# IMPLEMENTATION IN MODELICA OF A VIRTUAL-LAB FOR TESTING WASHING MACHINE DESIGNS

Carla Martín-Villalba
Alfonso Urquia
Sebastián Dormido
Departamento de Informática y Automática, UNED
Juan del Rosal 16, 28040 Madrid, Spain
{carla, aurquia, sdormido}@dia.uned.es

Félix Martínez
Ingeniería Mecánica, IKERLAN
J. M. Arizmendiarrieta 2, 20500 Arrasate, Gipuzkoa, Spain
Felix.Martinez@ikerlan.es

**KEYWORDS**
Interactive simulation, object-oriented modeling, hybrid-models, industrial processes

## ABSTRACT

The implementation of a virtual-lab for testing designs of drum-type washing machines is discussed. It is applied to the analysis of an industrial washing machine (120 Kg load capacity) manufactured by Fagor Industrial. The virtual-lab has been completely programmed using Modelica language. The washing machine dynamic model has been developed using *MultiBody* Modelica library. The interactive visual interface (i.e., the virtual-lab view) has been implemented using *VirtualLabBuilder* Modelica library. The virtual-lab supports interactive changes in the position and properties of the springs and the dampers, the properties of the inner and outer drums, and the mass and position of the load. Simulation results are in good agreement with the experimental data. The virtual-lab has demonstrated to be a valuable design and analysis tool, allowing the user: (1) to get insight into the system behavior; (2) to tune the system parameters in order to improve the dynamic behavior; and (3) to simulate special events, such as a component breakage.

## INTRODUCTION

Virtual-labs provide a flexible and user-friendly method to define the experiments performed on the model. Virtual-labs are composed of the *model* and the *view*. The *model* is the mathematical description of the system under study. The *view* is the user-to-model interface. It is intended to provide a visual representation of the simulated model behavior and to facilitate the user's interactive actions on the model during the simulation run. The graphical properties of the view elements are linked to the model variables, producing a bi-directional flow of information between the view and the model.

Modelica (Modelica 2007) is an object-oriented modeling language that facilitates the physical modeling paradigm. *MultiBody* (Otter et al. 2003) is a free Modelica library, providing 3-D mechanical components to model mechanical systems. An application of *MultiBody* library to the modeling of a household washing machine is described in (Donida et al. 2006).

An approach to the implementation of virtual-labs using Modelica language has been proposed in (Martin et al. 2006). This approach consists of:

- A systematic methodology to transform any Modelica model into a formulation suitable for interactive simulation has been proposed. Modelica models adapted according to this methodology can be used to set up interactive virtual-labs.
- *VirtualLabBuilder* Modelica library has been designed and programmed. It includes Modelica models implementing graphic interactive elements. These models allow the virtual-lab developer: (1) to compose the virtual-lab view; and (2) to link the visual properties of the view with the model variables. The interactive graphic interface is automatically generated during the model initialization process. The components of the library contain the code required to perform the communication between the view and the model.

Examples of virtual-lab implementation using this approach can be found in (Martin et al. 2007a, Martin et al. 2007b).

## VIRTUAL-LAB IMPLEMENTATION USING *VirtualLabBuilder* MODELICA LIBRARY

The steps to define the virtual-lab using *VirtualLabBuilder* Modelica library are the following:

**Step 1: Description of the Virtual-Lab Model**. The Modelica model has to be adapted to suit interactive simulation following the methodology described in (Martin et al. 2006).

**Step 2: Description of the Virtual-Lab View**. *VirtualLabBuilder* library is composed of the packages shown in Figure 1a. Some of them are intended to be used by the virtual-lab developers (i.e., *VirtualLabBuilder* users):

- `ViewElements` and `VLabModels` packages contain the classes required to implement the virtual-lab view and to set up the complete virtual-lab. The `ViewElements` package includes interactive graphic elements that can be used to compose the virtual-lab view.
- `Examples` package contains some tutorial material illustrating the library use.

On the contrary, the classes within the `src` package are not intended to be directly used by the virtual-lab developers.

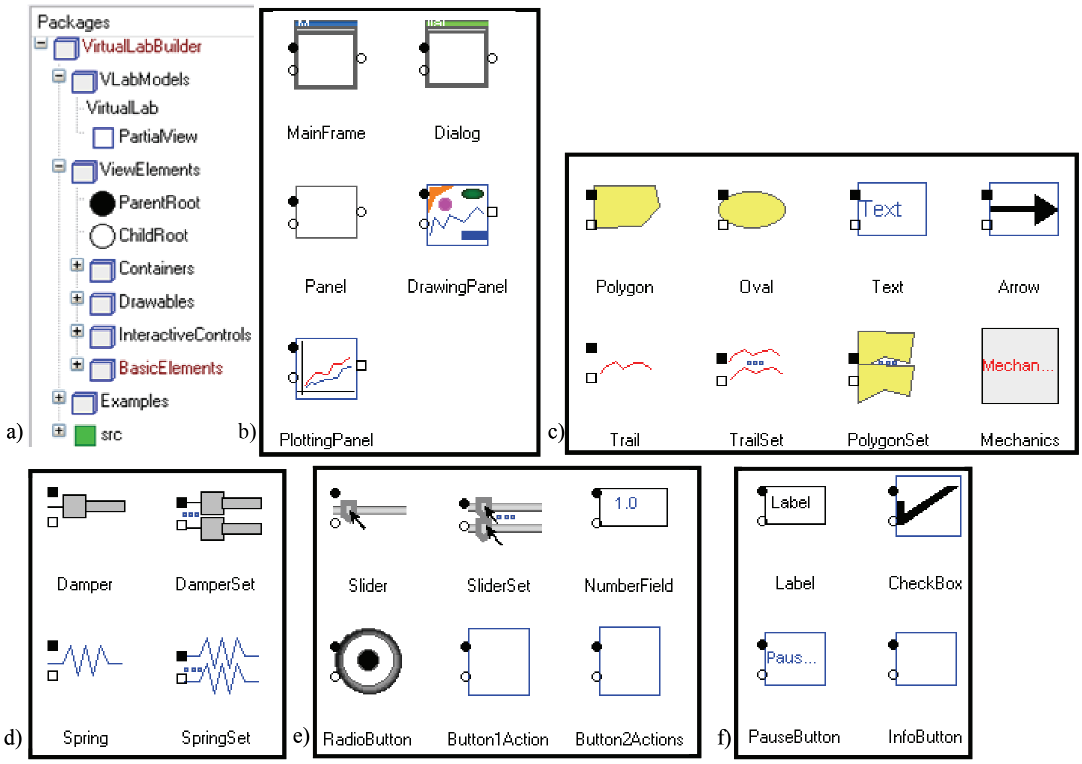


Figure 1: *VirtualLabBuilder* Library: a) General Structure; and Classes within the following Packages: b) `Containers`; c) `Drawables`; d) `Mechanics`; e) `InteractiveControls`; and f) `BasicElements`

The Modelica class describing the view must be a subclass of `PartialView` class. This class contains the code required to perform the communication between the model and the view. This code is valid for any model and view descriptions, and the virtual-lab designer only needs to set the length of the model-to-view communication interval.

The graphic components have to be connected by the virtual-lab developer. The "root" graphic component (i.e., the container component which hosts the rest of the components), named `root`, is pre-defined in the `PartialView` class. The connections among the graphic components determine their layout in the virtual-lab view. Modelica modeling environments – e.g. Dymola (Dynasim 2006) – allow defining in a drag-and-drop way the instantiation of the required *VirtualLabBuilder* library components and connecting them using the mouse.

**Step 3: Virtual-Lab Set Up**. The Modelica description of the virtual-lab has to be an instance of `VirtualLab` class. This class contains two parametrized generic classes: the class of the virtual-lab model and the class of the view. In addition, the virtual-lab designer has to specify, writing the required equations, how the variables of the model and the view classes are connected.

**Step 4: Translation to Executable Code and Launch**. The virtual-lab description is translated using Dymola and ran. As a part of the model initialization (i.e., the calculations performed to find the initial value of the model variables), the initial sections of the interactive graphic objects and of the `PartialView` class are executed. These initial sections contain calls to Modelica functions, which encapsulate calls to external C-functions that are Java-code generators.

As a result, during the model initialization, the Java code of the virtual-lab view is automatically generated, compiled and packed into a single jar file. Also, the communication procedure between the model and the view is set up. This communication is based on client-server architecture: the C-program generated by Dymola is the server and the Java program (which is automatically generated during the model initialization) is the client. Also, the communication procedure between the model and the view is set up. This communication is based on client-server architecture: the C-program generated by Dymola is the server and the Java program (which is automatically generated during the model initialization) is the client. Once the jar file has been created, it is executed. As a result, the initial layout of the virtual-lab view is displayed and the client-server communication is established. Then, the model simulation starts.

During the simulation run, there is a bi-directional flow of information between the model and the view. The model simulation (i.e., the server) sends to the view (i.e., the client) the data required to refresh the view. And the view sends to the model simulation the new value of the variables modified due to the user's interactive action.

## Interactive graphic elements

There are four packages within the `ViewElements` package (see Figures 1a), which contain the graphic elements that can be used to define the virtual-lab view:

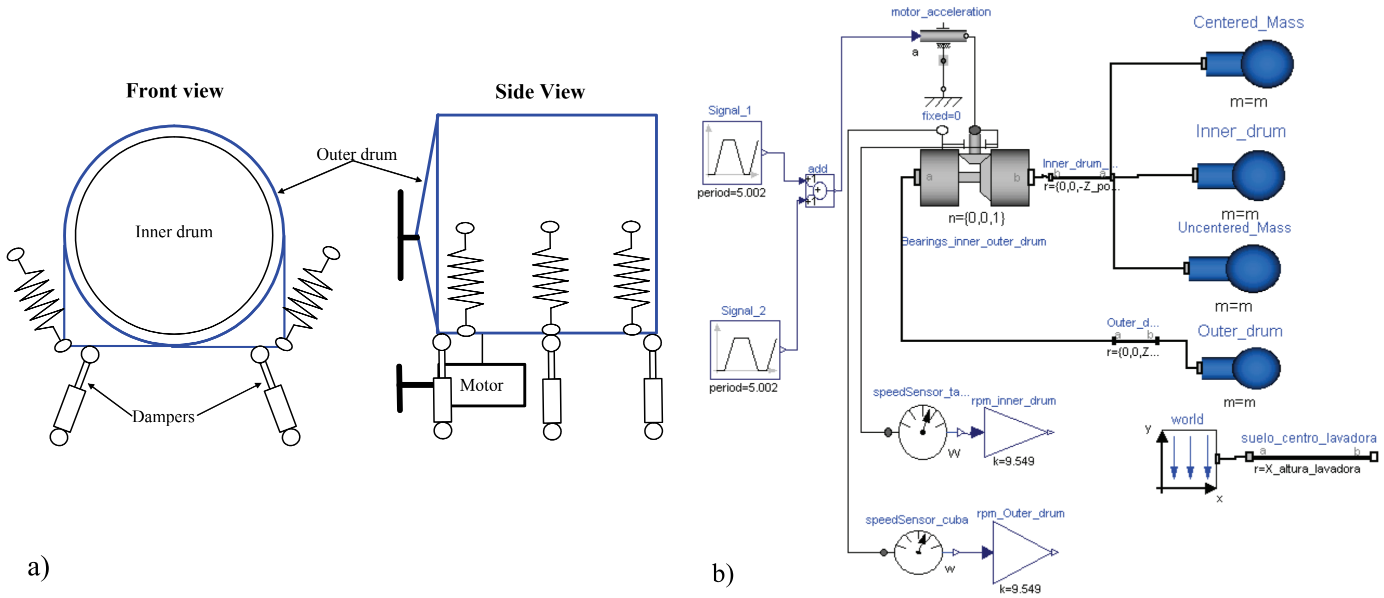


Figure 2: Drum-Type Washing Machine: a) Schematic Representation; and b) Diagram of the Modelica Dynamic Model

1. **`Containers`** package includes the following classes: `MainFrame`, `Dialog`, `Panel`, `DrawingPanel` and `PlottingPanel` (see Figure 1b). These graphic elements can host other graphic elements.
2. **`Drawables`** package includes the following classes: `Polygon`, `PolygonSet`, `Oval`, `Text`, `Arrow`, `Trail` and `TrailSet` (see Figure 1c); and also the `Mechanics` package (see Figure 1d), which contains the `Damper`, `DamperSet`, `Spring` and `SpringSet` classes. The elements of these classes must be placed inside containers, which provide the coordinate system. They are intended to be used for building an animated schematic representation of the system. The variables setting the geometric properties of these elements (position, size, etc.) can be linked to model variables.
3. **`InteractiveControls`** package includes six classes: `Slider`, `SliderSet`, `NumberField`, `RadioButton`, `Button1Action` and `Button2Actions` (see Figure 1e). Model variables can be linked to the variables defining the state of these interactive control elements. This allows changing the value of these model variables during the simulation run.
4. **`BasicElements`** package contains four classes: `Label`, `CheckBox`, `PauseButton` and `InfoButton` (see Figure 1f). Objects of these classes can be included inside a window or a panel. `PauseButton` class creates a button for pausing and resuming the simulation. `InfoButton` class creates a button for showing and hiding windows containing documentation in HTML format.

`Drawable` and `InteractiveControl` elements implement the information flow between the model and the view of the virtual-lab. The simulated value of the model variables modifies the properties of the drawable elements (i.e., model-to-view information flow). The user's interactive action on the interactive controls modifies the value of the model variables (i.e., view-to-model information flow).

## WASHING MACHINE VIRTUAL-LAB

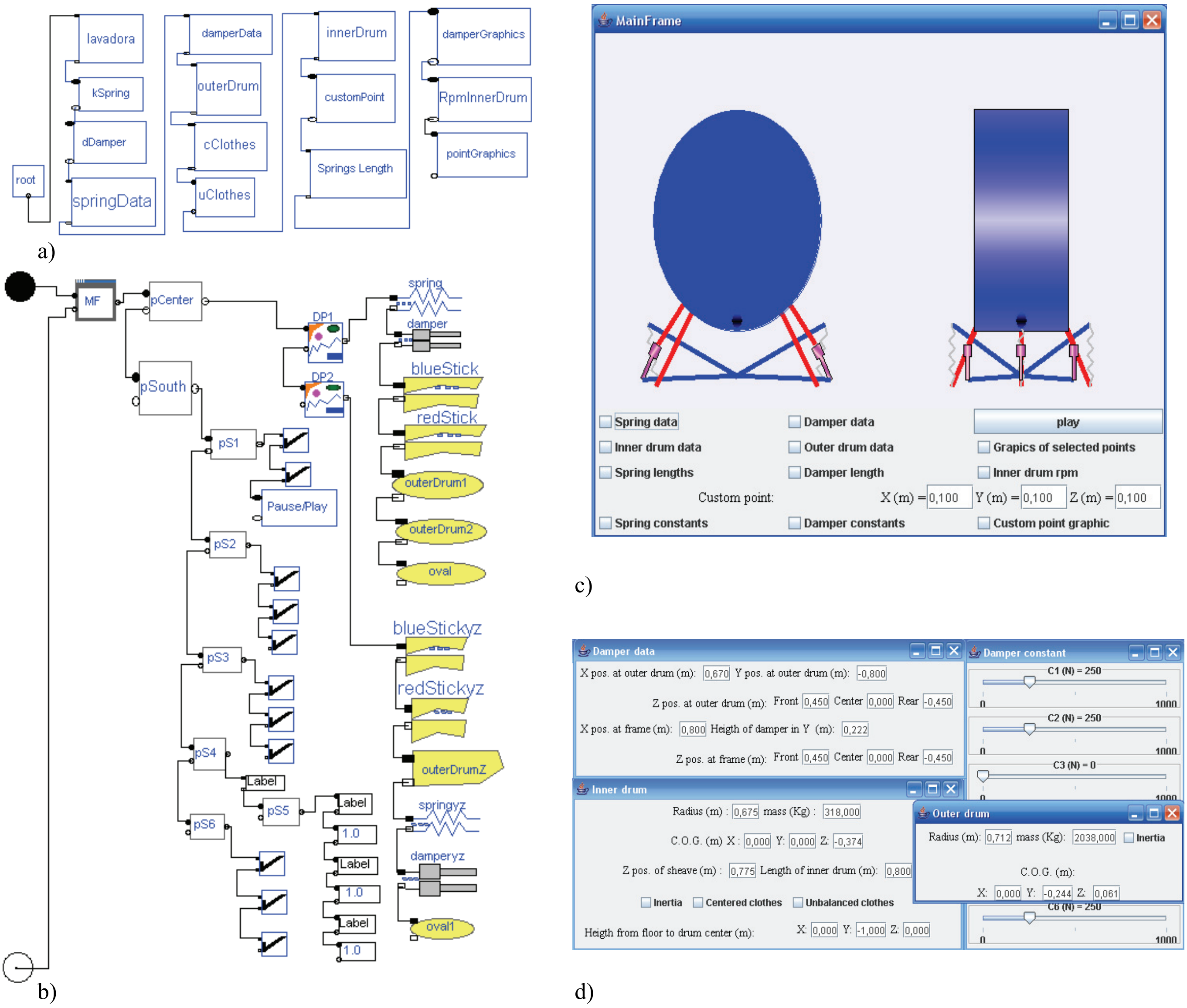
Drum-type washing machines are widely used in Europe. They are composed of an inner drum that rotates inside an outer drum, with a horizontal axis, making the clothes tumble upward and downward during washing cycle (see Figure 2a). During the drying cycle, clothes are submitted to both the gravity force (*g*) and the centrifugal force, generated by the inner drum rotational speed.

When the centrifugal force is bigger than "*g*", the clothes tend to stick to the inner drum wall. In some cases, it results in a non homogeneous distribution of the clothes' mass around the periphery of the inner drum. This is mainly due to the different composition of the tissues. Imbalance occurs when clothes' centre of mass does not coincide with the inner drum rotation axis, and it induces vibration to the outer drum.

In order to reduce the vibrations transmitted to the floor, the outer drum is suspended with springs. The forces transmitted to the frame (floor) can be drastically reduced if the resulting natural frequency (spring-drum) is very low. On the other hand, suspended drum movements can become uncontrollable when passing through the natural frequency and at low rotational speeds, which can cause collisions against the frame. Friction dampers are normally used to limit these movements.

Suspended drum movement depends on many factors. For example, the suspended mass inertia, the spring and damper positions and characteristics, the unbalanced mass value and location, and the spinning speed profile. All these parameters must be tuned for each new design, in order to minimize the drum displacements and the forces transmitted to the frame.

Accurate models of the drum dynamic, including unbalance load effects, can not be derived analytically due to the complexity of the dynamic behavior, influenced of all these parameters and their interactions. This limitation is even



a)

b)

c)

d)

☐ Spring data ☐ Damper data

☐ Inner drum data ☐ Outer drum data ☐ Grapics of selected points

☐ Spring lengths ☐ Damper length ☐ Inner drum rpm

Custom point: X (m) = 0,100 Y (m) = 0,100 Z (m) = 0,100

☐ Spring constants ☐ Damper constants ☐ Custom point graphic

X pos. at outer drum (m): 0,670 Y pos. at outer drum (m): -0,800

Z pos. at outer drum (m): Front 0,450 Center 0,000 Rear -0,450

X pos. at frame (m): 0,800 Heigth of damper in Y (m): 0,222

Z pos. at frame (m): Front 0,450 Center 0,000 Rear -0,450

C1 (N) = 250

C2 (N) = 250

C3 (N) = 0

C6 (N) = 250

Radius (m) : 0,675 mass (Kg) : 318,000

C.O.G. (m) X : 0,000 Y: 0,000 Z: -0,374

Z pos. of sheave (m) : 0,775 Length of inner drum (m): 0,800

☐ Inertia ☐ Centered clothes ☐ Unbalanced clothes

Heigth from floor to drum center (m): X: 0,000 Y: -1,000 Z: 0,000

Radius (m): 0,712 mass (Kg): 2038,000 ☐ Inertia

C.O.G. (m):

X: 0,000 Y: -0,244 Z: 0,061

Figure 3: Washing Machine Virtual-Lab: a) Diagram of the View Description in Modelica; b) Diagram of the Main Window Description in Modelica; c) Main Window; and d) Some Dialog Windows

more evident when analyzing big-size washing machines (40 to 120 kg load capacity), which are suspended by several couples of springs and dampers. The dynamic behavior of the suspended drum can be successfully analyzed using rigid-body dynamic modeling and computer simulation.

## Multibody model

*MultiBody* Modelica library (Otter et al. 2003) has been used for developing the dynamic model of the washing machine. All the bodies, except the springs, have been considered rigid. The diagram of the model is shown in Figure 2b. The suspended drum is composed of the following four bodies: outer drum, inner drum, centered and un-centered mass. These masses are attached to the inner drum. One rotational DoF is allowed between the inner and the outer drums.

The suspended drum has six DoF. Its dynamic behavior is governed by the forces generated by the mass of the un-centered clothes, the gravity, and the forces exerted by three pairs of springs and dampers. The springs and dampers are modeled as ideal elements (i.e., the force is proportional to the relative displacement or speed, respectively). Additionally, an external mass-free frame is considered. The springs and dampers are attached to this frame. This approach allows the computation of the floor reaction forces.

The model has been adapted – as it was proposed in (Martin et al. 2006) – to allow interactive changes in the position and properties of the springs and the dampers, the properties of the inner and outer drums, and the mass and position of the load. The model is intended to be used for tuning the value of these parameters, in order to improve the washing machine dynamic behavior. The evaluation of the suspended system displacement is accomplished for the following two critical test conditions: spinning start up and spinning at maximum speed. The dynamic behavior analysis is based on the following two key magnitudes: (1) the displacement of the suspended system with respect to the external frame; and (2) the forces transmitted to the floor. These forces can cause vibrations and relative displacements of the frame.

### Virtual-lab view

The Modelica description of the virtual-lab view has been developed modularly, extending and connecting the required graphic components of the *VirtualLabBuilder* library (see Figure 3a). The virtual-lab view contains a main window and 15 dialog windows. Each of these windows is described by a class. The classes defining the main and the dialog windows are briefly described below.

The diagram of the Modelica class that defines the main window is shown in Figure 3b. The `MF` component – of `MainFrame` class – generates the window shown in Figure 3c. The `MF` layout policy is set to *BorderLayout*, in order to allow selecting the position of the hosted elements (i.e., north, south, center, east or west positions). It has two components hosted inside it, `pCenter` and `pSouth`, both of `Panel` class.

- `pCenter` component is placed in the center of `MF`, and it contains two containers, `DP1` and `DP2`, of `DrawingPanel` class These two components contain the frontal and the lateral animated diagrams of the washing machine. These diagrams are composed of several drawable elements of `Polygon`, `PolygonSet`, `Oval`, `DamperSet` and `SpringSet` classes.
- `pSouth` component hosts several interactive controls of `PauseButton`, `CheckBox`, `Label` and `NumberField` classes. The checkboxes allow to show and to hide the dialog windows. The button allows to pause and to resume the simulation. The spatial coordinates of system points are set using `NumberField` class components.

There are two types of dialog windows: (1) the windows containing plots that display the time evolution of some model variables; and (2) the windows containing interactive controls that allow the user to perform interactive changes in the model variables. The following windows contain the interactive controls (see Figure 3d):

- "*Spring data*"/"*Damper data*" windows allow changing the position of the spring/damper extremities in relation to the frame and the outer drum.
- "*Inner drum*" window allows changing the value of the relevant properties of the inner drum, including radius, mass, length, center of gravity (C.O.G) position, center position and sheave position. Additionally, this window contains checkboxes that allow to show and to hide three dialog windows. These three dialog windows contain interactive control elements that allow changing the C.O.G. mass and position of the centered and unbalanced load, and the inertia matrix of the inner drum.
- "*Outer drum*" window allows changing the outer drum properties (i.e., radius, mass, inertia and C.O.G. position).
- "*Spring constant*"/"*Damper constant*" windows allow changing the value of the spring/damper constants.

The virtual-lab contains five plot windows displaying the time-evolution of the following magnitudes: (1) the damper lengths; (2) the spring lengths; (3) the position of a system point, which can be interactively chosen by the virtual-lab user; (4) the position of certain relevant points of the system; and (5) the rotational speed of the inner drum.

### Virtual-lab set up

The virtual-lab description is obtained as discussed in (Martin et al. 2006). It is translated using Dymola and executed. Then, the jar file containing the Java code of the virtual-lab view is automatically generated and executed, and the virtual-lab view is displayed. Simulation results are in good agreement with measurements of the suspended system displacements.

## CONCLUSIONS

*MultiBody* and *VirtualLabBuilder* Modelica libraries have been used to implement the virtual-lab of a drum-type washing machine. The virtual-lab has been successfully applied for testing alternative designs of an industrial washing machine (120 Kg load capacity) manufactured by Fagor Industrial. This virtual-lab allows: (1) faster parameter tuning; and (2) simulating special events – e.g. spring or damper breakage. It constitutes a very valuable tool for the designer, that can online explore "what happens if" with no need to run again the simulation, enhancing his knowledge of the system behavior.

## ACKNOWLEDGEMENTS

This work has been supported by the Spanish CICYT, under DPI2004-01804 grant, and by the IV PRICIT (Plan Regional de Ciencia y Tecnología de la Comunidad de Madrid, 2005-2008), under S-0505/DPI/0391 grant.

## REFERENCES

Donida F., G. Ferreti and F. Schiavo. 2006. "Modelling and Simulation of a Washing Machine", *In Proc. of the 50th Int. Congress ANIPLA 2006*, Rome (Italy).

Dynasim. 2006. "Dymola. User's Manual". Dynasim AB. Lund, Sweden.

Martin C., A. Urquia and S. Dormido. 2006. "An Approach to Virtual-lab Implementation using Modelica". *In Proc. of the 20th Annual European Simulation and Modelling Conference*, 137-141, Toulouse (France).

Martin C., A. Urquia and S. Dormido. 2007a. "Virtual-lab of a Solar House implemented using VirtualLabBuilder Modelica library". *In Proc. of the Conference on Systems and Control*, Marrakech (Morocco).

Martin C., A. Urquia and S. Dormido. 2007b. "Implementation of Interactive Virtual laboratories for Control Education using Modelica". *In Proc. of the European Control Conference,* Kos (Greece).

Modelica. 2007. http://www.modelica.org/

Otter M, H. Elmqvist and S.E. Mattsson. 2003. "The New Modelica MultiBody Library". *In Proc. of the 3rd Int. Modelica Conference,* 310-330, Linköping (Sweden).