MPTL14 2009
Udine 23-27 September 2009

WORKSHOP ON EASY JAVA SIMULATIONS AND THE COMPADRE DIGITAL LIBRARY

Francisco Esquembre, *Universidad de Murcia*

Wolfgang Christian, *Davidson College*

Bruce Mason, *University of Oklahoma*

1. Workshop Overview

The premise of the Easy Java Simulations and ComPADRE Digital Library workshop is that when teachers and students are actively involved in the modeling of physical systems they greatly increase the power of computer simulations to enhance teaching and learning of physics. Although the modeling method can be used without computers, the use of computers allows students to study problems that are more difficult and time consuming, to explore real-world applications, to visualize their results, and to communicate their results with others. This workshop report will benefit anyone wishing to use our ready to run computer models for teaching and learning. There are also links to material that will benefit computer-modeling instructors and computational physicists wishing to adopt Open Source Physics tools for their own research. We present the general pedagogical and technical issues that arise in the design of interactive computer-based tutorials as well as how EJS models can be adapted to meet local needs.

The workshop held at MPTL 14 provided an introduction to Easy Java Simulations (EJS) and the Opens Source Physics (OSP) Collection on ComPADRE and then was divided into three different threads designed to address the needs of different participants. We describe these threads next and then focus in this report on the basics of running, modifying, and accessing EJS models. Material is available online for further exploration of the EJS, OSP, and ComPADRE tools.

The first thread, “EJS 101”, is dedicated to newcomers who want to learn the principles of how EJS works. This is a hands-on EJS tutorial in which participants study, step by step, a simple example. They first explore to learn how this model is created and run, and then learn how to modify it and add new capabilities. This thread is recommended for participants without programming experience. Additional self study material, including the EJS program, examples, documentation, and sample curricular material can be downloaded from the Open Source Physics collection http://www.compadre.org/osp and from the Easy Java Simulations website http://www.um.es/fem/Ejs (Esquembre 2009).

The second thread, “Teaching Computation and Modeling with EJS”, is designed for EJS users who want to learn how they can incorporate computational physics into their teaching. This thread is recommended for users with some programming experience. This session is based on an EJS adaptation of *An Introduction to Computer Simulation Methods* computational physics textbook by Harvey Gould, Jan Tobochnik, and Wolfgang Christian. Preliminary chapters are available in a ComPADRE shared personal collection maintained by Wolfgang Christian (Christian 2009).

The third thread, “Using EJS-based curricular materials”, is designed to help instructors use the ComPADRE library tools to employ EJS materials in their classes. This thread explains the connection between EJS and the ComPADRE digital library, and describes ways to find, download, organize, and share physics-related curricular material based on EJS simulations. This thread is meant for teachers who are more interested in preparing or using EJS-based curricular material than in programming new simulations. Self-study material is available from a ComPADRE shared personal collection created by Bruce Mason, which includes both tutorial material and an example of the ComPADRE personal collections (Mason 2009).

This report is a brief introduction to help instructors find and operate EJS models. It will outline the collection of EJS resources available through the ComPADRE library and then focus on the basic structure and operation of EJS.

2. The ComPADRE OSP Collection

The Communities for Physics and Astronomy Digital Resources for Education (ComPADRE) digital library, a collaboration of physics professional societies, first opened to physics teachers and learners in the summer of 2003. The overarching goals of ComPADRE are to support physics learners and instructors at all levels and advance the infrastructure of physics and astronomy education through Web resources and services. Much more than a simple reference database, ComPADRE supplies numerous services for educators and learners in and outside of the classroom including: community-focused resource collections, communication tools, specialized information databases, workshop and conference Web sites, and resource hosting. ComPADRE, therefore, is not just a referatory (a collection of links to external sites), but also a repository (a collection of resources that are permanently housed on ComPADRE). Resource hosting is crucial for support of teachers to ensure that resources will be available when needed. Another crucial aspect of the ComPADRE effort is the building of partnerships and educational communities to help the library grow and thrive. The collaboration with the Open Source project is among the most productive.

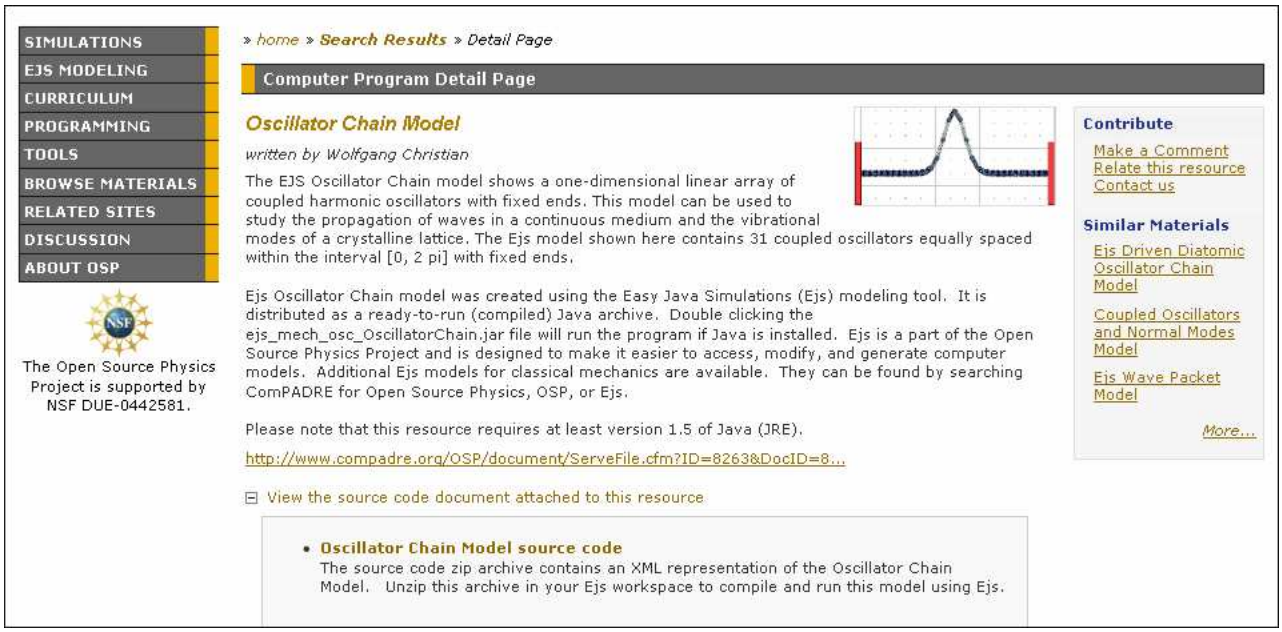

Figure 1: The coupled Oscillator Chain model as cataloged in the ComPADRE OSP Collection. The source code is a small XML file that can be read by EJS.

The Open Source Physics (OSP) collection within ComPADRE provides curriculum resources that engage users in physics, computation, and computer modeling. This collection is a repository of Java-based computational resources for teaching in the form of source code, executable simulations, and curriculum resources as shown in Figure 1. As with any good library, items within the collection are cataloged with information such as subject, author, format, resource type, and keywords. This allows the users to find relevant materials using common search or browse strategies. Unlike most digital libraries, however, the OSP collection distributes compiled programs that users can examine, modify, and redistribute with minimal effort. This report shows you how.

To begin, we suggest the reader go to the OSP collection within ComPADRE at http://www.compadre.org/osp. This web site is organized into sections focused on different Open Source tools and content. Information and resources relevant to this paper are in the "EJS Modeling" section linked on the right side of the page. To begin the exploration of EJS models, search the collection for topics that you regularly teach or that are related to your research field. For example, the search term "Coupled Oscillators" yields five items and "Elastic Collisions" yields eight at this time. Items can contain one or more subdocuments such as compiled programs, lesson plans, book chapters, and source code. Many of the items that you will find are ready-to-run (compiled) Java models created with EJS that can be downloaded and run by the user as an application (click on the jar file to execute) or as an applet embedded within a web page. To insure

that Java is properly installed on your computer, download a jar file from ComPADRE and run it. We next will examine how these simulations are created using EJS.

3. The EJS Learning Platform

The Easy Java Simulations (EJS) learning platform is a free, open-source Java modeling and authoring tool that simplifies the modeling process by breaking it into activities: (1) documentation, (2) modeling, and (3) interface design. EJS helps science teachers and students create interactive simulations of scientific phenomena. These simulations can then be used in computer laboratories with students to better explain difficult concepts, to motivate them to study science, or to let students work with the simulations or (for more advanced students) even create their own.

Let us begin by running Easy Java Simulations and inspecting a simulation created with it.[1] We use the EJS interface to open the file **ModelingScience\Ch02_Intro\MassAndSpring.xml**, which is installed with EJS in the source directory. The EJS interface will look like Figure 2.

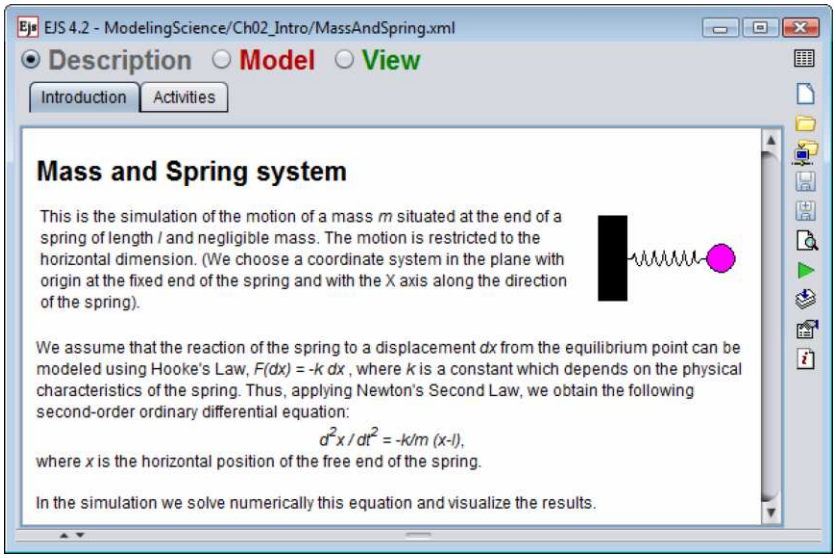

Figure 2: The description pages for the mass and spring simulation. Click on a tab to display the page. Right-click on a tab to edit the page.

Easy Java Simulations provides three workpanels for modeling. The first panel, *Description*, allows us to create and edit a multimedia HTML-based narrative that describes the model. Each narrative page appears in a tabbed panel within the workpanel and right clicking on the tab allows the user to edit the narrative or to import additional narrative text. Description pages are an essential part of the modeling process and these pages are included with the compiled model when it is distributed as a Java application or posted on a Web server as an applet.

3.1. Modeling with EJS

The second workpanel, *Model*, is dedicated to the computational modeling. We use this panel to create variables that describe the model, to initialize these variables, and to write algorithms that describe how this model changes in time.

In this simulation, we study the motion of a particle of mass m attached to one end of a massless spring of equilibrium length L. Our model assumes small oscillations so that the spring responds to a given (horizontal) displacement δx from its equilibrium length L with a force given by Hooke's law, $F_x = - k \, \delta x$. k is the elastic constant of the spring, which depends on its physical characteristics. We use Newton's second law to obtain a second-order differential equation for the position of the particle:

[1] Detailed installation and running instructions can be found at http://www.um.es/fem/EjsWiki/Main/Download.

$$\frac{d^2x}{dt^2} = -\frac{k}{m}(x - L) \tag{1}$$

We solve this system numerically to study how the state evolves in time.

When implementing a computational model, a good first step is to identify, define, and initialize the variables that describe the system. Figure 3 shows an EJS variable table. Each row defines a variable of the model by specifying the name of the variable, its type, its dimension, and its initial value.

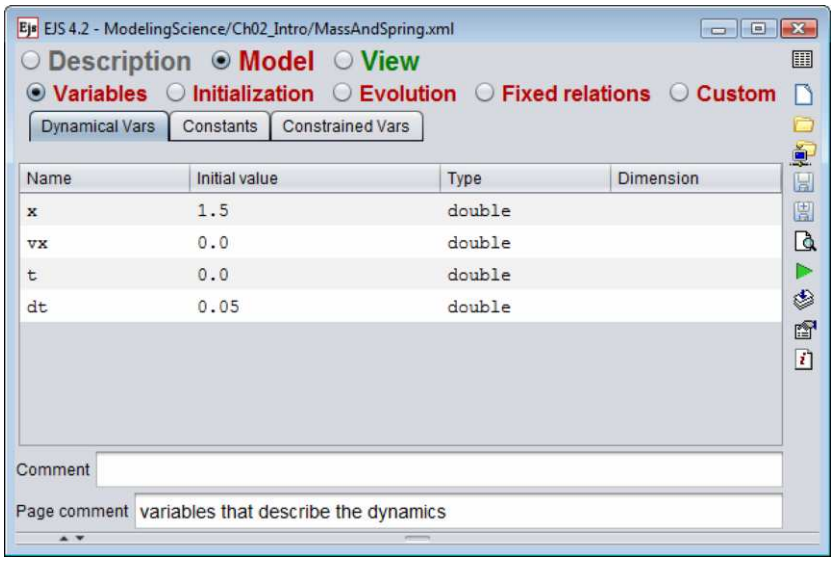

Figure 3: The Model *workpanel contains five subpanels. The subpanel for the definition of mass and spring dynamical variables is displayed.*

Correctly setting initial conditions is important when implementing a model because the model must start in a physically realizable state. Our model is relatively simple, and we initialize it by entering values (or simple Java expressions such as *0.5*m*vx*vx*) in the *Initial value* column of the table of variables. EJS uses these values when it initializes the simulation. Advanced models may require an initialization algorithm. The *Initialization* panel allows us to define one or more pages of Java code that perform the required computation.

The *Evolution* panel allows us to write the Java code that determines how the mass and spring system evolves in time and we use here the possibility to enter ordinary differential equations, such as (1), without programming. EJS provides a dedicated editor that lets us specify differential equations in a format that resembles mathematical notation and automatically generates the correct Java code. Because ODE algorithms solve systems of first-order ordinary differential equations, we recast our higher-order equation into a first-order system as shown in Figure 4.

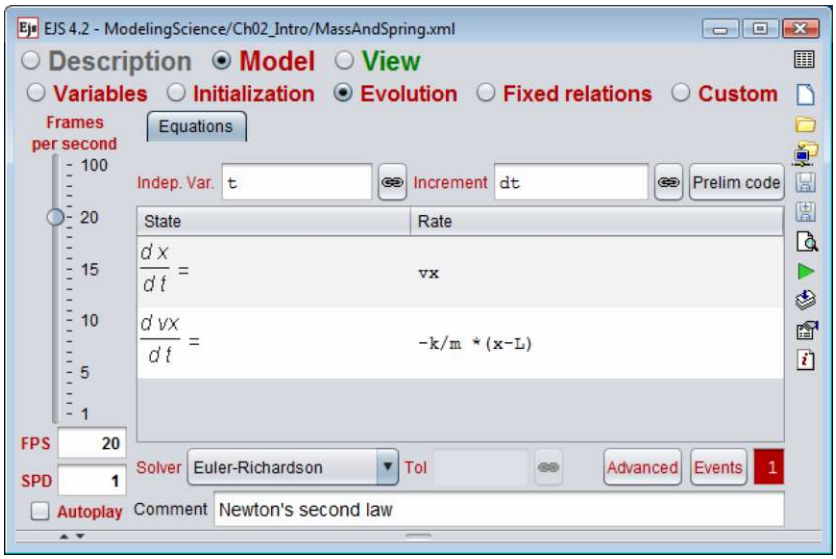

Figure 4: The ODE evolution panel showing the mass and spring differential equation and the numerical algorithm.

The evolution workpanel handles the technical aspects of the mass and spring ODE model without programming. The simulation advances the state of the system by numerically solving the model's differential equations using the midpoint algorithm. A dropdown menu at the bottom of the editor lets us select the ODE solver (numerical algorithm) that advances the solution from the current value of time, *t*, to the next value, *t* + *dt*. The left-hand side of the evolution workpanel includes fields that determine how smoothly and how fast the simulation runs.

The mass and spring model computes the kinetic, potential, and total energies of the system as output variables from the state variables in the *Fixed relations* panel as shown in Figure 5.

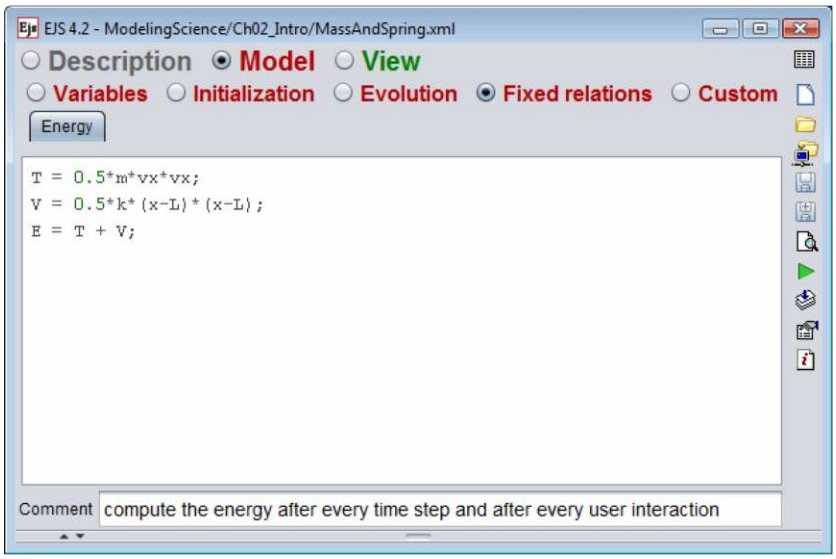

Figure 5: Fixed relations for the mass and spring model.

Finally, the *Custom* panel (empty for this model) can be used to define methods (functions) that can be used throughout the model.

3.2. Building user interfaces with EJS

The third workpanel, *View*, is dedicated to the task of building the graphical user interface that includes visualization, user interaction, and program control with a minimum of programming. The interface for our simulation was created by selecting graphical elements from the EJS palettes and adding them to the view's *Tree of elements*. The right frame of the view workpanel of EJS, shown

in Figure 6, contains a collection of *view elements*, grouped by functionality. View elements are building blocks that can be combined to form a complete user interface, and each view element is a specialized object with an on-screen representation. To create a user interface, we create a frame (window) and add elements, ranging from lines and shapes to input boxes, buttons, and graphs, using “drag and drop”.

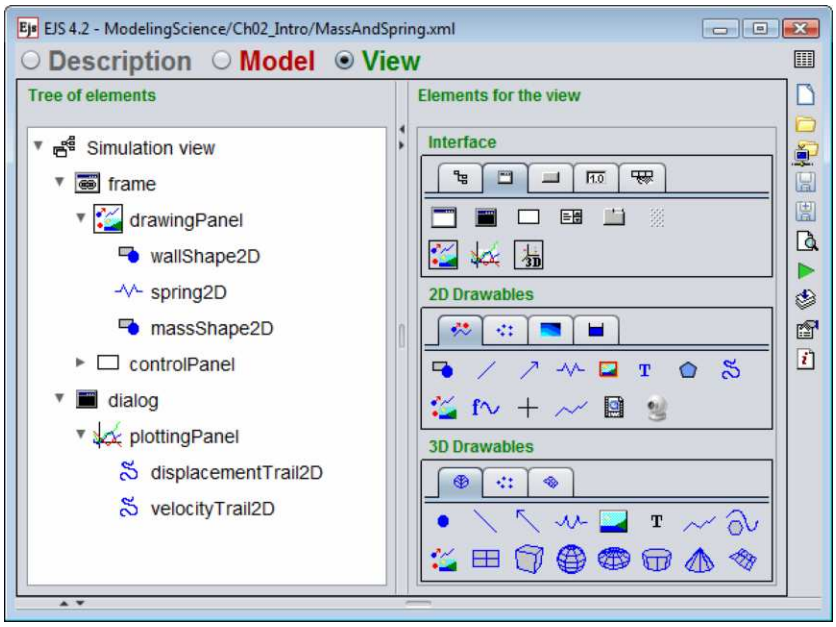

Figure 6: The View *workpanel showing the* Tree of elements *for the mass and spring user interface.*

Each view element has a set of internal parameters, called *properties*, which configure the element's appearance and behavior. We can edit these properties by double clicking on the element in the tree to display a table known as a *properties inspector*. Appearance properties, such as color, are often set to a constant value, such as *RED*. We can also use a variable from the model to set an element's property. This ability to connect (bind) a property to a variable without programming is the key to turning our view into a dynamic and interactive visualization. For example, the view element that represents the mass in our mass and spring simulation has the properties *Pos X* and *Pos Y* that are bound to the *x* and *y* variables of the model. This simple assignment establishes a bidirectional connection between model and view. These variables change as the model evolves and the shape follows the *x* and *y* values. If the user drags the shape to a new location, the *x* and *y* variables in the model change accordingly.

Elements can also have *action properties* which can be associated with code. User actions, such as dragging or clicking, invoke their corresponding action property, thus providing a simple way to control the simulation. For instance, the mass *On Release* property executes the following code:

```
vx = 0.0;                  // sets the velocity to zero
_view.resetTraces(); // clears all plots
```

3.3. Running the simulation

Easy Java Simulations is a powerful tool that lets us express our knowledge of a model at a very high level of abstraction. Notice that the three lines of code on the *Fixed relations* workpanel (Figure 5) and the two lines of code in the shape's action method are the only explicit Java code needed to implement the model. Easy Java Simulations creates a complete Java program by processing the information in the workpanels when the run icon is pressed. Click and drag the particle to a desired initial horizontal position and then click on the play/pause button. The particle oscillates about its equilibrium point and the plot displays the displacement and velocity data as shown in Figure 7.

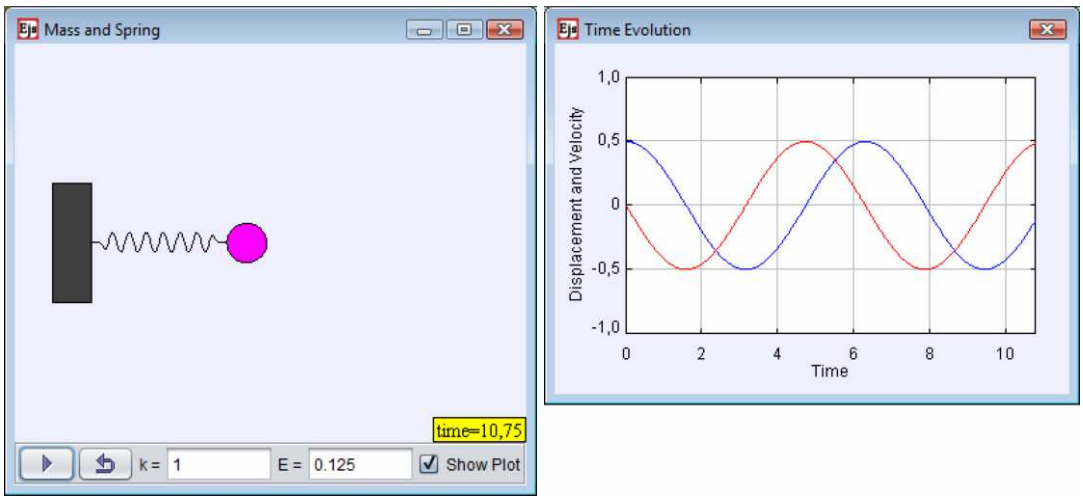

Figure 7: The mass and spring simulation displays an interactive drawing of the model and a graph with displacement and velocity data.

EJS is designed to be both a modeling and an authoring tool. We suggest that you now experiment with it to learn how you can modify existing models and even create your own. As a start, we recommend that you run the mass and spring simulation and go through the activities in the second page of the *Description* workpanel. Detailed instructions on how to modify this particular simulation can be found at the expanded version of this document at http://www.compadre.org/osp/items/detail.cfm?ID=7306.

4. Distributing the Simulation

Simulations created with EJS are stand-alone Java programs that can be distributed without EJS for other people to use. The easiest way to do this is to package the simulation in a single executable jar file by clicking on the *Package* icon. The stand-alone jar file thus created is ready to be distributed on a CD or via the Internet.

An important pedagogic feature is that this jar file is created in such a way that users can return to the EJS modeling environment at any time to examine, modify, and adapt the model. Students run the distributed **ejs_MassAndSpring.jar** file by double-clicking it. After working with the simulation, as instructed by their teachers, they can right-click in a drawing panel to bring the popup menu of Figure 8.

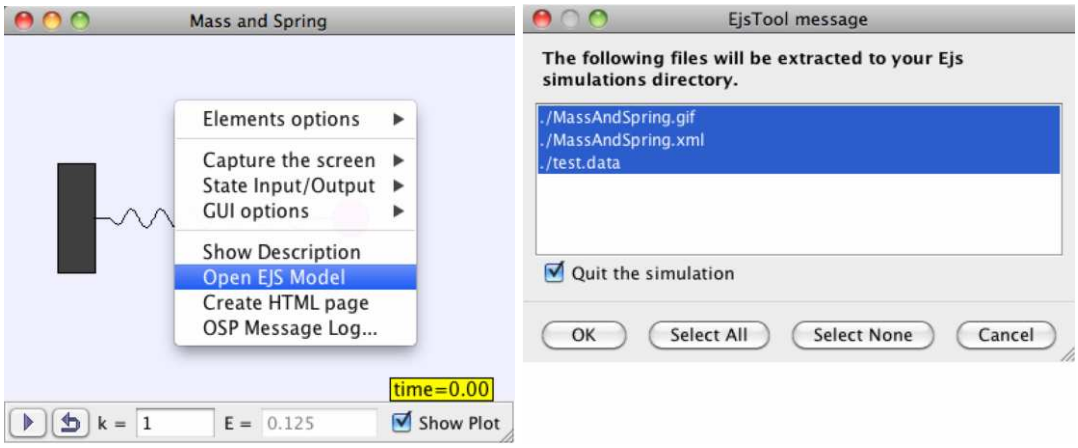

Figure 8: The "Open EJS Model" menu item (left) allows users to extract the model and open it using the local copy of EJS.

The jar file contains a small *Extensible Markup Language* (XML) description of each model in the jar and all associated data files (such as GIF images). Selecting the *Open EJS model* option of this menu extracts these files into the students' computer and opens the model using the local copy of

EJS, if EJS is installed. The student can then inspect, run, and modify the model following the teacher's instructions. A student can, for example, obtain an example or a template from an instructor and can later repackage the modified model into a new jar file for submission as a completed exercise.

5. Finding models

As you found in your initial exploration of the OSP Collection on ComPADRE, the library can be searched to find a wide range of EJS models created by many authors. Registered ComPADRE users (registration is free) can create their own personal, annotated collections on OSP containing EJS models along with any other learning material on ComPADRE. If desired, these personal collections can also be shared with others. However, there is also a service built directly into the EJS tool that allows the user to search for and download models without leaving EJS.

EJS comes installed with an extensive list of simulations created by other instructors. Because of EJS' simplified interface and common architecture, it is relatively easy to reuse examples created by other people. The source directory of the distribution's workspace contains some directories with sample simulations. But, even more important, repositories of EJS simulations available on the Internet that can be accessed directly from within the EJS interface.

The EJS digital libraries icon in the taskbar opens the window displayed in Figure 9, which contains a combo box with the available digital libraries. Select one of these libraries or click the *Get catalog* button to get the list of EJS models in it. All these libraries work in a similar way, and we use the OSP Collection to illustrate how they are accessed from within EJS.

Figure 9: The Digital Libraries window of EJS. Select one of the available repositories to retrieve the list of models available.

If you are connected to the Internet, select the *OSP collection on the ComPADRE digital library* entry of the top combo box and EJS will connect to the library to obtain the catalog of EJS models in the library. Currently there are 200 models organized by topic, and the collection will continue to grow. The left frame of the EJS library interface, shown in Figure 10, displays the subject hierarchy. When the name of a subcategory appears in red, double-clicking it will expand the node to show the list of models for that subject. Because many models have primary and secondary subject classifications, the check box at the top, right below the library combo box, allows you to decide whether you want the models to be listed uniquely under their primary classification, or appear in all matching categories (thus appearing more than once).

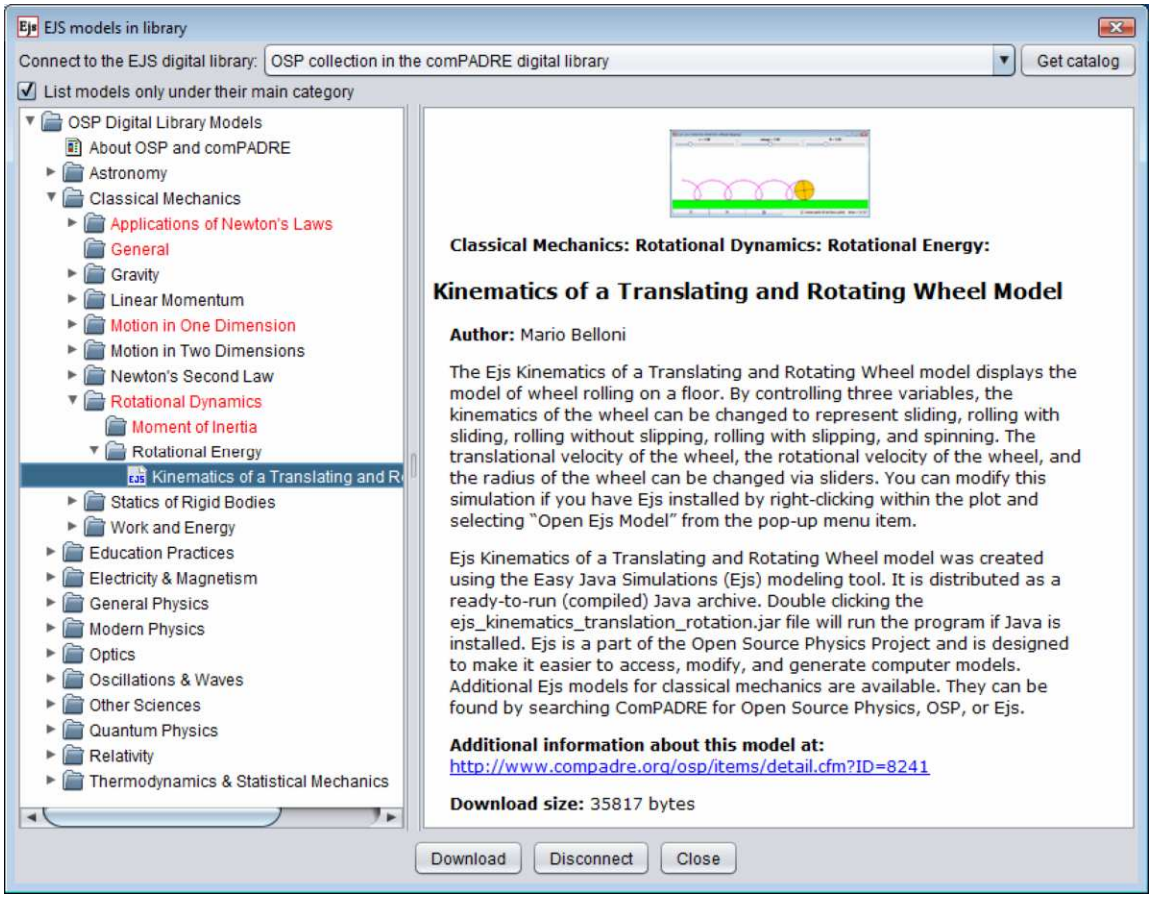

Figure 10: The OSP collection on the comPADRE digital library is organized in categories and subcategories. Each entry provides information about a model.

When you click a model node, with an EJS icon, the right frame shows information about the model obtained from the library. The information describes the model, and includes a direct link to the comPADRE library for further information. Double-clicking the model entry, or clicking the *Download* button, will retrieve the model and auxiliary files from the library, ask you for a place in your source directory of your workspace to download them, and open the model in EJS when the download is complete. Because source files are usually small, the download takes place almost instantly. Now you can inspect, run, or modify the model as we did, earlier in this document, for the mass and spring model.

6. Summary

Easy Java Simulations is a modeling and authoring tool expressly devoted to the task of creating simulations that use a computer to obtain numerical data from models as they advance in time, and to display this data in a form humans can understand. EJS has been designed to let you work at a high conceptual level, concentrating most of your time on the scientific aspects of the simulation, and asking the computer to automatically perform all the other necessary but easily automated tasks.

EJS structures the authoring process in three parts: creating a description, specifying the model, and building the view for the simulation. Each task has a dedicated workpanel that provides the needed functionality. The built-in ODE editor stands out as one of the most powerful aids in modeling dynamic processes. Also, building the view is made easier by the use of off-the-shelf, ready to use view elements, of which there are many different types, each one specialized for a given visualization or input task. We can use these elements to build views whose effectiveness and sophistication rival the work of a professional programmer. Once this high-level work is done, a single mouse click starts the internal engine that will generate a fully dynamic, interactive simulation from our specifications.

Every learning platform, including Easy Java Simulations in combination with the EJS connection to digital libraries such as ComPADRE, has a learning curve. The EJS modeling and authoring tool provides a simplified yet still powerful architecture appropriate for the skills of standard science instructors and students and the ComPADRE digital library provides capabilities for finding and reusing existing material. The combination of EJS with Internet connections to the library makes these models available instantly and will get you quickly up and running in creating the simulations you need for your classroom.

References

Esquembre F (2009) http://www.um.es/fem/EjsWiki/Main/Documentation, accessed 1/27/2010

Christian W (2009) Draft chapters of an EJS adaption of *An Introduction to Computer Simulation Methods: Applications to Physical Systems, 3rd Edition* (Addison Wesley, New York, 2006) by Gould H, Tobochnik J, Christian W is available in Wolfgang Christian's shared personal collection on ComPADRE. This filing cabinet is available at http://www.compadre.org/portal/filingcabinet/bookmarks.cfm?FID=17708

Mason B (2009) The OSP/EJS tutorial and an example file cabinet structure is available at http://www.compadre.org/psrc/filingcabinet/share.cfm?UID=5&FID=18499&code=AC03A8551C, accessed 1/27/2010.