Virtual Lab: Discovering through Simulation

Jean-Pierre Gerval, Yann Le RU

Institut Supérieur de l'Electronique et du Numérique - Brest
20 rue Cuirassé Bretagne - CS 42807 - 29228 BREST cedex 2 - FRANCE
Tel: +33 (0)2 98 03 84 07, Fax: +33 (0)2 98 03 84 10
E-mail: {jean-pierre.gerval, yann.le-ru}@isen.fr

Abstract

This paper sets out the design and the implementation of a Virtual Tutor. This Virtual Tutor is an avatar that "lives" in a distributed virtual reality application dedicated to practical activities in electronics: circuit design and simulation. The simulation of the circuit is done using the SPICE programme that is a general-purpose circuit simulation programme for non-linear dc, non-linear transient, and linear ac analyses. The implementation is based on VRML (Virtual Reality Modeling Language) and Java as languages and Cortona VRML plug-in from ParallelGraphics. The distribution of virtual worlds is obtained using DeepMatrix as environment server. Teachers use Concept Maps to design the behaviour of the Virtual Tutor. The control of the avatar is done using JESS (Java Expert System Shell). We describe in this paper a method that enables the creation of a Knowledge Base from a Concept Map.

Keywords*: Distributed Virtual Environments, Virtual Reality Modeling Language, Java, Concept Maps, Web-based Training.*

1. Introduction

Our Virtual Lab. has been experimented with a group of 40 students. This group represented half a class of 80. The target for this group was to prepare practical activities using the Virtual Lab. That is to say using virtual components and simulation by means of the SPICE programme (Gerval and Le RU, 2006).

The other half was preparing practical activities as usual. That is to say using paper and pens!

All these students were beginners in the field of electronics.

The main functionalities of the Virtual Lab. had been laid out to students during a short lesson. Then they had to prepare the practical activity by themselves. The students were expected to study various circuits that implemented operational amplifiers.

Our main goal through this experiment was to assess the relevance of the Virtual Lab. in the framework of the preparation of practical activities in electronics. Also, as we had split students into two different groups, we were expecting to make comparisons about the results of these two groups during practical activities. Benefits from Virtual Lab. vary according to the students' behaviour. Students who are eager to work get better benefits from the Virtual Lab. while the others get only lower gains.

Since the Virtual Lab. resource has been constantly available, well-motivated students have been encouraged not only to work the courses but also to look further.

Using the Virtual Lab. is a real added value for these students. As regards the other students, the Virtual Lab. remains a working tool like others. In this case, the Virtual Lab. does not increase schoolwork motivation.

On the other hand, this experiment emphasizes the fact that autonomous work using the Virtual Lab. cannot be applied so simply with the two types of student populations: naturally autonomous profiles and dependent profiles.

This state of fact is confirmed by their results:

- Naturally autonomous profiles are those who succeed better;
- Dependent profiles try to get a benefit to escape teacher monitoring.

In order to avoid that students who have a "dependent profile" escape teacher monitoring, we have decided to implement a Virtual Tutor. The main idea is to give students the feeling they are working in an autonomous way. But in the fact they are monitored and this way they can get a feedback about what they are doing.

2. Virtual world description

2.1. Basic components

The implementation of the virtual world is based on VRML (Virtual Reality Modeling Language) (VRML). Until now, we have implemented six different types of components (Fig. 1. and Fig. 2.), which are resistors, capacitors, inductors, diodes, transistors and operational amplifiers.

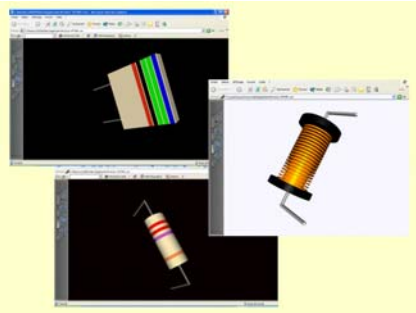

Figure 1. Passive components

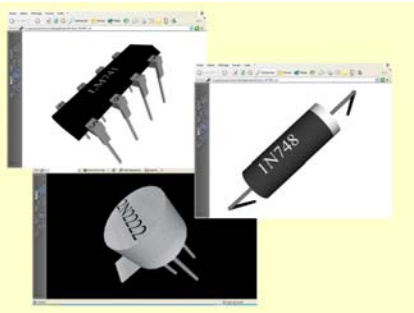

Figure 2. Active components

Students can choose a value for resistors or capacitors by selecting the right colours on the components according to colour codes. Concerning the other components, a menu has been implemented that enables students to choose a value.

2.2. Designing a circuit

Components are inserted into the virtual world by clicking on the corresponding icon (Fig. 3.). Students can move (or rotate) components by means of virtual axis (Fig. 4.) that represent the directions of the movement. After they have put components on the virtual PCB (Printed Circuit Board), students can build their circuit by clicking on

components' pins behind the virtual PCB (Fig. 5.). A link is created in the virtual world. A black line is drawn between components' pin. This line symbolizes a connection between two components.

Figure 3. Component choice

Figure 4. Moving components

Figure 5. Drawing the circuit

2.3. Devices and simulation

Two types of virtual electronics equipments are available: generators and oscilloscopes. Generators (Fig. 6.) enable students to set up a signal in terms of frequency, voltage and waveform. This signal will be applied to the circuit on the inputs selected by the student. Oscilloscopes (Fig. 7.) enable students to view circuits' outputs. That is to say: "simulation results". For each oscilloscope two channels are available. Students can adjust voltage and/or time scale.

3. Virtual world distribution

The server implementation is based on the DeepMatrix software (Reitmayr et.al., 1998) from GEOMETREK. This software enables users to enter 3D websites where they can interact with other users and objects. DeepMatrix implements client-server

architecture. On the server side, all messages are broadcasted in the same order to all clients. We have refined the proposed implementation from GEOMETREK, by introducing a filtering and pseudo dead reckoning mechanism (Singhal and Zyda, 1999) that permit a more friendly and flexible connection of users.

Figure 6. Generator

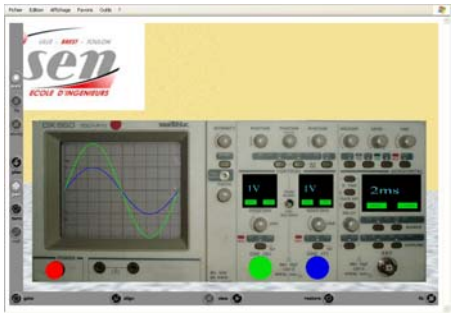

Figure 7. Oscilloscope

Clients are Java applets running in a HTML Browser. The communication between VRML world and client applet is made by use of External Authoring Interface (EAI) (Fig. 8). On the client side the EAI permits to achieve complex tasks by connecting the VRML Web Browser plug-in with a Java applet within the same web page.

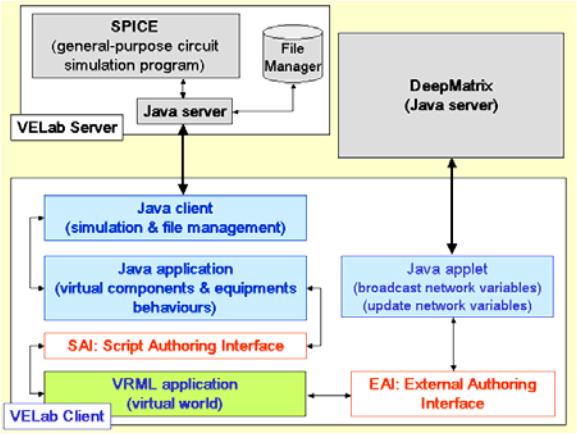

Figure 8. Distributed Virtual Lab. software architecture

EAI enables a two-way communication between the Java applet and the plug-in. The Java applet loads VRML content into the plug-in and adds avatar representation to the virtual world.

The avatars were designed on an approximate-body approach (Capin et.al., 1999), which provides frequently position and orientation information to remote hosts, taking into account a minimal set of joint points. The plug-in updates the Java applet about users position and orientation in the virtual world.

DeepMatrix offers network data structures, which enable clients to share data or to communicate together.

Concerning the Distributed Virtual Lab., VRML and Java code of each client are similar (Fig. 8.).

The main difficulty lies in the fact that these different clients must evolve in the same way according to users' actions in the different virtual worlds. That means that we have to know at the level of each java application if an event is local (a local user action) or if it is an update from network (another user action).

For example, when a user is changing the value of a resistor we have to change resistor's colours in the virtual world of this user and broadcast the new value of this resistor on the network. If the value of the resistor is changing because of a network update we just have to change the colours of this resistor. We do not have to broadcast anything else.

Another problem that is not solved by deepmatrix is the dynamic insertion of VRML code into a virtual world.

For example, when students proceed to virtual welding a line is inserted into the virtual world. If a new client connect to the virtual world, it is necessary to know if there was any welding before its connection. The same problem arises when a user requests a simulation. Simulation results are drawn on the screen of the virtual oscilloscope by means of VRML code, which is automatically generated and inserted into the virtual world.

Such data are saved into a file on the server side. This file keeps a trace of the state of the virtual world. This mechanism enables new clients to join old clients and share the same state.

4. Virtual tutor behaviour

4.1. Describing Virtual Tutor behaviour

The first step is to find or to define a tool in order to describe the behaviour of the Virtual Tutor.

This behaviour has to be designed by a teacher. The challenge is to provide a tool that is easy to use and easy to understand by users who are not specialized in computer sciences.

This tool must also be "content independent". That is to say that this tool must not be especially dedicated to the monitoring of practical activities in the field of electronics. The developed approach of virtual tutoring should be reused in various cases of practical activities.

Concept maps are widely used to describe experts' knowledge from various domains, for example in the field of electronics (Coffey *et.al*., 2003) or medicine (Michael *et.al*.). They can also be used to help students to integrate new concepts (Fernando Vega-Riveros *et.al*., 1998) even more they have been adapted with preschool children who can't read yet (Figueiredo *et.al*., 2004).

Concept maps are graphs that connect nodes with arcs. Nodes represent concepts and arcs represent relationships between nodes. It is an intuitive and visual representation

technique that seems to have *"more computational efficiency"* than any other forms of knowledge representation (Kremer, 1994).

According to the fact that in our Virtual Lab. most of behaviours have been developed in java language, we are naturally guided in choosing the same tool as an implementation language for the Virtual Tutor.

The behaviour of the Virtual Tutor is implemented by means of JESS (Java Expert System Shell). JESS is a Java implementation based on Clips (JESS). On the one hand, JESS has been used by other authors in order to control a virtual tutoring system and an architecture has been proposed in order to structure JESS rules (Gutl and Pivec, 2002).

On the other hand, concept maps have also been used to formalize JESS rules (Ciffey et.al., 2003).

But the originality of our work is to propose a generic approach (a content independent approach) that would enable the automatic generation of a Knowledge Base from a Concept Map.

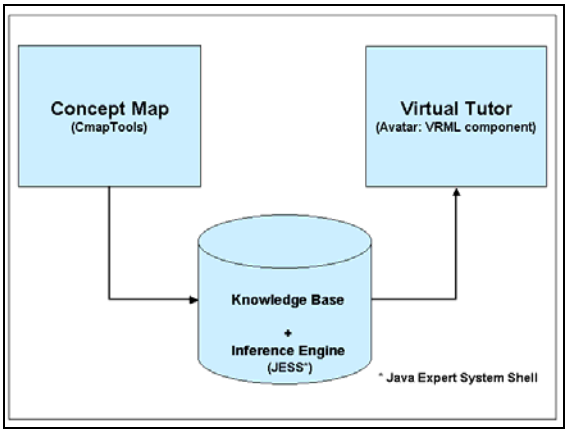

Figure 9. From Concept Map to Virtual Tutor behaviour

The different steps of our approach are showed Fig. 9:

1. Teachers design a Concept Map that represents the behaviour of the Virtual Tutor. Of course, this design must fit with the exercise that students have to achieve. Teachers are using CmapTools (CmapTools).
2. Rules are extracted from the Concept Map in order to feed JESS Knowledge Base.
3. The Virtual Tutor is a VRML avatar that will interact with students according to their actions in the virtual world. JESS takes in charge the control of the avatar.

4.2. Translating a Concept Map into rules

According to the fact that:

"Concept maps are not computational unless they have an associated semantics. That is, the maps' node and link types and their interconnections must be constrained to allow for computer support."(Kremer, 1994)

we have defined a basic semantics in order to be able to compute a Concept Map:

– Maps' nodes are predicates or actions.

– Link type is unique and the meaning of the link is "implies".

An example of such a map is given Fig. 10.

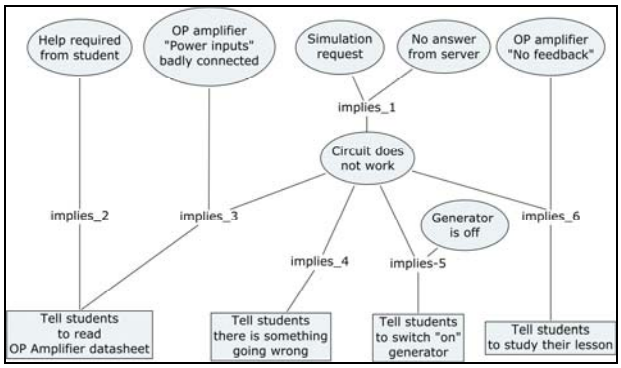

Figure 10. A map fragment on Operational Amplifier exercise

CmapTools enables the generation of an XML file describing the map. This XML file contains information concerning the topology and the semantics of the map (Fig. 11).

We use semantics data from the XML file in order to generate rules as following (Fig. 12):

1. Set up a 2D array with Linking-Phrases as row and Concepts as columns.
2. Assign weights to each connection.
   Weight = –1 if the connection goes from concept to linking-phrase.
   Weight = +1 if the connection goes from linking-phrase to concept.
3. Extract a rule from each row.
4. Write the rule in XML format for JESS: JESSML Language (JESS).

```
<concept-list>
        <concept id="1698" label="Generator is off"/>
        <concept id="674" label="Tell students to read OP Amplifier
datasheet"/>
        …
</concept-list>

<linking-phrase-list>
        <linking-phrase id="1366" label="implies_1"/>
        <linking-phrase id="1070" label="implies_3"/>
        …
</linking-phrase-list>

<connection-list>
        <connection id="1375" from-id="1366" to-id="1031"/>
        <connection id="1435" from-id="1031" to-id="1228"/>
        …
</connection-list>
```

Figure 11. XML fragment on Operational Amplifier exercise

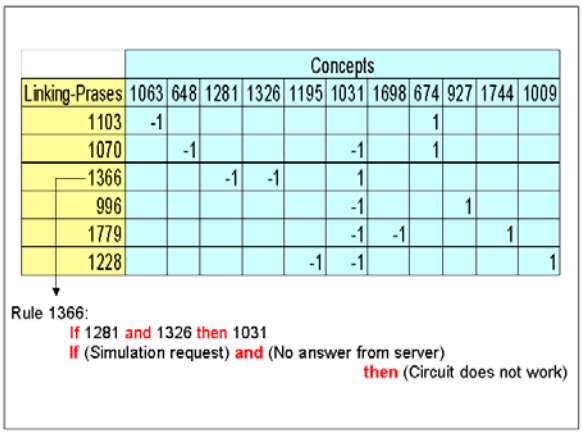

Figure 12. From Concept Map to rules

This method enables us to distinguish predicates and actions from the set of concepts. If there is a weight equal to -1 in a column that means this concept is a predicate. Otherwise it is an action.

5. Virtual tutor implementation

The Virtual Tutor is represented in the virtual world by means of an avatar, which is not connected to any users. This avatar is controlled by JESS on the server side (Fig. 13). Thus all clients that share the same virtual world are sharing the same Virtual Tutor.

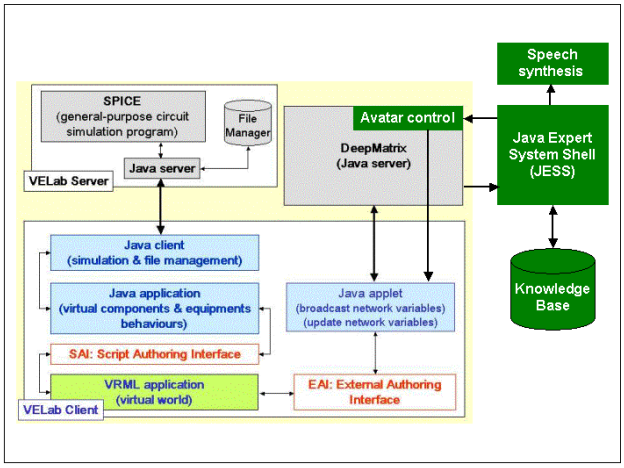

Figure 13. Virtual Tutor implementation

DeepMatrix offers network data structures, which enable clients to share data or to communicate together. DeepMatrix collects users' interactions. Events that are relevant to JESS rules are provided to JESS. Then JESS inference engine fires rules in order to select Virtual Tutor actions.

Virtual Tutor actions are both text messages and sentences that are stored on server side by means of mp3 files. When a comment has to be provided to students the Virtual Tutor speaks to students and, in the same time, a text message is broadcast to all students.

6. Conclusions and future works

This implementation of the Virtual Tutor has been experimented in the framework of an exercise dedicated to Operational Amplifier.

On a pedagogical point of view, it is really easy for a teacher to create a Concept Map and to generate rules for JESS. Such an approach could be used in other context.

But the integration of JESS to the Virtual Lab., on the server side, requires some hand works in order to link predicates to events collected by DeepMatrix. This point should be improved by means of a dictionary of events. The teacher could use this dictionary of events during the design phase of the Concept Map.

Semantics of our Concept Map should also be improved by means of new relations associated to linking-phrases. The same method could be used in order to generate rules for each type of relation.

On a technical point of view, DeepMatrix enables the use of avatar gestures. Experiments will help us to design and implement avatar gestures according to end-users' needs. This would help us to improve Virtual Tutor behaviours (Popovici, *et.al*, 2003).

We are also working on the integration of a speech synthesis module from MBROLA Project (MBROLA). This module will enable the Virtual Tutor to speak without needing any pre-recorded mp3 file.

REFERENCES

ANDERSON, R. E. (1992), Social impacts of computing: Codes of professional ethics. *Social Science Computing Review 10*, 2, 453-469.

CAPIN, T. K., PANDZIC, I. S., MAGNENAT-THALMANN, N., THALMANN, D. (1999), *Avatars in Networked Virtual Environments*, John Wiley & Sons, ISBN: 0-471- 98863-4.

COFFEY J. W., A. J. CAÑAS, T. REICHHERZER, G. HILL, N. SURI, R. CARFF, T. MITROVICH & D. EBERLE (2003), *Knowledge Modeling and the Creation of El-Tech: A Performance Support and Training System for Electronic Technicians*, Expert Systems with Applications, 25(4).

CmapTools, official Homepage, http://cmap.ihmc.us/

FERNANDO VEGA-RIVEROS, J., GLORIA PATRICIA MARCIALES-VIVAS, MAURICIO MARTÍNEZ-MELO (1998), *Concept Maps in Engineering Education: A Case Study*, Global J. of Engng. Educ., Vol. 2, No. 1.

FIGUEIREDO, M., LOPES, A. S., FIRMINO, R., SALOMÉ DE SOUSA (2004), *"Things we know about the cow": Concept mapping in a preschool setting*, Proc. of the First Int. Conference on Concept Mapping, Pamplona, Spain.

GERVAL, J-P., LE RU, Y. (2006), *VELab: A Virtual Lab for Electronics Virtual Experiments*, Advanced Technology for Learning, Volume 3, Issue 2, ACTA Press.

REITMAYR, G., CARROLL, S., REITMEYER, A., WAGNER, M. G. (1998), *DeepMatrix – An Open Technology Based Virtual Environment System*, White Paper, October 30.

GÜTL, CH., PIVEC M. (2002), *Virtual Tutor*, Proc. of ED-MEDIA 2002, Denver, USA, 668-672.

JESS: The Java Expert System Shell, official Homepage, http://herzberg.ca.sandia.gov/

KREMER, R. (1994), *Concept Mapping: Informal to Formal*, ICCS'94, Proceedings of the International Conference on Conceptual Structures, University of Maryland.

MBROLA, official Homepage, http://tcts.fpms.ac.be/synthesis/mbrola.html

MICHAEL J., ROVICK A., GLASS M., ZHOU Y. and EVENS M., *Learning from a Computer Tutor with Natural Language Capabilities,* Interactive Learning Environments, 11(3): 233–262.

POPOVICI, D. M., SERBANATI, L. D., GERVAL, J. P. (2003), *Virtual Perception Based Agents in Virtual Theater*, Technologies for Interactive Digital Storytelling and Entertainment, TIDSE 2003, Darmstadt, Germany, march 24-26.

SINGHAL, S., ZYDA, M. (1999), *Networked Virtual Environments*, Addison-Wesley, ISBN: 0-201-32557- 8.

VRML Standard Version 2.0, ISO/IEC CD 14772, 1996, http://vrml.org/VRML2.0/