Design and Implementation of VPL: A Virtual Programming Laboratory for Online Distance Learning

Alvin T.S. Chan[1*], Jiannong Cao[1], Chi-Kin Liu[1], and Weidong Cao[2]

[1]Internet and E-Commerce Laboratory,
Department of Computing
The Hong Kong Polytechnic University
Hung Hom, Kowloon, SAR of Hong Kong
[2]Department of Mathematics,
Jiangsu Institute of Education
Nanjing, Jiangsu, China
[*]Contact Author: cstschan@comp.polyu.edu.hk

Abstract. The virtual programming lab (VPL) project described in this paper offers is designed to facilitate Internet access to application software. It emulates a real computing laboratory environment that promotes group learning and project management. The laboratory resources are situated at the university and are centrally controlled. Users of the virtual programming laboratory include students, tutors and administrators of the system. Users can be located at different geographical locations and remotely access applications through the Internet. The virtual programming lab design is based on a distance education concept. This paper focuses on the design and development of the runtime modules within the VPL framework. These runtime modules provide underlying services that drive the launching of applications, file management and communications services. In addition, this paper presents an evaluation of the performance of the system.

1 Introduction

The growth of the Internet over the last few years has promoted many areas of web development including education. E-learning has become one of the fastest moving trends and is expected to play an ever more important role in the provision of distant education.

Web education makes distance learning effective and flexible. Students can learn according to their own schedules and, if necessary, follow a non-linear approach to acquiring knowledge based on their own capabilities. Students can access course materials anywhere at anytime. They can download lecture notes, communicate with the instructor via email, and sit examinations on the web. Already, there are popular e-learning software packages that support online lecturing and tutoring and considerable effort has been put into providing web-based learning and teaching.

All of this notwithstanding, it is not a simple matter to find reports that address the important issue of providing students with convenient online access to programming

W. Zhou et al. (Eds.): ICWL 2003, LNCS 2783, pp. 509–519, 2003.

facilities. Indeed, students who currently wish to access a particular item of software are invariably required to physically go to the computing lab to do it, and up until now, solutions to this problem have either been ad hoc or have required special-purpose web programming facilities [1][3].

The virtual programming laboratory system (VPL) enables users to launch programs at remote servers located in the laboratory and provides ease and better use of laboratory software resources. By providing wide area access through web infrastructure, VPL allows more students to remotely use the software located within the laboratory. An additional benefit of VPL is that it allows students to share resources and remote files with others. By providing a channel for users to communicate with their peers, VPL promotes collaborative learning.

2 Challenges in Laboratory-Oriented Learning

Advances in Internet technologies have led to new types of teaching and learning projects. Mostly, these projects have been limited to the dissemination of teaching materials, but some projects have tried to provide more attractive and Internet-informed features that support interactive, customized, and collaborative teaching and learning. Such Internet-informed projects have included providing a laboratory-learning environment on the web. This is an important step, as practical work plays a central role in learning science subjects. The aims of laboratory work are often synonymous with the aims of science courses [1] and aided learning. A laboratory with software access promotes learning because it allows students to apply concepts in practice. Yet the use of software in physical laboratory settings presents a number of challenges in terms of efficient and effective teaching.

Currently, little work is found on either ad hoc or special purpose web programming facilities that support virtual programming environment [2]. When students need to use lab-based software or have a discussion with project members, they are required to be physical present in a computer lab, causing students to waste time traveling and arranging meetings.

In short, laboratory resources are not leveraged effectively and uniformly. One reason for this is that software and application usage in laboratories often depends on course assignment workloads and project deadlines, so laboratories (and consequently software) have a very uneven usage pattern over a term. Similarly, usage tends to be very high during “normal” lab hours, while resources are under-utilized during off-peak hours, especially weekends. A further reason for ineffective and uneven leveraging of laboratory resources is geographical, in that the license usages are always distributed unevenly.

Instructor/tutor led laboratory sessions usually have an excess of materials to present. This may lead to students needing extended lab hours to complete their assignments. In these circumstances, instructors/tutors too, may face difficulty to managing their workloads, and find themselves unable to simultaneously run their courses and appropriately respond to students. Obviously, this can damage both teaching and learning and teacher and student morale.

A final problem is that software installation and configuration also consumes large amounts of time in the physical laboratory that could otherwise be spent on more productive learning activities.

The aim of the Virtual Programming Lab (VPL) [4] project is to solve each of these problems by enabling any student with access to a computer with a web browser and a connection to the Internet to gain access to laboratory resources, wherever, whenever. There are three desirable characteristics of the virtual learning environment [5] that are applicable to this project:

1. Supported and customized individual lab environment
2. Real-time and non-real time group usage of the virtual lab environment
3. Collaboration and learning between lab users

The VPL project supports these three characteristics. Users are assigned accounts to store their individual environment information. System support is provided for users and administrator to customize and manage their virtual lab environment. Within the virtual environment, users can interact with each other via built-in collaborative tools to promote interactive learning.

The VPL system also provides a statistical log of software usage patterns. An integrated mechanism to monitor the software usage patterns and user behavior are provided. The former is important for planning of resources, while the latter may be used as a basis to monitor students learning experience and behavior.

File management is also addressed in this project. The files in the VPL system can be stored in the local or remote machines. A simple job such as "opening a folder" involves many processes such as distinguishing the location of the file, connecting to the remote site, mapping to the remote user account and getting the file information. The file management function handles these complex tasks for the user. The user does not need to know the file location before copying the file. If a remote file is copied to local folder or vice versa, the file transfer between local and remote is necessary and should be performed transparently.

3 VPL Design

The purpose of the VPL system is to create a seamless laboratory environment that supports online distant learning. Because of the lack of laboratory simulation projects that allow remote program launching, the VPL system is designed to provide such functionality and a standard platform for user interaction.

Fig 1 shows the network infrastructure between clients and lab machines. The VPL system allows users to access the lab machine software through a thin client no matter what type of hardware, be a pc, laptop or PDA.

The overall system architecture adopts the typical 3-tier approach where the graphical user interface is located at the client tier, the virtual lab runtime API at the middle tier, and the database at the 3rd tier. The client side includes some components that provide access to the virtual lab runtime. The middle tier manages the virtual programming lab and provides APIs to access the database and the lab machine. The third tier stores information and resources in the system (see Fig 2).

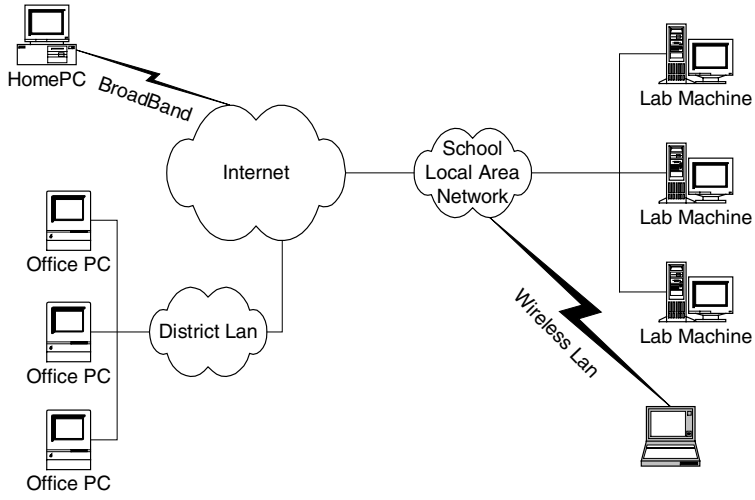

Fig. 1. System context of VPL

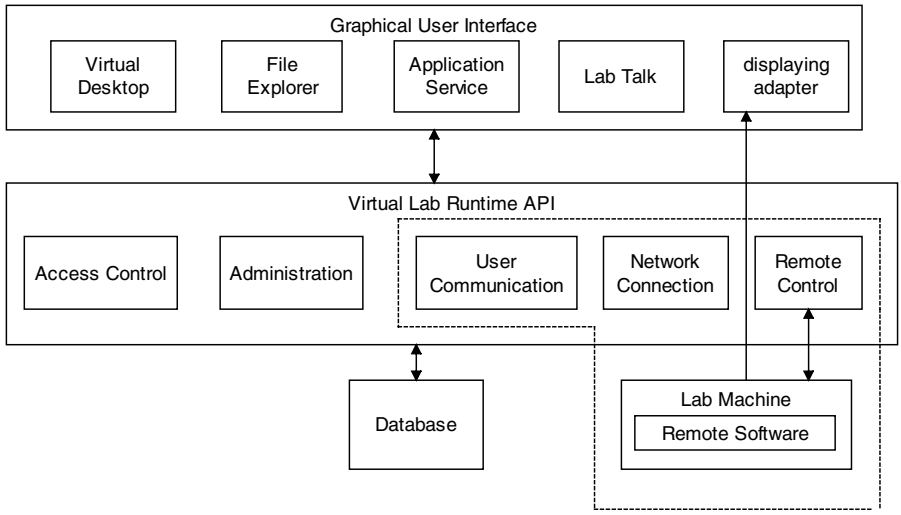

Fig. 2. Architecture of the virtual programming lab

This paper focuses on the virtual lab runtime part which is highlighted in dotted lines as shown in Fig 2. The runtime will appear as a set of software packages that bundled together to provide Java API access to service provided by the VPL system.

3.1 Design of VPL Runtime System

The design of the VPL runtime system can be broken down into five parts:

- File management subsystem—provides remote file management function for applications. The purpose of the subsystem is to enable the seamless transfer of files between local and remote systems operating in a graphical user interface. The component is designed to be modular so as to support open programming interfaces, where future applications can be built easily based on the API provided by this system.
- Communication subsystem—provides the communication function between users of the VPL system. In particular, it supports important functions to enable users to interact and collaborate with one another using tools such as messaging, chat and file attachments.
- Remote application launching—provides a remote software application launching function. This component forms an important part of the VPL system. It provides remote launching of software that is installed in the laboratory and to make it available to users accessing it via de facto standard web browsers. It is made up of a client applet and a corresponding terminal server which implement X protocol for remote terminal control.
- Usage monitor—designed for backend support and acting as a repository for real time usage monitoring. The component provides logging of activities that are performed by end users in terms of the pattern of software usage. This information is important to provide support for collaborative learning, where the usage monitor can inform end users who are concurrently using the same software. In such cases, users can choose to interact with one another. Knowledge of the usage patterns of users would also be useful in planning future software purchases and deployments.
- Administration subsystem—provides administrative functions that are similar to a real-life laboratory, providing facilities for users to be added or deleted, and application access rights to be granted.

The above sub-systems work in tandem to provide the collective functionality of the VPL runtime system.

4 Implementation and Performance Evaluation

A prototype of the VPL system is being implemented on a web platform hosted by the J2EE web application server. The implementation part follows the design as described in section 3. In this section, we discuss issues and techniques of implementing the VPL runtime system, including the file management sub-system, the communication sub-system, the usage monitor and the remote application launcher.

This project is implemented mainly in a Microsoft window and Linux environment. Java™ language is the main programming language used. The software

platforms used include the Oracle 8.1.7 relational database server, Oracle9i application server (J2EE compliance) and SonicMQ messaging server.

Also discussed are the collaboration with a backend database and case study on how to make use of the VPL runtime system.

4.1 Deployment and System Architecture

The system architecture of the VPL system includes a client pc connecting through the Internet to one web server and an application server.

The Client PC

The client pc contains two applets that are downloaded from the web server. Both of the applets are digitally signed.

The VPL applet contains necessary classes for running the virtual desktop graphical user interfaces and classes for connecting to backend services. It directly connects to the lab machine remote file system. In the deployment diagram, a remote file object is shown on the client PC. Physically, the remote file resides in the lab machine. The VPL applet is responsible for getting data from the database by connecting to the web/application server.

The displaying adapter applet is used to draw remote application graphical user interfaces on the client side. Shown in Fig 3 is a screen shot of the virtual desktop and the execution of a remote application GIMP.

The Web/Application Server

The web/application server is Oracle9i application server. It contains a web server to store web pages for the VPL system and an application server that is capable for running Java Servlet and Java Server Page (JSP). Java Servlet and Java Server Page are server side programming based on the Java standard enabling dynamic website building and connection to backend enterprise services. The message queue server is SonicMQ, it is used to provide message-based communication between client and server applications.

The Database Server

The database is based on the Oracle8i server. It is located within the school campus Intranet. In this way, both the web/application server and the lab machines will have direct access to the database by use the Java JDBC connection to the server.

The Lab Machines

Lab machines are computers providing remote software applications to remote users. They are located within the school campus Intranet. There are two system servers running in each of the lab machines: 1) a remote file server, a file server which provides remote file management functionality and 2) a usage monitor, which is a server program that keeps track of which program is being used and by which user. The usage status is updated to the database server.

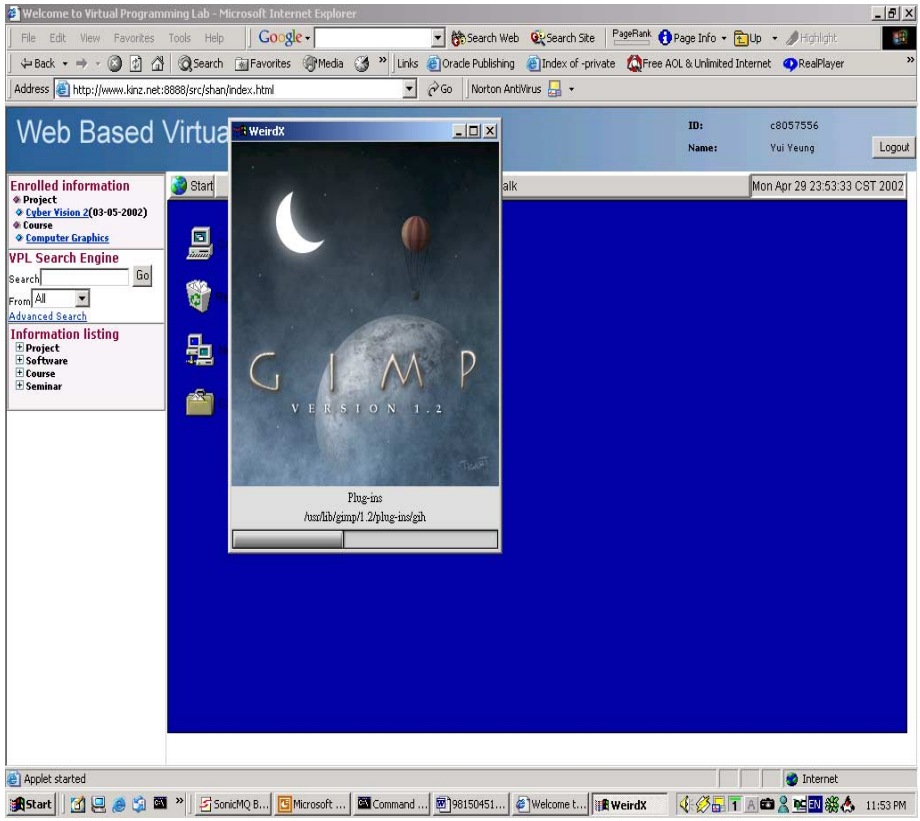

Fig. 3. Screen shot of VPL

The deployment diagram for the VPL system is shown in Fig 4. When the applet is started on the client side, it connects to the database though the web server and registers itself to the message queue server. There may be a number of lab machines working together to provide software usage monitoring. The usage monitor at the lab machine periodically connects to the database server to update the software usage status.

4.2 Experiments

To verify and evaluate the VPL system, we have set up an experimental test bed to measure the performance in terms of the delay in launching remote applications. The experiment focuses on the lab machine connection speed as shown in Fig 5.
The following software is used to measure the network performance on the client side and to measure the system resources usage on a Linux lab machine.

1. Network Smart Lite 1.0 (Build 385) is used to measure throughput, latency and, data sent and received. This application can be used on the connection with a LAN card.

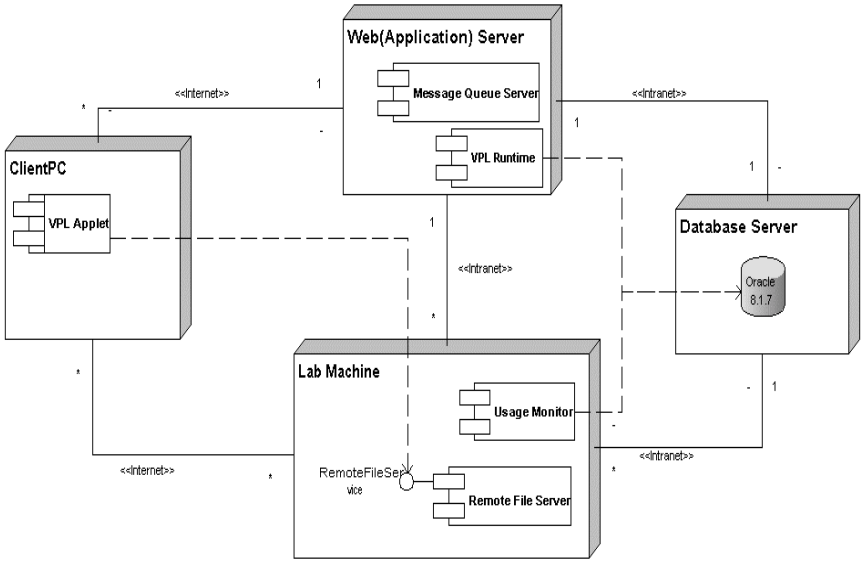

Fig. 4. Deployment diagram for the VPL system.

	Lab machine	Computer at school network	Computer outside school network.
Processors	Pentium III 500 MHz	Pentium III 500 MHz	Pentium III 600 MHz
Memory	256 MB PC-100	256 MB PC-100	512 MB PC-100
Network connection speed to Lab machine	Not Applicable	100 Mb/s LAN	1.5 Mb/s broadband or 56 kb/s dialup connection
OS	Mandrake Linux 8.0	Window 2000 professional edition	Window 2000 professional edition
Screen Resolution	Not Applicable	1024 * 768	1024 * 768
Color depth	16k bit color	16k bit color	16k bit color

Fig. 5. Confguration of experiment machine.

2. Qcheck version 2.1 is used to measure latency when using a dialup connection.
3. Xosview is used to monitor the system resource usage in a Linux system such as CPU time, physical memory usage and swap memory usage.

The experiment was conducted at three different connection speeds. The throughput and response time were measured using Qcheck version 2.1. The results are summarized below.

	Measure by Network Smart Lite and Qcheck	
	Throughput	**Latency**
School LAN	More than 10 Mbit/s	Less than 10 ms
Broadband LAN	Around 350 Kbit/s	Around 20 ms
Dialup connection	Around 30 Kbit/s	Around 150 ms

Fig. 6. Lab. machine throughput and latency

This experiment tests the launching times of three different types of software applications operating at different connection speeds. The purpose is to demonstrate the performance of the remote application as it launches in different network environments. The launching time is defined as that period between the moment of double clicking on an application icon and the moment when the application is ready to accept input. The launching time of each application, was measured five times. The average launching times is displayed in the table below.

Launching time	**JDeveloper 9i**	**Gedit**	**Xterm**
Local	24 sec	1 sec	0.7 sec
School LAN	28 sec	3 sec	2.5 sec
Broadband	54 sec	35 sec	33 sec
Dialup	> 5 minutes	75 sec	52 sec

Fig. 7. Program launching times s in different environments

Three applications were launched, Jdeveloper 9i, Gedit and Xterm. Of the three, Jdeveloper 9i is the most graphic- intensive application, followed by Gedit, then Xterm. This explains the exceptionally long delay in launching the Jdeveloper program as compared with others when operating over a dialup network (see Fig 13). The launching time difference decreases, however, as the connection speed increases and the bottleneck due to bandwidth availability becomes less of an issue.

5 Conclusion

This paper focuses on the development of runtime modules for VPL. With a well-defined interface established between the subsystems, the runtime modules have been developed to provide core mechanisms for driving the operations of the VPL system. We have successfully developed core modules that support remote application launching through the use of an X-protocol. To bind the remote application to a web terminal, a telnet mechanism was developed and implemented to transparently and remotely launch Unix-based applications over the web. To support seamless access of the files repository on both local and remote machines, a file management module was developed to allow the intuitive and efficient transfer of files between the

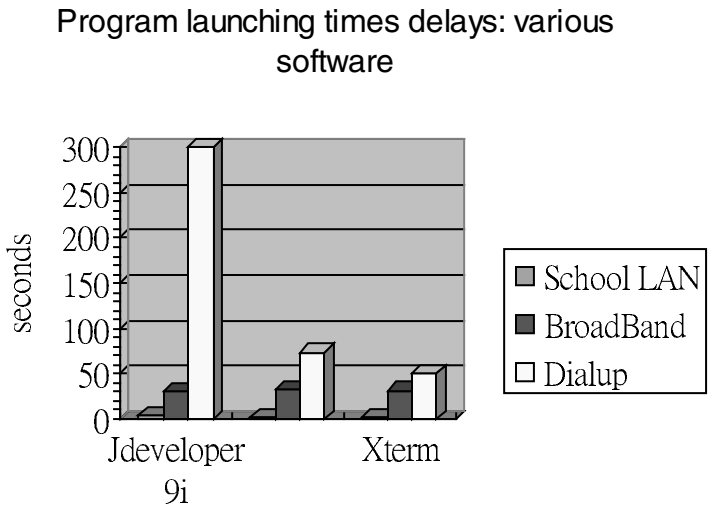

Fig. 8. Program launching time delays: various software

directories. The need to provide project-based messaging to allow student collaboration resulted in the development of the communication module. Based on the Java Messaging System, it allows a seamless exchange of messages between students and is configurable according to on topic/ project group.

While the core modules have been successfully developed, we believe there are several enhancements that can be incorporated to increase the robustness and usability of VPL. These include:

- **Firewall compatibility:** The VPL system should support users behind a firewall. This can be accomplished using the HTTP tunneling technique.
- **Intelligent load balancing:** The system implemented in this project involves only one lab machine. In order to support a larger number of users, a load balancing scheme based on lab machine CPU/RAM consumption levels and network utilization levels should be applied to the VPL system.
- **Automatic server discovery and monitoring:** the system involves several servers and is inherently complicated. Therefore, an automatic server monitoring and discovery platform is needed. A technique such as heartbeat listening could be applied to this system.
- **Peer to Peer computing:** The existing VPL system supports only message transfer between users. More intuitive communication channels such as web conferencing and white boards should be developed. More work should also be done to facilitate project collaboration such as screen and file sharing.

Acknowledgement. This project is supported by the HK Polytechnic University Teaching and Learning Grant LTG00-01\DLTC\COMP05.

References

1. Hboffstein A., Lunetta V., "The role of the laboratory in science teaching: neglected aspects of research.", Review of education research, 52, p.201–218.
2. Glenn W. Rowe, Peter Gregor, "A computer based learning system for teaching computing: implementation and evaluation", Computers & Education 33 (1999) p. 65–76, Pergamon
3. A. Di Stefano, F. Fazzino, L. Lo Bello, O. Mirabella, "Virtual lab: A Java Application for distance learning", 0-7803-4192-9/97/$10.00 © 1997 IEEE
4. Jiannong Cao, Alvin T.S. Chan, Weidong Cao, Cassidy Yeung, "Virtual Programming Lab for On-line Distance Learning", Proc. 1st International Conference on Web-based Learning (ICWL'02), Aug. 2002. Hong Kong. Lecture Notes in Computer Science (Springer-Verlag).
5. Sam K.P. Ma, Michael Rung-Tsong Lyu, "A web-based customized virtual learning environment" APWEB'99 http://www.cse.cuhk.edu.hk/~lyu/student/