments. The lines

```
restx<-restx[-positions];
resty<-resty[-positions];
money<-money[-positions];
```

will remove of (x,y) coordinates and the money counters for those restaurants identified in the vector called 'positions', and adjusts the length of the vector accordingly.

It should be duly noted at this point that R defines a matrix as an array with dimensions, which can be converted back to an array at any time. Just using one index when accessing a matrix will implicitly revert it into an array immediately, leading to propagating and hard-to-find errors in the script. Therefore one is on the save side using separate array like for x and y above when there is no real need for a matrix.

Finally, a combination command can be used to concatenate vectors. This used to place new restaurants. For example

```
money<-c(money,rep(0,plus));
```

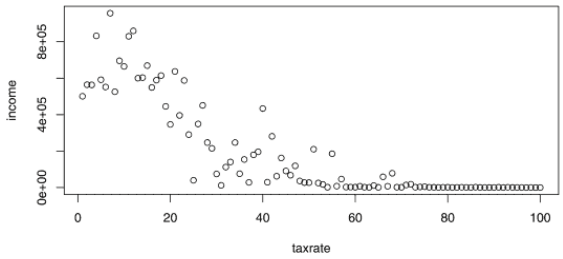


Figure 2. Overall Tax Income

adds as many new money counters as specified by the variable 'plus' behind the last existing one.

Using those tools, the whole model is implemented in less then hundred lines of code.

A-Task: Simulation of the one, five and ten year period is done easily by varying the variable for lentght of the outer loop. Plotting is done using one of the various routines offered by R for the graphical representation of data. Likewise, encapsulating the whole simulation routine into another loop allows 50 runs of the simulation, and one can use R statistical capabilities for analysis. This showed a mean value of 7.84 restaurants after 5 years, with a range between 3 and 14 surviving restaurants.

B-Task: If the tax rate climbs above 33 percent, the expected income from the two smaller towns gets too small to support a single restaurant. Therefore those 20 percent of the population nearly stop contributing to the overall tax income. In the simulation, the same starting Population was used to do five five-year runs for every tax rate between 1 and 100 percent. The average income per year depending on the tax rate is shown on Figure with a maximum at 7 percent, and the highest values between 4 and 12 percent, where the big cities are able to support 3 respectively 4 restaurants with a suitable safety margin, which cover most of their population.

C-Task: To keep track of the income of new restaurants lead new parameter vector has to be introduced. It is set to one in the first thirty restaurants, and zero in the newer ones. Therefore, just by computing the sum of this vector, the program keeps track of the old restaurants still existing. Together with the practice of adding the new restaurants after the last one already existing, the offset of the first new restaurant is calculated.

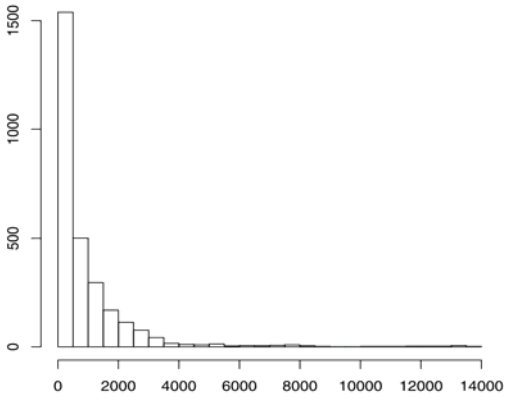


Figure 3. :Histogram of incomes

To create comparable data, the same starting population was used for each parameter, and the simulation time was set to 10 years, to even out the effects of the starting population.

The (rounded) summary in Table 1 shows the exponent of the distance has little effect on the overall expectation of income. This is because all new restaurants will start in the same square as the density is not updated after each restaurant, competing with each other for the customers available there, and quiet often closing down after one week. This leads to the data shown in the histogram of the income of the restaurants in their entire lifetime for d=6, with a mean of over 900, but a median of about 400, which equals less then two weeks of income about the profit threshold. A few survive, but most close down after just one week.

Resumé: As R is a statistical software with the aim of handling huge amounts of data, even quite a lot of operations on vectors of the length 3000 do not slow down the programm significantly compared to a test run with fewer people. As the timescale was quite large, simplification of the probability function did not alter the results in an unpredictable way.

Corresponding author: Florian Judex
Vienna University of Technology
Department of Analysis and Scientific Computing
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
florian.judex+E101@tuwien.ac.at

Received: January 20, 2007
Revised: February 25, 2007
Accepted: February 26, 2007

Direct Modelling and Programming of ODE – and CA – Results for ARGESIM Benchmark C17 'SIR-Type Epidemic' in Modelica/Dymola

Stefan Emrich, Ch. Simon, Th. Kadiofsky, J. Vilsecker, Vienna Univ. of Technology, Austria
Stefan.emrich@*tuwien.ac.at*

Simulator: Modelica is a free object-oriented modeling AND programming language designed for modeling large and complex systems in control, mechanics, electronics, etc and supports a-causal physical modeling. But Modelica allows also 'classic' programming with scalers, vectors and matrices. And in addition to direct text mode programming, graphical modeling is accessible through several front ends.

Simulations have been performed in Dymola, a 'full' Modelica simulator (in principle, Dymola has initiated the development of Modelica as 'standard' for physical modeling.

Model: Using the Modelica text mode, the given system of ODEs ([1]) can be solved numerically without difficulties by defining the model with the following equations:

```
equation
  der(S)= -r*S*I;
  der(I)= r*S*I-a*I;
  der(R) = a*I;
end SIR_epicemic;
```

Also the LGCA can be implemented with the Modelica language: The data structure used to represent the cells is a three dimensional array of integers. The first two dimensions contain the cells of the lattice and the third one represents all possible directions of particles within the cell, which count either 4, in the case of a HPP model, or 6 in the case of a FHP model. Each entry of this matrix has one of the following values: 0 empty, 1 susceptible, 2 infected, 3 recovered. Every time step consists of two phases: intercellular and inner-cellular motion. The former is realized by exchanging corresponding blocks of the matrix being aware of the given cell structure and the periodic boundary conditions, as shown in the following extract:

```
dim1:=size(World,1); dim2:=size(World,2);

// left-right-motion
World_new[:,1:dim2-1,2]:= World[:,2:dim2,5];
World_new[:,2:dim2,5]  := World[:,1:dim2-1,2];
World_new[:,dim2,2]    := World[:,1,5];
World_new[:,1,5]       := World[:,dim2,2];

// diagonal motion
for i in 1:dim1-1 loop
  // odd lines
  if mod(i,2) == 1 then
    World_new[i+1,:,6] := World[i,:,3];
    World_new[i,:,3]   := World[i+1,:,6];
    World_new[i+1,1:dim2-1,1]
                       := World[i,2:dim2,4];
    World_new[i+1,dim2,1]:= World[i,1,4];
    World_new[i,2:dim2,4]:=
                  World[i+1,1:dim2-1,1];
    World_new[i,1,4]     := World[i+1,dim2,1];
    // even lines
  else ..........
    ...
  end if; end for;
```

The inner-cellular motion is realised cell-wise. Concerning the FHP model there can be either a 3-particle head-on collision or a 2-particle head-on collision.

```
temp := World[i,j,:];
// 2-particle head-on collision
if ((temp[1]<>0) and (temp[4]<>0)
    and (temp[2]==0) and (temp[3]==0)
    and (temp[5]==0) and (temp[6]==0)) then ...
end if;
```

Recovery of infected individuals and infection of susceptible individuals happen according to specific rates. But a susceptible person only becomes infected with a certain probability, if there is at least one infected person in the same cell.

```
if I and S then
  k := 1;
  while PosS[k] <> 0 loop
    z := FHP.rand();
    if z < Inf_Rate*100 then
      World[i,j,PosS[k]]:=2;
    end if;
    k := k+1;
  end while;
end if;
```

In the last step of the inner-cellular activities, the particles move on through the cell along their directions, which means that the incoming particles become outgoing ones.

```
temp2 := World[:,:,1];
World_new[:,:,1] := World[:,:,4];
World_new[:,:,4] := temp2;
...
```

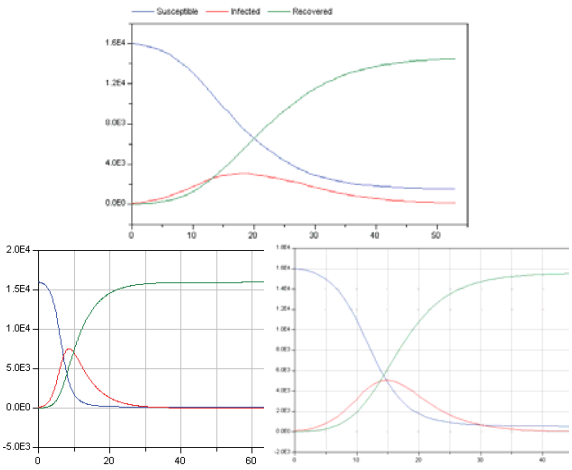


Figure 1. top: ODE, left: HPP, right: FHP solution

A-Task: Comparing the three plots in Figure 1 shows that the solutions of the HPP-model and the FHP-model deliver the same long-term-behavior. The main difference of the two models is the maximum number of infected individuals. The fact that this number is much higher in the FHP-model can be explained as follows: In the FHP-model there are more particles in one cell and hence more individuals can possibly be infected at once. The solution of the ODEs differs more significantly. Especially the velocity of propagation is much higher in the continuous model.

B-Task: To simulate the given vaccination strategies ([1]) in the FHP model, the initial distribution of the population was modified. Infected individuals were grouped together in one half of the domain and 4000 of the susceptible individuals were assumed to be vaccinated (recovered).

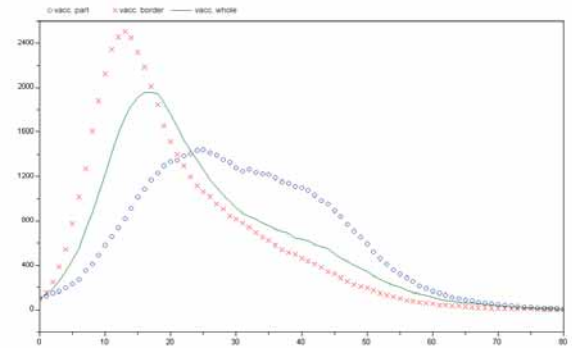


Figure 2: infected individuals – differtent strategies red: border vac., green: global vac., blue: partial vac.

Comparing the different strategies by opposing the number of infected individuals shows that vaccination in the 'epidemic' half of the domain yields a lower maximum peak of the number of infected people but at the same time the epidemic lasts longer. Border vaccination leads to a rapid breakout of the disease but also to a shorter lifetime. This fact can be explained by the initially unrestricted spread of the epidemic. The spatial isolation yields a fast decay of the disease.

C-Task: In contrast to the first tasks the whole population is redistributed uniformly after every time step. For this reason the infection process was modified here: In cells, which contain infected individuals, a certain fraction of the susceptible individuals becomes infected.

Figure 3 shows a comparison of the number of infected individuals in the ODE- (blue), DE- (red) and FHP-approach (green). The behaviour is not only qualitatively but also quantitatively similar. In contrast to task a, the velocity of propagation is nearly identical in all three approaches.

Resumé: The perhaps astonishing aspect of this solution is the fact, that Modelica can be used as technical programming language like MATLAB. The CA- models and update programs were directly translated from a MATLAB model / MATLAB algorithm. There, structures for vectors, matrices and their manipulation are very similar.

The Modelica-programmed algorithm is executed in a Dymola - ALGORITHM section. But some tricky modifications in the code were necessary to get correct results, because of ambiguities in interpretation of discrete equations.

Corresponding author: Stefan Emrich,
Vienna Univ. of Technology
Inst. for Real Estate Development and
Inst. f. Analysis and Scientific Computing /
Wiedner Hauptstrasse 8-10. 1040, Vienna
stefan.emrich@tuwien.ac.at

Received: January 17, 2008
Revised: June 20, 2008
Revised: November 5, 2008
Accepted: November 10, 2008

SNE News Section

Data & Quick Info

Contents

Simulation News Europe is the official journal of EUROSIM and sent to most members of the EUROSIM Societies as part of the membership benefits. Furthermore **SNE** is distributed to other societies and to individuals active in the area of modelling and simulation. SNE is registered with ISSN 1015-8685. Circulation of printed version is 3000 copies.

SNE at Web recent issues of SNE are also available via internet at `www.argesim.org`. Members of EUROSIM Societies may have access to the SNE Archive. This special *News Section* compiles date from EUROSIM and EUROSIM societies: addresses, weblinks, officers of societies with function and email. This *EUROSIM Data & Quick Info* is published regularly in SNE issues (except for special issues).

If you have any information, announcement, etc. you want to see published, please contact a member of the editorial board in your country or sne@argesim.org.

Editorial Information/Impressum - see front cover

Information EUROSIM

EUROSIM Federation of European Simulation Societies

General Information. *EUROSIM*, the Federation of European Simulation Societies, was set up in 1989. The purpose of EUROSIM is to provide a European forum for regional and national simulation societies to promote the advancement of modelling and simulation in industry, research, and development.

→ www.eurosim.info

Member Societies. EUROSIM members may be national simulation societies and regional or international societies and groups dealing with modelling and simulation. At present EUROSIM has eleven full members and three observer members:

ASIM	Arbeitsgemeinschaft Simulation *Austria, Germany, Switzerland*
CROSSIM	Croatian Society for Simulation Modeling *Croatia*
CSSS	Czech and Slovak Simulation Society *Czech Republic, Slovak Republic*
DBSS	Dutch Benelux Simulation Society *Belgium, Netherlands*
FRANCOSIM	Société Francophone de Simulation *Belgium, France*
HSS	Hungarian Simulation Society *Hungary*
ISCS	Italian Society for Computer Simulation *Italy*
PSCS	Polish Society for Computer Simulation *Poland*
SIMS	Simulation Society of Scandinavia *Denmark, Finland, Norway, Sweden*
SLOSIM	Slovenian Simulation Society *Slovenia*
UKSIM	United Kingdom Simulation Society *UK, Ireland*
CEA-SMSG	Spanish Modelling and Simulation Group *Spain, Observer Member*
LSS	Latvian Simulation Society *Latvia, Observer Member*
ROMSIM	Romanian Society for Modelling and Simulation, *Romania, Observer Member*

Contact addresses, weblinks and officers of the societies may be found in the information part of the societies.

EUROSIM board/EUROSIM officers. EUROSIM is governed by a board consisting of one representative of each member society, president and past president, and representatives for SNE and SIMPRA. The President is nominated by the society organising the next EUROSIM Congress. Secretary and Treasurer are elected out of members of the Board.

President	Mikuláš Alexík (CSSS), *alexik@frtk.fri.utc.sk*
Past president	Borut Zupančič (SLOSIM) *borut.zupancic@fe.uni-lj.si*
Secretary	Peter Fritzson (SIMS) *petfr@ida.liu.se*
Treasurer	Felix Breitenecker (ASIM) *felix.breitenecker@tuwien.ac.at*
SIMPRA Repres.	Jürgen Halin *halin@iet.mavt.ethz.ch*
SNE Repres.	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*

SNE – Simulation News Europe. EUROSIM societies are offered to distribute to their members the journal *Simulation News Europe* (SNE) as official membership journal. SNE is a scientific journal with reviewed contributions in the *Notes Section* as well as a membership newsletter for EUROSIM with information from the societies in the *News Section*. Publisher are EUROSIM, ARGESIM and ASIM.

Editor-in-chief	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*

→ www.argesim.org, menu SNE

→ www.asim-gi.org, menu International

EuroSim Congress. EUROSIM is running the triennial conference series EUROSIM Congress. The congress is organised by one of the EUROSIM societies. EUROSIM 2010 will be organised by CSSS in Prague, September 5-10, 2010.

Information	Mikulas Alexik (CSSS) *alexik@frtk.utc.sk*
Chair OC 2010	Miroslav Šnorek *snorek@fel.cvut.cz*

→ www.eurosim.org

ASIM
German Simulation Society
Arbeitsgemeinschaft Simulation

ASIM (Arbeitsgemeinschaft Simulation) is the association for simulation in the German speaking area, servicing mainly Germany, Switzerland and Austria. ASIM was founded in 1981 and has now about 700 individual members, and 30 institutional or industrial members. Furthermore, ASIM counts about 300 affiliated members.

→ www.asim-gi.org with members' area

info@asim-gi.org, admin@asim-gi.org

ASIM – Inst. f. Analysis and Scientific Computing
Vienna University of Technology
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria

ASIM Working Groups. ASIM, part of GI - Gesellschaft für Informatik, is organised in Working Groups, dealing with applications and comprehensive subjects:

GMMS	Methods in Modelling and Simulation Peter Schwarz, *schwarz@eas.iis.fhg.de*
SUG	Simulation in Environmental Systems Wittmann, *wittmann@informatik.uni-hamburg.de*
STS	Simulation of Technical Systems H.T.Mammen, *Heinz-Theo.Mammen@hella.com*
SPL	Simulation in Production and Logistics Sigrid Wenzel, *s.wenzel@uni-kassel.de*
SVS	Simulation of Transport Systems U. Brannolte, *Brannolte@bauing.uni-weimar.de*
SBW	Simulation in OR C. Böhnlein, *boehnlein@wiinf.uni-wuerzburg.de*
EDU	Simulation in Education/Education in Simulation W. Wiechert, *wiechert@simtec.mb.uni-siegen.de*

ASim Publications

SNE – Simulation News Europe. ASIM is publishing (co-publishing) SNE, which is regularly published and sent to all ASIM members (as part of their membership; 900 issues) and for promotion purposes (300 issues). Since 2006, the ASIM Working Groups publish *SNE Special Issues* with state-on-the-art reports on modelling and simulation in their workscope.

ASIM News. In December 2005, the ASIM Nachrichten has been replaced by an electronic news-letter - ASIM Newsletter. Editors are Th. Pawletta and C. Deatcu, Univ. Wismar, pawel@mb.hs-wismar.de.

ASIM Notes/ASIM Mitteilungen. The trademark ASIM Mitteilungen (ASIM Note) stands for all publications of ASIM and of the the ASIM Working Groups. Each publication gets an identification as ASIM Notes, independent of the publisher, and independent of the publication medium (printed books, CD, Web). ASIM Notes range from printed books (with CDs) published by Springer, via workshop publication published in SNE or ARGESIM, to compiled abstracts publishes at the ASIM weberver.

ASIM Books. ASIM co-operates with the SCS Publishing House e.V., with ARGESIM (Vienna University of Technology), and with Shaker Verlag Aachen in publication of two book series (Fortschritte in der Simulationstechnik - Frontiers in Simulation and Fortschrittsberichte Simulation - Advances in Simulation) and in publication of Proceedings. Publications in these series range from monographs via proceedings to PhD theses.

ASIM Board and Officers: The ASIM board consists of officers (elected all three years), of the chairpersons of the ASIM Working Groups (independently elected all three years), and of co-opted specialists.

Prsident	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*
Vice presidents	Sigrid Wenzel *s.wenzel@uni-kassel.de*
	Thorsten Peawletta, *pawel@mb.hs-wismar.de*
Secretary	Claus-Burkhard Böhnlein, *boehnlein@wiinf.uni-wuerzburg.de*
Treasurer	Ingrid Bausch-Gall, *Ingrid@Bausch-Gall.de*
Membership affairs	S. Wenzel, *s.wenzel@uni-kassel.de*
	W. Maurer, *werner.maurer@zhwin.ch*
	I. Bausch-Gall, *Ingrid@Bausch-Gall.de*
	F. Breitenecker (*mail address above*)
Universities	W. Wiechert *wiechert@simtec.mb.uni-siegen.de*
Industry	S. Wenzel, *s.wenzel@uni-kassel.de*
	K. Panreck, *Klaus.Panreck@hella.com*
Conferences	Klaus Panreck *Klaus.Panreck@hella.com*
	Albrecht Gnauck *albrecht.gnauck@tu-cottbus.de*
Publications	Th. Pawletta, *pawel@mb.hs-wismar.de*
	F. Breitenecker (*mail address above*)
Repr. EUROSIM	F. Breitenecker (*mail address above*)
Deputy	W. Wiechert *wiechert@simtec.mb.uni-siegen.de*
Edit. Board SNE	Thorsten Pawletta, *pawel@mb.hs-wismar.de*
Web EUROSIM	Anna Mathe, *anna.mathe@tuwien.ac.at*

CROSSIM – Croatian Society for Simulation Modelling

CROSSIM-*Croatian Society for Simulation Modelling* was founded in 1992 as a non-profit society with the goal to promote knowledge and use of simulation methods and techniques and development of education. CROSSIM is a full member of EUROSIM since 1997.

→ www.eurosim.info

✉ CROSSIM / Jadranka Božikov
Andrija Stampar School of Public Health,
Medical School, University of Zagreb
Rockefeller St. 4, HR-10000 Zagreb, Croatia

President	Jadranka Božikov, *jbozikov@snz.hr*
Vice president	Vesna Dušak, *vdusak@foi.hr*
Secretary	Vesna Bosilj-Vukšić, *vbosilj@efzg.hr*
Executive board members	Vlatko Čerić, *vceric@efzg.hr*
	Tarzan Legović, *legovic@irb.hr*
Repr. EUROSIM	Jadranka Bozikov, *jbozikov@snz.hr*
Edit. Board SNE	Jadranka Bozikov, *jbozikov@snz.hr*
Web EUROSIM	Jadranka Bozikov, *jbozikov@snz.hr*

CSSS – Czech and Slovak Simulation Society

CSSS -The *Czech and Slovak Simulation Society* has about 150 members working in Czech and Slovak national scientific and technical societies (*Czech Society for Applied Cybernetics and Informatics*, *Slovak Society for Applied Cybernetics and Informatics*). The main objectives of the society are: development of education and training in the field of modelling and simulation, organising professional workshops and conferences, disseminating information about modelling and simulation activities in Europe. Since 1992, CSSS is full member of EUROSIM.

→ www.fit.vutbr.cz/CSSS

✉ CSSS / Miroslav Šnorek, CTU Prague
FEE, Dept. Computer Science and Engineering,
Karlovo nam. 13, 121 35 Praha 2, Czech Republic

President	Miroslav Šnorek, *snorek@fel.cvut.cz*
Vice president	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Treasurer	Evžen Kindler, *ekindler@centrum.cz*
Scientific Secr.	A. Kavička, *Antonin.Kavicka@upce.cz*
Repr. EUROSIM	Miroslav Šnorek, *snorek@fel.cvut.cz*
Deputy	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Edit. Board SNE	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Web EUROSIM	Petr Peringer, *peringer@fit.vutbr.cz*

DBSS – Dutch Benelux Simulation Society

The Dutch Benelux Simulation Society (DBSS) was founded in July 1986 in order to create an organisation of simulation professionals within the Dutch language area. DBSS has actively promoted creation of similar organisations in other language areas. DBSS is a member of EUROSIM and works in close cooperation with its members and is further affiliated with SCS International, IMACS, and the Chinese Association for System Simulation and the Japanese Society for Simulation Technology.

→ www.eurosim.info

✉ DBSS / A. W. Heemink
Delft University of Technology, ITS - twi,
Mekelweg 4, 2628 CD Delft, The Netherlands

President	A. Heemink, *a.w.heemink@its.tudelft.nl*
Vice president	W. Smit, *smitnet@wxs.nl*
Treasurer	W. Smit, *smitnet@wxs.nl*
Secretary	W. Smit, *smitnet@wxs.nl*
Repr. EUROSIM	A. Heemink, *a.w.heemink@its.tudelft.nl*
Deputy	W. Smit, *smitnet@wxs.nl*
Edit. Board SNE	A. Heemink, *a.w.heemink@its.tudelft.nl*

FRANCOSIM – Société Francophone de Simulation

FRANCOSIM was founded in 1991 and aims to the promotion of simulation and research, in industry and academic fields. Francosim operates two poles.

- Pole Modelling and simulation of discrete event systems. Pole Contact: *Henri Pierreval, pierreva@imfa.fr*
- Pole Modelling and simulation of continuous systems. Pole Contact: *Yskandar Hamam*, *y.hamam@esiee.fr*

→ www.eurosim.info

✉ FRANCOSIM / Yskandar Hamam
Groupe ESIEE, Cité Descartes,
BP 99, 2 Bd. Blaise Pascal,
93162 Noisy le Grand CEDEX, FRANCE

President	Yskandar Hamam, *y.hamam@esiee.fr*
Treasurer	François Rocaries, *f.rocaries@esiee.fr*
Repr. EUROSIM	Yskandar Hamam, *y.hamam@esiee.fr*
Edit. Board SNE	Yskandar Hamam, *y.hamam@esiee.fr*

HSS – Hungarian Simulation Society

The Hungarian Member Society of EUROSIM was established in 1981 as an association promoting the exchange of information within the community of people involved in research, development, application and education of simulation in Hungary and also contributing to the enhancement of exchanging information between the Hungarian simulation community and the simulation communities abroad. HSS deals with the organization of lectures, exhibitions, demonstrations, and conferences.

→ www.eurosim.info

✉ HSS / András Jávor,
Budapest Univ. of Technology and Economics,
Sztoczek u. 4, 1111 Budapest, HUNGARY

President	András Jávor, *javor@eik.bme.hu*
Vice president	Gábor Szűcs, *szucs@itm.bme.hu*
Secretary	Ágnes Vigh, *vigh@itm.bme.hu*
Repr. EUROSIM	András Jávor, *javor@eik.bme.hu*
Deputy	Gábor Szűcs, *szucs@itm.bme.hu*
Edit. Board SNE	András Jávor, *javor@eik.bme.hu*
Web EUROSIM	Gábor Szűcs, *szucs@itm.bme.hu*

ISCS – Italian Society for Computer Simulation

The Italian Society for Computer Simulation (ISCS) is a scientific non-profit association of members from industry, university, education and several public and research institutions with common interest in all fields of computer simulation.

→ www.eurosim.info

✉ ISCS / Mario Savastano,
c/o CNR - IRSIP,
Via Claudio 21, 80125 Napoli, ITALY

President	MarioSavastano, *mario.savastano@unina.it*
Vice president	F. Maceri, *Franco.Maceri@uniroma2.it*
Secretary	Paola Provenzano, *paola.provenzano@uniroma2.it*
Treasurer	Pasquale Arpaia
Repr. EUROSIM	F. Maceri, *Franco.Maceri@uniroma2.it*
Edit. Board SNE	Mario Savastano, *mario.savastano@unina.it*

PSCS – Polish Society for Computer Simulation

PSCS was founded in 1993 in Warsaw. PSCS is a scientific, non-profit association of members from universities, research institutes and industry in Poland with common interests in variety of methods of computer simulations and its applications. At present PSCS counts 264 members.

→ www.ptsk.man.bialystok.pl

✉ PSCS / Leon Bobrowski, c/o IBIB PAN,
ul. Trojdena 4 (p.416), 02-109 Warszawa, Poland

President	Leon Bobrowski, *leon@ibib.waw.pl*
Vice president	A. Chudzikiewicz, *ach@it.pw.edu.pl*
Treasurer	Z. Sosnowski, *zenon@ii.pb.bialystok.pl*
Secretary	Zdzislaw Galkowski, *Zdzislaw.Galkowski@simr.pw.edu.pl*
Repr. EUROSIM	Leon Bobrowski, *leon@ibib.waw.pl*
Deputy	A.Chudzikiewicz, *ach@it.pw.edu.pl*
Edit. Board SNE	Z.Sosnowski, *zenon@ii.pb.bialystok.pl*

SIMS – Scandinavian Simulation Society

SIMS is the *Scandinavian Simulation Society* with members from the four Nordic countries Denmark, Finland, Norway and Sweden. The SIMS history goes back to 1959. SIMS practical matters are taken care of by the SIMS board consisting of two representatives from each Nordic country. Iceland will be represented by one board member.

SIMS Structure. SIMS is organised as federation of regional societies. There are FinSim (Finnish Simulation Forum), DKSIM (Dansk Simuleringsforening) and NFA (Norsk Forening for Automatisering).

→ www.scansims.org

✉ SIMS/Peter Fritzson, IDA, Linköping University,
58183, Linköping, Sweden

President	Peter Fritzson, *petfr@ida.liu.se*
Treasurer	Vadim Engelson, *vaden@ida.liu.se*
Repr. EUROSIM	Peter Fritzson, *petfr@ida.liu.se*
Edit. Board SNE	Esko Juuso, *esko.juuso@oulu.fi*
Web EUROSIM	Vadim Engelson, *vaden@ida.liu.se*

SLOSIM – Slovenian Society for Simulation and Modelling

SLOSIM - Slovenian Society for Simulation and Modelling was established in 1994 and became the full member of EUROSIM in 1996. Currently it has 69 members from both slovenian universities, institutes, and industry. It promotes modelling and simulation approaches to problem solving in industrial as well as in academic environments by establishing communication and cooperation among corresponding teams.

→ msc.fe.uni-lj.si/SLOSIM

✉ SLOSIM / Rihard Karba, Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, 1000 Ljubljana, Slovenia

President	Rihard Karba, *rihard.karba@fe.uni-lj.si*
Vice president	Leon Žlajpah, *leon.zlajpah@ijs.si*
Secretary	Aleš Belič, *ales.belic@fe.uni-lj.si*
Treasurer	Milan Simčič, *milan.simcic@fe.uni-lj.si*
Repr. EUROSIM	Rihard Karba, *rihard.karba@fe.uni-lj.si*
Deputy	Borut Zupančič, *borut.zupancic@fe.uni-lj.si*
Edit. Board SNE	Rihard Karba, *rihard.karba@fe.uni-lj.si*
Web EUROSIM	Aleš Belič, *ales.belic@fe.uni-lj.si*

UKSIM – United Kingdom Simulation Society

UKSIM has more than 100 members throughout the UK from universities and industry. It is active in all areas of simulation and it holds a biennial conference as well as regular meetings and workshops.

→ www.uksim.org.uk

✉ UKSIM / Alessandra Orsoni,
Kingston Business School, Kingston Hill,
Kingston-Upon-Thames, Surrey, KT2 7LB, UK

President	David Al-Dabass, *david.al-dabass@ntu.ac.uk*
Secretary	Alessandra Orsoni, *A.Orsoni@kingston.ac.uk*
Treasurer	B. Thompson, *barry@bjtcon.ndo.co.uk*
Membership chair	K. Al-Begain, *kbegain@glam.ac.uk*
Univ. liaison chair	R. Cheng, *rchc@maths.soton.ac.uk*
Ind. liaison chair	Richard Zobel, *r.zobel@ntworld.com*
Conf. venue chair	John Pollard, *j.pollard@ee.ucl.ac.uk*
Repr. EUROSIM	A. Orsoni, *A.Orsoni@kingston.ac.uk*
Edit. Board SNE	A. Orsoni, *A.Orsoni@kingston.ac.uk*

CEA-SMSG – Spanish Modelling and Simulation Group

CEA is the Spanish Society on Automation and Control In order to improve the efficiency and to deep into the different fields of automation, the association is divided into thematic groups, one of them is named 'Modelling and Simulation', constituting the group.

→ www.cea-ifac.es/wwwgrupos/simulacion

✉ CEA-SMSG / María Jesús de la Fuente,
System Engineering and AutomaticControl department,
University of Valladolid,
Real de Burgos s/n., 47011 Valladolid, SPAIN

President	María J. la Fuente, *maria@autom.uva.es*
Repr. EUROSIM	María J. la Fuente, *maria@autom.uva.es*

LSS – Latvian Simulation Society

The Latvian Simulation Society (LSS) has been founded in 1990 as the first professional simulation organisation in the field of Modelling and simulation in the post-Soviet area. Its members represent the main simulation centres in Latvia, including both academic and industrial sectors.

→ briedis.itl.rtu.lv/imb/

✉ LSS / Yuri Merkuryev, Dept. of Modelling
and Simulation Riga Technical University
Kalku street 1, Riga, LV-1658, LATVIA

President	Yuri Merkuryev, *merkur@itl.rtu.lv*
Repr. EUROSIM	Yuri Merkuryev, *merkur@itl.rtu.lv*

ROMSIM – Romanian Modelling and Simulation Society

ROMSIM has been founded in 1990 as a non-profit society, devoted to both theoretical and applied aspects of modelling and simulation of systems. ROMSIM currently has about 100 members from both Romania and Republic of Moldavia.

→ briedis.itl.rtu.lv/imb/

✉ LSS / Yuri Merkuryev, Dept. of Modelling
and Simulation Riga Technical University
Kalku street 1, Riga, LV-1658, LATVIA

President	Florin Stanciulescu, *sflorin@ici.ro*
Vice president	Florin Hartescu, *flory@ici.ro*
Secretary	Zoe Radulescu, *radulescu@ici.ro*
Repr. EUROSIM	Florin Stanciulescu, *sflorin@ici.ro*
Deputy	Florin Hartescu, *flory@ici.ro*
Edit. Board SNE	Florin Stanciulescu, *sflorin@ici.ro*

ASIM - German Simulation Society

ASIM (Arbeitsgemeinschaft Simulation) is the association for simulation in the German speaking area, servicing mainly Germany, Switzerland and Austria. ASIM was founded in 1981 and has now about 700 individual members, and 30 institutional or industrial members. Furthermore, ASIM counts about 300 affiliated members. Organisational details and contact addresses can be found in the ASIM short information (see before).

ASIM Publications

SNE - Simulation News Europe. ASIM is publishing together with ARGESIM the journal SNE, which is regularly published and sent to all ASIM members (as part of their membership; 900 issues) and for promotion purposes (300 issues). Since 2006, the ASIM Working Groups publish SNE Special Issues with state-on-the-art reports on modelling and simulation in their workscope.

ASIM News. In December 2005, printed ASIM Nachrichten has been replaced by an electronic newsletter - ASIM Newsletter. Editors are Th. Pawletta and C. Deatcu, Univ. Wismar, pawel@mb.hs-wismar.de. ASIM News can be downloaded from ASIM website www.asim-gi.org (members and non-members).

ASIM Notes - ASIM Mitteilungen. The trademark ASIM Mitteilungen (ASIM Note) stands for all publications of ASIM and of the ASIM Working Groups. Each publication gets identification as ASIM Mitteilung, independent of the publisher, and independent of the publication medium (printed book, CD, Web). ASIM Notes range from printed books (with CDs) published by Springer, via workshop publications published in SNE by ASIM, to compiled abstracts published at the ASIM webserver.

ASIM Books. ASIM co-operates with the SCS Publishing House e.V., with ARGESIM (Vienna University of Technology), and with Shaker Verlag Aachen in publication of book series (Fortschritte in der Simulationstechnik - Frontiers in Simulation and Fortschrittsberichte Simulation - Advances in Simulation) and in publication of Proceedings. Publications in these series range from monographs via proceedings to PhD theses.

Details on new ASIM publications will be published in the next SNE issue.

ASIM Working Groups

Working Group Structure. ASIM is part of GI - Gesellschaft für Informatik (Society for Informatics) and is itself structured into working groups (WG), which address various areas of modelling and simulation. As this news provide a detailed report on the activities of the ASIM working group *Simulation in Environmental Systems* (ASIM SUG), reports and details of the other working groups will be published in the next SNE issue. Workshops and conferences of the working groups are announced in the conference section.

ASIM Conferences

ASIM organises the up to now annual ASIM Conference, the ASIM working groups organise annual workshops (up to 150 participants) and bi-annual conferences (more than 150 participants. ASIM cooperates in organising the tri-annual EUROSIM Congress. Furthermore, ASIM co-organises local conferences, e.g. the ASIM Wismar Workshop. A special cooperation was established with the three-annual conference series MATHMOD - Mathematical Modelling in Vienna. Reports about workshops of 2008 and a report of the very successful conference Simulation in Production and Logistics (Berlin, October 2008) is scheduled for the next issue.

At present, the following conferences and workshops with ASIM as organizer, co-organizer or co-sponsor are scheduled for 2009:

- MATHMOD 2009, Vienna, February 2009
- Workshop GMMS Stuttgart, February 2009
- Workshop STS / GMMS Dresden, March 2009
- Workshop SUG Potsdam, March 2009
- ASIM Conference 2009: Symposium Simulation Technique, September 2009

MATHMOD 2009
Vienna Conference on Mathematical Modelling
February 11 – 13, 2009, Vienna, Austria
www.mathmod.at

ASIM is co-organizer of the tri-annual conference series MATHMOD in Vienna. The 6th MATHMOD Conference will take place February 2009 in Vienna, hosted by Vienna University of Technology, organised by Inge Troch (*Inge.Troch@tuwien.ac.at*) and

Felix Breitenecker (*Felix.Breitenecker@tuwien.ac.at*). ASIM members will organise Special Sessions, furthermore special contributions to Art & Modelling will be provided.

ASIM – GMMS WORKSHOP 2009
Survey on Methods and Application in Modelling
February 18 - 20, 2009, Univ. Stuttgart
www.simtech.uni-stuttgart.de/veranstaltungen/ASIM/
www.asim-gi.org

In 2009, these workshop series will be continued by a workshop in Stuttgart. Planned are, as in the year before, overview lectures and invited lectures, and PhD poster presentations.

ASIM – STS/GMMS WORKSHOP 2009
Simulation of Technical Systems
March 5 – 6, 2009 Dresden, Germany
www.eas.iis.fraunhofer.de/asim09
www.asim-gi.org

This workshop, organized by Division Design Automation of Fraunhofer Institute for Integrated Circuits Simulation, will concentrate on the following topics:

- Model-based development in automotive industry
- Simulation in medical engineering
- Modelling of mechatronic systems
- Simulation in robotics
- Simulation in electronic systems
- Modelling languages and modelling standards
- Methods and Algorithms

ASIM – SUG WORKSHOP 2009
Simulation of Environmental Systems
March 26 – 27, 2009 Cottbus, Germany
www.pik-potsdam.de/asim-workshop-2009
www.asim-gi.org

ASIM 2009 CONFERENCE
Symposium Simulation Technique
September 23- 25, 2009, Potsdam, Germany
www.tu-cottbus.de/umweltinformatik
www.asim-gi.org

The ASIM 2009 Conference will be organised by Albrecht Gnauck at Brandenburgische Technische Universität Cottbus, September 23 – 25, 2009. The conference themed ‚Simulation for Environment, Climate, Energy and Techniques' will provide six plenary lectures, parallel sections, workshops, poster exhibition, and tool exhibition.

Conference languages are German and English. Participants are invited to excursions to Spreewald, Surface Mining, Power Station, and to other social events.

SLOSIM – Slovenian Society for Simulation and Modelling

SLOSIM (Slovenian Society for Simulation and Modelling) was established in 1994 and became the full member of EUROSIM in 1996. Currently, it has 76 members from Slovenian universities, institutes, and industry. It promotes modelling and simulation approach to problem solving in industrial as well as in academic environments by establishing communication and cooperation among the corresponding teams. Organisational details and contact addresses can be found in the SLOSIM short information (see before).

→ msc.fe.uni-lj.si/slosim

✉ slosim@fe.uni-lj.si

Activities

In the last period we participated in organisation of annual ERK08 conference that took place in Portorož, Slovenia, from Sept. 29th to Oct. 1st. The society organised two sessions on modelling and simulations, while our members participated in the conference with 15 contributions.

Prof. Dr. Rihard Karba and Prof Dr. Borut Zupančič attended the EUROSIM board meeting in March in Cambridge.

On Nov. 27 2008 there was a meeting of executive board of SLOSIM as well as general assembly. Our member, Prof. Dr. Mojca Indihar-Štemberger presented modelling and simulation of business processes.

In December of 2008, the following lectures from the field of modelling in mobile robotics were organised:

- Prof. Dr. Ivan Petrović, University of Zagreb, Croatia: "Autonomous mobile robots navigation in unknown and dynamic indoor environments"
- Marija Seder, University of Zagreb, Croatia: "Dynamic window based approach to mobile ro-

bot motion control in the presence of moving obstacles"

- Mišel Brezak, University of Zagreb, Croatia: "Robust and accurate global vision system for real time tracking of multiple mobile robots"

Coming Events

Two special sessions are being organised on MATHMOD'09 conference by the members of SLOSIM as well as one plenary talk. The members of SLOSIM have been very active as reviewers for the upcoming MATHMOD conference.

SIMS − Scandinavian Simulation Society

SIMS is the Scandinavian Simulation Society with members from the four Nordic countries Denmark, Finland, Norway and Sweden. The SIMS history goes back to 1959. SIMS practical matters are taken care of by the SIMS board consisting of two representatives from each Nordic country. Iceland will be represented by one board member. Organisational details and contact addresses can be found in the SIMS short information (see before).

→ www.scansims.org

✉ esko.juuso@oulu.fi

SIMS Structure. SIMS is organised as federation of regional societies. There are FINSIM (Finnish Simulation Forum), DKSIM (Dansk Simuleringsforening) and NFA (Norsk Forening for Automatisering).

Activities. Many members are active in the organization Open Source Modelica Consortium.

Past Events

The 49th SIMS conference was held at Oslo University College in Oslo, in October 7-8, 2008. The programme included 2 invited speakers and 23 regular papers. The invited talks were on modelling and simulation of biological populations (professor Nils Christian Stenseth at the Department of Biology at the University of Oslo), and interval computations in numerical algorithms (Warwick Tucker, Research leader of CAPA − Computer-Aided Proofs in Analysis − at the University of Bergen). The 23 papers discussed topics including modelling techniques, fluidized beds, biological models, visualization and modelling, and industrial applications.

ModProd 2008 – 2nd Workshop on Model-based product development was held in Linköping in February 5-6, 2008. The workshop was organized by the Center for Model-based Product Development (MODPROD) at Linköping University. Programme included 2 keynote talks, 10 other presentations, a poster session and 6 tutorials in the area of model based tools and methods for mechanical systems, electronic systems and software, and unified approaches for model based design.

Scientific session on Energy saving and Green energy at Energitinget in Stockholm, March 12-13, 2008

International School on Finance, Insurance, and Energy Markets – Sustainable Development. Use of mathematics for financial and energy trading, Västerås, Sweden, May 5-9, 2008

PARA 2008 – 9th International Workshop on State-of-the-Art in Scientific and Parallel Computing was held May 13-16, 2008 on the Norwegian University of Science and Technology (NTNU) Campus in Trondheim, Norway. There were 5 invited speakers, 6 invited panel speakers and 114 speakers. Programme included 16 minisymposia and 4 tutorials. Topics were algorithms, software, tools, environments as well as applications for Scientific Computing, High Performance Computing, Parallel Computing, Grid Computing, and Computer Simulations including Interactive and Scientific Visualization. http://para08.idi.ntnu.no

Summer School on Sustainable Production and Energy: Catalysis by Nanomaterials, Catalytic Microreactors, Oulu, Finland, June 2-6, 2008.

Coming Events

SIMS 2009
50th Conference on Simulation and Modelling
October 6-8, 2009, Copenhagen, Denmark
www.scansims.org

The 50th Scandinavian Conference on Simulation and Modelling, will be organized by DKSIM in Copenhagen, Denmark, October 6-8, 2008.

The purpose of the SIMS conference is to cover broad aspects of modeling and simulation and scientific computation. It will be of interest for model builders, simulator personnel, scientists, engineers, vendors, etc. The scientific program will consist of

technical sessions with submitted and invited papers, and is open for poster sessions and vendor demonstrations. The focus area is modelling and simulation in energy systems.

Proceedings of the accepted papers will be distributed at the conference. Presented papers will be considered for publication in the EUROSIM scientific journal 'Simulation and Modelling - Practise and Theory (SIMPRA)' published by Elsevier Science. Especially Ph.D. students are encouraged to contribute with papers according to the conference themes.

NPCW '09

The 15th Nordic Process Control Workshop
January 29-30, 2009. Porsgrunn, Norway
www.hit.no/TF/npcw_09

The 15th Nordic Process Control Workshop, NPCW09 will be held at Telemark University College in Porsgrunn, Norway, 29-30 January 2009.

The aim of the NPCW is to provide researchers and practioners from industry and academia with a platform to report on recent developments in the newly emerging areas of technology and their potential applications to process automation. The NPC events have been fortunate to attract high quality papers from the key research groups of the Nordic countries and from the process industries as well.

OPENMODELICA ANNUAL WORKSHOP

February 2, 2009, Linköping, Sweden
http://www.ida.liu.se/labs/pelab/modelica/OpenModelicaWorkshop.html

The Open Source Modelica Consortium (OSMC) and Linköping University will organize the 1st OpenModelica Annual Workshop in Linköping.

MODPROD 2009

3rd Annual Workshop on Model-based Product Development
February 3-4, 2009, Linköping, Sweden
www.modprod.liu.se/

The Center for Model-based Product Development (MODPROD) at Linköping University focuses on model-based tools and methods for electronic systems and software. MODPROD 2009 is an annual workshop brings together expertise in these fields to discuss state of the art and the way ahead.

AUTOMATION XVIII SEMINAR

March 17-18, 2009, Helsinki, Finland
autom09.automaatioseura.fi

Automation XVIII Seminar, Helsinki, March 17-18, 2009, 2 special sessions on simulation.

PP&PS'09

The 15th Nordic Process Control Workshop
January 29-30, 2009, Porsgrunn, Norway
pppsc09.automaatioseura.fi/index.html

IFAC Symposium on Power Plants and Power Systems Control will be held in Tampere, July 5-8, 2009. The symposium is focused on the topics of novel research results and applications in control of energy related systems and processes. Energy production and delivery systems have to meet the new challenges coming from increased energy consumption and environmental aspects. The variety of energy sources and distributed power generation increase the need for more efficient control of power plants and power systems. Another issue beside the control of the balance of generation and consumption is the environmental aspect. Emissions control, energy saving technology, and development of new energy production systems will be discussed. Energy is one of the most important topics now and in the future. The scope of the symposium is to present and discuss the latest developments in energy related systems and processes.

LSS − Latvian Simulation Society

The Latvian Simulation Society had been found in 1990 as the first professional simulation organization in the field of modelling and simulation in the post-Soviet area. Its members represent the main simulation centres in Latvia, including both academic and industrial sectors, in particular, operating at Riga Technical University, Latvian University, the Latvian University of Agriculture, Transport and Telecommunication Institute, as well as at industrial companies DATI Exigen Group and Solvers, Ltd. At the EUROSIM Board meeting in Ljubljana on September 12, 2007, the Latvian Simulation Society became a full member of EUROSIM. Organisational details and contact addresses can be found in the LSS short information (see before).

→ www.itl.rtu.lv/imb

✉ merkur@itl.rtu.lv

Activities. The society performs regular activities at both national and international levels. It runs a series of scientific seminars where current research activities at involved organizations in the area of modelling and simulation are discussed. For instance, doctoral students regularly present here their research outcomes.

LSS members regularly participate at EUROSIM events. For example, at EUROSIM/UKSim2008, the 10th International Conference on Computer Modelling and Simulation, Latvian Simulation Society was represented with five papers.

LSS has hosted international conferences "Simulation, Gaming, Training and Business Process Reengineering in Operations" in 1996 and 2000), "Harbour, Maritime & Multimodal Logistics Modelling and Simulation", HMS in 1998 and 2003, as well as the 2005 European Conference on Modelling and Simulation, ECMS 2005.

Past Events. For instance, in 2008 it is co-sponsoring the following conferences: MBITS'08, The International Conference on Modelling of Business, Industrial and Transport Systems (May 7-10, 2008, Riga, Latvia); ECMS2008, 22nd EUROPEAN Conference on Modelling and Simulation (June 3 – 6, 2008, Nicosia, Cyprus), and I3M2008, the 5th International Mediterranean Modelling Multiconference (September 17-19, 2008, Amantea, Italy).

CROSSIM − Croatian Simulation Society

CROSSIM - CROatian Society for SIMulation Modelling was founded in 1992 as a non-profit society with the goal to promote knowledge and use of simulation methods and techniques and development of education and training in the field of simulation modelling. CROSSIM is a full member of EUROSIM since 1997.

→ www.eurosim.info

Publications. CROSSIM co-operates with the University Computing Centre, Zagreb, in publishing of the Journal of Computing and Information Technology (CIT). All information concerning CIT is available at CIT.SRCE.HR.

CIT Volume 16, Number 4, 2008 published selected papers of the ITI Congerence 2008.

Coming Events. The 31st International Conference Information Technology Interfaces ITI2009 will be held in Cavtat near Dubrovnik on June 22-25, 2009.

The conference has a long tradition (since 1974) of creating an inspiring, productive and pleasurable atmosphere for interdisciplinary communication among researchers, scholars and professionals from various subfields of ICT arena. ARGESIM and CROSSIM are among Conference co-operating institutions.

ITI 2009

31st International Conference

Information Technology Interfaces

with

BIOSTAT 2008

June 25 – 27, 2009

Cavtat near Dubrovnik, Croatia

The 16th Meeting of Researchers of Biometrics / Statistics *BIOSTAT 2008*, co-organised by Croatian Biometric Society, will be held in parallel with ITI 2009

Modelling and Simulation for Environmental Applications – Experiences of the ASIM EG 4.6.3 from 1998 to 2007

Jochen Wittmann, University of Hamburg, *wittmann@informatik.uni-hamburg.de*
Albrecht Gnauck, Brandenburg University of Technology at Cottbus, *albrecht.gnauck@tu-cottbus.de*

Modelling and simulation of environmental systems are complex tasks which cover broad ranges of operating environmental states and decision making. Environmental systems are complicated and complex systems. Therefore, different modelling and simulation methods are necessary to portray the non-linear behaviour of such systems. To express the differentiation of environmental systems the GI-TC 4.6 formed in 1992 two new working groups were established. One of these groups, the EG 4.6.3 dealing with tools of simulation and model building for environmental applications got a distinguished development to other expert groups of TC 4.6.3. The workgroup *Werkzeuge* für *Modellbildung und Simulation in Umweltanwendungen* founded in 1992 changed its status in 1996 to the Expert Group 4.6.3 and cooperates with ASIM, the GI-TC 4.5. Since 1999 the EG continues Grützner's activities in parallel with a yearly track in spring *Simulation in den Umwelt- und Geowissenschaften* guided by J. Wittmann, and with a yearly track in fall *Modellierung und Simulation von Ökosystemen* presented by A. Gnauck. The paper gives a short overview on the manifold of topics covered by activities of the EG between 1998 and 2007

Introduction

Since the beginning of the seventieth of last century modelling and simulation of environmental systems become more and more important for description of environmental states, to organise environmental knowledge and to manage environmental systems (Schnoor 1996). Environmental systems are complex, thermodynamically open systems with nonlinear interrelationships (Weber 2000). Holism and reductionism are two different approaches to study and analyse environmental processes and systems. Both approaches are needed for their study and management. On the one hand, holism attempts to reveal the properties of environmental systems by studying the system as a whole. Therefore, it is required that an analysis and management of an environmental system has to be occurred on systems theory scale. On the other hand, to simplify the study of an environmental system and to facilitate the interpretations of environmental processes the components of an environmental system have to be separated from the system level. This approach is useful to find governing interrelationships between real objects and has obvious shortcomings when the functioning of the entire environmental system is to be revealed.

The modelling procedure of environmental processes and/or systems can be understood as a synthesis of elements of knowledge about the environment. The model quality, that means its applicability to environmental processes and systems, depends on the quality of knowledge on the system and the amount and quality of available data on it. By means of mathematical models new knowledge about the reactions and properties of the entire system will be provided (Jørgensen 1992, Jakeman et al. 1993). Current approaches to environmental systems modelling and simulation are based on different scientific and engineering disciplines and methodologies. Informatic tools like data bases and data warehouses (Immon 1992, Denzer/Russel/Schimak 1995, Elmasri/Navathe 2000), environmental information systems (Günther 1998, Rautenstrauch 1999), geographical information systems (Brebbia/Pascolo 1998, Aspinelli 2001), management information systems (Brebbia/Pascolo 2000), and decision support systems (Wierzbicki/Makowski/Wessels 2000, Hobbs/Meier 2000) contribute to this subject as well as systems theory (Bossel 1992), environmental statistics (Mateu/Montes 2001, Haan 2002), environmental economics (Costanza 1991, Haasis 1996, Gilpin 2000) and advanced graph theory (Edwards 2001). Classifications of environmental models are often represented by the type of application area (Grützner 1995), by the type of medium (Grützner 1997) or by the type of simulation technology (Wisniewski/Klein 2001). Another classification can be given by the type of systems adaptability and stability (Straškraba/Gnauck 1985). Looking on environmental systems as controllable transfer systems notions of automata theory are of interest. Especially, cellular automata in combination with GIS led to numerous mod-

Year	Location	No. Participants	Papers	Applications	Informatic tools
1998	Witzenhausen	25	13	6	7
1999	Koblenz	25	16	7	9
2000	Hamburg	30	14	7	7
2001	Münster	30	14	4	10
2002	Cottbus	35	15	8	7
2003	Osnabrück	35	13	8	5
2004	Müncheberg	40	17	9	8
2005	Dresden	60	28	20	8
2006	Leipzig	45	24	14	10
2007	Berlin	55	19	14	5

Table 1. The workshop *Simulation in den Umwelt- und Geowissenschaften* from 1998 to 2007.

els for environmental planning (Bandini/Worsch 2001), for ecosystems description (Sonnenschein/Vogel 2001), and for environmental management (White/Engelen 1993, Thinh 2003).

The EG 4.6.3 founded in 1992 under the original name *Tools of modelling* and *simulation for environmental applications* was successfully presented by R. Grützner, University of Rostock, for many years. He organised a series of workshops on modelling and simulation of environmental systems under the title *Modelling and Simulation for Environmental Applications*. Since 1996 the EG 4.6.3 cooperates with the ASIM, the GI-TC 4.5. After the retirement of R. Grützner, J. Wittmann is the speaker of the EG since 2000. He undertook all the tasks and obligations from R. Grützner. According to new scientific and informatics related developments he changed the name of the Expert Group in 2000 to "*Simulation for Environmental Sciences and Geosciences*" and continued the series of workshops very successfully up to now.

Another parallel development took place at the Brandenburg University of Technology at Cottbus which was founded in 1991. In 1993, the Department of Ecosystems and Environmental Informatics was established under the full professorship of A. Gnauck. It was the first university department in Germany with the topic of environmental informatics contained in the name of the department. A. Gnauck organised yearly conferences and workshops related to theory, modelling and simulation of ecosystems since 1994. After some fruitful discussions R. Grützner and A. Gnauck agreed to combine both activities. Since October 1999 the EG 4.6.3 has two specialist groups and organises in parallel two yearly tracks on *Simulation in den Umwelt- und Geowissenschaften* guided by J. Wittmann in spring, and *Modellierung und Simulation von Ökosystemen* presented by A. Gnauck in fall. While the spring workshop takes place at changing locations the workshop in fall is carried out at the Island of Usedom each year.

The results of both workshops are published in books. R. Grützner published the papers of the workshops up to 1999 by Metropolis Publishing House at Marburg. J. Wittmann changed publications to Shaker Publishing House at Aachen. Up to 1998 A. Gnauck published the workshop papers by Blottner Publishing House at Taunusstein and Physica Publishing House at Heidelberg. Since 1999 he is publishing the results of the Usedom–Workshops at Shaker Publishing House at Aachen. Realising scientific changes in ecology he changed the title of the publications from *Theory and Modelling of Ecosystems* to *Modelling and Simulation of Ecosystems* in 2002.

The following chapters give a short impression on the activities of EG 4.6.3.

Simulation for environmental sciences and geosciences

The workshop on *Simulation in den Umwelt- und Geowissenschaften* organised and presented by J. Wittmann, University of Hamburg, takes place yearly in March at different locations. This principle was adopted from R. Grützner who organised these workshops before 2000. He wanted to integrate engineers and information scientists of research teams in Germany dealing with modelling and simulation for environmental applications. J. Wittmann followed this idea. Therefore, the workshop shifted from North to South and from West to East Germany. Table 1 contains an overview on workshop locations, number of papers presented, and number of participants from 1998 to 2007 as well as a classification of papers. As can be seen, there is more or less a balance between development of informatic tools and applications.

A classification of papers is very complicated. Different aspects and distinguished points of view characterise these workshop contributions. A few papers deal with certain aspects of theoretical informatics. Mostly, methodological developments are combined with practical applications on environmental media or on production processes. Because of the great variety of papers published in the workshop books they will be classified in a coarse manner by applications (in-

cluding environmental media and production processes) and by informatic tools (including data bases, environmental information systems, geographical information systems, decision support systems, software technology and development, methods and theory of informatics).

The increasing number of participants and papers is caused not only by people's interest on environmental knowledge and information, but also by harmonised or new national and European policies. Sustainable environmental development becomes more and more importance and needs support by informatic tools. This tendency is expressed by the increasing number of applications which can be noticed from table 1. The application oriented papers cover aspects of water management, land use and material flow of industrial processes. The informatic oriented papers contain new software developments, combinations of different software tools like GIS and data bases or GIS and DSS. Mostly, the papers contain a mixture of both aspects from a practical and a theoretical point of view.

Theory, modeling and simulation of ecosystems

The workshops on *Theory and modelling of ecosystems*, since 2002 known under the title *Modelling and simulation of ecosystems* are organised and presented by A. Gnauck, Brandenburg University of Technology at Cottbus, and takes place yearly in October at Kölpinsee/Island of Usedom. He extended and continued the original idea of R. Grützner to support modelling and simulation activities in the field of ecology. Theoretical developments on mathematical descriptions of ecological processes and data analysis as well as applications on aquatic and terrestrial ecosystems and for air protection are in the focus of the workshops. In opposite of the spring workshops of the EG 4.6.3 the target groups of this workshop are ecologists dealing with modelling and simulation of ecosystems, and practitioners and engineers working in the different fields of application of ecology and environmental technology. An overview on the activities of this specialist group of EG 4.6.3 is given in table 2 where the same classification is used. It can be seen from table 2 that the number of application oriented contributions presented at the workshops is greater or equal to the number of papers dealing with informatic tools since 2001. This is in agreement with the basic idea of the EG 4.6.3 to work out informatic tools of modelling and simulation for environmental applications, especially ecosystems.

Year	Location	No. Participants	Papers	Applications	Informatic tools
1997	Kölpinsee	16	10	4	6
1998	Kölpinsee	20	11	6	5
1999	Kölpinsee	18	8	3	5
2000	Kölpinsee	20	16	7	9
2001	Kölpinsee	28	18	10	8
2002	Kölpinsee	16	10	6	4
2003	Kölpinsee	27	18	10	8
2004	Kölpinsee	18	13	8	5
2005	Kölpinsee	28	18	9	9
2006	Kölpinsee	29	19	14	5
2007	Kölpinsee	32	26	19	7

Table 2. The workshop *Modellierung und Simulation von Ökosystemen* from 1997 to 2007

At the beginning of this series of workshops special attempt was made to mathematical interpretations of ecological processes and existing informatic tools. Now, more importance is given to applications, to combined software tools and new software developments. The ecological applications cover both limnic and terrestrial ecosystem processes. Additionally, they include aspects of hydrology and water quality of river basins, lake and reservoir ecosystem management, decision support for air protection, waste and wastewater management and land use. Other problems discussed are monitoring of ecosystems, environmental evaluation systems, spatio-temporal simulations of ecological systems, uncertainty, stability and 3D-visualisation of ecological processes, methods for visualising ecological processes and new developments of ecological modelling and simulation including neural networks and genetic algorithms.

Conclusions

The Expert Group 4.6.3 *Tools of modelling and simulation for environmental applications*, now named as *Simulation for environmental sciences and geosciences* has a good tradition. In 1992, R. Grützner formed a specialist group of information scientists and engineers by its valuable activities. The group discussed not only the development of new informatic tools for environmental applications, but integrated scientists from other disciplines as mathematics, chemistry, biology, and other environmental sciences. His excellent work was continued by J. Wittmann, who opened the group to a broader spectrum of

applications and informatic tools in a very communicative way. He invited new research groups to contribute actively to the topics discussed at the yearly workshops. Independent from the German Society of Informatics A. Gnauck created on the base of his university department a special discussion platform for ecologists and environmental engineers dealing with modelling and simulation. The combination of these two specialist groups gave the possibility to enlarge the number of topics to be discussed and to achieve more research groups working in the fields of environment. Now it can be stated that by combination of the two specialist groups an expert group of the TC 4.6 is formed really.

The future development of EG 4.6.3 will be seen in various directions. First, a close cooperation with other modelling and simulation groups which are organised in other GI-TC or outside of GI will be aspired. Especially, new methodological aspects of environmental systems modelling and simulation have to be considered. Second, workshop topics will be enlarged to stronger combinations of informatic tools and soft skills where sustainable management alternatives should be included. Third, new theoretical developments for visualising environmental states to support environmental decision making and to detect and to simulate adaptive environmental processes should be undertaken. For all these environmental features it is necessary to develop informatic tools as well.

References

Aspinelli, R. (2001): Environmental GIS and Data Analysis. Wiley, Chichester.

Bandini, S., Worsch, T. (eds.) (2001): Theoretical and Practical Issues on Cellular Automata. Springer, Berlin.

Bossel, H. (1992): Modellbildung und Simulation. Vieweg, Braunschweig Wiesbaden.

Brebbia, C. A. and P. Pascolo (eds.), 1998: GIS Technologies and their Environmental Applications. WIT Press, Southampton.

Brebbia, C. A. and P. Pascolo (eds.), 2000: Management Information Systems. WIT Press, Southampton.

Costanza, R. (ed.) (1991): Ecological economics: The science and management of sustainability. Columbia Univ. Press, New York.

Denzer, R., Russel, D., Schimak. G. (eds.) (1995): Environmental Software Systems. Chapman & Hall, London.

Edwards, D. (2001): Introduction to Graphical Modelling. Springer, Berlin.

Elmasri, R., Navathe, S. B. (2000): Fundamentals of database systems. 3rd ed., Addison-Wesley, Reading.

Gilpin, A. (2000): Environmental Economics. Wiley, Chichester.

Gronewold, A., Sonnenschein, M. (1998): Event-based modelling of ecological systems with asynchronous cellular automata. Ecol. Modelling **108**(1/3), 37-52.

Günther, O. (1998): Environmental Information Systems. Springer, Berlin.

Grützner, R. (1995): Environmental Modelling and Simulation – Applications and Future Requirements. In: Denzer, R., Russel, D., Schimak. G. (eds.): Environmental Software Systems. Chapman & Hall, London, pp. 113-122.

Grützner, R. (Hrsg.) (1997): Modellierung und Simulation im Umweltbereich. Vieweg, Braunschweig/Wiesbaden.

Haan, C. T. (2002): Statistical Methods in Hydrology. 2nd ed., Iowa State Press, Ames.

Haasis, H.-D. (1996): Betriebliche Umweltökonomie. Springer, Berlin.

Hobbs, B. F., Meier, P. (2000): Energy Decisions and Environment: A Guide to the Use of Multicriteria Methods. Kluwer, Boston.

Immon, W. H. (1992): Building the data warehouse. Wiley, New York.

Jakeman, A. J., M. B. Beck and M. J. M. J. McAleer (eds.) (1993): Modelling Change in Environmental systems. Wiley, New York.

Jørgensen, S. E. (1992): Integration of Ecosystem Theories: A Pattern. Kluwer Acad. Publ., Dordrecht.

Mateu, J., Montes, F. (2001): Spatial Statistics. WIT Press, Southampton.

Melli, P., Zanetti, P. (eds.) (1992): Environmental Modelling. Elsevier, Southampton.

Rautenstrauch, C. (1999): Betriebliche Umweltinformationssysteme. Springer, Berlin.

Sonnenschein, M, Vogel, U. (2001): Asymmetric cellular automata for the modelling of ecological systems. In: Hilty, L. M., Gilgen, P. W. (eds.): Sustainability in the Information Society. Part 2. Metropolis, Marburg, pp. 631-636.

Schnoor, J. (1996): Environmental Modeling. Wiley, Chichster.

Straškraba, M., Gnauck, A. (1985): Freshwater Ecosystems. Fischer, Stuttgart/Jena..

Thinh, N. X. (2003): Contemporary Spatial Analysis and Simulation of the Settlement Development of the Dresden City Region. In: Gnauck, A., Heinrich, R. (eds.): The Information Society and Enlargement of the European Union. Part 1, Metropolis, Marburg, pp. 253-261.

Weber, W. Jr. (2000): Environmental Systems and Processes. Wiley, Chichster.

White, R., Engelen, G. (1993): Cellular automata and fractal urban form: A Cellular modelling approach to the evolution of urban land-use patterns. Environment and Planning A, **25**, 1175-1199.

Wierzbicki, A. P., Makowski, M., Wessels, J. (eds.) (2000): Model-Based Decision Support Methodology with Environmental Applications. Kluwer, Dordrecht.

Wisniewski, M., Klein, J. (2001): Critical Path Analysis and Linear Programming. Palmgrave, Basingstoke.

Corresponding author: Jochen Wittmann,
University of Hamburg, Dept. of Informatics,
Vogt-Kölln-Straße 30, 22527 Hamburg, Germany.
wittmann@informatik.uni-hamburg.de

ASIM ASIM

ASIM - Buchreihen / ASIM Book Series

Fortschritte in der Simulationstechnik (FS) / Frontiers in Simulation (FS) Proceedings Conferences - Monographs, Proceedings:

I. Troch, F. Breitenecker (eds): *Proceedings MATHMOD 09 - Abstract Volume / Full Papers CD Volume*. Proc. 6th Conference Mathematical Modelling Vienna, February 2009, Vienna; ARGESIM Reports no. 34 & no. 35, ASIM / ARGESIM Vienna, 2009; ISBN 978-3-901608-34-6, ISBN 978-3-901608-35-3

M. Rabe (ed.): *Advances in Simulation for Production and Logistics Applications*. Proc. 13. ASIM-Fachtagung Simulation in Produktion und Logistik, October 2008, Berlin; Fraunhofer IRB-Verlag, Stuttgart, 2008, ISBN 978-3-8167-7798-4.

B. Zupančič, R. Karba, S. Blažič (eds.): *Proceedings EUROSIM 2007 - Abstract Volume / CD Volume*. Proc. 6th EUROSIM Congress on Modelling and Simulation, Sept. 2007, Ljubljana, Slovenia. ARGESIM Report no. 32, ASIM / ARGESIM Vienna, 2007; ISBN 978-3-901608-32-2.

S. Collisi-Böhmer, O. Rose, K. Weiß, S. Wenzel (Hrsg.): *Qualitätskriterien für die Simulation in Produktion und Logistik*. AMB 102, Springer, Heidelberg, 2006; ISBN 3-540-35272-4.

M. Rabe, S. Spiekermann, S. Wenzel (Hrsg.): *Verifikation und Validierung für die Simulation in Produktion und Logistik*. AMB 103, Springer, Heidelberg, 2006; ISBN 3-540-35281-3.

W. Borutzky: *Bond Graphs Methodology for Modelling Multidisciplinary Dynamic Systems*. FS 14, ISBN 3-936150-33-8, 2005.

Fortschrittsberichte Simulation (FB) - ARGESIM Reports (AR) - Special Monographs, PhD Theses, ASIM Workshop Proceedings

D. Leitner: *Simulation of Arterial Blood Flow with the Lattice Boltzmann Method* ARGESIM Report 16, ASIM / ARGESIM Vienna, 2009; ISBN 978-3-901608-66-7.

Th. Löscher: *Optimisation of Scheduling Problems Based on Timed Petri Nets*. ARGESIM Report 15, ASIM / ARGESIM Vienna, 2009; ISBN 978-3-901608-65-0.

R. Fink: *Untersuchungen zur Parallelverarbeitung mit wissenschaftlich-technischen Berechnungsumgebungen*. ARGESIM Report 12, ASIM / ARGESIM Vienna, 2008; ISBN 978-3-901608-62-9.

M. Gyimesi: *Simulation Service Providing als Webservice zur Simulation Diskreter Prozesse*. ARGESIM Report 13, ASIM / ARGESIM Vienna, ISBN 3-901-608-63-X, 2006.

J. Wöckl: *Hybrider Modellbildungszugang für biologische Abwasserreinigungsprozesse*. ARGESIM Report 14, ASIM / ARGESIM Vienna, ISBN 3-901608-64-8, 2006.

H. Ecker: *Suppression of Self-excited Vibrations in Mechanical Systems by Parametric Stiffness Excitation*. ARGESIM Report 11, ISBN 3-901-608-61-3, 2006.

C. Deatcu, P. Dünow, T. Pawletta, S. Pawletta (eds.): Proceedings 4. ASIM-Workshop Wismar 2008 – Modellierung, Regelung und Simulation in Automotive und Prozess-automation. ARGESIM Report 31, ASIM / ARGESIM Vienna, ISBN 978-3-901608-31-5, 2008.

J. Wittmann, H.-P. Bader (Hrsg.): *Simulation in Umwelt- und Geowissenschaften - Workshop Dübendorf 2008*. Shaker Verlag, Aachen 2008, ISBN 978-3-8322-7252-4.

A. Gnauck (Hrsg.): *Modellierung und Simulation von Ökosystemen - Workshop Kölpinsee 2006*. Shaker Verlag, Aachen 2007, AM 107; ISBN 978-3-8322-6058-3.

Available / Verfügbar: ASIM / ARGESIM Publisher Vienna - WWW.ASIM-GI.ORG
SCS Publishing House e.V., Erlangen, WWW.SCS-PUBLISHINGHOUSE.DE
Download ASIM Website WWW.ASIM-GI.ORG (partly; for ASIM members),
Bookstores / Buchhandlung, tw. ermäßigter Bezug für ASIM Mitglieder WWW.ASIM-GI.ORG

ARGESIM REPORTS ARGESIM REPORTS

Nachdem der Endabstieg der beiden Mars Rover unter Tausenden von atmosphärischen Bedingungen simuliert wurde, entwickelte und testete das Ingenieur-Team ein ausfallsicheres Bremsraketen-System, um eine zuverlässige Landung zu garantieren. Das Resultat – zwei erfolgreiche autonome Landungen, die exakt gemäß der Simulation erfolgten.
Mehr hierzu erfahren Sie unter:
www. mathworks.de/mbd

MATLAB®
&SIMULINK®

The MathWorks
Accelerating the pace of engineering and science

EUROSIM 2010

organised by CSSS

September 2010, Prague, Czech Republic

EUROSIM 2010

7th EUROSIM Congress on Modelling and Simulation

Eurosim Congress the most important modelling and simulation event in Europe

September 5-10, 2010, Prague, Czech Republic

Congress Venue
The Congress will take place in Prague, the capital city of Czech Republic, at the Congress Center of Masaryk College, part of Czech Technical University, in cooperation with the Faculty of Electrical Engineering of CTU.

About Czech Technical University in Prague
Czech Technical University celebrates 300 years of its history in 2007. Under the name Estate Engineering Teaching Institute in Prague was founded by the rescript of the Emperor Josef I of 18 January 1707 on the basis of a petition of Christian Josef Willenberg (1676-1731). This school was reorganized in 1806 as the Prague Polytechnic, and, after the disintegration of the former AustroHungarian Empire in 1918, transformed in to the Czech Technical University in Prague.

About EUROSIM
EUROSIM, the federation of European simulation societies, was set up in 1989. Its purpose is to promote, especially through local simulation societies, the idea of modelling and simulation in different fields, industry, research and development. At present, EUROSIM has 14 full members and 4 observer members.

Congress Scope and Topics
The Congress scope includes all aspects of continuous, discrete (event) and hybrid modelling, simulation, identification and optimisation approaches. Contributions from both technical and non-technical areas are welcome. Two basic tracks will be organized: M&S Methods and Technologies and M&S Applications.

Czech Republic - EUROSIM 2010 Host Country
The Czech Republic is a country in the centre of Europe. It is interesting for its 1,000-year-long history, rich culture and diverse nature. The country is open to new influences and opportunities thanks to a high level of industrial infrastructure, safety measures and plural media. The location of the Czech Republic in the very heart of Europe contributes to the fact that one can get there easily and fast. Usually all it takes to enter the country is a valid passport. The Czech Republic belongs to the Schengen zone. The need for a visas to enter the Czech Republic is very exceptional.

Prague - EUROSIM 2010 Host City
Prague is a magical city of bridges, cathedrals, gold-tipped towers and church spires, whose image has been mirrored in the surface of the Vltava River for more than a millennium.Walking through the city, you will quickly discover that the entire history of European architecture has left splendid representatives of various periods and styles. There are Romanesque, Gothic, Renaissance, Baroque and Classicist buildings, as well as more modern styles, such as Art Nouveau and Cubist. A poet once characterized Prague as a symphony of stones.

About CSSS
CSSS (The Czech and Slovak Simulation Society) has about 150 members in 2 groups connected to the Czech and Slovak national scientific and technical societies (Czech Society for Applied Cybernetics and Informatics, Slovak Society for Applied Cybernetics and Informatics). Since 1992 CSSS is a full member of EUROSIM.

Invitation
Czech and Slovak Simulation Society is greatly honored with the congress organisation and will do the best to organise an event with a high quality scientific programme with some other acompanied actions but also with some unforgettable social events.

Mikuláš Alexík, EUROSIM president,
Miroslav Šnorek, president of CSSS, EUROSIM 2010 Chair