Web-oriented educational system for digital circuits modeling and simulation

Marko Čupić
Faculty of Electrical Engineering and Computing
University of Zagreb
Unska 3, Zagreb, Croatia
Phone: +385 1 6129 548 E-mail: Marko.Cupic@fer.hr

Abstract – To foster students learning on digital logics and digital circuits design, theory lessons should be closely intertwined with practical exercises and experimentation. In this paper, we will present a web oriented educational system for digital circuits modeling and simulation – VHDLLab. Using VHDLLab students can experiment with simple or complex digital circuits and verify their knowledge on circuit construction and behavior. Most important advantages of this system when compared with commercially available VHDL environments and simulators are small size, no-licencing, free of charge, and design adjusted to students.

I. INTRODUCTION

When learning on digital logics and digital circuit design, students should be given the opportunity to practically verify what they have learned. This will not only provide them with a feedback on the width and depth of acquired knowledge, but will also enable them to experiment and deepen their understanding. On faculties, this opportunity is usually provided by the means of laboratory exercises. When dealing with digital circuits, exercises can be based on digital proto-boards, where students can work with actual integrated circuits and use wires to implement required circuits. But this is often not attainable, since it requires additional equipment, such as logical analyzers and wave generators, making cost of single working place unacceptable for large student population (courses with several hundreds or thousands of students). Alternative is a circuit modeling and simulation. Chip manufacturers such as Altera [1] and Xilinx [2] provide some tools which can be used for this purpose. Drawbacks, from the educational point of view, are extreme complexity of software, many options, problems with licenses and huge installation packages. To enable students to work and practice at home, at their own pace and time, students should each download a copy of software and go through licensing acquisition process – something that not many are willing to do.

To foster students' exploration and experimentation with digital circuits based on modeling and simulation, we have developed a web-oriented system named VHDLLab. This system is modular, based on client-server architecture and open source technologies, all of which are free. Its client is extremely small in size, and can be used within any java-enabled web browser.

This paper will describe VHDLLab and its capabilities from the aspect of support for learning and exploration of simple digital circuits. The paper is organized as follows. In section 2 we will describe organization of laboratory in courses on Digital electronics and Digital logic, and analyze some of present problems. We will than present a list of properties we consider important for beginners-oriented simulation software. In section 3 we will give an overview of VHDLLab system. In section 4 we will analyze supported tools for circuit specification built in VHDLLab. In section 5 we will discuss a short survey conducted on VHDLLab users. In section 6 we will give some future work directions. Conclusion is given in section 7.

II. LABORATORY ORGANIZATION AND PROBLEMS

During past five years we have been involved in organization of two courses on Faculty of Electrical Engineering and Computing connected with digital circuits: Digital electronics and Digital logic [3]. Digital electronics had an average of 600 enrolled students each academic year. Digital logic is its successor on a new FER-2 curriculum, and has an average of 900 enrolled students per academic year (peak was more that 1100 students in academic year 2005/2006 – first year the course was given).

During this time, both courses had similar laboratories. Students were given an opportunity to design simple and medium complex digital circuits, describe them using VHDL [4,5,6,7] (Very High Speed Integrated Circuit Hardware Description Language) and simulate its behavior. Given problems included both combinatorial and sequential circuits (simple logical gates, multiplexors, decoders, flip-flops, registers, counters and final state automata).

Organization of those laboratories proved to be very challenging. As a program of choice we selected Xilinx ISE WebPack bundled with ModelSim simulator. Those two programs occupy on disk more than 660 MB, and user is required to download installation packages and updates of also several hundred megabytes. In order to legally use it, each student is required to obtain a license (there is a free evaluation license available). Students that whish to practice at home must also separately download the whole installation package.

Since we are using shared faculty laboratories, installing the software also proved to be very difficult. On available computers, students have rather restricted user accounts (far from administrator permissions), so many programs simply do not work. This was a serious issue for us, since ModelSim tries to write files in Windows directory, which is off-limits for restricted users. This is the main reason why we use rather old ISE WebPack 6 – that was the last version we successfully installed under those restrictions.

Another issue was installation that (if user was not extremely cautious) added some sort of USB drivers on

Windows operating system, and disabled recognition and normal operation of users USB peripherals.

Finally, Project Navigator which is a part of ISE WebPack proved to be not very stable, and frequently crashed. All of this resulted with similar situation each year: at the beginning, students were given detailed instruction on how to obtain program, how to install it, how to obtain a license, to save their work as often as they can etc.

Complexity of user interface of those programs added additional burden on students. Those are excellent commercial programs giving its users means and options to perform many difficult tasks. However, this has its price with novices – instead to learn how simple logical AND gate works, student must spend a large amount of time trying to figure out where to add a new model, what to press to start simulation, which kind of simulation should be started (there are several available, ranging from simple behavioral model simulation to simulation after placement in FPGA circuit). And then student should also learn how to work with simulator itself.

On course Digital electronics we organized eight laboratory exercises, so we typically used the first one for introduction with ISE WebPack and ModelSim, leaving us with seven exercises for concrete work (this was a reasonable percentage of time spent learning the software itself). With introduction of course Digital logic, laboratory was reorganized into only three (longer) laboratory exercises, requiring from students to adopt programs and simultaneously complete given tasks from start. This proved to be a lot harder.

Taking all of the above considerations into account, we have started a project to design a new educational system for digital circuits modeling and simulation. Main goals were as follows:

- emphasize educational aspects of system
- support for adequate subset of VHDL
- simple and intuitive user interface with restricted number of options (or no options at all)
- easy to start
- easy to work with
- support for alternative circuit descriptions, such as drawing schemas and drawing final state automata
- easy conversion of alternative circuit description into VHDL
- no licensing
- free of charge
- small program size
- easy installation
- shared project storage accessible from Internet

Building a system with those characteristics would enable students to work and practice at home, at their own pace and time, and to complete laboratory exercises in known and common environment. The last requirement, shared project storage accessible from Internet means that system should not store project files on local file system, but on remote server. This will enable student to access his/her project from any networked computer, rendering additional benefit: student can complete laboratory exercise at home, with no time constraints. Later, during the formal laboratory time student can access his/her files and demonstrate designed circuits and simulation results.

III. VHDLLAB OVERVIEW

To design and build the described system, we have gathered a team of students already completed Digital electronics course, and after two years of work we had a functional system based on client server architecture, as illustrated in Figure 1. When developing server side and client side, there were several factors taken into account. For client side, we wanted a simple thin client capable of accessing project data on server and enabling user to describe digital circuits and display simulation results. Since many of our students use various operating systems such as Linux, Unix, and Microsoft Windows, a natural choice was Java programming language [8]. Initial client implementation was using Java Applet technology [9], requiring only a Java plug-in in user's web browser. However, since applets have a number of limitations, and

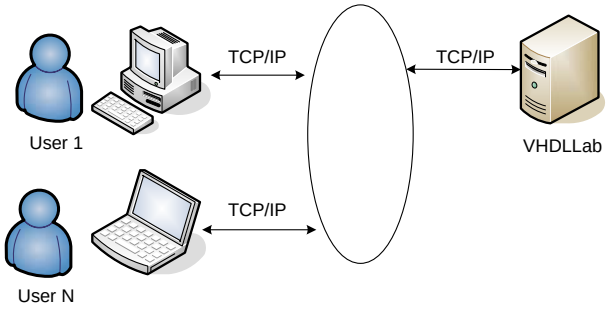

Figure 1. Client-server system architecture

require running web browser, we have moved to Java Web Start technology [10], enabling us to provide a user with desktop icon for starting client and no need for running web browser.

Considering the server side of the system, we wanted a program which will work on Linux based server, with option of easy porting to another operating system, and specifically to Microsoft Windows. Since this part of system works with relational database and uses operating system for starting new processes, once again, a natural choice was Java programming language. Architecture of server side is illustrated on Figure 2. Complete server side was implemented as a Java web application running inside of Apache Tomcat servlet container [11]. For data storage system uses the MySql relational database [12], and Hibernate object relational data mapper [13]. This

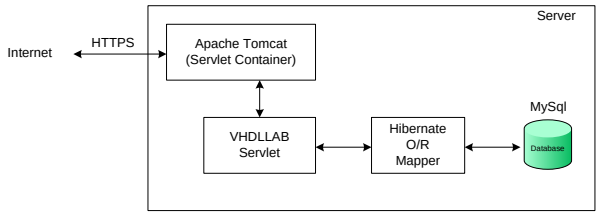

Figure 2. VHDLLab server architecture

configuration was then tested on Linux and Windows operating system and no problems were found.

As part of this project we have developed a simple VHDL parser which enabled us to track file dependencies. However, writing a full VHDL parser, compiler and simulator is rather demanding task. So instead, we searched the Internet for available open source free of

charge VHDL simulators which can be run from the command line, which provides no separate user interface but produce simulation result in file and which requires no licensing. The only available simulator which meets all of those constraints was GHDL [14] which we adopted. Simulation results produced by this simulator are stored in Verilog's VCD format (Value Change Dump).

Being designed as client server application, VHDLLab has its advantages and disadvantages. Its major advantages are already described as main goals of this project, all of which VHDLLab meets. Disadvantage is the need for constant connection on Internet during application usage. Namely, in order to work with VHDLLab, student must start its client, which connects to the VHDLLab server and retrieves student's projects. All modifications student makes are then immediately stored back on the server. When simulation is required, once again simulator is started on the server, simulation is performed and results are sent back to client. This enables a client to be extremely thin, but requires constant Internet connection. Another benefit is opportunity to finish all laboratory exercises at home, which students appreciated.

IV. SUPPORTED CIRCUIT DESCRIPTIONS

In order to have an educational value and to be as closely bounded to the Digital logic curriculum as possible, we have implemented three kinds of circuit description: using VHDL, drawing schema and drawing automata.

Description using VHDL is the foundation for all circuit descriptions in VHDLLab, since we use VHDL simulator for simulation tasks. An example of VHDL description of two input AND gate is shown in Figure 3.

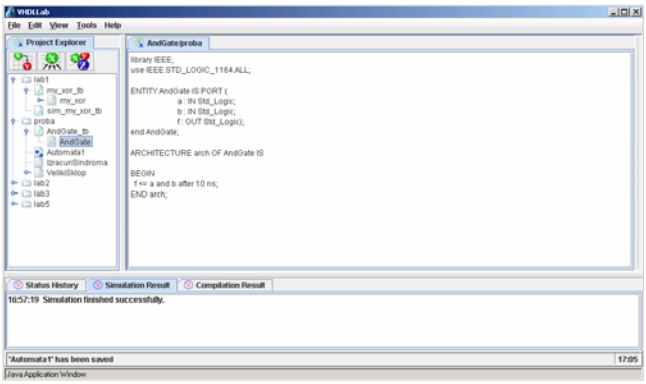

Figure 3. Writing VHDL in VHDLLab

As can be seen on Figure 3, VHDLLab client offers a simple but functional user interface divided in three areas. Left there is a Project Explorer displaying all of students projects and files inside project, hierarchically organized. Currently, student can have tree kinds of files: circuits, testbenches and saved simulations. Bottom area is for simulation/compilation/status messages, and central area is reserved for circuit editors. In this example, we can see a standard VHDL description of two input AND gate with propagation delay of 10 ns.

Once the student enters circuit design, a simulation should be performed either to explore what selected circuit does, or to verify its behavior. To perform this task student must add a testbench, and define desired inputs. This is shown on Figure 4.

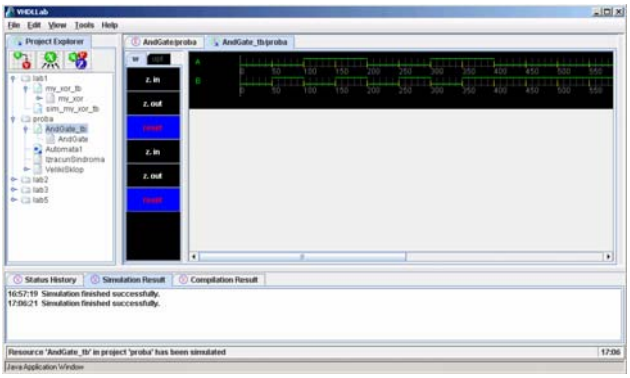

Figure 4. Creation of testbench for two-input AND gate

VHDLLab will enable students to create testbenches on two different ways. The simplest way is by clicking, and this is illustrated on Figure 4. The system will automatically recognize interface of circuit for which the testbench is being created and offer appropriate signals in drawer. More advanced users will eventually learn that testbench is just another VHDL module having empty interface, so VHDL support this kind of testbench creation as well.

Simulation can be performed by right clicking testbench and selecting Simulate. Simulation will be performed on server, and results will be displayed in editor, as shown on Figure 5.

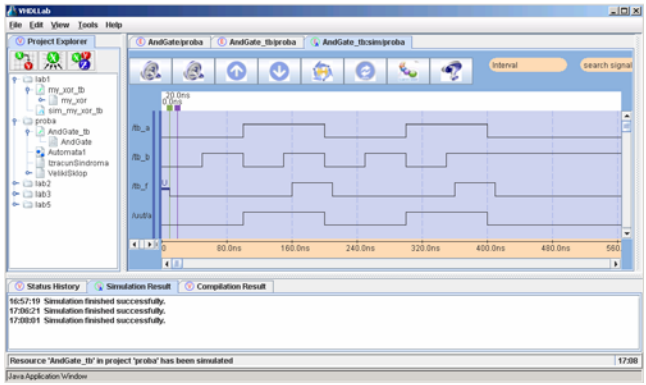

Figure 5. Simulation results for two-input AND gate

Component used for display of simulation result offers a range of commonly used tools in simulation result viewers. For example, there are tools for zooming, fast moving and cursors for interval measurements. Cursors can be automatically moved to next falling edge, next rising edge, previous falling edge or previous rising edge, given the selected signal. Signal naming used by result viewer is hierarchical, so that each signal can be uniquely identified.

When all simple circuits are described in VHDL, students can describe complex circuits by creating structural VHDL models. Structural models are models which are build from simpler models (for example, counter is built from flip-flops). Students are encouraged to do this on two different ways: first to create structural VHDL model themselves, and than to create a schema. Example of schematic usage is shown on Figure 6, for creating two-input XOR gate from ANDs, ORs and inverters.

Schematic offers a student a range of primitive gates which can be used for construction of more complex

circuits. Additionally, all circuits which student himself/herself defines in project will also be available for usage. Also, all primitive gates offered in schematic provides a range of parameters often missing in similar software, such as defining propagation delays. One reason for this omission is the fact that propagation delays specified by VHDL construct AFTER are not synthesizable. However, they are important for behavioral simulations and circuit explorations, so we decided to provide this capability.

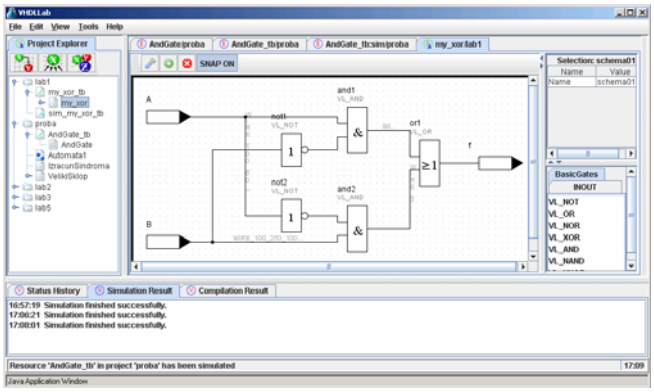

Figure 6. Schematic for two-input XOR gate

Another important aspect when learning VHDL is to recognize the fact that pure structural VHDL model and drawn schema are just two views of the same circuit. To emphasize this, VHDLLab offers student an option to view structural VHDL model for schema that student created (as illustrated in Figure 7). This is very educational experience, since by studying this, student can gain a better understanding of structural models, and how they are created.

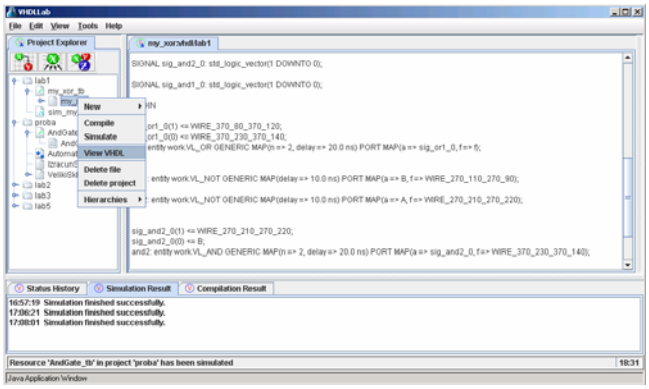

Figure 7. VHDL description of drawn schema

Finally, when it comes to sequential circuit design, students are taught to design circuits using finite state automata theory, and implement that design by, e.g. flip-flops. We believe that it is important to enable students to describe circuits directly on finite state automata level (for start), to get acquainted with this formalism. To foster this, we have implemented a third way to describe circuit behavior: by drawing automata. This is illustrated in Figure 8.

This example shows Moore finite state automata with states S0 through S4, with one input (a) and one output (y). Initial state is S0. Here also students can request to see

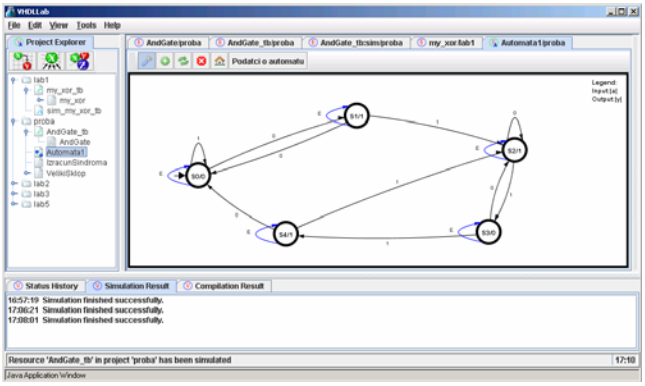

Figure 8. Specification of sequential circuit as automata

how would be this circuit be specified in VHDL. Example for automata from Figure 8 is given in Figure 9.

By studying this code, student can learn how Moore (or Mealy) type of finite state automata can be specified in VHDL. Given the fact that they learn to specify automata by defining new data type in VHDL automata

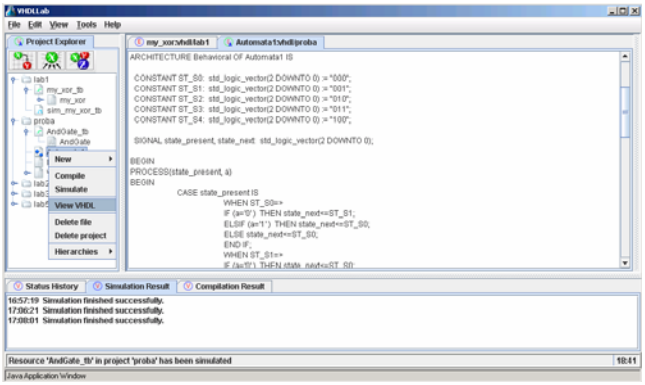

Figure 9. VHDL specification of automata from Figure 8

specification, interesting discussion can be developed on advantages of that kind of description over specifying each state as a constant, as illustrated in Figure 9.

V. SURVEY ON VHDLLAB

VHDLLab was first introduced into laboratory on course Digital logic in academic year 2007/2008. There were 850 enrolled students, and we offered them to use VHDLLab for laboratory exercises. From received applications, we have selected a test group of 42 students. Other students used ISE WebPack for laboratory exercises. All student followed the same curriculum, and had the same laboratory exercises. The only difference was in the program they used: ISE WebPack or VHDLLab. When semester ended, we have conducted a small survey to see how students responded to VHDLLab. From 42 students, 30 of them filled provided anonymous questionnaire.

On question "What do you think on possibility of doing laboratory exercises from home?" we obtained an average of 1.69 (where 1 was "absolutely satisfied" and 5 "absolutely unsatisfied") and 1.48 (where 1 was "extremely useful" and 5 was "extremely useless").

On question "How hard is the installation of VHDLLab?" we obtained an average of 1.93 (where 1 was "extremely easy" and 5 was "extremely difficult").

On question "What do you think on VHDLLab's user interface?" we obtained 1.97 (where 1 was "very easy to use" and 5 was "very hard to use") and 2.07 (where 1 was "intuitive" and 5 was "confusing"). This acknowledges our effort to keep user interface as simple and intuitive as possible.

Analyzing next questions, we found out that 96.67% (29 out of 30) of students find very educational VHDLLab's facility to view VHDL source for testbench, same amount of students find very educational VHDLLab's facility to view VHDL source for schema and 83.33% (25 out of 30) find very educational VHDLLab's facility to view VHDL source for automata.

On the question "If you had to do additional laboratory exercises, and if you could choose again, would you switch to ISE WebPack, or would you stay with VHDLLab, and why?", 29 students answered they would continue to use VHDLLab. Only one student answered he would switch to ISE WebPack, and provided interesting reason: "ISE WebPack is commercial tool used in industry; if I learn it now, I can benefit from it later.". Summing up answers of students which would stay with VHDLLab, we came to following arguments:

- It is simple.
- It is stable.
- I can work at home, when I want.
- It occupies almost no space on disk.
- There is no time limit on laboratory exercise. If I can not do something, I have all the time I need to figure it out.
- Group work. I can do laboratory exercise with the help of my friends.
- Consultations with others.
- It provides right amount of options for practical learning of Digital logics. Too many options only confuse beginners.

By analyzing those results, we believe we have developed a truly simple and educational system for digital circuits modeling and simulation, appropriate for students learning basic topics of Digital logics: Boolean algebra, simple combinatorial circuits such as multiplexors, decoders and adders, and simple sequential circuits, such as flip-flops, registers, counters and similar.

VI. FURTHER WORK

There are many small things which can be improved in VHDLLab, and after actively working with this software for one academic year we have collected a list of things to improve. However, there are also some new features which should be added. During the course of Digital logic students are taught on standard programmable circuits, such as CPLDs and FPGAs. When developing for FPGA, Xilinx ISE WebPack can show which CLBs (Configurable Logic Blocks) are utilized for specific circuit implementation, and provide a detailed overview (this latter, unfortunately, not with an evaluation license). For students learning on functions and circuits implementation, this can be extremely educational. So, this is set high on our to-do list: enable students to see, given some specific rather simple CLB, how would their design be implemented in FPGA.

VII. CONCLUSION

In this paper we have described a web-oriented educational system for digital circuits modeling and simulation: VHDLLab, which was built using open source and free of charge technologies. The main reason for its development was to design a system that emphasizes educational aspects, and is appropriate for beginners in Digital electronics and Digital logic.

VHDLLab has many limitations when compared with commercially available software, such as Xilinx ISE WebPack. However, many of those limitations were actually purposely imposed, to make the design and simulation of simple digital circuits as easy as possible, and especially appropriate for beginners.

From the other side, VHDLLab has some other features missing in commercially available software, whose sole purpose is to help student to learn and experiment with digital circuits and its design. The system was founded on VHDL, and for simulation purposes uses freely available command line VHDL simulator.

Given its small client footprint and shared project storage, VHDLLab enables students to work from home, when they want, and as long as they need to accomplish laboratory exercise. This alone resulted with very satisfied users. VHDLLab also makes client installation in faculty laboratories extremely simple. Just for comparison, three years ago, we had to install Xilinx ISE WebPack on 120 separate computers, which proved to be extremely time consuming. Installation of VHDLLab is unnecessary, thanks to Java WebStart technology, which only requires a Java to be installed (and this is true for all laboratory computers).

ACKNOWLEDGMENTS

The author would like to thank following students: Ivan Alić, Miro Bezjak, Davor Delač, Boris Ožegović, Aleksandar Prokopec and Tomislav Rajaković, who carried the biggest part of designing, programming, debugging and improving VHDLLab.

REFERENCES

[1] Altera, http://www.altera.com, visited on 2008-01-20.

[2] Xilinx, http://www.xilinx.com, visited on 2008-01-20.

[3] Digital logic, course, http://www.fer.hr/predmet/diglog, visited od 2007-12-10.

[4] P. J. Ashenden, *The Designer's Guide to VHDL*, 2nd Edn., Morgan Kaufmann, San Francisco, 2002.

[5] S. Brown and Z. Vranesic, *Fundamentals of Digital Logic with VHDL Design*, McGraw Hill, Toronto, Canada, 2000.

[6] P. Walsh, "Integrating vhdl into a first course in logic design," in *IEEE Canadian Conference on Electrical and Computer Engineering*, Edmonton, Canada, 1999, pp. 1531–1534.

[7] S. Areibi, “A First Course in Digital Design using VHDL and Programmable Logic.” *Proceedings of Frontiers in Education Conference*, 2001, pp. 19-23.
[8] Java, http://java.sun.com, visited on 2008-01-10.
[9] Java Applets, http://java.sun.com/applets/, visited on 2008-01-10.
[10] Java Web Start, http://java.sun.com/products/javawebstart/, visited on 2008-01-10.
[11] Apache Tomcat, http://tomcat.apache.org/, visited on 2008-01-10.
[12] MySql, http://www.mysql.com/, visited on 2008-01-10.
[13] Hibernate, http://www.hibernate.org, visited on 2008-01-10.
[14] T. Gingold, GHDL, http://ghdl.free.fr/, visited on 2008-01-10.