Teaching Kinematics and Dynamics of Multibody Mechanical System Using the Object Oriented Language Modelica

E. Seabra and J. Machado
University of Minho, Guimarães, Portugal

Abstract—**A new modeling language, called Modelica, for physical modeling is being developed in an international effort. The main objective is to make it easy to exchange models and model libraries for different domains, such as, mechanical, pneumatics, electrical, hydraulics, and others. The design approach builds on non-causal modeling with true ordinary differential and algebraic equations and the use of object-oriented constructs stemming from modern software development, (hierarchy, encapsulation) to facilitate reuse of models and model parts. This paper gives an overview of the use of the object oriented language Modelica with the mechanical Multibody Library to model and simulate three-dimensional mechanical systems.**

Index Terms—**Modelling and simulation, Multibody systems, Object oriented language, Modelica language.**

I. INTRODUCTION

It is well known that over the last few years, a number of commercially available software programs became powerful tools for engineers and designers, being an alternative to the traditional approaches based on analytical analysis.

Yet, in this paper, the computer capabilities are used in a teaching perspective. In fact, the computer aided analysis software can be used to study kinematic and dynamic characteristics of motion of simple and complex systems. Design software simulation allows testing design performance and simulating the behaviour of all components prior to building a physical prototype. This enables the determination of the most critical situation and, consequently, the design of the mechanical components and motion generator needs.

The traditional approach of teaching based on 'chalk and talk' attempts to transmit knowledge from the teacher to a passive recipient. However, during the last years, there is a growing awareness among engineering educators that while this style of instruction is suitable for teaching engineering analysis, it has some limitations when it comes to nurturing creativity, synthesis and engineering design, where different possible solutions have to be tested [1], [2] and [3]. Thus, it is essential to combine teaching by lectures and active learning techniques in order to have high motivation and participation of the students and, consequently, to reach a better understanding of the issues taught. In addition, the computer packages related to the mechanical systems in general become a powerful alternative to the classic approach, and promote students' creativity. Furthermore,

for 2D and 3D complex systems, the traditional mathematical analysis of the equations of motion can be a hard and difficult task, if not impossible in some cases. The modern software simulation programs provide an efficient way to help students in the active learning process [4]. Moreover, the reform movement in engineering education inspired by Engineering Criteria of ABET (Accreditation Board of Engineering Technology) is consistent with this method [5].

In this paper, the virtual reality capabilities are used in teaching kinematics and dynamics of multibody mechanical systems. The remainder of the paper is organized as follows. In section two the state of the art of the computational programs are presented. The main features of the object-oriented Modelica language are described in section three. Section four includes the computational modelling and simulation of multibody mechanism systems. Finally, in the last section, the main conclusions from this study are drawn.

II. COMPUTER-AIDED SIMULATION'S STATE OF THE ART

In recent years a number of computer-based aids for engineering has emerged, due not only to the development of hardware, but also owing to the improvements of software and mathematical tools.

There is a large amount of simulation software on the market. All languages and model representations are proprietary and developed for certain tools. There are general-purpose tools such as ACSL, SIMULINK and System Build. They are based on the same modelling methodology, input-output blocks, as in the previous standardization effort, CSSL, from 1967 [6]. There are domain-oriented packages: electronic programs (SPICE, Saber), multibody systems (ADAMS, DADS, SIMPACK), chemical processes (ASPEN Plus, SpeedUp) etc. With few exceptions, all simulation packages are only strong in one domain and are not capable of modelling components in other domains reasonably. This is a major disadvantage since technical systems are becoming more and more heterogeneous with components from many engineering domains.

Techniques for general-purpose physical modelling have been developed during the last decades, but did not receive much attention from the simulation market. The modern approaches build on non-causal modelling with true equations and the use of object-oriented constructs to facilitate reuse of modelling knowledge. There are already

several such modelling languages with such a support available from universities and small companies.

Examples of such modelling languages are ASCEND [7], Modelica [8, 9], gPROMS [10], NMF [11], ObjectMath [12], Omola [13], SIDOPS+ [14], Smile [15], U.L.M. [16] and VHDL-AMS [17].

A new modelling language, called Modelica, for physical modelling is being developed in an international effort [18, 19]. The main objective is to make it easy to exchange models and model libraries for different domains, such as, mechanical, pneumatics, electrical, hydraulics, and others.

The aim of the Modelica effort is to unify the concepts of these languages in order to introduce common basic syntax and semantics and to design a new unified modelling language for model representation.

The work started in the continuous time domain since there is a common mathematical framework in the form of differential algebraic equation (DAE) systems and there are several existing modelling languages based on similar ideas. There is also significant experience of using these languages in various applications. The short range goal is to design a modelling language based on (DAE) systems with some discrete-event features to handle discontinuities and sampled systems. The design should allow an evolution to a multi-formalism, multidomain, general-purpose modelling language [8].

It is highlighted that Modelica is a textual language, and as such more relevant for software tool builders than for the physical systems modellers. This is rather obvious, since Modelica is meant as a new standard for exchange of models, model libraries and experiments.

The present work was performed with the software Dymola (Dynamic Modelling Laboratory) that uses the language Modelica.

III.   MODELICA LANGUAGE

The design of Modelica builds on two relevant modern concepts in modelling and simulation, namely non-causal modelling and the use of object-oriented constructs (encapsulation, inheritance and hierarchy) originally used in software engineering [20-22] and now essential for physical systems modelling.

Since Modelica accepts non-causal models, bond-graphs can be translated to Modelica code as submodels (i.e. a-causally). Bond Graphs are a domain-independent graphical notion of physical systems modelling. During modelling, the edges in the graph denote the ideal exchange of energy between the submodels (vertices). One can state that bond-graph modelling is in fact a form of object-oriented physical systems modelling.

In Modelica language the models and submodels are declared as classes, with interfaces that are called connectors. A connector must contain all quantities needed to describe the interaction. Attributes can be used to specify how the connections are converted to computable code. Modification of a model definition is possible using the extends construct. In this way, for refining a generic submodel into a more specific one, only the 'new' specific parts need to be described. The common parts are inherited from the more generic submodel, and need to be specified only once.

The Modelica language has been introduced by giving small examples. Model classes and their instantiation form the basis of hierarchical modelling, connectors and connections correspond to physical connections of components. At the lowest level, equations are used to describe the relation between the quantities of the model.

The expressive modelling power of Modelica is large. Some of the more powerful constructs are modelling of, for example, multi-body systems, control systems and approximation to partial differential equations is done conveniently by utilizing matrix equations. Multi dimensional matrices and the usual matrix operators and matrix functions are thus supported in Modelica. It is also possible to have arrays of components and to define regular connection patterns.

The software Dymola (Dynamic Modelling Laboratory) that uses the language Modelica, is suitable for modelling various kinds of physical systems. It supports hierarchical model composition, libraries of truly reusable components, connectors and composite a-casual connections. Dymola uses a new modelling methodology based on object orientation and equations. The usual need for manual conversion of equations to a block diagram is removed by the use of automatic formula manipulation. Other highlights of Dymola are: handling of large, complex multi engineering models; faster modelling by graphical model composition; faster simulation; symbolic pre-processing; open for user defined model components; open interface to other programs; 3D animation; real-time simulation. In the software Dymola model libraries are available in many engineering domains, such as the multibody library [23, 24], that was used in the present work.

IV.   MULTIBODY SYSTEM SIMULATION

In this section are presented three examples of simulation of Multibody Systems. Before simulating these systems, the students are invited to define the mathematical modelling of each system and, in a second step, the respective equations are inserted on the Dymola Software, using Modelica modelling Language. If possible, all the components of the Library must be used; if not, some of these components must be defined (by students, with help from the teacher).

A.   Simple pendulum

A simple pendulum consists of a body and a revolute joint with linear damping in the joint. Fig. 1 shows its build up as Modelica composition diagram.

Every model that uses model components from the MultiBody library must have an instance of the Modelica.Mechanics.MultiBody.World model on highest level. The reason is that in the world object the gravity field is defined (uniform gravity or point gravity), as well as the default sizes of animation shapes and this information is reported to all used components. If the World object is missing, a warning message is printed and an instance of the World object with default settings is automatically utilized (this feature is defined with annotations and is, e.g., supported by Dymola).
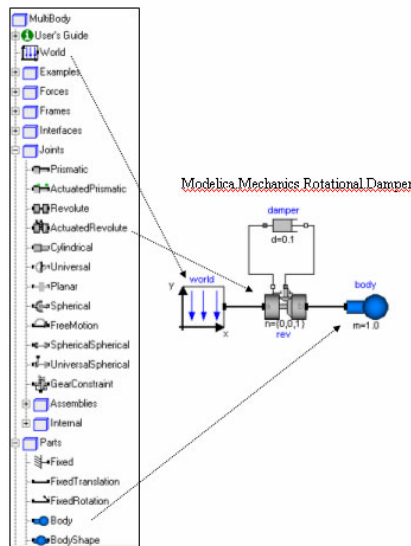
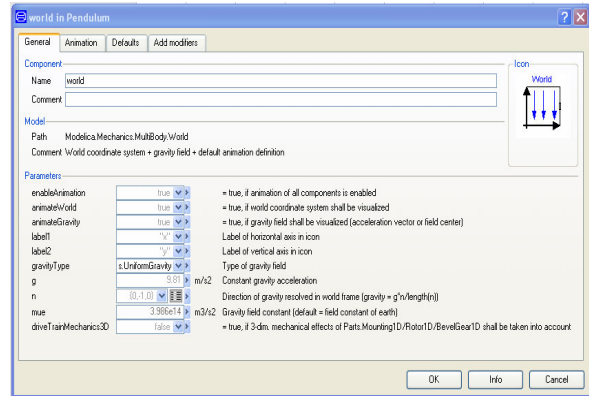Figure 1. Modelica diagram of the simple pendulum.
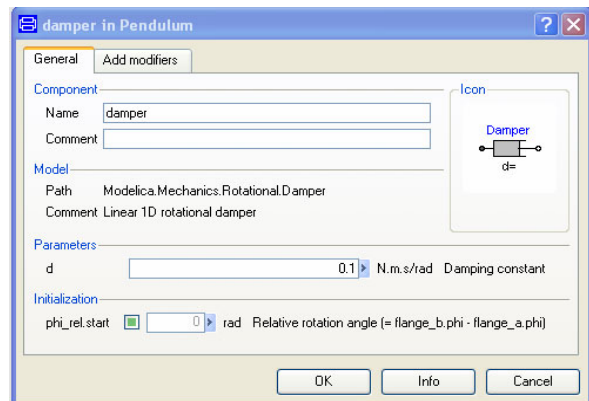

Figure 2. Parameter menus of the world

Figure 3. Parameter menus of the damper

Figure 4. Parameter menus of the revolute joint

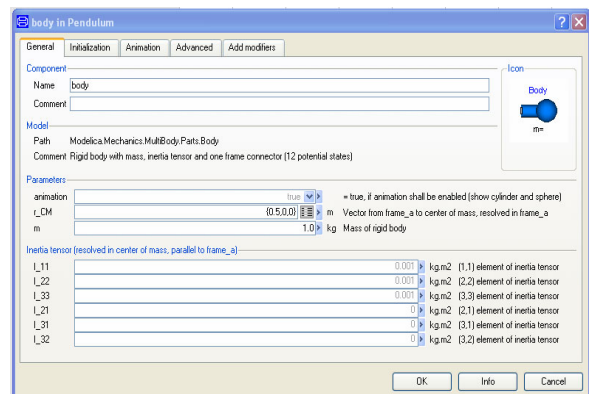Figure 5. Parameter menus of the body

In a second step the parameters of the dragged components need to be defined. Some parameters are vectors that have to be defined with respect to a local coordinate system of the corresponding component. The easiest way to perform this is to define a reference configuration of your multi-body model: in this configuration, the relative coordinates of all joints are zero. This means that all coordinate systems on all components are parallel to each other. Therefore, this just means that all vectors are resolved in the world frame in this configuration.

The reference configuration for the simple pendulum shall be defined in the following way: The y-axis of the world frame is directed upwards, i.e., opposite to the direction of gravity acceleration. The x-axis of the world frame is orthogonal to it. The revolute joint is placed at the origin of the world frame. The rotation axis of the revolute joint is directed along the z-axis of the world frame. The body is placed on the x-axis of the world frame (i.e., the rotation angle of the revolute joint is zero, when the body is on the x-axis). In Figs. 2, 3, 4, and 5 the definition of this reference configuration is shown in the parameter menus, respectively, of the world, damper, revolute joint and body.

Translate and simulate the model, e.g., with Dymola. Automatically, all defined components are visualized in an animation using default absolute or relative sizes of the components. For example, a body is visualized as a sphere and as a cylinder. The default size of the sphere is defined as a parameter in the world object. You may change this size in the "Animation" parameter menu of the body. The default size of the cylinder is defined relatively to the size of the sphere (half of the sphere size). With these settings Fig. 6 shows the animation. A Dymola feature is that all mechanical components have animation information with default sizes and colours.

The world coordinate system is visualized as a coordinate system with axis labels. The direction of the gravity acceleration vector is shown as green arrow. The red cylinder represents the rotation axis of the revolute joint and the light blue shapes represent the body. The centre of mass of the body is in the middle of the light blue sphere.
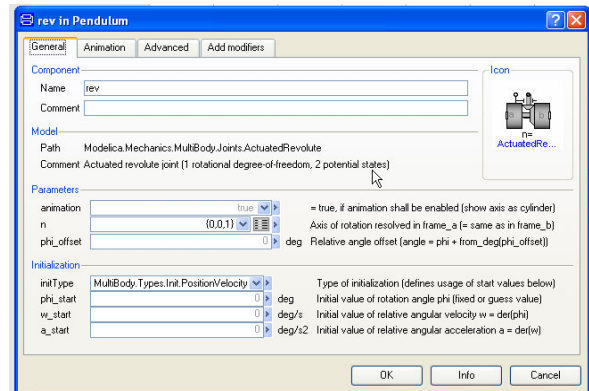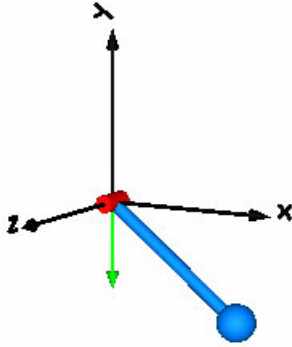
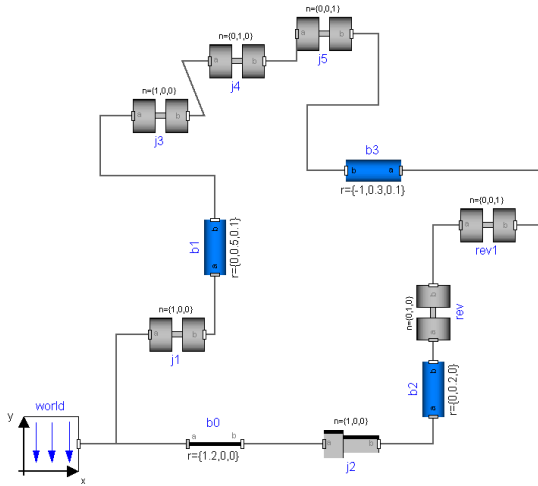Figure 6. Animation of the simple pendulum.

Figure 7. Modelica diagram of the four-bar mechanism

B. Four Bars Mechanism

This is a kinematic loop consisting of 6 revolute joints, 1 prismatic joint and 4 bars that is often used as a basic constructing unit in mechanisms. This four-bar mechanism was built with only revolute joints that correspond to 5 non-linear equations. Fig. 7 shows its build up as a Modelica composition diagram.

This case demonstrates that usually no particular knowledge of the user is needed to handle kinematic loops. Just connect the joints and bodies together according to the real system. In particular no cut-joints or a spanning tree has to be determined.

Fig. 8 shows the default animation. The axes of the revolute joints are represented by the red cylinders and the axis of the prismatic joint is represented by the red box on the lower right side. As an example, Fig. 9 shows kinematic results obtained by the Dymola software related with the displacement and velocity of the prismatic joint axis j2.

C. Cylinder engine

This is a model of the mechanical part of one cylinder of an engine. The combustion is not modelled. The "inertia" component at the lower left part is the inertia output of the engine driving the gearbox. Fig. 10 shows its build up as a Modelica composition diagram.

The engine is modelled solely by revolute and prismatic joints. Since these results in a planar loop there is the well known difficulty that the cut-forces perpendicular to the loop cannot be uniquely computed, as well as the cut

torques within the plane. This ambiguity is resolved by using the option planarCutJoint in the advanced menu of one revolute joint in every planar loop (joint B1). This option sets the cut-force in the direction of the axis of rotation, as well as the cut-torques perpendicular to the axis of rotation at this joint to zero and makes the problem mathematically well-formed. An animation of this case is shown in the Fig. 11.
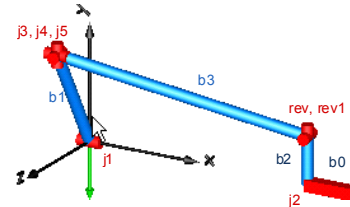

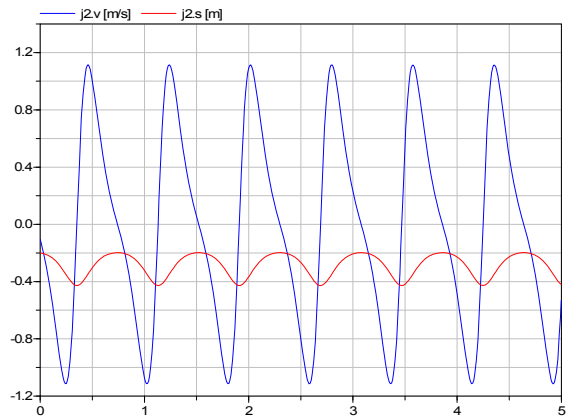Figure 8. Animation of the four-bar mechanism

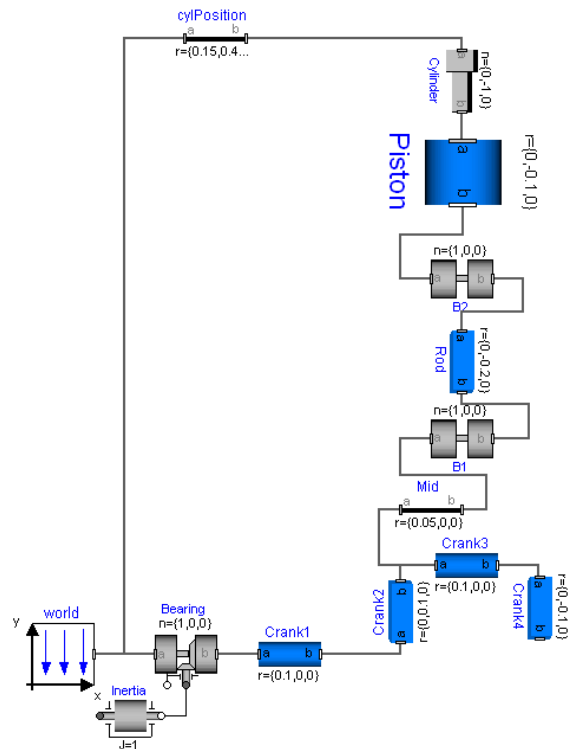Figure 9. Kinematics results of the prismatic joint axis j2 [axis xx – time(s)]
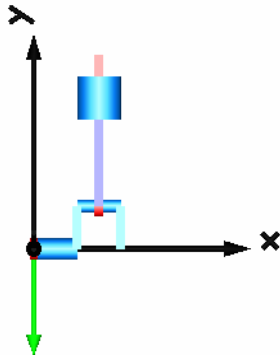

Figure 10. Modelica diagram of the one cylinder engine

Figure 11. Animation of the one cylinder engine

V. CONCLUSIONS

In this paper, an attempt to demonstrate the virtual reality capabilities in the teaching of kinematics and dynamics was done.

Thus, the visual display of the global motion produced using Dymola software provides an excellent correspondence between the abstract mathematics and a realistic animation of the physical reality. In general, students involved with this kind of project teaching have expressed enhanced understanding of the subject matter with the integration of the software tools. It is highly motivating, for the students, to see some animations and to observe the effect of all the mathematical equations that they developed before, They feel that all the developed effort is necessary and they also see the utility of all the complex models developed.

The teacher is, too, highly motivated because he can transmit the desired message, using this kind of modelling language and software. He can also demonstrate to the students that the hard tasks of mathematical modelling are needed and helpful and not so boring or of minor importance as some students suppose at the beginning of the study.

Graphical programming languages have many advantages compared to textual programming languages. They often allow programming in a style that closely mimics the way in which people model problems. Graphical languages fit users' existing perception of a problem, making the application easier to build, debug, document and maintain. An added benefit is the possibility to use colour and animation to provide feedback as the program executes.

This paper demonstrates the potentialities of the application of the software Dymola, using the graphical programming language Modelica, in the subject of the mechanical multibody systems.

REFERENCES

[1] A. Akay, "*New directions in Mechanical Engineering: Big-Ten-Plus Mechanical Engineering Department Heads*," Clearwater Beach, Florida, 2002.

[2] R.M. Lima, D. Carvalho, A.M. Flores, N. A. van Hattum-Janssen, "Case study on project led education in engineering: students and teachers' perceptions," *European Journal of Engineering Education*, vol. 32, no. 3, pp. 1-11, 2007. (doi:10.1080/030437 90701278599)

[3] J.C.F. Teixeira, J.F. Silva, P. Flores, "Development of Mechanical Engineering Curricula at the University of Minho," *European Journal of Engineering Education*, 2007.

[4] A. Akay, "The Renaissance Engineer: Educating Engineers in a Post-9/11 World," *European Journal of Engineering Education*, vol. 28, no. 2, pp. 145-150, 2003. (doi:10.1080/030437903100 0078979)

[5] G.D. Peterson, "Engineering Criteria 2000: a bold new change agent," *ASEE Prism*, September 1997, pp. 30-34, 1997.

[6] J.C. Strauss, "The SCi continuous system simulation language (CSSL)," *Simulation*, vol. 9, pp. 281–303, 1967. (doi:10.1177/003 754976700900601)

[7] P. Piela, T. Epperly, K. Westerberg, A. Westerberg, "ASCEND. An object-oriented computer environment for modeling and analysis: the modeling language," *Computers and Chemical Engineering,* vol.15, no. 1, pp. 53–72, 1991. (doi:10.1016/0098-1354(91)87006-U)

[8] H. Elmqvist, F. Cellier, M. Otter, "Object oriented modelling of power-electronic circuits using Dymola," *First Joint Conference of International Simulation Societies (CISS)*, The Society for Computer Simulation, ETH, Zurich, Switzerland, 1994.

[9] M. Pfennig, F. Thielecke, "Implementation of a Modelica Library for Simulation of High-Lift Drive Systems," *In Proceedings of the 6th International Modelica Conference*, Modelica' 2008, Bielefeld, Germany, 2008.

[10] P. Barton, C. Pantelides, "Modelling of combined discrete/continuous processes," *AIChE J.*, vol. 40, pp. 966–979, 1994. (doi:10.1002/aic.690400608)

[11] P. Sahlin, A. Bring, E.F. Sowell, "The Neutral Model Format for building simulation, Version 3.02," *Technical Report*, Department of Building Sciences. The Royal Institute of Technology, Stockholm, Sweden, 1996.

[12] P. Fritzson, L. Viklund, D. Fritzson, J. Herber, "High-level mathematical modelling and programming," *IEEE Software*, vol. 12, no. 3, 1995.

[13] E. Mattsson, M. Andersson, K. Åström, Object oriented modelling and simulation," In LINKENS, Ed., *CAD for Control Systems*, Marcel Dekker Inc, New York, pp.31–69, 1993.

[14] A.P. Breunese, J.F. Broenink, "Modelling mechatronic systems using the SIDOPS+ language," *In Proceedings of ICBGM'97*, 3rd International Conference on Bond Graph Modelling and Simulation, Simulation Series, The Society for Computer Simulation International, pp. 301–306, 1997.

[15] M. Kloas, V. Friesen, M. Simons, "Smile - A simulation environment for energy systems," *In Sydow*, Ed., Proceedings of the 5th International IMACS-Symposium on Systems Analysis and Simulation (SAS'95), Gordon and Breach Publishers vol.18–19 of Systems Analysis Modelling Simulation, pp. 503–506, 1995.

[16] A. Jeandel, F. Boudaud, P. Ravier, A. Buhsing, "U.L.M: Un Langage de Modélisation, a modelling language," *In Proceedings of the CESA'96 IMACS Multiconference*, IMACS, Lille, France, 1996.

[17] IEEE, "Standard VHDL Analog and Mixed-Signal Extensions," *Technical Report* IEEE 1076.1, 1997.

[18] S.E. Mattsson, E H. Elmqvist, J.F. Broenink, "Modelica™ - An international effort to design the next generation modelling language," *Journal A,* vol. 38, no. 3, pp.16-19, 1997.

[19] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen, *Object oriented modelling and design*, Prentice Hall, 1991.

[20] J. Åkesson, M. Gäfvert, H. Tummescheit, "JModelica—an Open Source Platform for Optimization of Modelica Models*," In Proceedings of the 6th Vienna International Conference on Mathematical Modelling*, MATHMOD 2009, Vienna, Austria, 2009.

[21] J. Åkesson, T. Ekman, G. Hedin, "Implementation of a Modelica Compiler using JastAdd Attribute Grammars," *Science of Computer Programming*, 2009.

[22] K. Stavåker, A. Pop, P. Fritzson "Compiling and Using Pattern Matching in Modelica," *In Proceedings of the 6th International Modelica Conference*, Modelica' 2008, Bielefeld, Germany, 2008.

[23] M. Höbinger, M. Otter, "PlanarMultiBody - A Modelica Library for Planar Multi-Body Systems," *In Proceedings of the 6th International Modelica Conference*, Modelica' 2008, Bielefeld, Germany, 2008.

[24] M. Höbinger, "PlanarMultiBody - A Modelica Library for Planar Multi-Body Systems," *In Proceedings of the 6th International Modelica Conference*, Modelica' 2008, Bielefeld, Germany, 2008.

AUTHORS

E. Seabra is with the Mechanical Engineering Department, University of Minho, 4800-058 Guimarães, Portugal (e-mail: eseabra@dem.uminho.pt).

J. Machado is with the Mechanical Engineering Department, University of Minho, 4800-058 Guimarães, Portugal (e-mail: jmachado@dem.uminho.pt).