A Virtual Laboratory for Digital Signal Processing

Chyi-Ren Dow, Fu-Wei Hsu, Tsung-Kai Yang and Jin-Yu Bai
Department of Information Engineering and Computer Science
Feng Chia University, Taichung, Taiwan
crdow@fcu.edu.tw, {sam, toycat, ohright}@pluto.iecs.fcu.edu.tw

Abstract

This work designs and implements a virtual digital signal processing laboratory, VDSPL. VDSPL consists of four parts: mobile agent execution environments, mobile agents, DSP development software and DSP experimental platforms. The network capability of VDSPL is created by using mobile agent and wrapper techniques without modifying the source code of the original programs. VDSPL provides human-human and human-computer interaction for students and teachers, and it can also lighten the loading of teachers, increase the learning result of students, and improve the usage of network bandwidth. A prototype of VDSPL has been implemented by using the IBM Aglet system and Java Native Interface for DSP experimental platforms.

***Keywords:** Digital signal processing, virtual laboratory, e-learning, mobile agent, wrapper.*

1. Introduction

Digital signal processing (DSP) [9, 17] is one of the most powerful technologies in the twenty-first century and is a growing subject area in Electrical, Computer Science and other Engineering/Science disciplines. DSP is closely linked to our life and is widely applied in many fields such as: telecommunications, robotics, consumer electronics, medicine, military, instrumentation, aerospace industry, and automobile. Each of these areas has developed a deep DSP technology, with its own algorithms, mathematics, and specialized techniques.

Although DSP is the trend of future technology development, the learning of DSP is not an easy task for novices. Not only the DSP hardware architecture, but also the flexible and powerful instruction sets of DSP chips are difficult for students. Thus, fast and convenient CAI tools for the DSP learning are necessary. Currently, most DSP learning tools are standalone. This kind of learning approach has only human-computer interaction and lacks of human-human interaction [4, 6] such as teacher to student and student to student. In order to add human-human interactions, it is necessary to create network capability for DSP-learning tools. A network enabled DSP learning environment can support multiple users and allow them to interact with each other to increase their interests in learning DSP in any place and at any time via the Internet.

In addition to the network capability, a DSP virtual laboratory should support the features of multimedia and multi-level. Through multimedia demonstrations, students can easily understand various DSP theories. We can use the multimedia technology to enhance an experimental environment for students. Furthermore, a DSP course material should be organized in multiple levels so students can select DSP studying materials according to their ability to reduce the frustrations when learning and deepen their impressions about DSP.

This work designs, develops and implements a Virtual DSP Laboratory, VDSPL using mobile agent and wrapper techniques. The autonomous feature of mobile agents can be used in the virtual laboratory to substitute for a teacher's behaviors and actions in a practical laboratory. Mobile agents could guide several groups of students in different places simultaneously. When a student needs to interact with the teacher, the virtual laboratory can dispatch a mobile agent to perform this function. For a student, the mobile agent can play a learning guide and arrange the learning activities, to improve the learning efficiency in a virtual laboratory.

The rest of this paper is organized as follows. First of all, in Section 2 we discuss the background materials and related work. Section 3 describes the system architecture of our work. The system implementation and prototype are presented in Sections 4 and 5, respectively. Conclusions are made in Section 6.

2. Related Work

There are many research areas related to our work, including virtual laboratory, digital signal processing, mobile agent techniques and wrapper concept. These topics are described in this section.

Distance education can be done in a wide variety of styles via different learning models. The virtual laboratory is one of the important components for macro university architecture [1, 4]. Students are required to learn some courses through online experiments and simulations, and the virtual laboratory is provided for the students to conduct course related experiments and simulations via networks. Based on the equipment and user access in each experiment, laboratories can be classified into four types [4]. The first type of laboratory is the practical lab. This is a traditional laboratory. The second type of laboratory is the remote lab. This kind of laboratory uses physical experimental equipment and allows users to remotely access the equipment and instruments. The third type is the micro lab which provides some virtual equipment and allows only local access. Traditional computer-assisted instruction (CAI) tools belong to this type. The fourth type is the macro lab which consists of one or more micro labs and allows remote access through the Internet. Some web-based learning environments belong to this type. The virtual laboratory proposed in this work is a hybrid of the remote lab and the macro lab.

The theorems of DSP use the mathematics and the algorithms to manipulate the signals [7, 14] (ex. seismic vibrations, visual images, sound waves) after they have been converted into a digital form. Currently, there are some DSP electronics manufactures (such as TI, Motorola, NEC, and Analog Device) to develop their own series of DSP chips. For instances, TI developed a series of high performance DSP chip called TMS320™ DSPs. In the past few years, it is a challenge for student to learn the DSP concepts, theorems and algorithms without any auxiliary simulating/emulating tools in a lecture class. With the rapid technological changing, there are several powerful simulation software tools (e.g., MATALAB, MATHCAD [2, 15]) for student to learn DSP. Through this learning method, students cannot practice DSP experiments with DSP hardwares; they can only simulate digital signal processing by the simulation tools.

The mobile agent [3, 5, 8, 10-11, 13] is an emerging technology that can be applied in many fields, including electronic commerce, personal assistance, secure brokering, distributed information retrieval, telecommunication networks services, workflow applications and groupware, monitoring and notification, information dissemination and parallel processing, etc. [4, 8]. The use of mobile agents can bring several advantages [8, 11], including the reduction of the network traffic and latency in client/server network computing paradigm, protocol encapsulation, dynamic adaptation, heterogeneity, robust and fault tolerance. In the past few years, there are several contemporary mobile agent systems [8, 12, 16] developed, and two main categories of mobile agent systems can be identified: systems based on the Java language (e.g. Mole [10], Aglets [8, 16], Odyssey, Concordia, and Voyager. [10]) and systems based on scripting languages (e.g. Agent Tcl [10], Ara Tcl-based Ara, and TACOMA).

One important problem we face when building a virtual laboratory is where to place an extra function for a stand-alone learning tool without knowing its source code. To be included in a virtual laboratory and used via networks, these stand-alone learning tools have to be modified. In our approach, we use the wrapper concept to implement our virtual laboratory. Wrapper [4, 13] is a technique that provides a convenient way to expand upon existing functions of an application program, without modifying its source code. Wrappers intercept function calls, method invocations, and messages to the application software that they wrap, redirecting or doing pre- and/or post-processing of input/output. Wrappers provide a way to compose applications from different parts. The fact that a mobile agent is wrapped should be transparent to other mobile agents in the system, and potentially to the agent itself.

3. System Architecture

The system architecture of our virtual laboratory and the functions of each component of the system are described in this section.

3.1. System Overview

In our proposed framework, we use the mobile agent techniques to construct the VDSPL. Figure 1 shows the system architecture of VDSPL. There are four major components in our virtual laboratory. These components are the mobile agent execution environments, mobile agents, DSP development software and DSP experimental platforms. Various mobile agents are designed to assist a teacher. The mobile agents in the teacher side can be dispatched to the student side to represent the actual teacher and interact directly with the students. The development tools are standalone programs on the teacher and student sides. Furthermore, some middlewares are designed and wrapped into our agents to provide interactive functions and rules for mobile agents and the development tool.

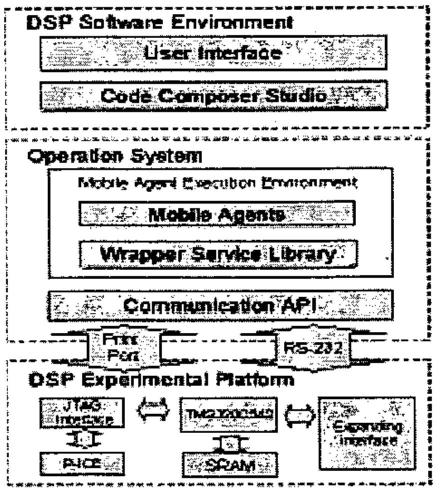

Figure 1. System Architecture

3.2. Mobile Agent Execution Environment

A mobile agent requires an execution environment called the mobile agent execution environment (MAEE) [8]. This environment must be installed on the student and teacher sides to provide a necessary runtime environment for agents to execute. The environment's basic facilities include mobility, communications, naming and location, and security. All mobile agents are received and executed in the environment and we also regard it as an entry point or operating system for mobile agents. Furthermore, four important roles exist in MAEE, including the engine, resources, location and principal. The engine serves as a workhorse or virtual machine for MAEE and mobile agents. The resources include networks, database, processors, memory and other hardware, and software services. The location can be typically written as an Internet Protocol (IP) address and a port of the engine with a MAEE name attribute. Principals like agents that have the responsibility for the operation of MAEE. The MAEE is implemented by using the Java language. Therefore, MAEE is a java application that runs on the Java virtual machine (JVM) and has the following good properties: platform independence, secure execution, dynamic class loading, multithread programming, object serialization and reflection.

3.3. Mobile Agent

The mobile agent is a principal role in the virtual laboratory. Different mobile agents such as the guide agent, demo agent, learning agent, monitor agent, homework agent and assessment agent can be designed for our learning environment. A teacher can use various mobile agents to assist students to learn. A guide agent can be used to provide an interactive interface between the teacher and the student. On the teacher side, the guide agent provides various assisting functions for the teacher. On the student side, the learning agent has some predefined FAQ rules and it will reply appropriate answers from a knowledge base when the students ask some common questions or the user's behavior matches certain rules. The monitoring agent could act as the teacher to monitor the student's actions and learning status. The homework agent could act as the teacher to dispatch homework to the student and record the student's homework execution status. The assessment agent could give an assessment to check the student's learning results and provide different levels of assessment materials. The demo agent helps the teacher to demonstrate the steps of the experiment and let the student have an overview of the experiment.

3.4. DSP Experimental Environment

The DSP experimental environment contains two parts: hardware environment and software environment. In the DSP software environment part, the software is the DSP program development software which is an existing application software without network capability and provides a powerful integrated environment and have several necessary analysis tools to develop DSP programs. The software makes it easier and faster to implement DSP programs using C as opposed to the assembly language. The software also includes the debugging and real-time analysis capabilities. Currently, there are many DSP software environments such as Code Composer, Matlab, Altera, etc.

In the hardware part, the DSP experiment platform adopts the digital signal processor from the DSP chip manufacturer. The hardware platform consists of a DSP emulator and debuggers. They can support the user in debugging the DSP program code through a standard parallel port or PCI slot. Through the integration of the software and hardware environments, we can develop, debug, modify and execute our DSP programs.

4. Implementation

This section describes our system implementation. The mobile agent and learning platforms are presented first. Expanding the network capability for the virtual laboratory system is described next. Then, agent models and on-line learning implementation are presented.

4.1. Platform

Our virtual laboratory system consists of two platforms. The first is the mobile agent platform and the second is the DSP experimental platform. These two platforms are installed on the teacher and student sides. The mobile agent platform is Aglets, which was developed by the IBM Research Laboratory in Japan. The Aglets Software Developer Kit (ASDK) requires the JDK 1.1 or higher to be installed and is the first Internet agent systems based on the Java classes. The ASDK provides a modular structure and an easy-to-use API for the programming of Aglets. The Aglets are Java objects and can travel from computer to computer via networks. The migration of Aglets is based on a proprietary Agent Transfer Protocol (ATP). An aglet that executes on a host can suddenly halt execution, be dispatch to a remote host, and resume execution. When the Aglet moves, it takes along its program code as well as the states of all of the objects that it is carrying. The security mechanism of Java virtual machine and Aglet makes a host safe when receiving the Aglets data.

The DSP experimental platform is composed of TI's integrated development tool CCStudio, Dmatek PRO-OPEN TMS320C542 DSP Controller and PICE-DSP ICE 320C542 [18]. CCStudio software is a fully integrated development environment and supports TI's leading DSP platforms. It integrates all host and target tools in a unified environment, including TI's DSP/BIOS™ kernel, code-generation tools, debugger, and Real-Time Data Exchange (RTDX) technology to simplify DSP system configuration and application design. CCStudio also has an open architecture that allows TI and third parties to extend the IDEs functionality by seamlessly plugging-in additional specialized tools. Through the CCStudio, the students can learn DSP from multimedia presentation of real-world signals and system theory. Dmatek DSP Controller is an experimental board based on TI's TMS320C542 DSP chip and design for users to realize the function of DSP chip and its peripheral device. PICE-DSP ICE 320C542 is in-circuit emulator for DSPs.

4.2. Network Capability

The network-enabled VDSPL capability is implemented by using Aglet design patterns and the wrapper concept. Design patterns are reusable components and have been proven to be very useful in the object-oriented field to achieve good application designs. The wrapper concept is used to expand new capabilities for an existing tool without modifying the original source code. The implementation of wrapper concept uses Aglet design patterns and the Java native interface (JNI). The Aglets design patterns include traveling patterns, task patterns and interaction patterns. We add the network capability for the virtual laboratory by inheriting the traveling patterns. These patterns can deal with various aspects of managing the movements of mobile agents, such as routing and quality of service and they also allow us to enforce encapsulation of mobility management that enhances reuse and simplifies aglet design. Furthermore, the traveling patterns include three traveling models, including Itinerary pattern, Forwarding pattern and Ticket pattern. In our approach, we use the Itinerary pattern and Forwarding pattern.

4.3. Agent Models

In order to remotely control VDSPL, we use the JNI to connect the Win32 API in the initializeIterface. The Java native interface and Visual C++ are used to bind the Win32 API such as the "jni2c.dll" dynamic link library (DLL). An interface is initialized between other mobile agents and VDSPL for the wrapper agent. Moreover, the wrapper agent can execute a doCommand function that can be called by other mobile agents to control and monitor VDSPL. The wrapper agent can also respond to the results based on a wrapper script. Figure 2 shows the trigger of Windows API using JNI.

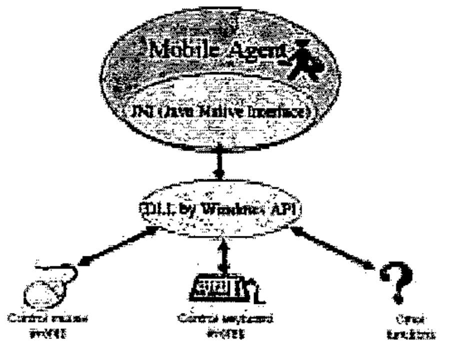

Figure 2. Trigger of Windows API using JNI

In our system, there are six mobile agents implemented, including guide agent, monitor agent, demo agent, assessment agent, homework agent, and learning agent. These agents are designed for the platform on the teacher side and the student side, and each mobile agent has different ability. The guide agent, assessment agent, demo agent, homework agent work in the foreground. Above agents have user interface to allow the user to interact directly with the system. Other agents without the awareness of their existence by the user work in the background.

There are three basic patterns for an agent, the Aglet class object, wrapper class object and guide class object. The Aglet class allows the mobile agent to execute in the Aglet agent execution environment. This object class provides VDSPL the network capability. The wrapper class object provides mobile agent a way to interact with the wrapper agent. The guide class allows agents to communicate and interact with the user. This class provides function calls for the wrapper script. Each type of mobile agent uses different teaching and learning knowledge-based rules. If the predicate of each rule is satisfied, the mobile agent will take predefined actions.

5. System Prototype

A prototype of VDSPL is presented in this section. As shown in Figure 3, when the mobile agent platform starts running, it will first initiate an experimental platform and provide an agent list for the teacher. If the teacher needs an agent service or wants to communicate with students, the guide agent can be used to do so. After the guide agent clones a learning agent for students, the learning agent will carry the learning materials which are determined by the teacher. When the learning agent starts, the students will receive a message informing them and the learning program will start and then load the DSP learning materials.

Figure 4 is a snapshot of the prototype when the demo agent starts, and the demonstration example will be presented step by step according to the demo script. In VDSPL, we have also created a web site (http://pluto.iecs.fcu.edu.tw/~dsp/index.htm)that provides news, DSP introduction, and DSP material zone, on-line learning, download, discussion board, and related links.

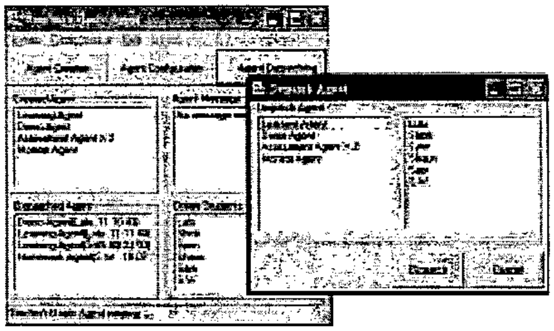

Figure 3. Guide Agent

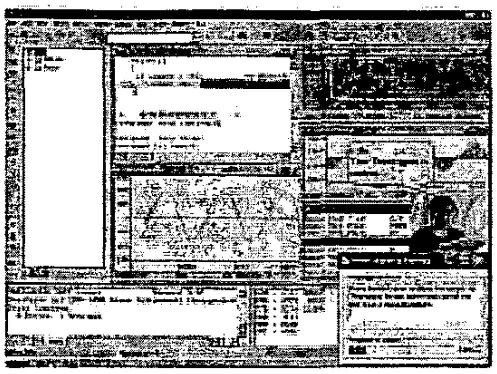

Figure 4. A Snapshot of the System

6. Conclusions

In this paper we present the VDSPL, a mobile agent-based virtual digital signal processing laboratory. Our system incorporates mobile agent techniques with DSP development tool to provide teachers and students with various instructions and interactions. The mobile agent and wrapper techniques are used to enable the network capability of standalone DSP development tools and improve the teacher-student interaction for distance DSP learning. Furthermore, the students can get guidance and learn in the personalized environment through mobile agents. In addition, the mobile agent and design patterns are also used to perform software re-engineering and provide a virtual laboratory.

Reference

[1] S. K. Chang, T. Arndt, S. Levialdi, A. C. Liu, J. Ma, T. Shih, and G. Tortora, "Macro University A Framework for a Federation of Virtual Universities," International Journal of Computer Processing of Oriental Languages, Vol. 13, No. 3, pp. 205-221, September 2000.

[2] A. Causen, A. Spanias, A. Xavier, and M. Tampi, "A Java Signal Analysis Tool for Signal Processing Experiments," Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, Vol. 3, pp. 1849-1852, 1998.

[3] A. I. Concepcion, J. Ruan and R. R. Samson, "SPIDER: A Multi-Agent Architecture for Internet Distributed Computing System," Proceedings of the ISCA 15th International Conference on Parallel and Distributed Computing Systems, pp. 147-152, September 2002.

[4] C. R. Dow, C. Y. Lin, C. C. Shen, J. H. Lin, and S. C. Chen, "A Virtual Laboratory for Macro Universities Using Mobile Agent Techniques," The International Journal of Computer Processing of Oriental Languages, Vol. 15, No. 1, pp.1-18, 2002.

[5] C. R Dow, C. Y. Lin, and F. W. Hsu, "A Mobile Agent-based Virtual Language Learning Laboratory," Proceedings of the International Conference on Chinese Language Computing, pp. 98-103, Taichung, Taiwan, July 2002.

[6] A. K. Dey, "Enabling the Use of Context in Interactive Applications," Proceedings of the 2000 Conference on Human Factors in Computing Systems, pp. 79-80, April 2000.

[7] W. S. Gan, Y. K. Chong, W. G. and W. T. Tan, "Rapid Prototyping System for Teaching Real-Time Digital Signal Processing," IEEE Transactions on Education, Vol. 43, No.1, pp. 19-24, February 2000.

[8] D. B. Lange and M. Oshima, "Programming and Deploying Java Mobile Agents with Aglets," Addison-Wesley, Reading, MA, 1998.

[9] S. H. Mousavinezhad and I. M. Abdel-Qader, "Digital Signal Processing in Theory and Practice," Proceedings of the 31st ASEE/IEEE Frontiers in Education Conference, October 2001.

[10] V. A. Pham and A. Karmouch, "Mobile Software Agents: an Overview," IEEE Communications Magazine, Vol. 36, No. 7, pp.26-37, July 1998.

[11] A. Silva and M. M. da Silva, J. Delgado, "AgentSpace: A Next-Generation Mobile Agent System," Lecture Notes in Computer Science, September 1998.

[12] L. M. Silva, G. Soares, P. Martins, V. Batista, and L. Santos, "Comparing the Performance of Mobile Agent System: A Study of Benchmarking," Technical Report, JAMES Project, 1999.

[13] N. P. Sudmann and D. Johansen, "Supporting Mobile Agent Applications Using Wrappers," Proceedings of the 12th International Workshop on Database and Expert Systems Applications, pp. 689-695, September 2001.

[14] H. T. Wu, T. C. Hsiao, C. L. Chen, C. M. Su, J. C. Su and J. C. Jiang, "An Integrated Teaching and Learning DSP Lab. System," Journal of Science and Technology Vol. 10, No. 1, pp. 29-36, January 2001.

[15] S. Wolfram, "Mathmatica: A System for Doing Mathmatics by Computer," Addison-Wesley, Reading, MA, 1988.

[16] "IBM's Java Aglet," http://www.trl.ibm.com/aglets/

[17] "Texas Instrument," http://www.ti.com.tw/

[18] "DMATEK Co. Ltd.," http://www.dmatek.com.tw/