



American Control Conference 2000  
Chicago, Illinois, USA  
June 28-30, 2000

## A WEB-BASED SYSTEM FOR CONTROL ENGINEERING EDUCATION

C. Schmid, A. Ali

Control Engineering Laboratory  
Department of Electrical Engineering and Information Sciences  
Ruhr-Universität Bochum, Germany  
cs@esr.ruhr-uni-bochum.de

**Keywords:** Education, Learning, System Dynamics, Web

### Abstract

This contribution describes a Web-based system developed for control engineering education. Simulation and visualisation of dynamical systems in the environment of a standard Web-browser are made possible by extending its capabilities. The communication with a powerful computation/simulation engine is realised. Virtual Reality Modelling Language (VRML) is used to animate the virtual 3-D models of dynamical systems. Tools for online graphical representation of simulated signals and live mathematical relationships are developed using Java. The courseware is classified into tutorials, exercises and virtual experiments

### 1. Introduction

The conventional educational courses are offered according to some specific schedule. A student has to be at the right place at right time to attend such courses. This - *knowledge by chance* - is not the only one drawback of such an educational system. There is normally one instructor for a large group of students. This situation leads to a lack of individual attention. Students, normally being the passive listeners, in such lecture sessions lose motivation and interest with the course of time. These courses normally fail to take into account the discrepancies in the intellectual levels and attitudes of different students towards a specific subject and are presented with a pace set by the instructor or, sometimes, by some dominant students.

Education in areas of engineering sciences especially control engineering faces severer problems because the ideas and phenomena involved in such areas are complex and hard to demonstrate on the conventional blackboard. The importance of laboratory experiments in engineering education can never be overemphasised. As, sometimes, the resources (personnel, equipment etc.) are scarce, it is hard to provide everybody the opportunity and freedom of experimentation.

The World Wide Web demonstrates how current technology can support information sharing among large and widely dispersed groups. The hypertext coupled with wide area network (WAN) functionality in the Web is the major incentive for exploiting this technology for educational purposes to eradicate the above mentioned hurdles of the

conventional education. The Web provides significant new functionality in transmitting information to the students and provides an effective mechanism for integrating tools into a single user interface. A Web based educational system due to its *round the world* and *round the clock* accessibility (as long as the administrator authorises) is independent of time and location. The students can access the course anywhere and at any place they like and can follow it with their own pace. The presentation of learning material using Hyper-Text-Markup-Language (HTML) gives the students a great flexibility and interactivity. Interconnections among different topics and ideas can be explored using the hyperlinks, and material can be presented in a relatively more effective way. Due to the existing Multimedia support by the Web browsers, all these media, like videos, audio clips, and animations, can be deployed to visualise and demonstrate the complex ideas or phenomena involved in engineering sciences.

The commercially available Web browsers are normally capable of handling multimedia like hypertexts, audio-visual clips and animated sketches and diagrams without any additional programming. These features enable a browser to present a good and interesting multimedia show, but if the intention is to use it as a serious medium of instruction for education in control then these capabilities seem to be insufficient to meet the goal. In order to use a Web browser as a user interface for education in control, its capabilities have to be extended. Such an interface should be able to present mathematical formulas and equations on the Web pages and solve algebraic and differential equations. Numerical/symbolical computations, analysis and simulation of dynamical systems involved in control and presentation of these computed or simulated data in graphical form for further investigations and comments should also be the responsibility of the browser. It should be in a position to animate 3-D models of the simulated dynamical systems in order to perform virtual experiments.

One possible solution for the above problem is to implement the whole computational engine and graphical user interface in form of Java applets, which would have been a big project from the point of view of the programming expense. All the numerical and symbolical analysis had to be implemented in Java. The outcome of the whole laborious work would have been a computationally inefficient system.

The solution chosen by us uses a powerful and very popular computational tool (MATLAB) for carrying out simulations and numerical or symbolical computations. In section 2 of this paper we give some technical details of the learning system. The next section deals with didactic concepts and curricular matters. Section 4 deals with graphical user interface and page layout and navigation. In the last section we will give some concluding remarks.

## 2. Technical Realisation and Software Components

The architecture of the developed learning system is shown in Fig. 1. All the curricular stuff is held on the server side, and delivered on demand to the clients via the Web. This approach has an advantage that updates and revisions are done centrally on the server and the latest version is always available to all the clients. A standard Netscape Web Browser is the only user interface on the client side. This saves the extra effort of learning "How to use the system". The simple explanatory stuff is presented as formatted text with hyperlinks, animated images and audio-visual clips using HTML. Additional software components in form of browser plug-ins and Java applets required on the client

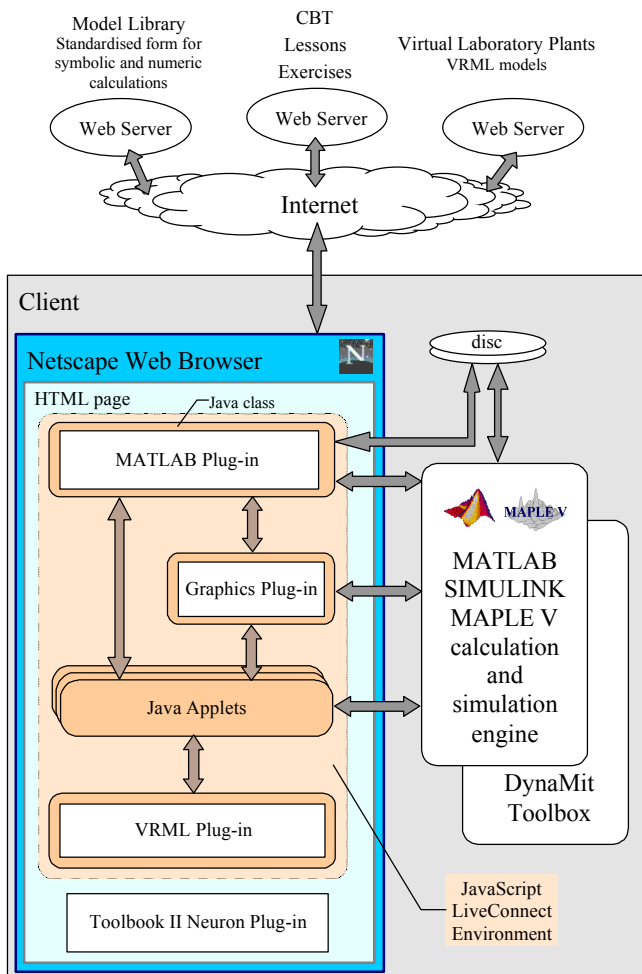


Fig. 1: System architecture

side are shown in Fig. 1. In the following we give a brief description of each of these components:

### **MATLAB/SIMULINK/MAPLE**

All the numerical and symbolical computations are performed by MATLAB. It must be installed on the client machine in order to avoid heavy data traffic between the server and the client. SIMULINK is used to simulate the behaviour of dynamical systems and MAPLE for symbolic computations.

### **DynaMit Toolbox**

This toolbox contains some MATLAB routines which are frequently used by the learning system. In order to cut download times during learning sessions this toolbox is installed on the client machine.

### **MATLAB Plug-in**

A browser plug-in called MATLAB Plug-in is developed to link the computational engine of MATLAB with the browser. This plug-in is used to read or write the values of variables in the MATLAB workspace and to call MATLAB commands from the Browser. As this plug-in comes into action only *on-call*, it is not a suitable channel for data streams.

### **Graphics Plug-in**

A browser plug-in called Graphics Plug-in is developed to present the static graphics created by MATLAB or other software on the web page.

### **VRML Plug-in**

Real-looking three dimensional models of dynamical plants and the visual set-up of a virtual laboratory are developed using VRML. These models are animated on the Web using the VRML Plug-in [Cos00]. Dynamical behaviours of these virtual plants are simulated externally by the computational/simulation engine using the corresponding simulation models.

### **ToolBook II Neuron Plug-in**

The authoring tool Toolbook II [Tol00] offers some good features, like interactive animations of block diagrams etc., which are from the didactic point of view of great interest. In order to view this stuff in Web Browser as it runs under Toolbook II runtime environment, the Neuron Plug-in is required on the client side.

### **LiveConnect Environment**

This environment comes with the Netscape Browser and associates JavaScript as part of the Java environment. Therefore JavaScript, Java and plug-ins can be integrated smoothly into a Web page. JavaScript can use public methods from Java classes. By extending the plug-ins runtime interface, Java classes can be added, such that from any

Java class the plug-in methods can be used. This is a powerful feature, as objects on a page can communicate with each other. Such objects may be Java applets, plug-ins, forms, buttons, images, etc.

### **Java Applets**

In addition to the plug-ins some general Java applets are implemented:

*HotEqn* is an applet family and developed to present mathematical expressions and equations on the Web. Such an applet can be parameterised using the famous LaTeX format. Different *HotEqn* applets on a Web page can communicate with one another and with the computational engine. This component is equipped with interactive features and is useful for substitutions and symbolic or numerical evaluations of expressions.

*HotEqnEd* is a symbolic editor for entering symbolic expressions based on *HotEqn*. The application is for exercises, where the student has to enter formula. The applet consists of a separate free configurable keyboard window. The input is displayed directly in the applet window on the Web page and send to MATLAB.

The *Slider* applet is developed to implement variables or parameters, which can be varied or tuned by the user on the web page. This applet communicates with MATLAB via the MATLAB Plug-in and invokes JavaScript and MATLAB routines when its value is changed.

The *HotAnim* applet is developed to establish a live communication channel between the simulation engine and the VRML plant models. This applet is a free configurable communication interface using the methods of the External Authoring Interface (EAI) [Mar97] according to ISO/IEC 14772-1:1997 standards. One possible way of such an inter-application communication is Microsoft's Dynamic Data Exchange (DDE) channel. As the new Java version supports sockets, which offer a relatively faster more reliable and platform-independent solution for data exchange, the *HotAnim* applet uses datagram (UDP) sockets for communication with the simulation engine. The other end of this communication link is realised as a dynamic link library (DLL) and linked in the simulation engine as an S-function. An additional advantage of this approach is that it can be modified easily to the situations, where the simulation engine is installed on a remote machine. The virtual sensors, actuators and events of the whole set-up are configured using a script file. Signals like set points, user-operated switches and disturbances are communicated from virtual plant sensors to the simulation kernel via MATLAB Plug-in methods, and simulated signals are carried back to the virtual actuators through the socket channel.

The *HotScope* applet implements a SIMULINK-like scope on the Web page for online observation of simulated sig-

nals. This applet also uses the same communication channel with SIMULINK as *HotAnim*.

## **3. Didactic Concepts and Curricular Set-up**

Reflecting the conventional educational set-up, the Web based curriculum is composed of three sections: tutorials, exercises and laboratory experiments. The tutorial acts as a summary of basic concepts, methods and approaches. In the exercise section problems are proposed. The students have to solve these problems using the skills learned in corresponding tutorial lesson. The virtual control laboratory simulates a laboratory environment, so that small real world experiments can be performed. Here follows a brief description of these components.

### **Tutorial**

The tutorial, the 'lecture' part, may be used either independently to learn the basics of systems and control theory, or as an interactive facility for solving the exercises. The tutorial is not intended to be a substitution of the conventional lectures of the curriculum held by the tutors, but to be an extension of them. Both ways of using the tutorial are supported by the proposed structure. Therefore, the tutorial section is divided into independent lessons. For a better understanding, each lesson is structured in a uniform way. In the beginning of each lesson, the fundamentals and prerequisites are outlined. A simple technical example is presented to illustrate the problem and as a motivation for the lesson. Several alternative methods for solving the problem are proposed and compared. At this stage a decision has to be made which way to solve the problem should be pursued. In the following section the details of the best solution are given. This part of the lesson is the most important one. At this point the application of the abstract methods to the real problem is elaborated. In the next section the reliability of the solution is discussed from the point of view of a real task. The last section consists of an overview of the lesson.

### **Exercises**

The exercise section consists of a set of problems to be solved. As the level of student's knowledge is not previously known, a Web-based hypermedia learning environment has to be designed very skilfully. A key issue for the success of an exercise is testing the knowledge. A trivial method is to check keywords, which leads in the extreme to drill-and-practice exercises. This type of knowledge evaluation is not appropriate for education, because the student is requested to try different answers until the correct one is found. The benefit with respect to a comprehensive view of a topic of this approach is regarded to be very small. To motivate both skilful and unskilled learners, a compromise between a guided exercise, where each subtask

is well defined and an unguided exercise, where only the main goal is formulated has to be worked out.

### **Laboratory Experiments**

In the virtual laboratory a laboratory environment is simulated, so that small, real-world experiments can be performed virtually. During a virtual control laboratory experiment the major aim is to explore the dynamical behaviour of a plant using simulation studies under variable conditions. This enables the students to get a deeper insight in the behaviour of a real plant.

An example is the VTOL emulator from Fig. 2. This unstable plant with non-linear dynamical behaviour is a considerably challenging configuration for testing control operation in 3-D configuration of this kind. The student can test the controller by simulation and interaction with the plant. A virtual joystick gives the beam angles. The propellers are rotating due to the speed given by the controller. Also a variable noise generated by the motors and propellers contributes to a realistic environment.



Fig. 2: VTOL example from the laboratory section

### **Hyperlinks and context-sensitive help**

The tutorial can be used in two different ways according to the prior knowledge of the student and his learning preferences. For a less knowledgeable student, the subchapters of the tutorial can be worked out successively in order to learn the fundamentals of systems and control theory. For a skilled student, who already has a fundamental knowledge of the basic theory, only special lessons are of interest. These special issues can be found either in the table of contents or by the help mechanism implemented in the exercise section to accomplish a given subtask. There are at least two ways to look up special topics of the tutorial section. Firstly, the complete section can be accessed by an hyperlink, which links the appropriate chapter of the tutorial to the current exercise. This becomes necessary if the

student has never heard of the method, which should be used. For example, the student is unable to remember the transformation of a time-domain state-space model into the frequency domain formulation.

An unguided or unstructured lesson or exercise may result in confusion in the case of unskilled learners, whereas a guided exercise may bore skilful learners. Thus, we propose a help mechanism, which is available for the student at any time. This on-line help assists the student in solving the particular task and furthermore provides links to the appropriate tutorial lessons if desired.

### **Context-free help (Glossary)**

A glossary is available from everywhere within the learning environment. This glossary is a collection of keywords. Each keyword is linked to a page, which describes the selected keyword briefly. This approach becomes important, if the student had already learned a topic but forgot some of its important aspects.

### **General aspects**

Almost all the computer-based training and learning systems are subject to the risk of being treated as mere television shows. To cope with this problem special measures are taken while designing the lessons and exercises of the course. The presentation is made simple and precise. The lessons proceed with a suitable rate of increment in the level of difficulty. A lesson starts with a well-known example from the everyday life and tries to entangle the student by its explorative and interactive nature. Special care is taken to avoid the possible danger of writing a book on the electronic media. The stuff is presented in a way that the students do not lose interest with the course of time as the problem reaches its climax of complexity and difficulty. But, in spite of all the efforts, the role of a human teacher can never be overemphasised. That is why such Web-based curricula should not be considered as a replacement but a supplement to the conventional education.

## **4. User Interface and Navigation**

A proper graphical user interface is very important for a computer-based learning system. It influences the student's concentration and interest in the lesson. A simple, pleasant and user-friendly environment attracts the students and can enhance the efficacy of the programme. Another important aspect to be considered is the navigation. Of course an electronic courseware may be more effective than its conventional counterpart due to the great flexibility and availability of hot hyperlinks and cross-references. But excess of everything is bad. The navigation in a learning system must be controlled for the sake of didactic purposes. The learning system under consideration is realised in three navigation levels, see Fig. 3. The Level 0 is the starting page. In this central navigation document one can find hy-

perlinks to Level 1 documents, which contain starting pages for the three curricular sections and additional information pages.

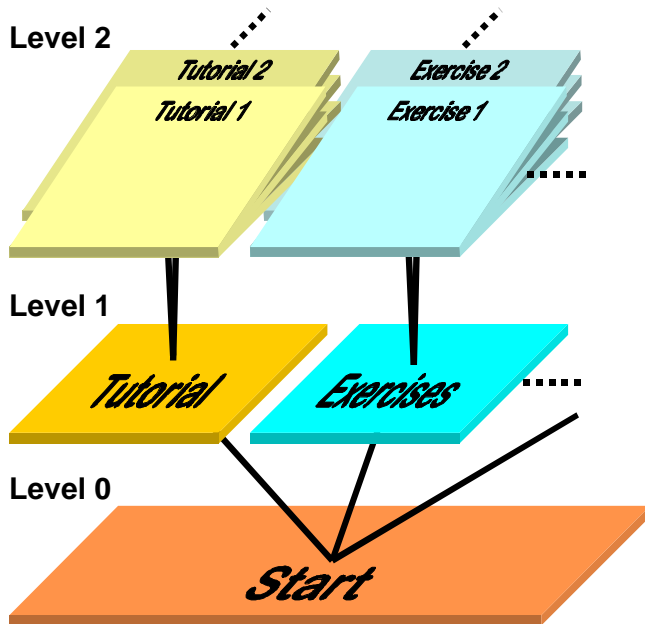


Fig. 3: Navigation levels in DynaMit

For the sake of uniformity, tutorials and exercises make use of the same layout structure. The Level 1 pages give some further information about using the tutorial/exercise section and contain a table of contents, leading to different chapters. Tutorials, exercises and laboratory experiments are accessible at Level 2. A flexible navigation bar is realised at this level. Sometimes, it is desirable to disable subsequent pages of a lesson until a specific condition is met. This 'guidance' for the user either avoids inconsistencies of the underlying model data or forces, for example, some didactic aims to be met before the next topic is presented. This requirement is achieved by configuring the navigation bar dynamically according to the desired navigation scenario. The system always indicates the current status for user orientation. This is important to prevent the user from getting 'lost-in-cyberspace'.

As the tutorial and exercise sections are divided into small lessons, which are normally self-consistent, it is not desirable that a student confuses himself by starting from the mid of a lesson. Each lesson should be started from the first page and followed successively. Of course the links for context-sensitive and context-free help are always available.

To achieve an environment, with the above features, the navigation structure from Fig. 3 is mapped into the frames as shown in Fig. 4. The Main frame contains the actual learning contents, whereas the Navigation frame makes the navigation through the environment easy. The contents of this frame are configured dynamically as desired by the

main pages. Shared scripts are located in a JavaScript library.

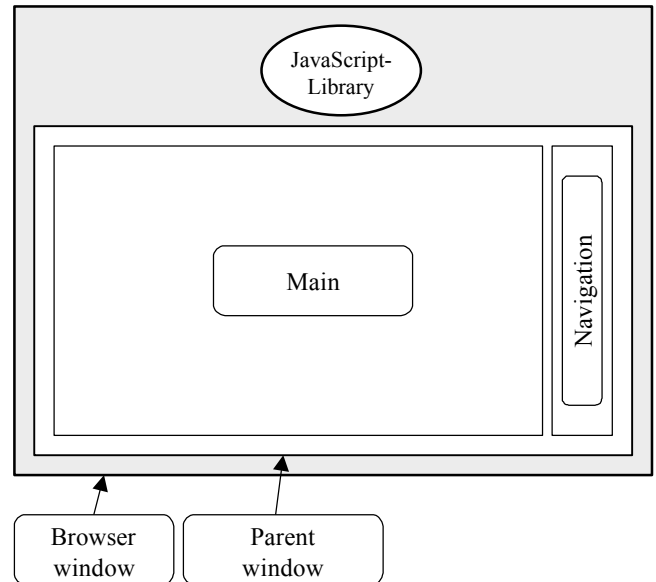


Fig. 4: Frame layout of the DynaMit Web pages

## 5. Conclusions

This contribution presents a Web-based learning framework, which addresses students and learners of control engineering. Such learning systems should be considered as a supplement rather than a substitution of conventional courses. Basic methods of systems' analysis and controller design are taught on the Web in tutorial, exercise and laboratory sections. The use of standard Web browsers as well as the integrated use of commonly used mathematical tools such as MATLAB and MAPLE makes the handling of the learning environment easy and flexible enough to tackle tasks of different levels of complexity. A client/server structure was chosen to cut download times.

## Acknowledgement

The support of this project by the *Universitätsverbund MultiMedia des Landes Nordrhein-Westfalen* (Project No. 6-12-97) is gratefully acknowledged.

## References

- [Cos00] Cosmo Homepage: <http://www.cai.com/cosmo> (Mar. 2000).
- [Mar97] Marrin, C.: Proposal for a VRML 2.0 Informative Annex, External Authoring Interface Reference. <http://www.vrml.org/WorkingGroups/vrml-eai/ExternalInterface.html> (Mar. 2000).
- [To100] The Toolbook II Instructor Homepage: <http://www.asymetrix.com/products/toolbook2/instructor/> (Mar. 2000).