DEVS as Part of and Integrating Tool in a Course on System Dynamics.

Ernesto Kofman[(1)] and Sergio Junco[(2)]
kofman, sjunco @fceia.unr.edu.ar
[(1), (2)] Departamento de Electrónica, FCEIA, Universidad Nacional de Rosario, Argentina.
[(1)] CONICET.

Abstract

This paper describes the introduction of the DEVS formalism in a course on System Dynamics in the computer science career at the National University of Rosario. The experience, which started in 2001, attempts to show that DEVS can play an important role not only in the context of discrete event modeling and simulation but also as a tool which helps to integrate the main concepts of general system theory in a language which is familiar to computer science students. Besides giving a detailed description of the concepts developed in the course, we also present the lab works, homework problems, and term project assigned to the students, as well as the software tools used in the course.

Keywords: DEVS, System Dynamics, Education, Differential Equations.

1. INTRODUCTION.

The DEVS formalism [Zeigler 1976] is one of the most important tools in the context of discrete event modeling and simulation. Its original scope has been extended in several directions, among which the provision of a unified framework for the representation of hybrid systems [Praehofer 1991; Zeigler et al., 2000], and the discrete-event modeling of numerical approximations of differential equations [Zeigler and Lee 1998; Kofman and Junco 2001] are the most relevant to this article.

This modeling and simulation paradigm is easy to understand by students of computer science because of several reasons: DEVS-models are described with a language that looks like the grammars usual in automata theory; the hierarchical structure of coupled DEVS-models immediately reminds the main concepts of object oriented programming; and most of the simple examples often used to introduce DEVS come from the computer science area (the classic queue and processor system, for instance).

Being conceived within the realm of Systems Theory, the general specification of DEVS is similar to the axiomatic definition of continuous dynamical systems [Kalman et al., 1969]. This fact allows for a quite straightforward conceptual connection at the highest level of abstraction between DEVS and state equations models of continuous systems, when the latter are considered in terms of the transition functions provided by the formal solutions of the differential equations.

The above mentioned properties motivated us to incorporate DEVS in the syllabus of an undergraduate course in computer science that was originally intended to give the students some exposure to the basic issues of differential equations and their usage as models of continuous dynamical systems. The goal was to transmit –in the limited time-frame allotted to the course- the specifics of continuous, discrete-time and discrete-event dynamical systems, to present them as particular instances of a more general concept of dynamical system, and to simultaneously discuss some general issues related to systems theory (like structure and behavior, modularity, coupling, hierarchy, etc.).

This one-semester course starts with a concise introduction to ordinary differential equations, including basic techniques of quadrature and numerical integration, and continues presenting the analysis of dynamic systems in the state space (general solution of the state equation, phase portraits, linearization, Lyapunov stability theorems).

Matlab/Simulink-supported simulation labs are included at this stage to provide confidence with numerical methods and dynamical behavior. Then, the DEVS formalism is introduced and, finally, all the concepts are integrated together in a term project which consists in the modeling and simulation of a complex hybrid system. This hybrid system consists of an elevator control system involving a DC-motor (which is modeled by a Runge-Kutta approximation of its differential equations), sensors, actuators, and an asynchronous controller to be designed by the students in order to meet a set of specified behavioral specifications. The simulations are done using PowerDEVS [Kofman et al., 2003], a software which is previously used in a practical work by the students to simulate a simple "queue-processor" system.

In that way, DEVS is used not only with modeling and simulation purposes but also as a tool that helps to generalize some system theoretical concepts, to make the students familiar with classic numerical methods, with automatic control issues and with the application of system theory in realistic problems.

This paper describes in detail the mentioned course, its lab-exercises and term projects in order to transmit the experience to other professors and to open a discussion about the advantages and disadvantages of introducing DEVS in undergraduate courses.

2. DESCRIPTION OF THE COURSE.

The course begins with an introduction to differential equations following a classic approach (theorem of existence and uniqueness of solutions, first order equations, order reduction , etc.) and goes on showing their application to the study of continuous system dynamics. Taking into account the background of the students, the examples are mainly related to mechanical systems.

After the basic definitions, the main techniques for quadrature of first order differential equations are introduced (separation of variables, linear equations, homogeneous cases, Bernoulli equations, etc.). Together with the reduction of order, state equations and block diagrams are presented as alternative representations of n-th order ordinary differential equations, which are not further treated. A short overview of the numerical approach to ODE-solving is giving, wherby the most classical methods approximations are presented (Euler, Runge-Kutta, Predictor-Corrector methods including implicit and variable step versions), and important issues are discussed, like accuracy and stability, stiff systems and some remarks about differential-algebraic equations.

These topics complete the first part of the course, which is complemented with a lab work in which the students simulate different systems of differential equations using the Matlab/Simulink environment, including an example consisting in a filter for audio signals. After a first middle term exam evaluating all these subjects, we consider that the students have acquired the necessary background to be introduced into the basic concepts of continuous system dynamics.

The second part of the course is intended to introduce the basic tools for the analysis of dynamical systems in the state space. It begins with the study of general properties of the solution of (possibly) nonlinear, time-varying state equations, with succesive restrictions to the linear and linear time-invariant (LTI) cases. Next, for second order LTI-systems, a detailed study of their state space trajectories and invariant properties of their eigenspaces is carried-out, emphasizing the role of the evolution and transition matrices.

After that, the course comes back to nonlinear systems showing how to analyze their local behavior around equilibrium points. The technique of linearization is introduced as a preambule to the first method of Lyapunov for stability analysis and to the Hartman-Grobman theorem for approximate trajectory determination. Finally, the second method of Lyapunov, complemented with the Invariance Principle of LaSalle, is introduced.

This preview of the theory of continuous system dynamics concludes with a lab work that involves the analysis and simulation of different dynamical systems: A Lotka-Volterra model, a pendulum, a nonlinear spring--mass system, a Van der Pol oscillator, an oxidation process and a Duffing equation. In these examples the students deal with most of the features which appear related to continuous dynamical systems: multiple equilibrium points; stable, marginally stable and unstable equilibria; arriving to chaotic behavior in the (controlled) Duffing example. A second middle term examination evaluates this part of the course.

The last part of the course begins with an introduction to the theory of systems starting from a review of the general solution of the state equation. This is taken to introduce the axiomatic definition of a continuous dynamical system [Kalman et al., 1969] which is then linked with the general definition of a dynamical system [Zeigler et al., 2000].

This part is mainly based on Zeigler's book "Theory of Modelling and Simulation" and after treating the basic concepts of system theory, the DEVS formalism is introduced and developed.

The formalism is first presented in its atomic version, including some classic examples like processors and queues but also showing other examples which then play a key role in the simulation of continous systems like static functions and integrators [Kofman and Junco 2001]. Some theoretical topics like legitimacy and the relationship between a (generic) DEVS-model and the general dynamical system that it defines are also shown at this stage.

In this way, two different connections between DEVS and differential equations are seen. The first one corresponds to the fact that both formalisms define dynamical systems, as it can be seen at their most abstract level of specification. The second one has to do with the fact that DEVS models can represent approximations of differential equation systems.

Coupled DEVS models are then introduced simultaneously with the concept of hierarchical coupling. Although we start introducing the coupling with translation functions, we then emphasize the use of input and output ports. The property of closure under coupling and its proof are then treated. We deal with this proof not only to show formally the hierarchical features of DEVS but also to explain in deeper detail the way in which coupled DEVS models work. Understanding this proof considerably helps to the comprehension of coupled DEVS dynamics.

These theoretical concepts are illustrated with examples including the typical queue-processor systems and also some approximations to continuous systems.

The last topic which is treated in the course is related to DEVS simulation. Here, we introduce the basic simulation scheme for coupled DEVS models as it is developed in [Zeigler et al., 2000] and we introduce the software environment PowerDEVS [Kofman et al., 2003]. This software is first used in a lab work involving all the topics related to the theory of DEVS developed during the course.

The course concludes with a term project which integrates the main concepts that were treated. This project consists in the design, modeling, simulation and analysis of an hybrid control system for an elevator. The model includes the continuous dynamics of the elevator, its electrical drive (a DC Motor), and an asynchronous discrete control law, all of them interfaced by sensors and actuators. Solving the project requires manipulating the differential equations, obtaining numerical approximations, modeling the discrete parts with DEVS, implementing the DEVS models in PowerDEVS and analyzing the simulation results.

A detailed description of the DEVS lab work and the term project is given in the next section.

3. DEVS LAB WORK AND TERM PROJECT

As we mentioned before, the lab work and the project are based on the use of PowerDEVS. This software environment offers all the features which are needed to perform the simulations with a direct translation from the DEVS models.

The lab work consists in analyzing via modeling and simulation a system consisting in several queue-processor sub-systems which are fed by jobs coming from a random generator and distributed by different switches. The students should build the atomic models of the generator, queue and processor and then program them as blocks of PowerDEVS.

Figures 1 and 2 show the PowerDEVS model of the system. There, the Queue-Processor sub-model makes use of hierarchical coupling.

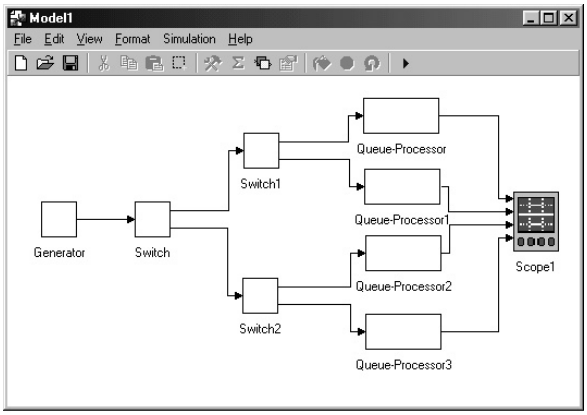

Figure 1. PowerDEVS model of a parallel processing system.

The first goal of the simulations is to establish the relationship between the mean time between consecutive jobs and their mean processing time so that all the jobs are processed (the queues are assumed to have a finite capability so they can loss jobs if they arrive faster than they can be processed).

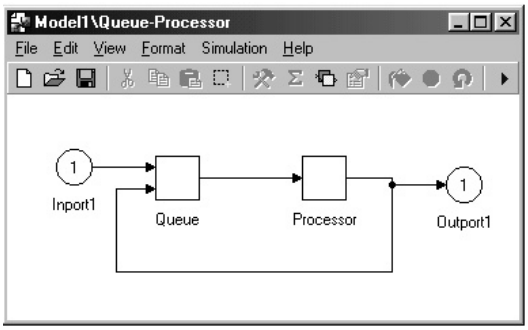

Figure 2. Queue-Processor sub-system.

A legitimacy analysis of the atomic models is also performed. Then the connections are modified so that the output of the processors is fed back to the system input showing that legitimacy is not closed under coupling (the whole model becomes illegitimate despite the fact that all its sub-models are legitimate).

After doing this lab work, the students have developed skills related to DEVS modeling and analysis and they have become familiar with the usage of PowerDEVS and we consider that they are ready to get involved with a more complex system.

The term project was conceived to integrate the main concepts of the course. Using the DEVS formalism as a tool of representation of the different subsystems, the students must deal with differential equations and their numerical approximations, they should design and obtain the DEVS model of an asynchronous control system, they have to model sensors and actuators, and they must analyze simulation results which involve stability and other continuous system dynamic features.

As already said, the system consists of an elevator (driven by a DC motor), a control subsystem, sensors and actuators. There is also a “generator” which simulates the users that call the elevator from different floors in a random way.

A first scenario implies a kind of open-loop control system, as sensors are provided that can detect the presence of the elevator only at discrete places: they are placed within a distance of 1m to each other. The height of the floors is 3mts. The control uses the information given by the sensors (events provoked when the elevator passes through them) and it can only take three different actions: setting the armature voltage of the DC motor to +V, to -V or applying a brake. The brake can be only applied when null speed is detected by another specific sensor.

A state equation-model of elevator and motor is given and the first thing to be done by the students is to approximate it with a numerical method (fourth order Runge-Kutta is suggested) and to simulate it in order to determine the delay necessary after passing a sensor to invert the voltage so that the elevator stops in the following floor.

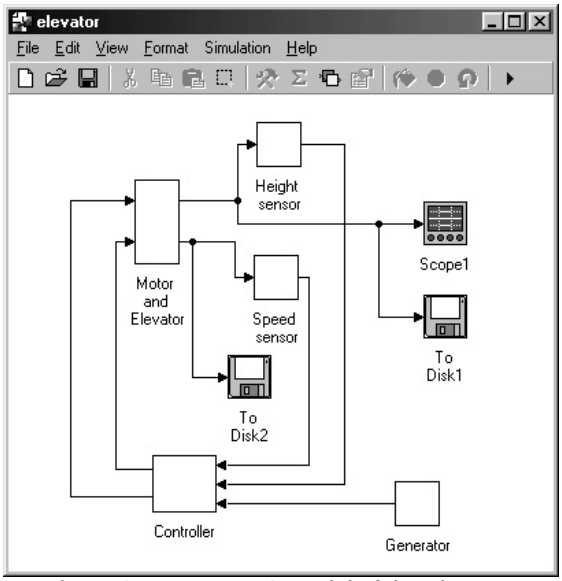

Figure 3. PowerDEVS model of the elevator.

Using this idea the students must design the controller. The system is completed with the model of the sensors (which produce events when the elevator height reaches integer numbers) and the random generator (Fig.3).

The model is validated with different simulations under different conditions (weight and voltage variations). As this is a sort of open loop control, its performance is limited and variations of the input voltage produce undesired behaviour.

In order to improve the performance of the control system the hypothesis are changed: it is supposed that the height can be continuously measured and that the armature voltage can be continuously modified. In that way, it is possible to implement a proportional control, which is analyzed for different parameters and situations. This new scenario allows to introduce (only heuristically) some concepts of closed-loop control, and to give a taste of its power.

Both, the lab work and the term project, are approved with reports including the complete models, the simulation results, the analysis and conclusions.

4. CONCLUSIONS AND FUTURE DIRECTIONS

We showed in this paper the way in which we included the DEVS formalism in the syllabus of an undergraduate course on system dynamics for computer science students.

We evaluate it as a fruitful experience which allows to integrate the main concepts of system dynamics using a language which is familiar to the students.

Besides being a tool which connects theoretical topics, DEVS gives the students the possibility of implementing the practical issues related to the theory in applications that they find interesting.

Up to this moment, we introduced only classic numerical methods. Beggining from this year, we will also suggest the students to simulate and compare the results with quantization based methods which are best suited for the simulation of hybrid systems [Kofman 2004] and are already implemented in PowerDEVS.

It would be also desirable to include a short introduction to discrete time systems showing also the way in which they can be represented by DEVS. Although the students do it in the term project (they represent by DEVS a Runge-Kutta approximation) we are not exploiting this from a theoretical point of view.

In spite of the integration role it plays, it is still quite hard to exaplain the connection between DEVS and the other topics. It usually happens that the students become aware of the relationship after they finish the course.

A solution to this problem might consist in reordering the topics. However, the lack of previous knowledge on differential equations enforces to pay special attention to the continuous systems leaving only a small fraction of the course dedicated to general system theory.

5. REFERENCES

R. Kalman, P.L. Falb and M. A. Arbib. 1969. *Topics in Mathematical System Theory*, McGraw-Hill.

Ernesto Kofman and Sergio Junco, 2001, “Quantized State Systems. A DEVS Approach for Continuous Systems Simulation". Transactions of SCS, 18, no.3, 123-132.

Ernesto Kofman, Marcelo Lapadula and Esteban Pagliero, 2003, “PowerDEVS: A DEVS–Based Environment for Hybrid System Modeling and Simulation”, Technical Report LSD0306, LSD, Universidad Nacional de Rosario. Submitted.

Ernesto Kofman, 2004, “Discrete Event Simulation of Hybrid Systems”. SIAM Journal on Scientific Computing 25 no.5, 1771-1797.

Herbert Praehofer, 1991, “System Theoretic foundations for combined Discrete-Continuous System simulation.” PhD thesis, J.Kepler University of Linz.

Bernard Zeigler, 1976. *Theory of Modeling and Simulation*. John Wiley & Sons, New York.

Bernard Zeigler and J.S.Lee, 1998, “Theory of quantized systems: formal basis for DEVS/HLA distributed simulation environment”. In SPIE Proceedings, 49 –58.

Bernard Zeigler, Tag Gon Kim and Herbert Praehofer, 2000. *Theory of Modeling and Simulation*. Second edition. Academic Press, New York.