Università
della
Svizzera
italiana

Facoltà
di scienze della
comunicazione

**Istituto di
Comunicazione
Istituzionale
e Formativa
ICIeF**

newMine
eLab

Nr. 6

FOSLET 2006

Proceedings of the Workshop on Free and Open Source Learning Environments and Tools Hosted by OSS 2006 (Como, June 10, 2006)

Edited by Luca Botturi, Riccardo Mazza and Stefano Tardini

Lugano, June 2006

Table of Contents

Foreword

Web-based Learning Environments supported by Course Management Systems (also known as Learning Management Systems, LMSs) have become a valid solution for institutions, schools and universities wishing to deliver eLearning or support blended-learning activities. Learning Environments are used to distribute information and content material to students, prepare assignments and tests, engage in discussions, and manage distance classes without any time and space restrictions.
During the last few years, several institutions have moved from commercial/proprietary solutions to Free and Open Source Software (FOSS) environments. The increasing popularity that FOS solutions for eLearning are enjoying is partly due to the absence of license costs, and partly to the great adaptability and interoperability of FOSS solutions, also in relationship with the development and adoption of Learning Technology Standards, such as the Shareable Content Object Reference Model (SCORM) and the IMS Global Content Packaging specifications. These latter aspects can make it easier to integrate FOS eLearning solutions in the institutional setting and foster the adoption of innovative teaching and learning solutions.

In this context, the idea of promoting a workshop on FOSS solutions for e- and blended learning activities arose from the experience gained by the editors of the present proceedings at the eLab (eLearning Lab: www.elearninglab.org) of the University of Lugano (USI) and of the University of Applied Sciences of Italian Switzerland (SUPSI), where in 2004 an OS LMS was introduced in order to support the educational activities and to promote the use of eLearning in the teaching and learning practices of both institutions.
The introduction of e-Courses (this is the name of the platform introduced, based on the Moodle technology; see http://corsi.elearninglab.org) soon raised interesting issues concerning the installation of the platform, its integration into the different existing universities' systems, its customization according to both institutions' needs, the promotion of the new platform among faculty members and students of USI and SUPSI and its evaluation, which showed high satisfaction levels. Hence the interest for other similar experiences and the chance of promoting within the Second International Conference on Open Source Systems (OSS 2006) a workshop focusing on two perspectives: (a) interoperability, including course content migration, metadata, and standards implementation; (b) adaptability and integration, including LMS fine-tuning to specific organizational requirements, integration into the teaching activity, costs and management.

The accepted papers, which are published in the present proceedings in alphabetical order, show a worldwide interest in the issue and present relevant case studies:

1. The first paper, by Ernesto Damiani, Fulvio Frati and Davide Rebeccani of the University of Milan, Italy, presents an OS virtual lab designed in order to provide students with a complete training environment accessible via web directly from web browsers.
2. In the second paper, by Ivan Nagy, Jitka Homolová and Evgenia Suzdaleva of the Academy of Sciences of Prague, Czech Republic, a system used to present mathematical texts in a close connection with software realization of the results is presented, which was used to teach the basis of Bayesian statistics.
3. The third paper, by Takayuki Sekiya and Kazunori Yamaguchi, shows a case of integration of different information systems at the University of Tokyo, Japan, centered around an OS LMS called CFIVE.
4. In the fourth paper, by Rahul Upakare and M. Sasikumar of the Center for Development of Advanced Computing of Mumbai, India, another case of integration is presented, namely the transformation of a standalone automated program grading system called Parikshak into a Web services framework using an OS toolkit.
5. The last paper, by Martin Weller of the Open University, UK, provides an approach to Virtual Learning Environments (VLEs) from the point of view of their adopters, showing that OS VLEs represent – with respect to commercial VLEs – a compromise that can meet the needs of both lead and conventional users.

Finally, we would like to thank the OSS 2006 promoters for giving us the chance of organizing the FOSLET workshop, and to invite you to FOSLET 2007.

Luca Botturi
Riccardo Mazza
Stefano Tardini

The Open Source Virtual Lab: a Case Study

Ernesto Damiani
Department of Information Technology
University of Milan
Italy
damiani@dti.unimi.it

Fulvio Frati
Department of Information Technology
University of Milan
Italy
frati@dti.unimi.it

Davide Rebeccani
Department of Information Technology
University of Milan
Italy
rebeccani@dti.unimi.it

ABSTRACT

E-Learning is becoming a standard issue for Information Technologies degree courses. Video lessons, on-line exercises, and didactic forums, with interactions with tutors and teachers are already provided by current on-line degree courses. Within this scenario, there is a lack of virtual environments to allow students to make real experiences on network programming and configuration. In order to plug this gap, within the on-line degree course of "Security of Informatics Systems and Networks" provided by University of Milan, our group designed and realized the Virtual Lab, to offer to students a complete training environment accessible via web directly from normal web browser. This paper describes our fully open source solution and supplies a road map for other future works on virtual laboratories and online teaching.

Keywords

e-Learning, On-Line degree courses, Open Source, Virtual Laboratories, Xen

1. INTRODUCTION

Providing hands-on experience by using computer and making exercises on network configuration and security-related issues are essential for current Information Technologies (IT) education. In fact, IT degrees rely on laboratory activities, especially for courses that aim to provide an experience on network programming. Many universities, therefore, developed security laboratories, from which realistic experience of security technologies, network programming and tools become available for students. Thanks to Internet technologies, it is now possible to offer courses and laboratories on the web that satisfy all functionalities of conventional laboratories. However, some issues remain to be solved. For instance, conventional laboratories usually are implemented on an isolated network which allows students to exercise on network programming, firewall design, etc. Although this isolation, that prevents students from performing some dangerous operations, there are several problems, like system failures, that can be recovered manually only if they take place in a conventional laboratory. The computational resource requested and the fact that virtual machines must be used with administrator privileges, make virtual laboratories very difficult to develop. Nevertheless, they are of paramount importance in case of remote teaching and e-Learning experience, especially in Information Technologies related courses. This idea, coupled with the e-Learning Strategies of University of Milan[1] and the Italian legislation (ministerial decree 17.4.2003, `http://www.mininnovazione.it/ita/normativa/allegati/Decreto17_04_03.pdf`) that defines the possibility of online university degrees, constitutes the bases of University of Milan's CdL (Corso di Laurea, degree course) online project. This online course, focused on security related issues, provides an e-Learning experience with a full support of a virtual laboratory. This kind of e-Learning experience is not only an online distribution of course related material: CdL online is not a simple "online version of a normal university course" but an "online focused course" with a totally re-minded and redesigned teaching context. The e-Learning model used is a "blended" model based on the presence of different teaching situations: traditional lessons, online video-lessons, forum activities, online exercises and laboratory exercises. In this context, we develop the Virtual Lab project basing only on open source technologies. This fully open source system has been developed using Xen platform on top of Gentoo Linux distribution. In this manner we overcome all problems related to virtual networking laboratory and networking programming on virtual network using powerful Linux scripts. The purpose of this paper is to present our fully open source virtual laboratory for e-Learning on University course of IT, and to propose an ideal road map for other future works on virtual laboratories and online teaching. This paper is organized as follows. Section 2 presents related works about existing implementations of remote laboratories and available virtualization techniques. Section 3 globally describes the entities of our environment. Section 4 shows the Virtual Lab system framework and the implemented technologies. Section 5 describes in details how we configured the Virtual Lab environment. Section 6 lists some expected enhanced of

[1]This strategies is well delineated in international congresses: "e-Learning: una sfida per l'università. Strategie, metodi, modelli" (11-13 November 2002) and the following meetings "e-Learning 2003 ANEE" (30 June 2003) and "Qualità nella gestione di progetti di e-Learning" (June 2004)

our project, and, finally, Section 7 presents our conclusions.

2. BACKGROUND

2.1 Related Work

Remote laboratories represent the translation of in-situ laboratory experiments to distance learning (or e-Learning), offering remote access to real laboratory equipments and real instruments. New way for e-Learning is a *virtual laboratory*, where a simulation system commonly replaces the real system. Virtual laboratories typically were born for simulation software such as *Matlab+Simulink* in case of [6] or *LabView* for [15] or web-based training system for Information Technology Security named Telelab [10, 11], that provides to students virtual machines related on a particular security exercise. Yet, one has to take care that such software can be also used for real system control. One can find remote (or virtual) laboratory experiments in various scientific and technical topics such as automatic control in [7], electronics, chemicals and mechanicals in [8] and robotics in [5, 14].

2.2 Related Techniques for Virtualization

A Virtual Machine (VM) is a duplicate of a real machine running in an simulated environment. This simulated environment is exposed by the virtual machine monitor, a software component that presents a layer functionally identical to the underlying hardware, manages the VMs instances and controls system resources access. Thanks to this hardware replication the applications running on a VM are not aware of running in a simulated environment. Applications have thus at their disposal an environment populated with virtual devices such as virtual disks, emulated processors, virtual network adapters, and so on. This approach ensuresa VM isolation.

The Virtual Machine Monitor (VMM) has the complete control over the resource pool allocated to the VMs; is impossible for an application running in a simulated environment to use resources not explicitly assigned to its container.

There are mainly two approaches to virtualization: *full virtualization* and *paravirtualization*. The full virtualization strategy aims to create a virtual execution environment for running full unmodified operating system images, replicating the original guest operating system platform behavior and facilities on the host system. The main drawback of this approach regards the intrinsic slowness of the implementation of this type of solution, and the need to mimic a real complete system behavior and to enforce resource isolation lead to the realization of a complex "resource-hungry" software component. Full virtualization is featured in products such as VMWare [19], Bochs [12], and QEMU [3]. Paravirtualization approach addresses the performance problem demanding that the guest operating system must be modified to run in the virtual environment. The need of applying changes or patches to the kernel requires obviously the operating system source to be available. At times, the two strategies have been mixed in some emulators to take the best of each approach.

3. SYSTEM DESCRIPTION

Fundamental requirement of e-Learning platforms is to make contents accessible on the web and give to students a privileged communication channel with didactic tutors and teachers [18]. To achieve this goal, e-Learning platforms share a common three-tiers structure composed by *a user interface*, represented by user web browser, connected to a *web server*, that provides services based on *data repositories* and connects users to *resources*. CdL online e-Learning environment follows this architectural paradigm and its structure is depicted in Figure 1.

The system is composed by three major entities:

1. *the e-Learning platform*, that allows the access to courses contents and to the Virtual Lab, and the interactions with tutors and teachers;
2. *the Virtual Server*, that implements an informatics laboratory which provides students with a real working environment; it consists of a VMs pool, a virtualization server and a firewall. The pool contains VMs which are assigned to students.
3. *the student web browser*, that allows to connect and interact with the platform and runs applet to connect, by ssh connections, to the assigned VM.

Each user is identified by username and password whereby access to the platform by means of the web browser. The services supplied by the platform are fourfold:

1. *Communication services*, where students can communicate with their coursemates and didactic tutors with dedicated forums;
2. *Community services*, where students can find profiles of teachers, tutors and students themselves;
3. *Teaching services*, that give access to contents (videolessons and exercises) related to courses students are enrolled on;
4. *Calendar services*, to remind students deadlines and engagements.

Within the teaching services, we focus on the implementation of the Virtual Lab, where each student can interact with an own Linux VM with all administrator privileges. They could access to it following specific links in the platform; these links will establish a connection with the Virtual Lab server and open a ssh shell on the student's VM.

As described in Figure 1, the e-Learning platform is connected to the Virtual Lab server through a dedicated firewall that filters and manages all incoming connections, ensuring virtual server security and isolating the VMs pool from external environment. In fact, each VM operates in a sealed environment: students can modify the overall Linux system configuration and can interact with all VMs that are active at the same time; communication between students and their own VM is allowed only by a ssh connection through predefined ports. Teaching activities are shaped basing on Virtual Lab: exercises consist of making simple application about network programming, as for instance Socket and RPC (Remote Procedure Calls) libraries, and discussions in forums stimulate students to try what they learned during lessons, as for instance configuring system firewall, routing

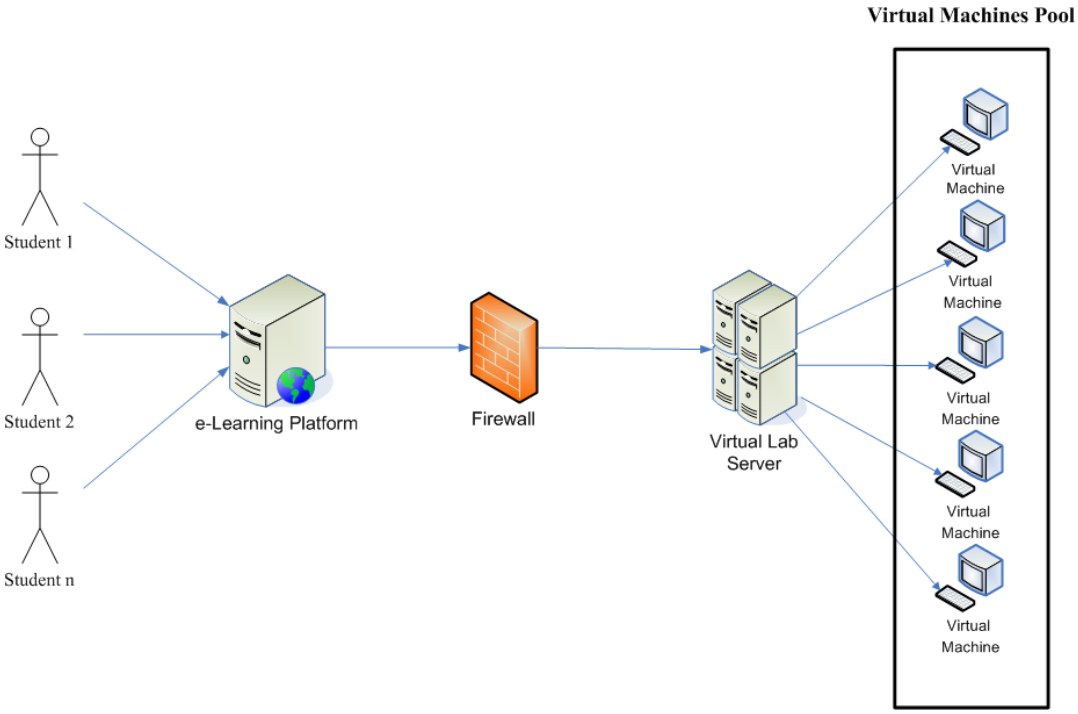

Figure 1: System structure

tables, and network interfaces. In this way, students, in addiction to learning theoretic notions, are faced with real world systems and can try to resolve real world problems.

4. THE VIRTUAL LAB FRAMEWORK

The system framework of Virtual Lab can be examined focusing on three aspects:

- the Hardware of the Virtual Server and the Firewall;
- the Virtualization software;
- the Virtual Machines.

Each of these aspects is explained in details in the following sections.

4.1 Hardware

In order to implement a reliable and scalable environment, the requirements that drive our choices about hardware were essentially two: a spacious storage unit, that allows us to give to students a complete development environment, suitable also for different courses that could need additional software, and a big RAM memory, indispensable to allow the system to manage as much as VMs at the same time.
Following these requirements, we choice to implement the virtual server with these characteristics: a Fujitsu-Siemens Primergy RX-300 S2 with 2 Intel Xeon EM64T CPUs at 3.20Ghz, 8 Gb RAM memory and four 300 Gb SCSI U320 hard disks in RAID 5; the server is connected to our internal net with a Broadcom Corporation NetXtreme BCM5721 Gigabit Ethernet PCI network interface. The firewall is implemented on a separate machine to increment system security from external attacks and to preserve virtual server performance. The firewall has the following features: a Fujitsu-Siemens Primergy RX-100 S2 with an Intel Pentium 4 CPU at 3.00 Ghz, 1Gb RAM memory and two 80Gb SATA hard disks. Further, it is connected to the external net with an Intel Corporation 82541 GI/PI Gigabit Ethernet network interface.
The maximum number of VMs running simultaneously on our system is currently about 90, and the storage occupation is currently of about 300 Gb, that allows us to fully manage all students of current and future academic years.

4.2 The Virtualization Software: Xen

For our system, we choose Xen as virtualization framework. Xen is a virtual machine monitor created by the University of Cambridge [2, 21] and released under the GNU GPL license that embraces the paravirtualization approach. Xen supports x86/32 and x86/64 platform, and, using the hardware CPU virtualization provided by Intel VT and AMD SVM technologies, has the ability to run an unmodified guest operating system kernel. As said previously, Xen requires the guest operating system kernel to be ported to the environment x86-xeno architecture. Its paravirtualized virtual machine monitor or hypervisor permits to achieve performance close to native hardware. More in detail, a Xen system is composed by multiple layers as shown in Figure 2. Each isolated virtual execution environment is called *domain*. Xen hypervisor [4] manages the scheduling operation related to the execution of each domain, while each guest operating system manages the VM application scheduling. During system boot, a domain with special privileges, *Domain 0*, is automatically created. Domain 0 can initialize other domains ($DomU$) and manages their virtual devices. The majority of all management and administration tasks

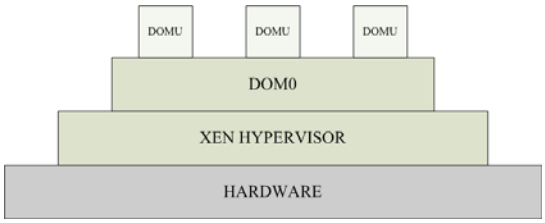

Figure 2: Xen system layers.

are carried out through this special domain. Xend, a process that runs in domain 0, manages VMs and provides access to their console. Common Xen usage scenarios include kernel development, operating system and network configuration tests, server consolidation and server resources allocation.

4.3 Virtual Machines

The implementation of virtual machines required a further analysis based on some additional considerations.
First, a VM has to be an efficient, isolated duplicate of a real machine [13]. Every VM works in a sealed environment, to isolate its disk and memory address space and to protect system integrity from possible VM failures. Second, we had to supply to students a complete and up to date operating system, in order to give them all the instruments to develop simple programs and get experiences on system configurations. Further, some hardware constraints, in particular the 64 bit implementation of the server, and the support of Xen, restrict the set of possible acceptable operating systems. Following the points depicted above and basing to our previous experiences, we chose to implement our VMs based on the Linux distribution Gentoo, that is described in Section 4.3.1.

4.3.1 Gentoo

Gentoo is a Linux distribution [9, 17] with several unique characteristics that fit our requests about VMs. First of all, the major feature of Gentoo distribution is its high adaptability, thanks to a technology called *Portage*. Portage performs many key functions: *i) software distribution system*, that permits to developers to install and compile only the needed packages that can be added at all the time without reinstalling the entire system, *ii) package building and installation*, Portage automatically builds a custom version of the package optimizing it for the underlying hardware, and *iii) automatic updating* of the entire system. Second, Gentoo is fully open source and distributed under the GNU General Public License. Finally, Gentoo supports our 64 bit architecture and implements the Xen environment.

5. VIRTUAL LAB IMPLEMENTATION

The implementation of Virtual Lab follows three major steps: the *Network configuration*, the *Firewall configuration*, and the *Platform connection*. In the following sections, these steps are explained without enter into details. A more specific description of configuration technical details is out of the scope of this paper.

5.1 Network Configuration

The initial stage of VMs network configuration is the creation of VM images. In fact, each VM is composed of a root image, that represents the available disk space and contains the operating system and the installed packages, and a swap image, used to manage memory swaps. We create an initial 2 Gb image over which we install a basic version of Gentoo. This image contains all essential services and, in particular, a complete *gcc* compiler and tools such as iptables, text editors, Perl and Python. The swap image is created by the `mkswap` command and has a size of 256 Mb. At this point, we have to associate each student with the relative root and swap images. Students root and swap images are automatically created by a shell script, that reads students id and copies the original images naming them by the concatenation of the string “root” and “swap” with the student id plus 10000.

```
kernel = "/home/kernels/vmlinuz-2.6"
memory = 512
name = "vm-10058"
vif = [ ’’ ]
ip = "10.0.0.1"
netmask = "255.255.255.0"
gateway = "10.0.0.254"
disk = [’file:/vm/root-10001,hda1,w’,
        ’file:/vm/swap-10001,hda2,w’]
root = "/dev/hda1 ro"
```

Figure 3: Xen virtual machine configuration file example.

The next stage is to create for each VM the relative Xen configuration file, that contains all the settings specific to this VM. Figure 3 shows an example of Xen configuration file created by a shell script for the student id “1”. Except for small differences, all the VMs share the same settings [20]:

- *kernel*: indicate which kernel load at boot time;
- *memory*: indicate the amount of server memory to assign to the VM. This value is tuned basing on VM purposes and installed packages;
- *name*: indicate the name to assign to the VM. Since it has to be unique within the Xen environment and it is used to refer to the specific VM, it is created concatenating the string “vm-” with the student id plus 10000.
- *vif*: allow to specify additional options for the virtual network interface, as for instance the VM MAC address;
- *ip*: set a static IP address to the VM. IP addresses are assigned sequentially and all students of the same year share the same subnet. For the next year, IP addresses will be in the form of 10.0.1.x;
- *netmask*: set the netmask, allowing students to communicate only with VMs owned by students of the same year. In particular, they can communicate only with machines that share the same subnet;
- *gateway*: set the VM common gateway. The gateway is responsible for the management of the packets through

```
$IPTABLES -t nat -A PREROUTING -p tcp -i
  $EXTERNAL_INTERFACE -d $VM_PUBLIC_IP
  --dport 10001 -j DNAT
  --to-destination 10.0.0.1:22

$IPTABLES -A FORWARD -p tcp
  -i $EXTERNAL_INTERFACE
  -o $INTERNAL_INTERFACE
  -d 10.0.0.1 --dport 22 -j ACCEPT
```

Figure 4: Example of firewall rules.

the subnet. The gateway IP is the same than the Xen-bridge (see Figure 5), a virtual network interface that supervises all communications between VMs and the external net;

- *disk*: indicate where root and swap images are located into the host system and in which logical partition, of the guest system, mount them (in the example *hda1* and *hda2*);
- *root*: indicate which partition, specified in the previous disk field, has to be considered as the root partition.

At this point, the server is ready to work as a virtualization server and VMs can be started, accessed locally via console or ssh, and destroyed. Our goal is a little more complicated, since we have to give to the student the access to her VM via ssh through the e-Learning platform, and, at the same time, isolated virtual server from the external net, except for the ssh channel. These features has been achieved through particular a firewall configuration.

5.2 Firewall Configuration

In the Virtual Lab project, the firewall has three main goals: protects the server from external attacks, isolates VMs from the external net, and forwards connections from student computer, through the e-Learning platform, to the relative VM via ssh. Note that each VM has a local IP address and can not be accessed from the external without some specific firewall configurations. Figure 5 shows the communication flow between the platform, with its public IP address, accessed by a student, to the virtual server through the firewall. In particular, the figure explains how the student, whose id is equal to 1, through the platform, accesses to the firewall; the port number used to connect identifies univocally the VM owned by the student. Based on this port number, firewall rules forward the incoming connection to the right local IP, identifying the VM, to the well-known ssh port 22. Looking at the example in Figure 5, the incoming communication on port 10001 is forwarded to local IP 10.0.0.1 on port 22, hence to VM 1. Figure 4 supplies an example of firewall rules, created by a shell script, that forward connections from port 10001 to the local IP 10.0.0.1 on the port 22 and vice versa. The first statement in the example literally says to add to the *NAT table* of the firewall a rule to intercept all incoming packets from the external interface, on the TCP protocol, on the 10001 port and to forward them to the local IP 10.0.0.1, the VM running on the virtual server, on the port 22. The second statement add a rule to the *FORWARD chain*; this rule accepts all incoming connections from the external interface to the internal interface that are directed to the local IP 10.0.0.1 on the port 22 [16].

5.3 Platform Connection

The final step of Virtual Lab configuration resides in defining the communication protocol between the e-Learning platform and the virtual server. As said before, the interaction between students and VMs happens via ssh protocol. To increase server performance, VMs are not loaded at boot time; they are started only when requested by the platform. Every time a student activates her VM, the platform calls the script in Figure 7 to start a specific VM; in the same way, when she closes the connection, the platform calls again the script to shutdown the VM. The script we used accepts two parameters: the first one can be `start` or `stop`, respectively, to load and shutdown the VM, and the second contains the id of the student who asks for the operation. Looking to the script in details, in case that the first parameters is equal to "start", the system calls the Xen command `xm` to load the VM associated to the configuration file and relative to the student specified by the second parameter. If the second parameter is "stop", the system closes the VM and looks for unmounted loop devices, detaching, with the Linux command `losetup`, the file associated with the specified loop device. Once the platform has started the corrected VM, the direct connection with the student can be finalized. The platform then opens an applet that supplies the ssh shell; the applet we choose is MindTerm [1], a pure Java client that implements the SSH1 and SSH2 protocols and is released free for non-commercial use. To summarize the communication flow between all actors (Student, e-Learning Platform, Firewall, Virtual Server, Virtual Machine), Figure 6 shows the UML sequence diagram of a communication session, starting from the user platform login and including the VM activation and concluding with the VM closing.

6. FUTURE WORKS

There are essentially two categories of new features that will be implemented in future releases of the Virtual Lab. The first category comprises enhancement in the management of VMs: we plan to create scripts for the nightly back-up of VMs home directory and for software updates or new packages installation. The second category will include the need to create new and more complex network topologies, where each student will own a private subnet with, for example, three VMs that will act as client computer, web server, firewall, to test applications for a real net environment.

7. CONCLUSIONS

In this paper we presented our e-Learning experience within the more comprehensive project of on-line degree course of "Security of Informatics Systems and Networks" of University of Milan. Virtual Lab manages a pool of 120 users and, actually, registered an average of 30 contemporaneous accesses. We implemented a Virtual Lab that supplies to each student a own VM accessible every time by a ssh connection, instead of different approaches, as for instance [10][11], that do not provide personal VMs. We described step by step how we realized all the framework to give to interested organizations a road map to build a cost effective, fully open source solution that allows them to provide to students a complete and available working environment.

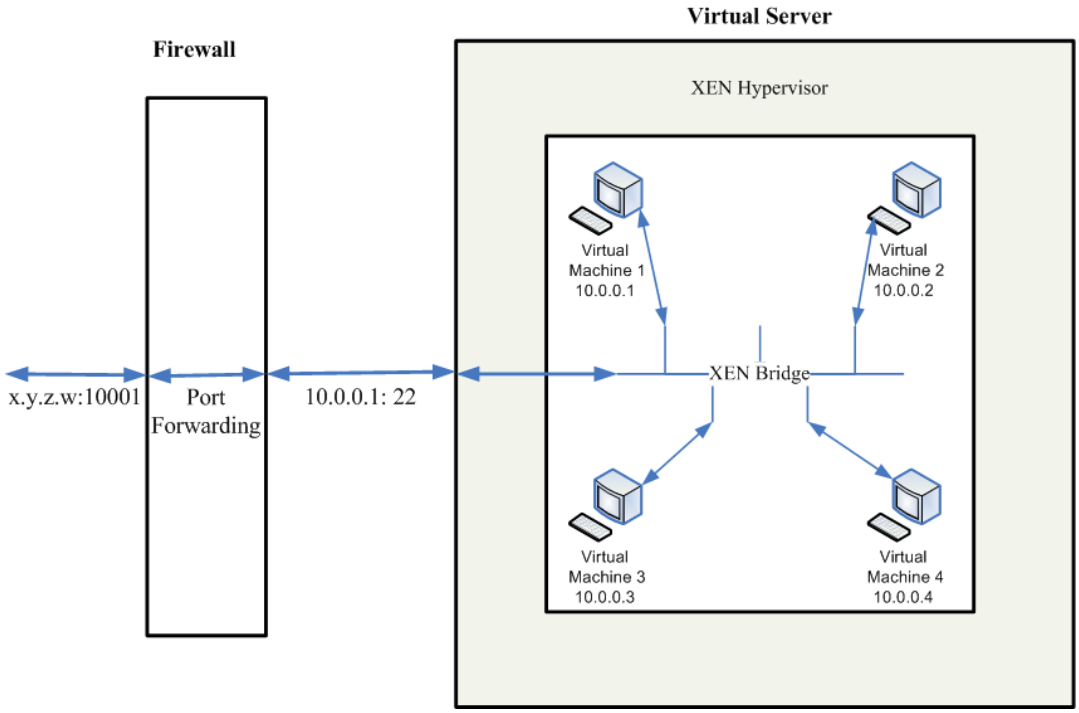

Figure 5: Communications between virtual machines and the external net through Xen-bridge.

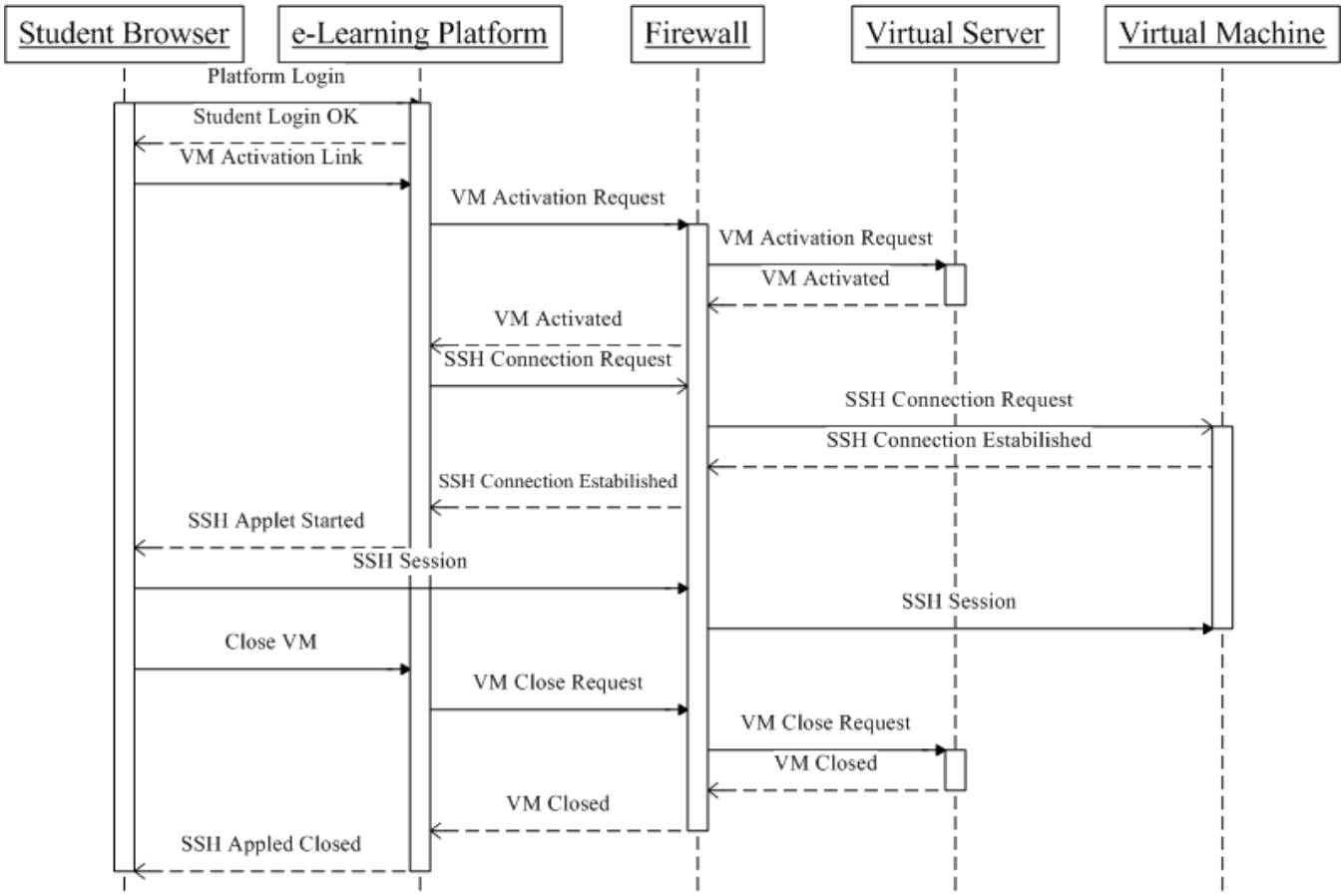

Figure 6: Virtual Lab communication UML sequence diagram.

```
#!/bin/bash
XM="/usr/sbin/xm"
MACHINES_CONFIG="/vm"
CONFIG_PREFIX="config-"

case "$1" in
'start')
        sudo $XM create
          $MACHINES_CONFIG/$CONFIG_PREFIX$2
;;
'stop')
        sudo $XM shutdown
          `sudo xm list|grep $2|awk {'print $2'}`
        sleep 10
        if [ -z `sudo losetup -a |
            grep root-$2 |
            cut -d ":" -f 1` ];
        then
                exit;
        else
                sudo losetup -d
                  `sudo losetup -a |
                    grep root-$2|
                    cut -d ":" -f 1`
                sudo losetup -d
                  `sudo losetup -a |
                    grep swap-$2|
                    cut -d ":" -f 1`
        fi
;;
esac
exit 0
```

Figure 7: Shell script to load and shutdown virtual machines.

Acknowledgments

This work was supported in part by the European Union within the PRIME Project in the FP6/IST Programme under contract IST-2002-507591 and by the Italian MIUR within the KIWI and MAPS projects.

8. ADDITIONAL AUTHORS

Additional authors:
Marco Anisetti (Department of Information Technology, University of Milan, email: anisetti@dti.unimi.it),
Valerio Bellandi (Department of Information Technology, University of Milan, email: bellandi@dti.unimi.it), and
Umberto Raimondi (Department of Information Technology, University of Milan, email: uraimondi@crema.unimi.it).

9. REFERENCES

[1] AppGate Network Security - MindTerm, http://www.appgate.com/products/80_MindTerm/, April 2006.

[2] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and Andrew Warfield: Xen and the Art of Virtualization, In *Proceedings of the nineteenth ACM symposium on Operating systems principles, Operating Systems Review*, volume 37, 5, pages 164–177, New York, 2003.

[3] F. Bellard: QEMU CPU emulator., http://fabrice.bellard.free.fr/qemu/.

[4] M. Ben-Yehuda: The Xen hypervisor: virtualizing a machine near you, http://www.mulix.org/lectures/xen-haifux-apr-2005/xen-haifux.pdf, IBM Haifa Research Labs, 2006.

[5] A. Bicchi, A. Coppelli, F. Quarto, L. Rizzo, F. Turchi, and A. Balestrino: Breaking the Labs Walls Tele-Laboratories at the University of Pisa, In *Proc. of IEEE International Conference on Robotics and Automation*, May 21-26, 2001, Seoul, Korea.

[6] C. Bonivento, L. Gentili, L. Marconi, and L. Rappini: A Web-Based Laboratory For Control Engineering Education, In *Proc. of 2nd International workshop on tele-education using Virtual Laboratories*, August 8-9, 2002, Sherbrooke, Canada.

[7] C. Chiliculita and L. Frangu: A Web Based Remote Control Laboratory, In *6th Multiconference on Systemic, Cybernetics and Informatics*,July 14-18 2002, Orlando, Florida, USA.

[8] S.K. Esche: Remote Experimentation One building Block in Online Engineering Education, In *Proc. of the 2002 ASEE/SEFI/TUB International Colloquium on Global Changes in Engineering Education*, October 1-4, 2002, Berlin, Germany.

[9] Gentoo Linux, http://www.gentoo.org/, 2006

[10] J. Hu, M. Schmitt, and C. Meinel: Virtual Machine Management for Tele-Lab "IT-Security" Server, In *Proc. of 10th IEEE Symposium on Computers and Communications (ISCC 2005)*, 2005.

[11] J. Hu, C. Meinel, and M. Schmitt II: Tele-Lab IT Security: An Architecture for Interactive Lessons for Security Education In *SIGCSE04*, March 37, 2004, Norfolk, Virginia, USA.

[12] IA-32 Emulator Project, Available from http://bochs.sourceforge.org/,2001

[13] G. J. Popek and R. P. Goldberg: Formal requirements for virtualizable third generation architectures, *Communications of the ACM*, 17(7), 1974.

[14] R. Safaric, D.W. Calkin, R.M. Parkin, and C.A. Czarnecki: Virtual environment for telerobotic, *Integrated Computer-Aided Engineerings*, Apr. 2001, vol. 8, no. 2, pp. 95-104.

[15] C. Salzamann, H.A. Latchman, D. Gillet, and O.D. Crisalle: Requirements for real-time laboratory experimentation over the Internet In *Proc. of 1998 International Conference on Engineering Education*, 1998, Rio de Janeiro, Brazil.

[16] The NetFilter/IpTables Project, http://www.netfilter.org/projects/iptables/index.html, 2006.

[17] G. K. Thiruvathukal: Gentoo Linux: The next generation of Linux, *j-COMPUT-SCI-ENG*, 6(5):66–74, Sept./Oct. 2004.

[18] G. Totkov: Virtual Learning Environments: Towards New Generation, In *Proceedings of International Conference on Computer System and Technologies (CompSysTech'2003)*, Sofia, Bulgaria, 2003.

[19] VMware Inc. VMware GSX Server, Available from `http://www.vmware.com/`, 2002.

[20] Xen User Manual - v3.0, `http://www.cl.cam.ac.uk/Research/SRG/netos/xen/readmes/user.pdf`, 2006.

[21] Xen Virtual Machine Monitor - University of Cambridge, `http://www.cl.cam.ac.uk/Research/SRG/netos/xen/`

Teaching System for Advanced Statistics

Ivan Nagy
Faculty of Transportation Sciences CTU
110 00 Prague 1
Prague, Czech Republic
nagy@utia.cz

Jitka Homolová
Faculty of Transportation Sciences CTU
110 00 Prague 1
Prague, Czech Republic
jkratoch@utia.cz

Evgenia Suzdaleva
Institute of Information Theory and Automation
182 08 Prague 8
Prague, Czech Republic
suzdalev@utia.cas.cz

ABSTRACT

A teaching system for presentation mathematical texts in a close connection with software realization of the results is presented. For creating texts, the "pdf-version" of mathematical text editor LaTeX is used. For software application, the interactive programming language GNU Octave is exploited. Both the texts and programs are free, the subroutines used are open. The system is used for teaching the basis of Bayesian Statistics.

Categories and Subject Descriptors

H.7 [**Good practices of integration of FOS solutions into real institutional settings**]: Miscellaneous

General Terms

Teaching or self-teaching based on hypertexts with application of free and open software.

Keywords

Statistics, textbook, free hyperref text, free software subroutines

1. INTRODUCTION

Free learning systems based on the Internet are very popular, nowadays. This system of teaching, either aimed at self-study or to be combined with some regular school study, brings about many advantages. The most evident merits are: (i) the students who missed some lecture or did not understand fully its content have a background where it is possible to fill in the information or to exercise topics that need practical skill; (ii) the students, who prefer to stay at home and to study alone or those who do not study and just want to learn something from the subject, have all they need for their self-study; (iii) the teachers, whose range of lessons is currently cut, become time independent and can rely on the database of information stored on web and so within the reach of everybody.

The situation is complicated if the subject to be taught is of a mathematical nature. Up to now, mathematical texts on web pages are restricted. In our system, a systematic use of "pdf-files" with "hyperrefs" created in the mathematical editor LaTeX proved to be a satisfactory solution.

Another trouble occurs, when the taught subject needs programming and running results on a computer. In this respect, there are two main problems: (i) How to combine web and running a program? (ii) How to describe programs implementing mathematical formulas?

The subject we deal with is Statistics. Evidently, it involves both the above problems - complex mathematical description needing hypertexts and running the results in some suitable (free and open) program.

The content of the work further described is a continuation of the work supported by the Minerva EU grant Edukalibre: Libre Software Methods for E-Education [1, 2, 3] and it is based on the theory of Bayesian Modelling Estimation and Control systematically developed in the Institute of Information Theory and Automation, Czech Academy of Sciences [4, 5]. Here we prepared the basic methodology for creating such system and here we also prepared most of the texts and necessary subroutines. This system was preliminary tested on chosen PhD students and now, it started to be used for teaching.

The teaching system mentioned is by no means a unique one. Especially in connection with Statistics, there is a lot of such projects. We can divide them into several groups. They are:

1. **Components of encyclopedias**, e.g. Wikipedia, PlanetMath, Encarta, etc. They bring information but they are not systematic.

2. **Professional programs** like Statistica, Statgraphics, Excel, Matlab. They involve relatively wide range of problems and possess high comfort for users. Nevertheless, they are not open, so you can use only offered tasks. Moreover, they are very expensive.

3. **Free programs** as for example First Bayes [6], STEPS [7], Sila [8], Statistical Java [9], MacAnova [10], StatCalc [11], Statistics101 [12], SciLab [18], ViSta64 [13]. There is a lot of these systems, the above choice is

more or less accidental. Mostly they are very useful for practical experiments. The drawback is that they are not directly connected with any teaching text and they are not so easy to extend.

4. **Syllabi of universitas.** They are usually in a standard extend, some go deeper but often at the expense to their range. Some offer the contents, only, some give at disposal also the teaching texts. In any case, they serve to the purposes of the individual schools.

5. **Books on Statistics.** There is a lot of excellent textbooks. However, they usually are separated from programs and they are rather expensive.

6. **Web applications** as can be for example found under the titles Statistical Computing Resources [14], Maths, Stats & OR Network [15] or The Math forum & Drexel [16], which are most close to our demands. They are composed of a continuous presentation of the theory, suitably filled in by applets, demonstrating the specific problems. They give nice demonstration but their modification is not easy. From our point of view, they are practically closed.

The main demands to our teaching system are: (i) connection of systematic exposition of statistical theory with a suitable program; (ii) usage of hyperrefs not only in the text but also in connection with the programs; (iii) the demand that the system is open and free. The presented system is nothing more than an attempt of a union of the demands mentioned.

2. PROBLEM FORMULATION

The goal is to construct a teaching system for training PhD students in the basis of Bayesian Statistics. As the subject is rather broad and theoretically difficult, only four most important basic tasks have been chosen for presentation. They are

- **Modeling and simulation**, where a description of dynamic system under uncertainty is introduced.
- **Estimation**, which assigns parameter values to a structurally built model so that it would reflect the investigated system.
- **Prediction**, which shows the future behavior of the system.
- **Control**, which computes optimal values of the model controlled variable so that the measured system output would minimize some given criterion.

The teaching system has to

- enable accepting high quality mathematical texts;
- be at disposal at the Internet for easy reach;
- support statistical programs that are
  (i) free, (ii) open, (iii) well described.

By "free" we mean to be at disposal for everybody without a necessity to pay for it; "open" denotes a system, where all basic programs are prepared and there is a possibility to add easily new programs based on the existing ones; "well described" guarantees that all mathematical formulas and algorithms contained in the program are clearly described by a mathematical text.

3. THEORETICAL BACKGROUND

The Bayesian Statistics is a powerfull tool for modelling, estimation, prediction and control of dynamic systems under uncertainty. Here, we sketch only the basis of the named and most important tasks.

First of all, let us introduce a **notation** used. We monitor a process and measure its variables in a discrete time $t = 1, 2, \ldots$. The time instants $0, -1, \ldots$ denote measurements performed before the beginning of monitoring, which bring a prior information about the process and serve together with measured data for identification. By y_t and u_t we denote the actual (current) values of system output and input. By star, e.g. y_t^*, ve denote all possible values of the variable, here of y_t. The couple of the actually measured data is $d_t = \{y_t, u_t\}$. A set of all measured data up to the actual time instant t (including the prior ones) is denoted $d(t) = \{d_t, d_{t-1}, \ldots\}$. *Note*: Sometimes these sets are interpreted as vectors.

3.1 Modeling

A general description of dynamic system under uncertainty is the following conditional probability density function (PDF)

$$f(y_t|u_t, \phi_{t-1}, \Theta), \tag{1}$$

where

$\phi_{t-1} = \{d_{t-1}, d_{t-2}, \ldots, d_{t-n}\}$ is a regression vector where n is a depth of memory of the model,

θ is a model parameter.

The meaning of such description is following: for given control u_t, regression vector ϕ_{t-1} and parameters Θ the model gives probabilistic description (i.e. PDF) of the output y_t. According to the nature of the random variables entering, the PDF can be either of continuous or discrete nature. Subsequently, the model is called either continuous or discrete.

3.2 Estimation

Bayesian probabilistic description of unknown parameter Θ at time t is the conditional PDF

$$f(\Theta|d(t)) \tag{2}$$

This is so called posterior PDF which includes all the information brought by measured data $d(t)$ (including the prior information) into the parameter description. The PDF $f(\Theta)$ is called prior and it is built only on the prior knowledge.

The proces of collecting information and using it for improving the description (2) in estimation is given by the Bayes rule

$$f(\Theta|d(t)) \propto f(y_t|\phi_{t-1}, \Theta) f(\Theta|d(t-1)), \tag{3}$$

where the principle of Natural Conditions of Control saying that $f(\Theta|u_t, d(t-1)) = f(\Theta|d(t-1))$ (see [4]), is used.

Note: The Bayes rule (3) says that recursive building of information into the parameters description is obtained by repetitive multiplying the parameter PDF by the model PDF. For practical computations, it is very important to choose so called reproducing form of the prior PDF which would preserve its form during the proces of estimation. If not, the computations become infeasible.

3.3 Prediction

The task of one-step-ahead prediction is to forecast the coming output value of the monitored process. Its probabilistic description is again in the form of a conditional PDF

$$f(y_t|u_t, d(t-1)), \tag{4}$$

where knowledge of the parameter Θ is not admitted.

Construction of the predictive PDF (4) using the parameter PDF (2) is following

$$f(y_t|u_t, d(t-1)) = \int_{\Theta^*} f(y_t|u_t, \phi_{t-1}, \Theta) f(\Theta|d(t-1)) d\Theta \tag{5}$$

Note: If we predict more than one step ahead, we talk about a multi-step prediction. This task is more general but also more complicated for computation. Here the output i-steps-ahead is forecasted, based on the knowledge of past data and inputs, or some their model, on the prediction interval, i.e. for time instants t, $t+1$, ..., $t+i$.

Using the predictive PDF (4), the optimal point prediction $\hat{y}_t$ can be computed. For quadratic criterion of optimality, the point prediction is given as the conditional mean

$$\hat{y}_t = E[y_t|u_t, d(t-1)] = \int_{y_t^*} y_t f(y_t|u_t, d(t-1)) dy_t. \tag{6}$$

3.4 Control

The most general description of the feedback control variable has again the form of conditional PDF

$$f(u_t|d(t-1)). \tag{7}$$

Practically, the optimal control variable u_t^* minimizing quadratic criterion $\sum_{t=1}^{N} \omega_t$ with $\omega_t = y_t^2 + q u_t^2$, where q is a penalty imposed on inputs and N is length of control horizon, is deterministic and given by the following formula

$$u_t^* = \arg\min_{u_t} E[\omega_t + \Phi_{t+1}|u_t, d(t-1)] = \tag{8}$$

$$= \arg\min_{u_t} \int_{y_t^*} (y_t^2 + q u_t^2 + \Phi_{t+1}) f(y_t|u_t, d(t-1)) dy_t,$$

where Φ_{t+1} is the partial minimum from the previous step of minimization $\Phi_t = \min_{u_t} E[\omega_t + \Phi_{t+1}|u_t, d(t-1)]$ with $\Phi_{N+1} = 0$ and the needed PDF is the predictive one (4). If the parameter Θ of the model is known, this PDF coincides with the model one (1). If Θ is unknown, the predictive PDF has to be constructed according to (5).

Note: Similarly as for the prediction, the task of optimal control on a finite horizon is more practical and also complicated. Here, the well known principle of dynamical programming must be used. However, such control synthesis is not feasible if an optimal solution is demanded for unknown model parameter Θ - so called dual control. For practical applications, some approximations have to be accepted.

4. TEACHING SYSTEM

The theory sketched in the previous section on a general PDF level can be further elaborated for the whole family of specific distributions of the model. In the teaching system considered, two distributions are considered. The first one is normal, the second one is general discrete distribution. All the theory mentioned is rather sophisticated and the computations following from it are not solvable without using a computer. From the facts mentioned above, it follows:

- a thorough and well structured theory description with hyperrefs is highly desirable,
- free and open software implementing the algorithms following from the theory is necessary,
- as close as possible connection between the theory and the programs is essential.

The teaching system should respect all these demands.

4.1 Hypertexts

As it has been mentioned, understanding of such a complex subject as Bayesian Statistics is impossible without intelligible and well organized text. Not only does the text have to be written carefully but it also has to include automatic references. They provide the reader quick approach to referenced equations or explanation of some basic notions. They also can evoke the subroutines implementing the algorithms and thus they connect the theory and the corresponding program.

There are two main types of hyperrefs. Firstly, they are ordinary references to equations or sections fully supported and used practically in all text editors supporting mathematics. Secondly, they are references to a certain database. There are approximately three types of databases used in the teaching system.

Database of notions

To understand the theory correctly, it is very important to remember well the precise meaning of all defined notions. It is obviously difficult, especially for students. To help them, all the basic notions are made as hyperrefs and after clicking they show a brief description of the corresponding notion from the prepared database. The database of notions contains also inter-references to other terms of the database and the whole database is equipped with index. It enables the students to use the database itself also as a teaching text just through surfing it and looking for meaning of notions and their connections.

Database of subroutines

The basic statistical algorithms are supported by subroutines. They can be trivial (e.g. for model simulation) but also rather complex (e.g. for control synthesis on a finite interval). In any case, they need a good and thorough description. As the subroutines implement mathematical algorithms, they need also mathematical description. Internal program comments are totaly insufficient. A good solution is to create a database of subroutine descriptions, each description connected to its subroutine by a hyperref. Naturally, the hyperrefs are written also across the database thus enabling to browse the database and to learn the links between the subroutines.

Database of tasks
The theory results into specific problems that are called tasks. The implementation of these tasks can be built from the selected subroutines. E.g. a task of prediction the output of a dynamic system can use subroutines for simulation, statistics collection, computation of parameter point estimates and prediction. Similarly as for subroutines, these tasks deserve good mathematical introduction that is done through the database of the tasks descriptions. Both the databases (for subroutines and tasks) are very important parts of the theoretical text.

An ideal solution for writing such texts is to use the free mathematical editor LaTeX (in its "pdf-variant") which it is able to generate mathematical texts of high professional quality. After using the package "hyperref" the texts are supplied not only by automatic references to equations and sections, but it is also possible to make named anchors in the text and to produce automatic references to them as well as to other "pdf-files" and anchors in them.

4.2 Software

There is a lot of statistical software at disposal with various quality in communication and presentation of results. Some of them are really excellent. But, as far the authors know, they all have the same common drawbacks: they are not open and they are rather expensive.

Openness of the software
By openness we mean a possibility of easy programming new subroutines and using them for a construction of new tasks. If this feature is missing, nomatter how broad the basis of the subroutines is, a student can always come to its border and this is crucial to him. He is restricted and cannot solve his problem that is out of range of program possibilities.

That is why, it is necessary to use some programmable language as a statistical software. In this software, a basic collection of subroutines as well as tasks must be prepared with a sufficient luxury of setting the tasks and at least average quality of both numerical and graphical outcomes. These subroutines and tasks must be supplied in a source code to (i) provide a student information about its content, (ii) be a guide how to program further subroutines or tasks.

It is clear, that programming in the chosen language must be as simple as possible not to task student's mind by other thing than Statistics. From this point of view, the interactive program language GNU Octave seems to be an ideal choice. It is not only simple and powerfull but also free of charge.

Price of the software
A crucial problem for students, studying Statistics, is that they are instructed in some statistical program which is hired to them for exercising during the seminars. A great deal of work is done by a student to master the program. Even an examination often tests the level of mastering the programm. But after finishing the course of Statistics, the program becomes inaccessible to a student because it is too expensive to buy. That is why a big stress is put on the fact, that the software used for teaching Statistics is free.

4.3 Connection between theory and software

Statistics is the subject where neither theory nor practical application can be suppressed. If only the theory is stressed, the students can memorize formulas and propositions, but they are not able to solve practical tasks. That is why the approach of many statistical courses is to teach software, first of all. Nevertheless, this approach can be even more dangerous, because students learn only clicking the mouse on the buttons of the program. They learn just to solve several standard situations in the program and very often they are even not able to interpret the result.

Good connection between theory and software is necessary from the following reasons

- the text of a subroutine can be a good help in understanding the corresponding theoretical algorithm,
- the theoretical text is necessary for a correct choice of the proper subroutine,
- the theoretical description of subroutines is unreplaceable help for their correct setting as well as interpreting the obtained results.

An idea of solution the problem is following:

At the top level there is the theoretical text, dealing with the four general problems on the level of pdfs without any specification to a particular model distribution. The problems are modeling, estimation, prediction and control. Each problem has pointers to the specific tasks.

Under the topmost level there is a level of tasks. Here the general problems are specified for a particular model distribution. Here the mathematical specification of the task as well as its program implementation are described. An important component of this file is also a reference to the program implementation of the task as well as all the related subroutines or tasks. The collection of all task descriptions creates a database of the tasks, represented by its index.

The lower supporting level is a collection of subroutines, used for a construction of the tasks. The philosophy of their description is the same as for the tasks. Each subroutine has its "pdf-file" with a description. This description is on one hand connected to the tasks using the particular subroutine on the other hand and above all it is bound to the subroutine itself. A collection of all subroutine descriptions creates a database of subroutines, again with its index.

Besides this "pdf-description", each task and subroutine has its own comments as already mentioned above.

5. EXAMPLE

Let us demonstrate the teaching system on some simple example. Consider a static discrete system with the output $y \in \{1, 2, \ldots, n\}$. Its model is parametrized by the vector parameter $\Theta = \{\Theta_1, \Theta_2, \ldots, \Theta_n\}$. The task is to perform a point estimation of items of the model parameter (in the following text, the <u>underline</u> means hyperref).

5.1 Theory

Model

In the theoretical description, the <u>model</u> (1) specifies to

$$f(y_t|\Theta) = \prod_{i=1}^{n} \Theta_i^{\delta(i,y_t)}, \tag{9}$$

where $\delta(i, y_t)$ is the Kronecker function which is equal to one for $y_t = i$ and is zero otherwise and $\Theta_i \geq 0$ for $i = 1, 2, \ldots, n$, and $\sum_{i=1}^{n} \Theta_i = 1$.

This model directly expresses probabilities that the system will be in one from his n states determined by the specific value of his output.

Estimation

The general Bayes rule (3) gets the form

$$f(\Theta|d(t)) \propto f(y_t|\Theta)f(\Theta|d(t-1)) \tag{10}$$

with the model PDF (9) and the prior PDF in the self reproducing form

$$f(\Theta|d(t-1)) = \prod_{i=1}^{n} \Theta^{V_{i;t-1}-1}, \tag{11}$$

where V_{t-1} is a vector statistics at time $t-1$. The statistics V_0 is the prior statistics expressing an expert knowledge.

Inserting (11) and (9) into (10) we obtain the rule of updating the statistics by sequentially measured data

$$V_{i;t} = V_{i;t-1} + \delta(i, y_t), \tag{12}$$

for $i = 1, 2, \ldots, n$ and on the whole time interval for which the estimation is to be performed.

Note: The meaning of the previous update is clear and it fully agrees with the statistical definition of probability. Each time a new data item comes we find to which state (level of output) it belongs and we increment the corresponding item of the statistics by one.

After updating the statistics we can compute the point estimates of the parameter items. The most widely used choice for optimal selection of these point estimates is to minimize the following quadratic criterion

$$E[(\Theta - \hat{\Theta}_t)'Q(\Theta - \hat{\Theta}_t)|d(t)],$$

where $\hat{\Theta}_t$ is the point estimate involving data up to time t and Q is some penalization matrix.

Minimization of this criterion gives the point estimate in the form of conditional expectation

$$\hat{\Theta}_t = E[\Theta|d(t)]. \tag{13}$$

Prove as a homework or see point estimates.

This result applied to our example gives

$$\hat{\Theta}_{i;t} = \frac{V_{i;t}}{\sum_{j=1}^{n} V_{j;t}}, \tag{14}$$

where V_t is the vector statistics from (11). Proof: see here.

Database of notions

Here is an example of the database of notions mentioned. Some of the notions involved in the theory example are used here for illustration.

Point estimate is a number which estimates the value of the unknown parameter. The formula for computation of an optimal point estimate depends on the choice of optimality criterion. E.g. for quadratic criterion the point estimate is a conditional expectation conditioned by all measured data.

Quadratic criterion at time t for variable X and reference value $\hat{X}_t$ is

$$E[(X - \hat{X}_t)'Q(X - \hat{X}_t)|d(t)],$$

where Q is a weighting matrix - for no weighting it is a unit matrix.

Statistics is a (vector) variable that comprises all the collected information necessary for parameters estimation. By the information we mean the prior (expert) information and that carried by the collected data.

Weighting matrix is matrix entering the quadratic criterion. The magnitude of its diagonal items can stress suppress significance of some items of the penalized variable. The nondiagonal items can even penalize products of different items of the variable - e.g. the increments of the output variable.

Database of subroutines

Here, a sample of the database of subroutines is presented. Each item of the database has two parts. Firstly, it is a descriptive part, written in a "pdf-form", describing the subroutine mainly from theoretical point of view. Secondly, it is the subroutine itself with the comments inside it, describing mainly the meaning of input and output parameters.

Subroutine simDisc

The subroutine performs simulation with a discrete static model of the form

$$f(y_t|u_t, \varphi_t\Theta) = f(y_t|\Theta) = \Theta_{y_t}.$$

This model chooses values of $y_t = i$ from a set {1,2,...,n} so that $Pr(y_t = i) = \Theta_i$.

Program implementation is following:

```
Th=cumsum(th);
```

performs cumulative summation e.g. for `th=[.3 .2 .5]` it gives `Th=[.3 .5 1]`.

Output generation is done by the command

```
y=sum(Th<rand)+1;
```

where `rand` is a generator of uniformly distributed random variable and `Th` is a cumulative sum of the discrete model parameter.

Example: Let `th=[.3 .2 .5]` then `Th=[.3 .5 1]`. Now, let `rand=.392`. Then `rand<Th=[1 0 0]`. And `y=2`, what is correct as `rand>.3`, and `rand<.3+.2` and `rand<.3+.2+.5`.

Call of the subroutine: `y=simDisc(th)`

Show the subroutine in the editor: push here

Similar subroutines: `simDiscDyna`, `simCont`, `simContDyna`

and here is the corresponding subroutine

```
function y=simDisc(th)
% y=simDisc(th)
% simulation of discrete system
%
% y  system output
%    = 1,2,...,length(th)
% th system parametr
%    = prob. of items of y
%
Th=cumsum(th);     % cummulative probs
y=sum(Th<rand)+1;  % output generation
```

Subroutine statDisUpdt

The subroutine performs updating of a discrete statistics

$$V_{i;t} = V_{i;t-1} + \delta(i, y_t)$$

for $i = 1, 2, \ldots, n$ where n is length of the statistic V.

This update follows from the Bayes rule when substituting the static discrete model and corresponding (self-reproducing prior PDF).

Call of the subroutine: `V=statDisUpdt(V,y)`

Show the subroutine in the editor: push here

Similar subroutines: `statConUpdt`

and here is the corresponding subroutine

```
function V=statDisUpdt(V,y)
% V=statDisUpdt(V,y)
% discrete statistics updata
%
% V  statistics
% y  level of measured output
%    = 1,2,...,length(V)
%
V(y)=V(y)+1;       % statistics update
```

Subroutine ptestDisc

The subroutine performs computing of parameter point estimates based on the estimated parameters PDF (posterior PDF). The computation of the point estimates for quadratic criterion of optimality is following

$$\hat{\Theta}_t = E[\Theta|d(t)] = \int_0^1 \Theta f(\Theta|d(\Theta|d(t)))d\Theta =$$

$$= \frac{B(V_{t-1} + \delta(i, y_t))}{B(V_{t-1})} = \frac{V_{y_t}}{\sum_{j=1}^{n} V_j}.$$

The following formulas have been used: beta function, multivatiate beta function.

Call of the subroutine: `th=ptestDisc(V)`

Show the subroutine in the editor: push here

Similar subroutines: `ptestCont`

and here is the corresponding subroutine

```
function th=ptestDisc(V)
% th=ptestDisc(V)
% dicrete quadratic point estimates
%
% th  point estimates
% V   statistics
%
th=V/sum(V);       % point estimates
```

Database of tasks

Similarly to the database of subroutines, each item of the database of tasks consists of the similar two parts, too. They are the descriptive part and the executive one. The descriptive part provides complete description of the task. Mainly, it uses a reference to the general theory and then describes a specialization to the specific task.

Then, some difficult programming steps are discussed and explained.

After it, a serial of experiments is recommended. Each experiment is accompanied by a comment, saying what remarkable can be seen or what changes can occur when some parameter is varied.

Conclusion summarizes the meaning of the task and points out other similar tasks.

Here is an example of the task prepared throughout this Example - point estimation of parameters of discrete model. (Again, the underlined text means hyperref.)

Task EstPtDisc

Theoretical solution

The program solves the task of point estimation of parameters of a static discrete model. The general form of a model is given in (1). Its discrete variant is specialized in (9). If as a prior PDF, the self reproducing prior (11) is chosen then the statistics update runs according to (12). The point estimates for quadratic criterion of optimality are computed using the formula (14).

Software solution

The task is constructed from three subroutines:

simDisc that performers simulation of discrete static system,

statDisUpdt that collects statistic from th incoming data and based on a specified prior knowledge,

ptestDisc that computes point estimates for the quadratic criterion of optimality.

Recommended experiments

1. Set various parameters of the simulated system.

   Those items of the parameter that have small probability of occurrence are more difficult to estimate. The reason is a lack of evidence for them.

2. Try various prior probability for estimation. You can do it by changing the coefficient `npri` in the range from 1 to 100.

   For small `npri` the prior (uniform) information is suppressed. The bigger value of `npri` starts with the uniform prior information which calms the start, but brings a slight inaccuracy into the estimation.

Conclusions

The task of prediction is very important from both theoretical and practical point of view. Theoretically it shows how to construct the predictive PDF, that is further used for all tasks of control. These tasks are very complicated and we often have to restrict ourselves to some approximation, where instead of the predictive PDF the point estimates of the model parameters are used. In applications, even the task od prediction itself plays a significant role. For example in transportation problems, it is very important to know the future evolution of transportation flows in the monitored traffic region.

and here is the corresponding task

```
clc,clear all
% Task: Parameter estimation of static discrete r
% ---------------------------------------------
% model f(y(t)|th) = th_y(t), y(t)=1,2,...,n
%
% Data for setting
ndat=2000;                    % number of data
npri=10;                      % number of prior d
th=[.1 .4 .3 .2];             % model parameters

% Data for initialization
V=ones(size(th))*npri;
The=[];

% Time loop
for i=1:ndat
  y(i)=simDisc(th);           % simulation
  V=statDisUpdt(V,y(i));      % statistics update
  The=[The; ptestDisc(V)];    % point estimates
end

% Results - prints
disp('Original parameters')
th
disp('Parameter estimates')
the=ptestDisc(V)

% Results - plots
fig,plot(The)
hold on
s=fix(2*ndat/3):ndat;
plot(s,th(1)*ones(size(s)),s,th(2)*ones(size(s))
  s,th(3)*ones(size(s)),s,th(4)*ones(size(s)))
hold off
st1='Evolution of parameters estimates ';
st2='(stight lines at the end are tru values)'
title([st1,st2])
xlabel('Time')
ylabel('Estimates')
axis([1,ndat,min(th)-.05,max(th)+.05])
```

Illustration of results

Results of the tasks can be both numerical and graphical. Here we demonstrate a graphical one which illustrates evolution of point estimates of the parameters of the investigated static discrete model. In the two figures the influence of different prior information is demonstrated.

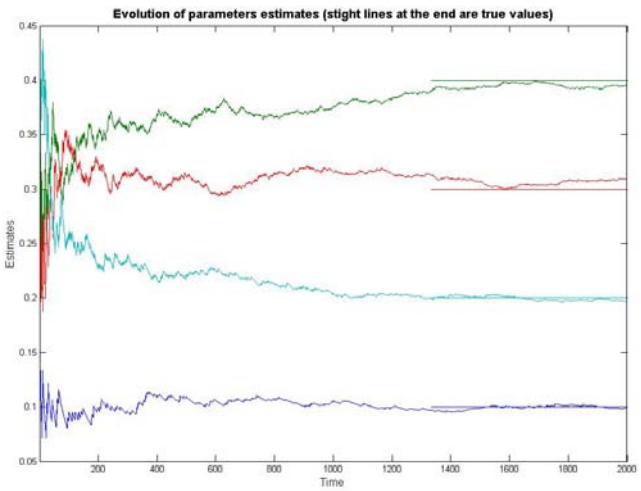

Figure 1: Evolution of point estimates with week prior

In this figure, a sequential refinement of the values of point estimated during the parameter estimation of a static discrete model with four output levels is shown. The prior information given the estimation at the very beginning of the process is uniform distribution (all parameters have the same values) and this information is rather week (as it is gained just from four measurements). That is why the beginning of estimation is rather chaotic.

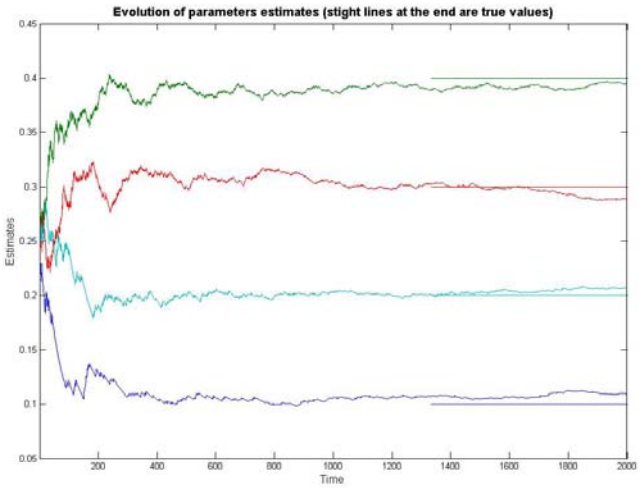

Figure 2: Evolution of point estimates with stronger prior

Here, similar situation is presented. The prior distribution is uniform again, but its weight is as if it would be extracted from 40 data items. Thus the information gained from the initial really measured data is compared with the relatively strong prior information and thus, the jumps given by the randomness of the initial data are reduced.

6. CONCLUSIONS

The teaching system described grows from our demands as lecturers of STATISTICS. The topic presented concerns rather complicated mathematical area of BAYESIAN STATISTICS focused to PhD students. As the final target of the course is a practical application, the subject is closely bound with programming. This is what makes the presented subject characteristic.

This year, we started to use the system for teaching on our Technical university, Prague, in the Czech Republic. Even this one-year experience with the system has shown some of its imperfections. The most serious of them is that we must pay much higher attention to examples from the practice throughout the whole explanation. E.g. we should speak not generally about a measured variable but specifically for instance about an occupancy of the traffic flow measured on a detector in a specific point of communication or not about a model needed for prediction but about a model that could predict the traffic flow intensity.

From the fact, that we are the teachers, it follows

- we have permanent audience (each year about 20 persons) that will use the system;
- we will have a stable feedback from this people indicating what is wrong or at least not so easy to understand;
- we will have to work continuously on improving the system as, naturally, each teacher wants to teach as best as possible.

Only years of practice with using the system can acknowledge our hope in its usefulness and to focus it directly on the problems our PhD students are going to meet.

Nevertheless, the significance of the mentioned system is not only for teaching STATISTICS. It shows new methodology of constructing such a system by those who are not programmers by their profession. It is easily applicable for any other subject with similar characteristics.

7. REFERENCES

[1] *Edukalibre: Libre Software Methods for E-Education*, SOCRATES Programme, Minerva - Promotio of Open and Diastance Learning - Information and Communication Technologies in the Field of Education., 110330-CP-1-2003-1-ES-MINERVA-M.

[2] E. Suzdaleva, I. Nagy, L. Pavelková, and Jitka Homolová, "Edukalibre. guide", Tech. Rep. 2145, ÚTIA AV ČR, Prague, October 2005.

[3] E. Suzdaleva and I. Nagy, "Bayesian decision making and e-learning: Interactive examples", Tech. Rep. 2133, ÚTIA AV ČR, Prague, July 2005.

[4] V. Peterka, "Bayesian approach to system identification", in *Trends and Progress in System Identification*, P. Eykhoff, Ed., pp. 239–304. Pergamon Press, Oxford, 1981.

[5] M. Kárný, J. Böhm, T. V. Guy, L. Jirsa, I. Nagy, P. Nedoma, and L. Tesař, *Optimized Bayesian Dynamic Advising: Theory and Algorithms*, Springer, London, 2005.

[6] First Bayes, *http://www.tonyohagan.co.uk/1b/*.

[7] STEPS, *http://www.stats.gla.ac.uk/steps/*.

[8] Sila, *http://www.eco.rug.nl/medewerk/knypstra/sila.html*.

[9] Statistical Java, *http://www.stat.vt.edu/ sundar/java/applets/*.

[10] MacAnova, *http://www.stat.umn.edu/macanova/*.

[11] StatCalc, *http://www.ucs.louisiana.edu/ kxk4695/StatCalc.htm*.

[12] Statistica101, *http://www.programplaz.com/company/statistics101-net.html*.

[13] ViSta64, *http://blogs.zdnet.com/microsoftvistas/?p=32*.

[14] Statistical Computing Resources, *http://www.ats.ucla.edu/STAT/*.

[15] Stats & OR Network Maths, *http://mathstore.ac.uk/*.

[16] The Math forum & Drexel, *http://mathforum.org/*.

[17] J.M. Bernardo and A.F.M. Smith, *Bayesian Theory*, John Wiley & Sons, Chichester, New York, Brisbane, Toronto, Singapore, 1997, 2nd edition.

[18] SciLab, *http://www.scilab.org/*.

Case Study of Information Systems Cooperation Centered in Open Source LMS, CFIVE

Takayuki Sekiya
Information Technology Center
The University of Tokyo
3-8-1 Komaba, Meguro, Tokyo, JAPAN
sekiya@ecc.u-tokyo.ac.jp

Kazunori Yamaguchi
Information Technology Center
The University of Tokyo
2-11-16 Yayoi, Bunkyo, Tokyo, JAPAN
yamaguch@mail.itc.u-tokyo.ac.jp

ABSTRACT
In universities, an increasing number of information systems such as educational computer systems and online student information systems are constructed. Often interoperability of these systems is not well examined, and some of their functions and data overlap causing inefficiency and inconvenience. In this paper, we show an integration of some of these information systems centered around our Leaning Management System called CFIVE. CFIVE is an open source software and can be used in other universities.

Categories and Subject Descriptors
H.4 [**Information Systems Applications**]: Miscellaneous

1. INTRODUCTION
In universities, an increasing number of information systems such as educational computer systems and online student information systems are constructed. Often interoperability of these systems is not well examined, and some of their functions and data overlap causing inefficiency and inconvenience. In this paper, we show an integration of some of these information systems centered around our Learning Management System called CFIVE.

Our center, Information Technology Center of the University of Tokyo, provides university's staff and students with computer network infrastructure. As an infrastructure for supporting learning, we developed CFIVE[7] and provides a service of CFIVE to the university's staff and students since 2004. We are distributing CFIVE under GNU GPL (version 2).

As the integration, we have finished two projects. One is the integration of CFIVE with a student information system, and another is the integration of CFIVE with a smart card reader system. Through the integration, the following questions are clarified.

1) What kind of integration of systems should be employed?

2) How different policies governing the systems can be adjusted so that they can collaborate smoothly?

3) How the systems can be made friendly to staff and students?

After the half year of operation, we performed enquiries. We think that these integrations are successful because the result of the enquires is favorable. In this paper, we report what we have learnt from these two projects.

This paper is organized as follows. First, we discuss cooperation among information systems at universities in Section 2. Then, we explain the CFIVE that is a Leaning Management System we developed in Section 3. The two projects are detailed in Section 4. The conclusion is in Section 5.

2. COOPERATION AMONG INFORMATION SYSTEMS

2.1 Necessity of Cooperation among Information Systems
In our university it is getting popular to use information systems for ordinary work. We use a web-based accounting system for managing research budget. Some faculties have online student information systems, and the university is developing a university-wide student information system in a few years. Even though these systems have their own raisons d'être, we observed some problems about these systems.

(1) An account is independently issued for each system.
Account management is a costly operation of administrators and multiple accounts are inconvenient for users.

(2) The information handled partially overlaps.
Handling such common information independently is inefficient.

2.2 Important Points
Solving the above-mentioned two problems, we have to think about the following issues.

(1) Security
By sharing account information, the danger of the exposure

of a single account information increases. For sharing data, we have to transfer data among systems. The possibility of eavesdropping increases and the vulnerability of one system endangers the others. For making a cooperation among systems, it is important to device a method to alleviate the danger.

(2) Policy
Administration policies vary among systems. For example, some administrator issues a renewed password after two days from application while other administrator issues it promptly. Some administrator works days and nights if its system is in trouble, while other administrator works nine-to-five even in such case. For making a cooperation among systems, it is important to make these issues clear and adjust responsibilities on both sides.

2.3 Existing Solutions

Identity management systems[8] are already in market, which may solve the problem (1). Some universities such as Drew University[4] introduced the identity management system. However, the identity management system is a large system and it costs much. So, our university does not have a plan to acquire it.

No monolithic solution exists for solving the problem (2) as far as we know.

2.4 Our Solution

Because education is one of the most important tasks of university, various data are related with LMS. So, to solve the problem (1) and (2), it is reasonable to place LMS at the center of information systems.

If an administrator may want to have more security on the account, he/she may issue a secondary password for accessing the system he/she manages. Because the shared account guarantees some security, the secondary password can be delivered in a less strict manner.

In order to reduce the danger of eavesdropping, we transfer sensitive data just once a day. A check is performed on the transfered data so that falsified data is not used. The transfer line is protected by both SSH and VPN to reduce the danger of security holes.

In order to alleviate the problem of policy difference, we make the systems loosely coupled so that if one of the system fails, the others can run independently. Also, responsibilities of both sides are documented before starting cooperation.

3. CFIVE

3.1 Our Needs for LMS

Our center provides all faculty staff and students of our university with IT infrastructure. With LMS to be such IT infrastructure, the following features are important.

(1) Scalability
In our university, some lectures have several hundreds of students. So, the LMS should be highly scalable.

(2) Adaptability
Teaching methods depend on faculties and staff. So, the LMS should be compatible with wide varieties of teaching methods. Therefore, the LMS should be highly adaptable.

(3) User Support
The user support is a very important function of our center. Because unused functions make user support more complicated in vain, the functions of our LMS should be kept minimum.

No proper function is available to fulfill the need of faculty staff, then we have to develop tools to satisfy the need. So, at least the API of the LMS should be open, and it is desirable that a source code of the LMS is also open.

(4) Interoperability
Our center and other faculties of the University of Tokyo provide information systems such as educational computer systems and student information systems. We often have to make the LMS to share some data with such systems, and the interoperability is crucial for it.

(5) Sustainability
The system should be cost effective.

3.2 Existing LMS's

According to the survey[2], more than 80% American higher education institutes use LMS. In North American and European countries, commercial CMS such as Blackboard/WebCT and Jenzabar are quite popular. Because these software have many functions, it will take long to master these software failing to satisfy the requirement (3). Also, these software are expensive failing to satisfying the requirement (5). Therefore, the commercial LMS's are not our options.

Sakai[9] is an attractive free software. Sakai is internationalized in the sense that messages in local languages including Japanese can be shown. However, the school year of the most of Japanese universities is different from that assumed in Sakai. For example, in Sakai, academic terms such as "SPRING 2006","SPRING SUMMER 2006", "SUMMER 2006" are automatically set up. But the school year of the most Japanese universities is divided in two semesters. Also, the numbering custom of courses at Japanese universities is different from that assumed in Sakai. Sakai expects teachers to input three type information, "Subject (five alphabetical words)", "Class (a three-digit number)", and "Sections (a three-digit number)", but in our university a course number consisting of 5-6 digits to distinguish each course is used. ULAN PROJECT[10] has been trying to improve Sakai, but it has not yet finished.

Because the existing software does not match the teaching methods adopted at universities, many LMS's have been developed. For example, Kansai University developed CEAS[3], and National Institute of Multimedia Education (NIME) developed exCampus[5]. Though these systems are useful for some specific teaching method, they are less customizable failing to satisfy the requirements (1), (2), and (4). In conclusion, there were no LMS satisfying our needs.

3.3 Design Policies of CFIVE

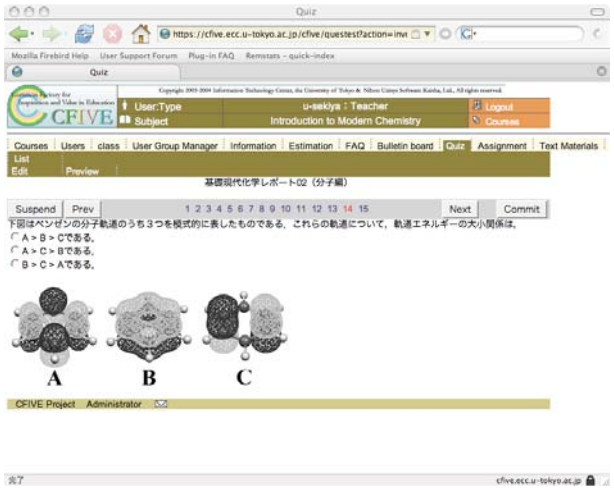

Figure 1: The screenshot of CFIVE

The Information Technology Center at the University of Tokyo, and Nihon Unisys Solutions, Ltd jointly developed an original LMS called "CFIVE" (short for Common Factory for Inspiration and Value in Education") in April 2004. CFIVE is distributed as an open source software under GNU GPL(version2). In order to satisfy the requirements mentioned in Section 3.1, we set the following design policies for developing CFIVE.

Scalability
We adopted the Java Servlet architecture for efficiency. Script languages such as Perl or PHP are not used.

Interoperability
In order to make it easy to share authentication information between CFIVE and other information systems, LDAP is used. For making it possible to manipulate data directly, a standard SQL-based database is used.

Standards
CFIVE does not support SCORM[1]. This is because there is no need to use SCORM compatible learning materials at present, and its implementation and maintenance cost is high. Now that SCORM engine[6] is available, we may adopt it in our future version of CFIVE.

Open Source
We keep a code proper to CFIVE as small as possible by utilizing existing open source software such as PostgreSQL, OpenLDAP, Tomcat, Torque, Log4J, SiteMesh, and JSTL to make the maintenance easier.

Multilingual
In order to guarantee that multilingual characters are treated properly in every function, Unicode is used for both the source codes and the data in a database of CFIVE without exception. In CFIVE, the language of the menu is determined by the preference on language of the accessing Web browser.

Role
The role of a person varies depending on courses. For example, a person may play student in one course and teaching assistance in other course. In CFIVE, a user and a role are separated and a single user can take various roles depending on courses.

Table 1: Major Functions of CFIVE

News	News
FAQ	Frequently asked questions
Assignment	Report submission
Download	Educational material delivery
Quiz	Quiz management and challenge. Answer types include single choice, multiple choice, fill-in-the-blank, and free text. Answers except free text are automatically graded.
BBS	Bulletin boards for discussion
Evaluation	Evaluation feedback
Class	Class schedule and attendance record management

In our university, some courses take a project-based learning method. In CFIVE, a group management functions is equipped to support such project-based learning method.

3.4 Features of CFIVE

Following the policies stated in Section 3.3, we developed CFIVE in 2004. Major functions of CFIVE are listed in Table 1. A sample screenshot is shown in Fig. 1.

3.5 Current Status of CFIVE

Our center offers a service of CFIVE to students and faculty staff members of our university, and more than 50 courses have used the service so far. We are continuously improving the system responding to the requests from staff and students, and the cost is moderate.

As reported in the next section, we loosely coupled CFIVE with a student information system, and extended "Class" function of it so that a portable reader of student ID cards can be used. These are accomplished with a relatively small amount of effort proving that our design is effective.

4. CASE STUDY

4.1 Collaboration with Online Student Information System

The college of arts and sciences at the University of Tokyo is a faculty to provide all the freshmen and sophomores with liberal education. In 2004, the college decided to introduce an online student information system named UTask in order to automate course enrollment and grade report management. However, the college did not have a function to manage account information for students, which our center has for CFIVE. So, the college and our center started to talk about collaboration. UTask is operated by the college and CFIVE is operated by our center independently. Because the operation policies are different, we avoid tight-coupling.

(1) The account information of students in CFIVE is provided for UTask-Web.

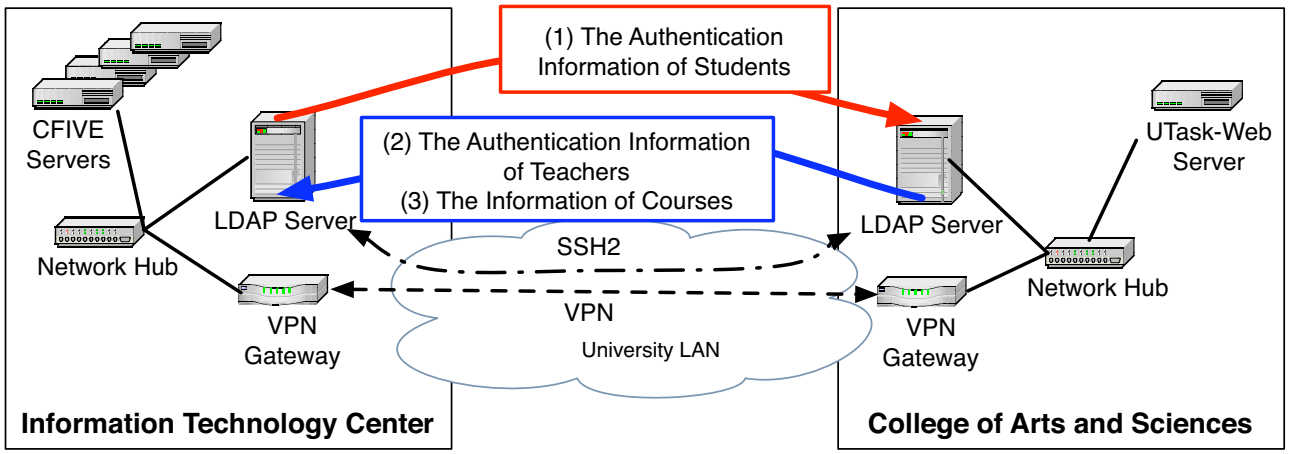

Figure 2: UTask-Web and CFIVE

Our university has about 7,000 freshmen and sophomores. Our center issues accounts for them so that they can use educational campuswide computing system (ECCS) in learning. The accounts are also valid for CFIVE.

(2) The account information of staff in UTask is provided for CFIVE.
The center has a function to manage account information of staff as well as students. However, because staff are not forced to apply for accounts, only a few staff actually have accounts. On the other hand, it is easy for the college to issue accounts for staff, because the college send and receive information to/from staff frequently. Consequently, contrary to the students' case, the college provides the authentication information of staff in UTask for our center.

(3) The enrollment information in UTask is provided for CFIVE.
Student enrollment information is primarily managed by UTask for student grade management. However, the center needs the information for controlling the visibility of courses in CFIVE.

Now, we will explain the mechanism of transferring information between CFIVE and UTask-Web. The information transferred between the two systems is important, and so it must be transferred through a secure path in order to avoid information leakage or eavesdropping. In order to secure the path, hardware-based VPN is introduced, and for transferring the files for exchanging information stated above, SSH is employed as illustrated in Fig. 2. Using both VPN and SSH seem to be a bit excessive, but we adopted this scheme because it is difficult to prove that the student record is free from falsification if vulnerability of SSH as in the case of SSH1 is found.

UTask and CFIVE have their own LDAP servers so that each of the systems can work separately even if the other system is down. The authentication information of students is exported into an LDIF file in the LDAP server of CFIVE, and the file is transferred to the other system every midnight. The other system imports the file into its LDAP server if the file seems to be correct as to its timestamp and size. The time and frequency of copy is decided so that it causes no excessive load to the systems.

By adopting loose coupling, we minimized the influence of the other system and that of the administration policy of the other system. However, there is some influence as follows. At the beginning of summer term which is the first term for freshmen, our center issues accounts to freshmen. Usually it takes more than one week to issue accounts to all of them. Because the students to have to enroll some courses in the first week using UTask and now the accounts are needed for UTask, we had to shorten the time with the help of the college.

The college thinks that student record should be more secured than the information in CFIVE because it contains the student information such as course record and academic achievement. So they issue PIN (Personal Identity Number) which is used along the account we issue.

The college and our center signed a memorandum of cooperation between UTask and CFIVE about the basic scheme explained above, responsibilities, and restrictions on sensitive information.

No serious problems occurred for about one year since we started the cooperation of two systems. The scheme works smoothly. The online information exchange reduced the workload of us. There is an enquiry why the password change in CFIVE is not reflected on UTask promptly, but it is just a matter of explanation.

4.2 Attendance Management System based on a Smart Card

The identification card of staff and students at the University of Tokyo is a smart card called "UTcard." We use UTcard for a check out at the university coop stores, entrance gates at the university libraries, and so on.

In the University of Tokyo, a student ID is valid just for two or three years. When a student promotes from a sophomore to a junior, a new student ID is assigned to him/her. Such "volatile" ID is useful for manual handling because it

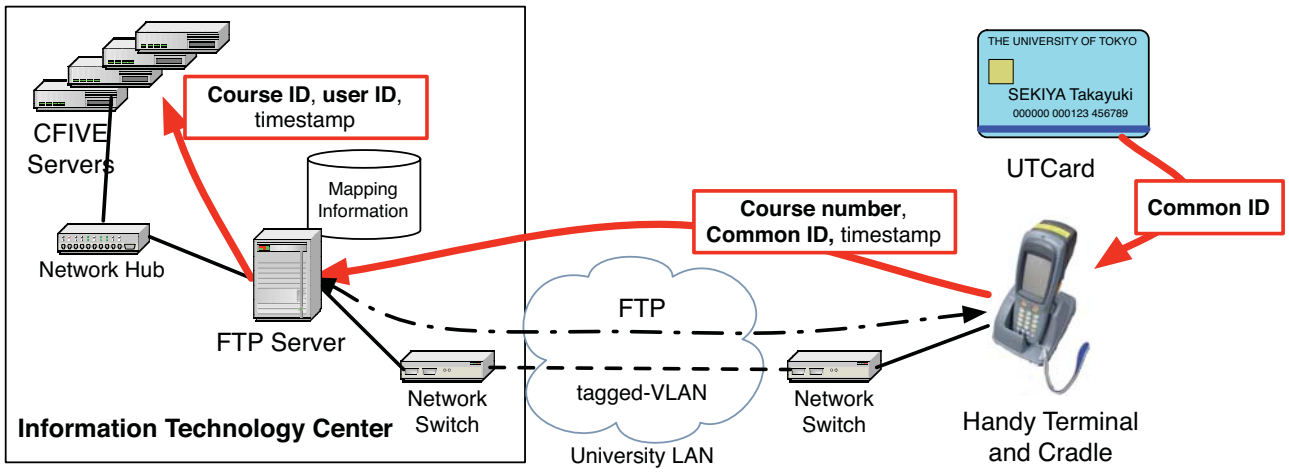

Figure 3: The Attendance Management System based on UTCard

Figure 4: HT and cradle

is usually short and consecutive. However, for automatic transaction, such a short ID causes a trouble, because some different IDs correspond to a single student, and the unity is not easy. Also, because the combination of such short IDs is limited, they are re-used. As a result, a student is not identified just by a short ID in a long period. In order to overcome these problems for automatic transaction, we introduced an identification number called "common ID" four years ago. Common ID is an identification number uniquely assigned to a person lifelong. From the requirement that common ID should be unique to potential members of the University of Tokyo for the coming hundred years. a rather long number, 18 digits with tentatively active 10 digits, is used. Manual handling of such a long number is painful, and for automatic handling of common ID we need a carrier. So, the UTcard is used as a carrier of the common ID for automatic handling. The attendance management system which will be explained here is one of the application of using the common ID in UTcard.

In the University of Tokyo, class attendance information is gathered for some scheduled classes and final examinations, inside or outside(field study or physical), with the size of the class ranging from 10 to 1000. For a large classroom used frequently, it may be better to equip UTcard readers at the entrance doors of the classroom. It is not cost effective for small classrooms not so frequently used. So, we developed a portable device for reading UTcard which can be carried to such small classrooms. The portable device, which we call "Handy Terminal" (HT, for short) is developed by us jointly with NTT Communications Corporation, which is the vendor of the UTcard, and DENSO CORPORATION, which is the vendor of the HT. HT is shown in Fig. 4

Next we will explain a mechanism of the attendance management system as shown in Fig. 3. We make the function of HT as simple as possible in order to reduce the development cost. It has only two modes: course number setup and UTcard read-in. The two modes toggle by pushing a single button. In the course number setup mode, you can set a course number just by typing the number followed by ENT(ENTER key). In the UTcard read-in mode, you can read UTcard and write its common ID in a file indexed by the course number set in the previous course number setup mode.

Data in HT is sent to a predetermined ftp server by pushing "Server" and "Confirm" buttons through a cradle. For security, we use a tagged-VLAN for a network connection from a cradle to the ftp server. Only a new data is sent to an ftp server and once the data is sent to a server, it is put in a deletion queue. Data in the deletion queue is removed in a FIFO order. No operation is required to remove data explicitly. When something goes wrong and the data sent is lost, then if data is still in the deletion queue, it can be sent again by a special sequence.

There is no button to delete data on HT. The common ID it has read never disappear until it is sent to an ftp server and the data gathered afterward pushes it out from the deletion queue. So, there is no room for doubt that staff removes the data read accidentally. Because users are usually notices in such equipments, such a kind of fool-proofness is very important.

We cannot expect that the data gathered is perfect. Some

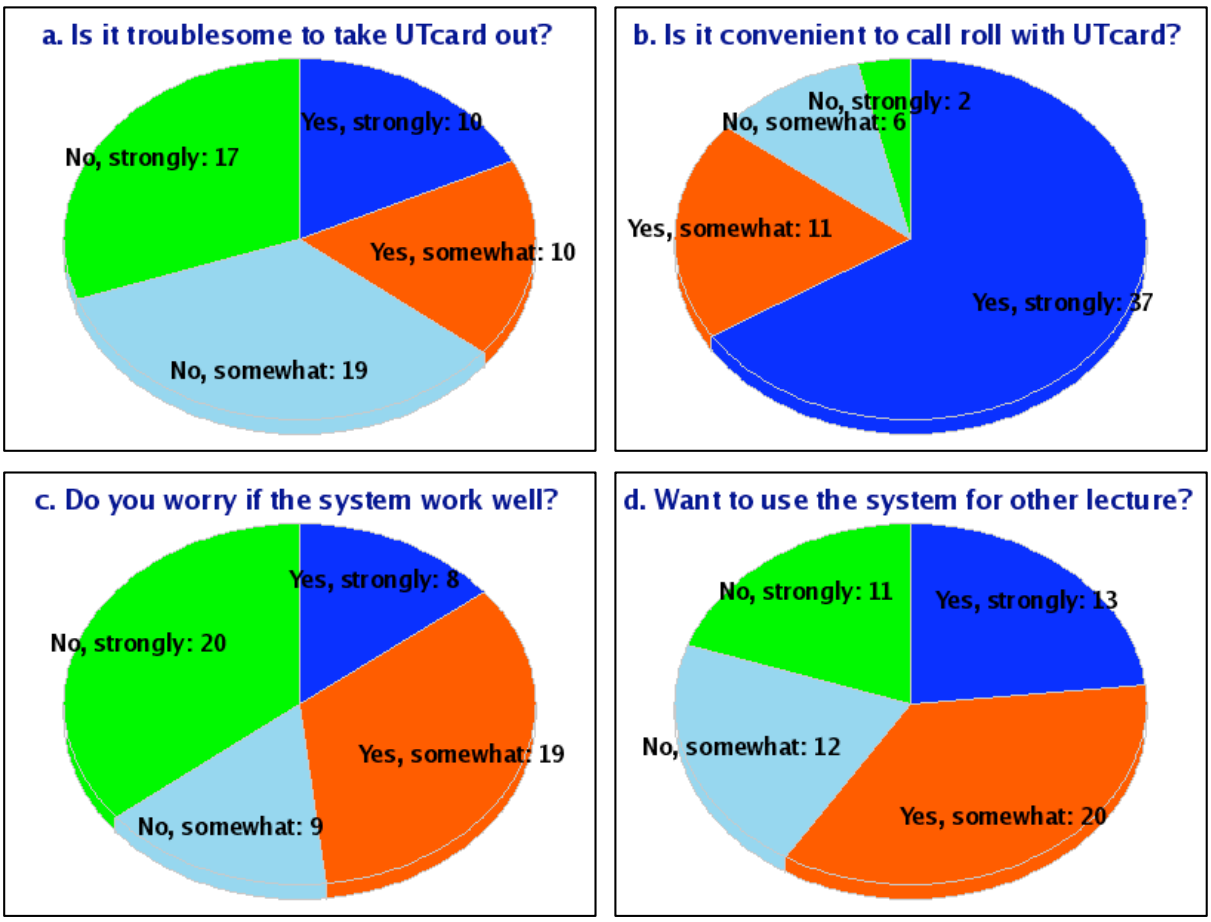

Figure 5: The result of the questionnaire

students attend without UTcards and there may be troubles in reading UTcards. So, we need some mechanism to manage the data gathered. It is not effective to develop an independent system for this purpose only. Fortunately, CFIVE has the function to manage attendance record. So, we decide to use CFIVE for the management of the attendance information. For doing it, there are two problems.

1. HT can read common ID, while a user in CFIVE is identified by a user ID which is based on a student ID.
2. For a course, many classes may be registered, and there may be a case that no class is registered. How the attendance data for the course should be used?

We have designed HT and the ftp server, and modified CFIVE to cope with these problems. 1) We get a mapping information from common ID to student ID from the administrator of UTcard. We have a mapping information from student ID to user ID of CFIVE. Common ID is translated into the corresponding user ID at the ftp server using the information. 2) A course ID is registered for each course in CFIVE. We have a mapping information from course number to course ID. Because the data in HT is divided into files indexed by course number set by a teacher, it is easy to translate course number into course ID and to associate the data to a course in CFIVE. In order to make it possible to locate a class to which the record is related, HT is designed to store common ID with the timestamp of its acquisition. We show how the timestamp is used by an example. Suppose that there are scheduled classes "May 5, 13:00-14:30", "May 12, 13:00-14:30", ... , and the record is "g99999 May 12, 13:44". Then, it is registered that g99999 attended the class "May 12, 13:00-14:30". If no class is scheduled for the time period including the timestamp of the record, then a class is automatically created for the day. For example, if there is no class entry and the record is "g99999 May 12, 13:44", then a dummy class "May 12, 00:00-00:00" is created, and g99999 is registered that he/she attended the class.

So, the entire flow is quite simple.

1. A teacher creates a class in CFIVE. (This can be omitted if the automatically created class entry is sufficient.)
2. A teacher sets a course number at HT and read UTcards of students by it.
3. A teacher puts HT on a cradle and push "Send" and

"Confirm." (This has to be done only when a teacher want to see the attendance record at CFIVE.)

This system is developed in just one month. Because the attendance record format in CFIVE is open, we can put the attendance record by issuing SQL command to the database without bothering with complex protocols.

We used this system for one course in the Winter semester started on October 2005. The number of enrollment for the course is about 400. In the course, teaching assistants read UTcard of the students at the entrance of the classroom. The system worked quite well for this usage. We experienced no information loss. There were some UTcards from which HT could not read common ID. This is not a big problem, because we have to have a way to deal with students without UTcards, and the above case can be dealt similarly. Such exception is rare and majority is handled very efficiently.

We show some results of enquiries conducted at the end of the course in Fig. 5. 85% of the students think that this attendance system is convenient. 50% of the students have concern about the correctness of the attendance record. A part of this anxiety is caused by the fact that there is some UTcards which HT can not read, and we expect that this will be eventually resolved.

5. CONCLUSION

In this paper, we reported two projects of cooperation of information systems centered in CFIVE. Through the projects, we could carry out flexible cooperation among information systems centered in CFIVE because CFIVE is an open source software and the API of CFIVE is open. We found technical issue was not a big problem, preferably we made a success of the projects due to paying attention to security and administration policy.

6. ACKNOWLEDGMENT

We would like to thank Nihon Unisys Solutions, Ltd. for their help in the initial development of CFIVE, NTT Communications Corporation and DENSO CORPORATION for the development of the portable smart card reader, and the college of arts and sciences of the University of Tokyo for the cooperation in the student information system. We are grateful to our colleagues, Dr. Yoshiyuki Onoue, Mr. Takaaki Sato, and Ms. Yuki Terawaki, who have never-failing enthusiasm in improving CFIVE. Special thanks go to all the staff and students for using CFIVE and giving valuable suggestions to us.

7. REFERENCES

[1] Advanced Distributed Learning (ADL). SCORM 2004, 2004. http://www.adlnet.gov/scorm/.

[2] K. C. Green. Campus computing 2003, 2003. The 14th National Survey of Computing and Information Technology in American Higher Education, The Campus Computing Project.

[3] T. Koyama, K. Koyama, Y. Ueki, M. Arakawa, and M. Fuyuki. Web-based coordinated education activation system ceas which aims to fit japanese educational environment — system reconstruction based on java framework —. In *IPSJ SIG Technical Reports 2005-SIGCMS-1*, pages 51–58, 2005. (in Japanese).

[4] E. A. Larsson. A case study: implementing novell identity management at drew university. In *SIGUCCS '05: Proceedings of the 33rd annual ACM SIGUCCS conference on User services*, pages 165–170, New York, NY, USA, 2005. ACM Press.

[5] National Institute of Multimedia Education. excampus.org, 2004. http://excampus.nime.ac.jp/.

[6] NTT Open Source LMS Project. Opensource LMS, 2004. https://www.oss.ecl.ntt.co.jp/lms/,http://osl.xcalat.com/.

[7] T. Sekiya, Y. Terawaki, Y. Onoue, and K. Yamaguchi. Cfive: Open source learning management system. *Journal of Multimedia Aided Education Research*, 1(2):73–81, 2005. (in Japanese).

[8] Sun microsystems. Identity Management Solutions, 2006. http://www.sun.com/emrkt/idmanagement/.

[9] The Sakai Project. About sakai, 2006. http://www.sakaiproject.org/about.html.

[10] ULAN PROJECT. Ulan: Ubiquitous learning architecture for next generation, 2004. http://www.ulan.jp/.

Converting Legacy Application to Web Services using an Open Source Toolkit: A Case Study

Rahul Upakare, M Sasikumar
Open Source Software Resource
Centre, CDAC, Sector 7, Raintree
Marg, CBD Belapur, Navi Mumbai,
400 614, India, +91-22-27565303

{rahulu, sasi}@cdacmumbai.in

ABSTRACT
Today a large number of software tools are available to use computer to support traditional on-campus based courses. One such tool that we are using for automated program grading in our on-campus diploma programmes is Parikshak system. Today many Learning Management Systems (LMSs) are available for delivering courses online or in blended mode. Tools which are useful for on-campus course students can be easily integrated with the chosen LMS, to make them available for off-campus students. But, most of the times, these tools (like Parikshak) gets implemented as a standalone application without any inherent mechanism to allow seamless integration with other applications or frameworks. In this paper, we investigate how well Web services meet the interoperability requirements by transforming a real example of Parikshak system into a program grading Web services framework using the open source gSOAP toolkit and providing standard Web service interfaces for seamless integration to other applications.

Keywords
Program grading, Web services, Interoperability

1. INTRODUCTION
Programming assignments are a critical part of programming courses. It is difficult and time consuming to evaluate programming assignments manually, because of the complexity and sheer numbers of students involved in such courses. It is also difficult to ensure uniform standards of evaluation. To address such problems, we have a system called Parikshak – an automated program grading system - which provides a secure and restricted development environment for students to develop and submit their programming course assignments and to receive immediate feedback. It provides course staff tools to monitor and control the process and the ability to generate student assignment submission reports to track students performance.

Today many Learning Management System (LMS) are available for delivering courses online or in blended mode. Tools such as Parikshak need to be integrated with the chosen LMS, to make them accessible seamlessly under a single environment. But, most of the times, these tools (like Parikshak) gets implemented as a standalone application without any inherent mechanism to allow seamless integration with other applications or frameworks.

Web services is an emerging technology and useful to implement Service Oriented Architecture (SOA) [4]. Web services offer a strong foundation for software interoperability through the core open standards of eXtensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL), and Universal Description, Discovery and Integration (UDDI). SOAs are transforming monolithic applications into services, thus allowing portability, reusability and on-demand access. Instead of thinking about applications residing on a specific server, we should think of ubiquitous services that can be used and shared more dynamically. The implied benefits are simplified design, code reuse and a giant step toward "business agility" [3]. SOA solutions are composed of reusable services, with well-defined, published and standards-compliant interfaces. SOA can be used as a mechanism for integrating existing legacy applications regardless of their platform or language.[1]

Concepts like Service based software [2], Software as a Service [1], etc. are emerging rapidly and has given innovative vision for future software architecture and development. Software as a service (SaaS) is a software distribution model in which applications are hosted by a service provider and made available to customers over a network, typically the Internet. SaaS is becoming an increasingly prevalent delivery model as underlying technologies that support Web services and SOA mature and new developmental approaches, such as Ajax, become popular. There is interest in moving many applications from their existing framework to run over the web under a Web services framework.

Parikshak is currently a standalone system in the typical legacy application mode. We believe that integration of such a legacy software, into a LMS will be smoother if it is transformed into a Web service. Further we believe that Parikshak Web service is best seen – from machine perspective – as a collection of services, each service performing a specific task. Such an approach enables many natural enhancements to Parikshak and also provides many advantages, as discussed in subsequent sections.

The remainder of this paper is organized as follows. Section 2 discusses the Parikshak system which is being used at our campus for automated program grading. The open source gSOAP toolkit is briefly discussed in Section 3. Section 4 discusses Parikshak and Web services and our phased approach to transform Parikshak

[1] Quotes from http://serviceorientation.org/

into a program grading Web services framework. In section 5, we discuss the program grading APIs, Conclusion and future work is discussed in section 6.

2. ABOUT PARIKSHAK

Parikshak is a system for testing problem solving skills through programming, which provides a restricted environment for students to develop and test their program. Parikshak is a legacy system and has evolved from bunch of shell scripts to a collection of C programs. It has been built and maintained in-house. Our centre and affiliates are using Parikshak system in programming modules of our diploma programs to automate the program grading.

2.1 Parikshak – A User Perspective

Parikshak is used in two modes: assignment and evaluation. Assignment mode is used to solve the assignments and evaluation mode for programming tests. We use Parikshak for Object Oriented Programming using Java (OOPJ) and Data Structures and Algorithms (DSAL) modules of our post graduate diploma programmes.

Parikshak Administrator can install assignment package for a particular course module. An assignment package contains one or more programming assignments to be solved by students. Students have to submit these assignments within the specific period set by the Administrator. Parikshak keeps log of all the submissions done by students and does not allow any submission after or before the specified period.

Parikshak is used in the evaluation mode for conducting a Machine Graded Programming Test (MGPT) [6]. MGPT is conducted in two sessions. The first session is conducted in a classroom under manual supervision. In this session, students get a problem statement to read, understand and develop the solution on paper along with suitable test cases. Access to computer facility is not allowed during the first session. In the second session, students use computer to invoke Parikshak in the evaluation mode and then they can write and submit the solution for grading. Students get limited time during both the sessions.

From a user (student) perspective, Parikshak environment is essentially a restricted Linux shell. Many of the Linux commands have been disabled in this environment from examination security point of view and to discourage copying, etc. and some additional commands have been provided. The Parikshak environment is invoked, using the command *testme*. Command *showproblem* shows the problem statement. Users can use *vi* or *nano* editor to write the program solution and can submit it for testing using *submit* command. When the program is submitted for testing, Parikshak will compile the program, listing the compilation errors if any. If the program solution compiles successfully, the system will run the submitted program and give it test inputs and evaluate the output.

2.2 Parikshak Architecture

Parikshak system is not a single executable. It is a collection of C program executables. Each of these executables act as a module to provide the specific functionality. The major modules and their relation to various commands given earlier are shown in Figure 1.

The different modules include assignment and test packing module to create the installable package, assignment and test unpacking module to install the installable package, users module to control the access, and report generation module to generate reports.

Here, we attempt to reorganise this Parikshak system to make it interoperable with other systems using web services based approach.

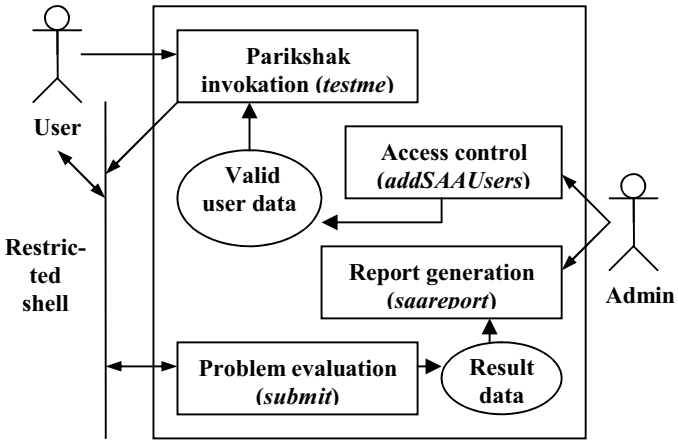

Figure 1. Parikshak as collection of executables

3. INTRODUCTION TO OPEN SOURCE gSOAP TOOLKIT

The gSOAP compiler tools provide a SOAP to C/C++ language binding to ease the development of Web services and client application in C and/or C++. gSOAP is one of the few SOAP toolkits that support the full range of SOAP 1.1 RPC encoding features including sparse multidimensional arrays and polymorphic types [5]. For example, a remote method with a base class parameter may accept derived class instances from a client. The gSOAP toolkit also supports streaming DIME attachment transfers, which allows exchange of binary data. The gSOAP 2.0 and higher is thread safe and supports the implementation of multithreaded standalone services in which a thread is used to handle a request.

To offer application as Web service, it must implement a set of SOAP-compliant RPC functions to expose the service on the Web for remote invocation. The gSOAP toolkit completely automates this task. The gSOAP compiler can also generate a WSDL document describing the service in detail; this would be useful to discover the Web service.

The deployment of a Web service as a CGI application is one way to provide service on the Internet. gSOAP services can also run as standalone services on any port by utilizing built-in HTTP and TCP/IP stacks. We have used gSOAP toolkit to implement our system, thanks to the strengths mentioned above.

4. PARIKSHAK AND WEB SERVICES

This section discusses challenges and issues in transforming Parikshak into Web services, and a phased approach to transform Parikshak, a legacy standalone program grading system, to a program grading Web services framework.

Web services are self-contained applications that are published, located, described and invoked over the Internet or intranet. A Web service can be built from the ground up as a new application or an existing legacy system can be re-engineered to make it Web service enabled. Offering application like Parikshak as a Web service will help to access it not only through the Internet, but will also make it platform and programming language independent. This means the Parikshak service invocation can be easily integrated with any Learning Management System like "Moodle" written in *PHP* or Integrated Development Environment like "Eclipse" written in *Java* and can be accessible transparently by the users. A Web service model widens the reach of Parikshak to other devices which can support development of Web services clients like Hand-held devices, mobile handsets etc.

Conversion of Parikshak into a Web services framework is achieved in phases. In phase 1, programmatic access to user/student related functionalities of the system is provided. In phase 2, Parikshak is reorganised into a collection of services, and finally a service discovery component is added in phase 3.

4.1 Challenges and Issues

Before designing Parikshak as a Web service, let us find out the challenges and design issues involved in implementing the Program grading Web service.

4.1.1 Long response time

For every program grading request, Parikshak service requires compiling and grading the solution against 6-8 pre-defined test cases. Therefore the response time for handling program grading requests by the service will be long. This means that if the service is serving the program grading request from one client, other clients may not be able to use Parikshak Web service at the same time.

4.1.2 Time restrictions

When we use Parikshak for Machine Graded Programming Test (MGPT) [6] students get 90 minutes on computer in the second session to write and submit the solution for grading to Parikshak as described in section 2.1. Such time restrictions needs to be imposed fairly and consistently by Parikshak program grading Web services framework also.

4.1.3 Malicious code

The program grading service compiles and executes the source code submitted by the user. Since, program will be getting executed on the server, there is a possibility that someone can try to submit a malicious code to bring the service or server down. For example, calling fork system call to create a new process infinitely on Unix machine, can bring down the server. Also, one can try to delete or modify the existing files.

4.1.4 Interoperability

Web services technology can provide a new level of interoperability between software applications. There are many frameworks are available to enable software with SOAP, WSDL, and UDDI capabilities. But, not all frameworks are yet interoperable with each other. Also, these frameworks might support different version of W3C recommended specifications for SOAP, WSDL etc. Also, to ensure the interoperability, it is important to decide the data types of parameters which will be used in Web service function calls.

4.2 Phase 1: Program Grading Web Service APIs for Parikshak

In phase 1, we have enabled programmatical access [7] to Parikshak including problem set enquiry, problem set report etc. For this, we have made all student related functionality of the system available as program grading Web services APIs, which enables seamless integration of Parikshak with other application just by implementing a Web service client.

Figure 2 shows Parikshak transformed, this way, into a Web service for a student/user. A service has been written as a multithreaded standalone server. A client application can be implemented to invoke the Parikshak functionalities using the provided APIs. W3C recommended Standard Object Access Protocol (SOAP) is used for messaging.

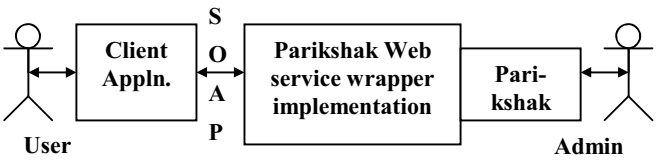

Figure 2. Parikshak System as a Web Service

Web services framework is implemented as a wrapper over existing Parikshak system, with some of the main components of Parikshak re-implemented to suit the Web services model. Using the WSDL document of the service, the service consumer can get the details of the data types definitions related to the messages being exchanged, messages name, parameters, message binding, and service location. The user then can send appropriate request using those APIs. The APIs allow user to request for available problem set names, details about any particular problem set, a particular problem statement within a set, etc. Users can write a solution for the selected problem and send it for evaluation to Parikshak. Also, they can send request to get the problem submission status for a given problem set.

A Web service client can be written in any programming language and on any platform using these APIs.

4.3 Phase 2: Conversion of Parikshak into a Collection of Services

While a human user tends to perceive Parikshak as a single well-integrated service, we believe it is best seen – from a machine perspective – as a collection of services, each service performing a specific task.

In Phase 2, we separate the independent functionalities within, as independent Web services. As we can see in Figure 3, user authorisation, program grading, result processing and malicious

code detection are all almost independent functionalities within Parikshak. Therefore, we can view Parikshak as a loose coupling of such functionalities by implementing those as separate Web services.

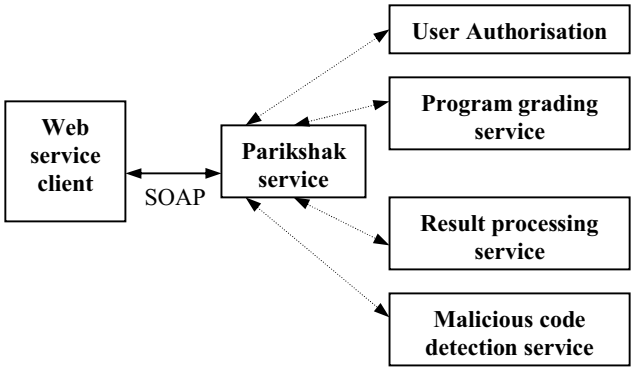

Figure 3. Parikshak system as a collection of loosely coupled services

For external user, it continues to be a single interface i.e. a Parikshak service, but internally it will be a collection of services loosely connected with each other. In fact, due to this loose coupling, service implementations can be scaled, evolved and changed dynamically, as long as each continues to implement the specified service contract.

Also, in case better service for any of the components is available elsewhere providing the matching functionality, then it can be integrated with Parikshak seamlessly. This also enables Parikshak to run across multiple platforms with the core engine running on one platform, and the compiler/evaluator for different languages running on different machines under different operating systems. Each of the component services can also be further specialized into another level of services e.g. Java malicious code detection service, C malicious code detection service, etc.

4.4 Phase 3: Web Services Oriented Architecture for Program Grading

In phase 2, we have seen Parikshak as a collection of loosely coupled services. Here, we extend the organisation of Parikshak as a collection of loosely coupled services which are discoverable using a local service registry as shown in Figure 4.

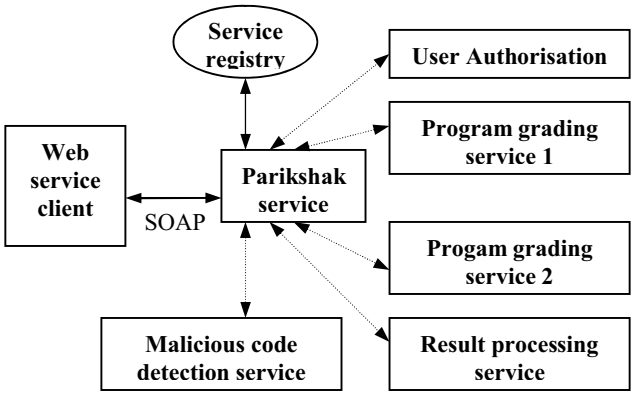

Figure 4. Service oriented architecture for Parikshak

A service consumer need not have to worry about a particular service location. Parikshak Web service can make an appropriate choice on behalf of the consumer using the service registry.

The service registry is implemented as another Web service. The Parikshak service can invoke the service registry APIs to get the appropriate location of the User authorisation service, Program grading service, etc. Currently, service registry can provide a service location for requested service. It also allows service provider to publish and un-publish their services.

5. OBSERVATIONS

During the implementation of all three phases, we found the gSOAP toolkit very useful. To address the long response time issue as discussed in section 4.1.1, we have implemented Parikshak service as multithreaded server. The gSOAP supports implementation of multithreaded standalone services and we found it very easy to implement.

As shown in Figure 3 and Figure 4, Parikshak Web service acts as a service for Web service clients and as a client to program grading service, malicious code detection service, service registry etc. Implementing client functionality within the Web service can be easily achieved using the gSOAP toolkit.

Web services are stateless and as discussed in section 4.1.2, program grading service requires maintaining the time restrictions. We address this issue, by implementing our own session management. When a user requests a problem statement, service adds entry to session list and returns a session key. A user needs to send this session key for all subsequent calls.

We did not face any problem in deciding the data types of messages as mostly they were of type strings or integer. Only in case of problem statement file, we have used baes64 binary format to support statement file in various formats like PDF, DOC etc. Also, gSOAP support the use of XSD type encoded messages to improve the interoperability.

6. PARIKSHAK WEB SERVICE APIs

In this section, we briefly describe some of the major service APIs that we have implemented as part of the phase-1 transformation.

6.1 Program Grading Related Requests

The student/user needs to communicate with the service to get the problem statement, submit the solution for evaluation, getting the problem submission status for a set etc. Some of the requests provided for student/user are listed below.

6.1.1 Problem Set Names Request

As the name suggest, this requests problem set names to Parikshak program grading Web service and receive in return an array of problem set names.

6.1.2 Problem Set Details Request

Problem set details request submits a name of problem set to Parikshak program grading Web service and receive in return the list of problem number and corresponding problem title, in the requested problem set.

6.1.3 Problem Statement Request

Problem statement request submits a name of problem set, a problem number and expected problem statement file format (text/html, text/plain, text/pdf etc,) and receive in return the problem statement for the requested problem number in selected problem set and in the specified file format. The problem statement is sent as base64 encoded text. This request also returns a session key which user need to send with every further request (regarding this problem) to the program grading service.

This request has two variants, one for assignment mode and the other for test mode.

6.1.4 Compile Problem Request

Compile Problem request submits a source code of implemented solution and programming language to Parikshak program grading Web service and receive in return the compilation status of the submitted solution or listing of compilation error if any.

6.1.5 Submit Problem Request

Submit Problem request submits a user ID, problem set name, problem set number, session key and source code of implemented solution to Parikshak program grading Web service and receive in return the grading status of the submitted problem or compilation error if any.

Also, there is *Get Remaining Time Left Request* to check remaining time and *Get Problem Set Submission Status Request* to check problem submission status.

6.2 Other Requests

The other requests include requests which are sent among the Parikshak program grading service and the individual web services mentioned in phase 2 or which are not specifically related to program grading functionality. Some requests like *get problem set report request* and *is malicious code request* can be invoked directly by service consumer, provided the service details are known and accessible to Web service client.

6.2.1 Get Problem Set Report Request

Get problem set report requests submit a problem set name to result processing service and receive in turn the problem set report containing detailed information about student problem submission status for selected set. This request is restricted to Parikshak administrator only. Currently HTTP authentication is used to authenticate the Administrator in addition to secret registration key security.

6.2.2 Is Valid User Request

Is valid user request submits a user ID and registration key and problem set name to authorisation service and in turn receive true or false based on whether the user is valid or not, for the particular assignment/test.

6.2.3 Get Service Location

Get service location request submits a name of service and in turn receive the location of requested service.

7. FUTURE WORK

As discussed in our phased approach, we have successfully implemented all the three phases. As a future work we will implement multiple instances of the various component services running to share the load among the available program grading service instances. This multiple service provider idea can be used for any service which takes longer response time or generally gets loaded heavily. This also helps to make sure the high availability of services, because if one service provider is not available then the request can be passed to other available provider offering similar service. This means there are less chances of total service failure. We also plan to convert more of the legacy components of Parikshak into a more Web service complaint structure. These two will significantly enhance the scalability and extensibility of the Parikshak system.

8. CONCLUSION

In this paper, we have described our approach to reorganise Parikshak under a Web service based service oriented architecture to enhance its accessibility and scalability and to make it interoperable with other tools. As discussed in section 4, we are aiming to develop a complex service oriented architecture. Our choice of open source gSOAP toolkit for the implementation proved successful. The gSOAP stub and skeleton compiler provides a good SOAP to C++ language binding for development of SOAP enabled Web services and clients. Overall our experience with Web service technologies when developing the all the three phases and planning the system evolution has been positive.

9. ACKNOWLEDGMENTS

This work was carried out under the Open Source Software Resource Centre (OSSRC) project, which is a joint initiative of IBM, Indian Institute of Technology Bombay, Mumbai (IITB) and Centre for Development of Advanced Computing (CDAC). We also thank OSSRC colleagues for their valuable comments and suggestions on this paper.

10. REFERENCES

[1] Turner, M., Budgen, D., and Brereton, P. Turning Software into a Service. *IEEE Computer,* (Oct. 2003), 38-44.

[2] Bennett, K. H., Layzell, P. J., Budgen, D., Brereton, P., Macaulay, L., and Munro, M. Service-based Software: The Future for Flexible Software, *Proceedings of 7th Asia-Pacific Software Engineering Conference*, IEEE Computer Society Press, (Dec. 2000), 214-221.

[3] Foster, I. Grid's place in the service-oriented architecture, http://www.computerworld.com/softwaretopics/software/apps/story/0,10801,97919,00.html?from=story_kc, November, 2004.

[4] Thomas Erl, Service-Oriented Architcture: Concepts, Technology, and Design, Prentice Hall, 2005.

[5] Robert van Engelen, *gSOAP User Guide*, http://www.cs.fsu.edu/~engelen/soap.html, February, 2006.

[6] *CDAC Mumbai course Handbook*, http://www.cdacmumbai.in/education/coursehandbooks/2005/, 2005.

[7] Petinot, Y., Giles, C.L., Bhatnagar, V., Teregowda, P. B., Han, H., and Councill, I. A service-oriented architecture for digital libraries, ICSOC, (Nov. 2004), 263-268.

Technology succession and open source VLEs

Martin Weller
Institute of Educational Technology
The Open University
Milton Keynes, UK
44 1908 653712
m.j.weller@open.ac.uk

ABSTRACT
In this paper, the academic audience of Virtual Learning Environments (VLEs) is divided in to lead and conventional users. These users have different requirements, but as institutions move towards enterprise solutions for VLEs they are required to use the same systems. Currently commercial VLEs are the most heavily deployed in the education sector, despite some criticisms of the approach these systems foster. Using the analogy of plant succession, it is suggested that commercial VLEs have suitably altered the environment to the extent that open source VLEs can now be deployed. Such systems represent a compromise that can meet the needs of both sets of users. Some case studies that have recently adopted open source solutions are then provided to demonstrate this process.

Keywords
Virtual learning environments, e-learning, learning management systems, open source, technology adoption

1. INTRODUCTION
The seminal work in the adoption of technology is Rogers *Diffusion of Innovation* [11]. In it he describes the manner in which innovations have an S-shaped curve of adoption. The gradient of the curve can vary, depending on the innovation in question (the innovation needn't be technological, but the pattern and categories often apply to the uptake of a technology), and factors such as how much benefit, or commercial advantage it provides, the audience in which it is being adopted, the ease with which it can adopted, and so on. The key point is that there is usually a period where the innovation is used by a relatively small community, and then as it enters the steep part of the curve, uptake increases dramatically. This is also characterized as the 'tipping point', when an innovation gains a critical mass of users. There is also a flattening off of the curve, which suggests that beyond a certain point, a lot of time (and usually resource) is required to increase the number of users.

Similarly, Riggs and von Hippel [10] looked at innovations developed by users and those by manufacturers. They found that innovations by users generally enabled instruments to do new things, while those by manufacturers allowed users to do the same thing but more conveniently or reliably. These two groups again reflect the different demands of the revolutionaries and democrats. Von Hippel [7] goes on to differentiate between 'lead users' and more conventional users. Lead users are often ahead of market trends, and expect to gain relatively high benefits from a solution to the needs they have encountered there. They tend to modify products, and seek out products that can be modified. These modifications in turn benefit others, and the lead users in turn get the most benefit from any modifications.

Virtual Learning Environments (VLEs) may not be the most innovative educational technology to be found in use today, but they are one of the most pervasive, with 86% of respondents from UK HE institutions reporting the presence of a VLE in their institution [4]. This is perhaps why many researchers and educational technologists hold them in something resembling disdain. There are a number of charges often leveled at the more popular VLEs, and particularly commercial ones, which can be summarized as:

- They are content focused
- They have no strong pedagogy
- They are based around a teacher-classroom model
- They combine a number of average tools, but not the best ones
- They do not feature a particular tool
- They operate on a lowest common denominator approach
- They do not meet the needs of different subject areas
- It is difficult to exchange content between them, despite claims to interoperability

There is an element of truth in many of these claims, but the problems they represent are not as catastrophic to successful elearning as many of their proponents suggest.

There is something very familiar about this debate. If one substitutes the word 'Microsoft' for 'commercial VLEs' then many of the arguments sound similar to those leveled at a number of Microsoft products, principally the Windows operating system, but also tools such as Word, Excel and server technologies such as NT. The argument is actually about any large corporation with proprietary software, but it is best embodied in the debate around Microsoft, which acts as a proxy for all such companies. The similarity with the VLE market is strengthened when one considers that Microsoft have recently bought a large stake in the company providing the commercial VLE, Blackboard, and that in 2005 Blackboard acquired the other main VLE company, WebCT. This makes the scenario of a 'Microsoft' for educational software, ie a very powerful provider who has a near monopoly, all the more likely.

The products that serve the majority of any audience that reside in the middle part of the normal distribution curve (Von Hippel's conventional users) are almost, by definition, not the sort of tools that those who occupy the leading edge (the lead users) find interesting and suitable. The key accusation leveled against such products is their lack of flexibility. But flexibility often arises from a deep understanding of how such tools operate, and what they can be extended to do. This level of complexity is unsuitable for the conventional users. And such flexibility often leads to instability in the hands of the less knowledgeable.

2. CURRENT USE OF VLES

A 2004 survey conducted by the Organisation for Economic Cooperation and Development looked at e-learning in tertiary education in thirteen countries and a smaller survey by the Observatory of Borderless Higher Education reveal a good deal about the current situation regarding VLEs [9]. The survey showed that only 37% of respondents had a single institution wide VLE, while the remainder had a mixture of systems, often with one institutional and then a number local versions, although 90% expected to have an institution-wide system in next 5 years. Just over half of the institutions used a proprietary system, often with some open source systems in conjunction.

The move towards institution-wide systems means that increasingly the two audiences identified above, the lead users and conventional users, are forced to cohabit in the same virtual space. The question then is to what extent can a system be found that meets both of their needs?

The OECD survey seems to strengthen the position of commercial VLEs, but this may reflect the history of VLE uptake rather than its future direction. Three factors may see this position gradually undermined:

- Open standards – the development of open standards presents something of a dilemma for commercial VLEs. Customers expect the systems to comply with standards, and yet in doing so the commercial system begins to lose its unique selling point.
- Convergence of functionality – as system converge in terms of functionality, there is little to choose between commercial and open source options.
- Reliability of open source solutions – since 2004 a number of open source solutions have gained momentum to become serious rivals, most notably SAKAI and Moodle.

Perhaps of greater interest is what the survey reveals about the other systems that form part of the wider managed learning environment. Only 6.6% of respondents reported an institution-wide content management system (CMS) while 31% reported an institution wide portal, with a further 24% expecting to implement one within a year. Compare these figures with the almost total adoption of VLEs (only one respondent reported no VLE).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

3. TECHNOLOGY SUCCESSION

What this demonstrates is that VLEs have achieved a level of uptake and penetration that has been rapid, but has not necessarily caused major disruptive changes. Most VLEs seek to match current practice, certainly much more closely than a CMS, which requires a number of contextual assumptions to be in place before it can be put to effective use:

1. Most content is available digitally – this is not too controversial, as all institutions have a good deal of information available digitally, but this is often administrative in nature and may not be the case for the majority of educational material. It may follow then that there is a requirement for all information to be produced in this way.
2. Content is in appropriately sized chunks – without absolutely requiring a learning object approach, for a CMS to be useful it needs to store chunks of learning content that can be aggregated together in different packages (usually courses). The granularity of the resources therefore needs to be suitably small to permit this, which has implications for how academics produce material.
3. Reuse of material is encouraged – while an CMS can be used to create content, and is particularly useful when doing so collaboratively, there is an assumption that the resources within an CMS will be reused in different contexts. If reuse is not part of the culture then the value to the institution of the CMS will be diminished.
4. E-learning plays a significant role in the overall educational strategy – a CMS is an expensive and sophisticated system, which requires a critical mass of resources for it to be worth the investment. This assumes that the CMS is used to support teaching, and is not an institutional CMS for storing and managing mainly internal, administrative documents.

From this perspective then we can ask to what extent can VLEs be seen as a Trojan horse for other e-learning applications and practices that begin to more seriously change the nature of higher education? Portals and CMSs are, arguably, more significant change factors (whether for good or ill), but the VLE can be seen as the *sine qua non* for the implementation of such systems.

There is an analogy with plant succession here. When there is a new environment, for example barren rock, a few pioneer species, such as lichens begin to grow. The acid from these decomposes some rock particles, and their own death creates a coarse soil. This is suitable for mosses, which require little soil, and in turn these decompose to enrich and deepen the soil, until it is suitable for some grasses to grow. The process ends with the establishment of a stable, climax community.

In e-learning terms, VLEs, and in particular commercial VLEs have acted as the pioneer species, moving in to the new environment and creating slight changes which make the habitat suitable for secondary colonizers. Commercial VLEs have done this *precisely* because they match the current model of practice, and match the standard purchasing and support model ie not in spite of the accusations leveled at them, which were set out above, but because of these.

However, in a succession model the role of colonizers is to adapt the environment for secondary colonizers. This is where current open source VLEs now come in to play, as well as closely integrated systems such as portals and eportfolios. The kind of environmental changes wrought by commercial VLEs include general acceptance of the e-learning approach, integration with administrative systems, staff development, recruitment of enthusiasts, changes in assessment practice, acknowledgement of tools already used by students, and so on.

Once secondary systems have been established, then the environment would be more receptive to systems that require more radical changes in practice, such as CMSs and Personalized Learning Environments (PLEs).

There seems to be a good case for the open source approach in the development of educational software. Firstly, there is a natural affinity between the open source and academic communities. The process of contributing code has been compared with that of academic review process. [1] suggest that "you give away your knowledge, not because you are altruistic, but because that is the way of career progression within the academic field. You give away knowledge and information in return for status and reputation. The acceptance of a gift by a community implies recognition of the status of the donor and the existence of certain reciprocal rights. Scientific contributions are gifts, as authors normally do not receive royalties or other payments for the publication of results in a journal…. The open source communities are driven by similar norms. You write a piece of software and provide it to the community. Your contribution is peer reviewed by the owners of a software development project and, if it is good enough, you get your credits in the open source gift economy. A good idea is usable in further research but also gives the owner credits." pg 318

Secondly, many open source contributors are employed in education and many projects start as educational projects, for example Moodle began life as part of founder's Martin Dougiamas' PhD. Thus it would make sense that in the area of VLEs, which are so central to the education process in the 21st Century, that a successful open source solution could be found. It has only been very recently though that open source VLEs have been able to compete with commercial ones in terms of usability and reliability.

With the development of more robust and user-friendly open source solutions, then it is possible that an open source VLE has moved some way towards becoming a mainstream technology and can thus satisfy the conventional users. At the same time, because they are open to development and modification, they still satisfy some of the requirements of the lead users. They thus represent a reasonable compromise between these two audiences, and in succession terms are ideally placed to take advantage of the colonizing work done by commercial VLEs.

4. SOME CASE STUDIES

4.1 The UK Open University

The UK Open University (UKOU) is a distance education university, and often operates with large student numbers, for instance there are around 300,000 registered users on its discussion systems and some courses have cohorts in excess of 10,000. As such, the requirements it has of educational technologies are not the same as though of more traditional, campus-based institutions. This has led to the UKOU developing a history of innovation and implementation of ICT in its teaching materials, but often this has occurred at the course, rather than institutional level. The UKOU thus faced the sort of tension detailed above, most academics had been accustomed to developing their own specific tools, and were acting in revolutionary mode, while as an institution the University recognized the need to make e-learning provision part of the mainstream and to offer a uniform quality of experience for students with regards to the technology they encountered on different courses.

In 2004 the UKOU launched a VLE project. It was in the unusual situation of having developed or bought in a number of tools and systems that commonly constitute a VLE, without having these integrated in to a recognizable VLE architecture. The tools it already possessed were:

- Discussion and conferencing - through OpenText's FirstClass system
- Authentication – handled through an LDAP compliant in-house system that allowed single sign on across all OU systems.
- Template driven content delivery – via and in-house system, Promises.
- Blogging – available on some courses through MovableType
- Audio conferencing – Lyceum, and in house product had been successfully deployed on a number of courses, particularly in languages (e.g. [6]).
- Assignment handling – a large scale system had been developed in-house to match the UKOU's award process.
- Assessment – a combination of QuestionMark Perception and an in-house product, Open Mark, were used, although there was no enterprise solution, and practice varied.

As well as identifying areas where the existing provision could be improved, for example compliance with IMS Content Packaging for content delivery, the systems audit revealed a number of gaps, for example in terms of student tracking. What was perhaps most lacking however was the conception of these components as parts of a larger system.

After an extensive review and consultation process it was concluded that a service oriented architecture (SOA) approach that integrated existing applications and the development (or procurement) of tools to fill existing gaps represented the best option.

However, while SOAs have gained a good deal of attention, there are relatively few examples in operation. One is the Tasmanian LeAP project [8] which uses a service oriented approach to create a flexible VLE. Perhaps the best known of such approaches is the SAKAI initiative (http://www.sakaiproject.org), which aims to deliver the following components as open source.

So, while a SOA represented a good architectural vision and was a worthy goal, there were also more pragmatic needs regarding the timely roll-out of the VLE and also the need to provide clarity to a number of related projects which would be interfacing with the VLE. For example, an eportfolio review was under way and this needed cognizance of the implementation of any VLE and detailed technical integration methodology.

A further review concluded that the adoption of the open source VLE Moodle represented a practical middle-ground between a fully developed SOA approach, and a proprietary VLE. The advantage of the Moodle option were as follows:

- Its existing tool provision allowed the UKOU to shortcut the development of some tools
- The system could be integrated with existing systems
- Access to the source code meant the system could be adapted to our specific needs and to our development plan, rather than waiting for releases.
- It mapped on to the UKOU's strategic directives, particularly that of being a leader in modern pedagogy and technology
- The UKOU could contribute to and benefit from an existing Moodle community.

It is estimated that the adoption of Moodle will reduce the implementation time of a service oriented VLE by 25% in the UKOU. Currently Moodle has been integrated with the existing authentication system, assignment handling and FirstClass. The existing assessment tools within Moodle will be utilized, although in some instances integration with OpenMark will also be deployed. The content delivery, management and navigation functions of Moodle will be adopted wholesale, although with customization to a UKOU look and feel.

4.2 State University of New York

The State University of New York (SUNY) has 64 campuses distributed over New York state. It also offer an extensive online programme through SUNY Learning Network, which has over 100,000 students, 3,000 staff and 40 of the campuses participate. Any VLE system therefore needs to support purely online, blended and campus based education, over a widely distributed system.

In 2005 they embarked on an extensive review programme to find the solution for their next generation VLE (having used the IBM Lotus Notes/ Domino system for a number of years). Their approach comprised four main stages [12]:

1. Assessments and Assumptions – this established the foundational data that would be required of any VLE technology candidates. These reviews included technical and IT environment assessments, assumptions on requirements, and assertions for long-term trends in VLE development. The conclusions from this process were that the current system could no longer meet their needs and that a portal was "the best technology foundation for a modern LMS."
2. Analysis of Task Force Recommendations - a Task Force made recommendations for a single VLE system across all campuses for teaching, learning, and research. The recommendations of that task force were then analyzed in order to form the necessary criteria for evaluating candidates for a new VLE solution. Five key evaluation criteria were the produced for use in the next stage:
    a. Strong support for integration of new teaching and learning tools via open standards.
    b. Student-centric rather than course-centric application design.
    c. Support for the IMS Learning Design Specification.
    d. Native interoperability with SUNY's portal environment.
    e. Strong integration capabilities with campus IT systems
3. Evaluation of Potential Solutions - using both the assessment studies and the analysis of the Task Force recommendations, potential solutions were evaluated. Once a strong solution had been identified, the team prepared an overview as well as a snapshot of a functional specification for production of that solution. The products they evaluated were Blackboard, WebCT, ANGEL, Academus, Moodle, Sakai, dotLRN, as well as the combinations of Sakai + Moodle + uPortal, Sakai + Academus + uPortal and Sakai + LAMS + uPortal. Their final recommendation was for a component approach, which combined uPortal, LAMS and a range of other open source tools, which they believe "Provides a much richer feature set than any currently available single-platform LMS."
4. Implementation Strategy - an implementation strategy for the proposed LMS solution was formulated. This is based around an 'agile' development plan with regular updates and some outsourcing of development.

The SUNY solution is summarized as "a component strategy, as no single-platform LMS solution exists today to meet our needs. This powerful component strategy would integrate several carefully chosen Open Source projects, each with strong technical compatibility, resulting in a whole far greater than the

sum of its parts." This is unusual in a number of respects. Firstly, it places the portal at the centre of the system, rather than a VLE. Secondly, their process places a strong emphasis on the Learning Design specification, with it being one of the five key criteria that was used to determine the final system. This leads us on to the next noteworthy point about the SUNY solution, namely the selection of LAMS as their main VLE tool. While LAMS has gained a lot of attention and been successfully deployed in local contexts, it is rarely employed as the central system. The SUNY implementation will be an interesting test of how well LAMS manages this promotion to centre stage. The last point of note from the SUNY study is the conclusion of a component strategy, as with the UKOU a service oriented architecture was recognised as the optimal solution in terms of pedagogic requirements and flexibility, but existing open source solutions provided a convenient means to achieve this in a short timescale.

4.3 New Zealand Open Source VLE

Moodle was selected by The New Zealand Open Source VLE project to form the basis of their collaborative development. The project is a coalition of twenty tertiary education establishments in New Zealand who have committed themselves to using and developing an open source VLE. This is driven by a desire to share the costs of e-learning development. This made an open source option the most logical choice, so it was not a choice between open source and proprietary but rather a choice between open source alternatives.

Their objectives of the project are [13]:

- Significantly reduce the total cost of ownership at a system wide-level
- Select and contribute to open source communities
- Encourage collaboration and user networks
- Reduce to barriers to entry: technology, support & professional development
- Accommodate flexible pedagogical approaches
- Support localisation - including Maori and Pacific Island languages
- Advocate for interoperability
- Catalyst for innovation

They evaluated three open source options in detail: Moodle, ATutor and Ilias. They used two frameworks for their evaluation: Chickering and Ehrmann's [5] seven principles of pedagogy and technology selection and Britain and Liber's [2] Framework for the pedagogical evaluation of eLearning environments.

They chose Moodle in 2004 because they felt that it offered:

- An open and active community with a critical mass of developers.
- A modular system architecture
- Relatively easy integration with other systems
- A course / student focus rather than tool-centric
- Adaptability

Using Moodle as the basis, each of the participating institutions creates a distinctive and localized version. The second stage of the project is focusing on the development of additional tools such as a personalized portal, personal development planning (PDP) tools, e-portfolio, simulations and instructor support tools.

5. DISCUSSION

As e-learning moves from a peripheral to mainstream activity (e.g. [3]) Higher Education institutions in particular face a difficult dilemma. They need to move to the provision of standard, robust e-learning technologies that are operated institution-wide so that support can be centralized, teaching quality can be audited, staff development programmes can be conducted, resources can be allocated, and so forth. This consolidation of e-learning services however can lead to the alienation of those academics who have developed specific applications and who have very specific demands regarding educational technology. There are two main audiences for the VLE within an institution, the lead users and the conventional users. The requirements of the two audiences are very different, yet they are required to both operate within the constraints of the institutional VLE.

For the conventional users a commercial VLE has thus far been the preferred option as it provides a robust solution, with appropriate support and training material. Such systems have been designed specifically around meeting the needs of such users and although they may be limited in some respects, they provide the main functions within an easy to use framework.

For the lead users, a solution based around a more service oriented architecture is likely to be appealing. This is both technically more interesting and also more flexible so they can develop a best of breed approach, integrating their particular tools as required.

The experience of the case studies suggests that an open source VLE option, represents a compromise between these two options, that can potentially satisfy the needs of both audiences. The system is sufficiently robust, and there is a sufficiently large user community for it not to be viewed as a research tool. It is however also flexible, and adaptable to the needs of any particular institution, so those with specific technical requirements can use the VLE as the 'backbone' of a service oriented solution. Due to much of the work done during the implementation of commercial VLEs, they have ironically, altered the environment suitably so that open source VLEs are now a viable enterprise solution.

It should be recognised though that an open architecture, open source VLE represents only one half of the interoperability equation. The other half relates to content, and being able to populate such environments with a range of content that suits the needs of different learners. In this respect there are a number of important developments. The first is the development of open standards relating to content, such as metadata for describing content, and content packaging for exchanging structured resources. The second important development in this area is that of open educational resources. This work was initiated by MIT's

Open CourseWare project, which aims to make all MIT teaching materials freely available. Many other universities have followed suit, including some in Japan, China and Latin America. The open educational resource movement also builds on the work of learning object repositories, such as MERLOT. This work is potentially significant for open VLEs because it helps to blur boundaries between institutions and aids the personalization process by creating a much wider range of resources to draw upon.

Although the concept of reusing material, particularly in the form of learning objects, has been around for a few years now, such reuse has not been apparent in most institutions. However, as the move towards open VLEs becomes more evident, so the notion of reuse becomes more prevalent. The initial work on reuse and standardization focused on content, for example metadata and content packaging were amongst the first specifications to be produced, while software interoperability is relatively recent, for example with the development of SAKAI and the IMS Tools Interoperability Profile. It may be that tool interoperability is in fact a bigger driver for reuse than content, and reuse of content will follow once open learning systems have been established, because following the concept of technology succession, such systems change the environment in which they operate, in this case making reuse a more acceptable concept.

6. REFERENCES

[1] Bergquist, M and Ljungberg J (2001) The Power of gifts: organizing social relationships in open source communities. *Information Systems Journal* 11 pp. 305-320

[2] Britain, S. and Liber, O. (2004). *A Framework for the Pedagogical Evaluation of eLearning Environments.* http://www.cetis.ac.uk/members/pedagogy/files/4thMeet_framework/VLEfullReport

[3] Brown, M. V. and Galli, A. L (2005) 'E-learning goes mainstream'. *Power* 149:3 pp. 30-39

[4] Brown, T. & Jenkins, M (2003) *VLE Surveys: A longitudinal perspective between March 2001 and March 2003 for Higher Education in the United Kingdom.* http://www.ucisa.ac.uk/groups/tlig/vle/index_html

[5] Chickering, Arthur and Stephen C. Ehrmann (1996), 'Implementing the Seven Principles: Technology as Lever,' *AAHE Bulletin*, October, pp. 3-6. http://www.tltgroup.org/programs/seven.html

[6] Hampel R. and Hauck, M. (2004) 'Towards an effective use of audio-conferencing in distance language courses' *Language Learning & Technology.* Vol. 8(1), January 2004, pp. 66-82

[7] Hippel, E. v (2005) *Democratizing Innovation*. MIT Press, Cambridge, Mass.

[8] LeAP Project Case Study (2004) *Implementing Web Services in an Education Environment*. Version Date: 20 July 2004. http://www.education.tas.gov.au/admin/ict/projects/imsdoecasestudy/LeAPProjectCase.pdf

[9] OECD (2005) 'E-learning in Tertiary Education: Where Do We Stand?' *Education & Skills 2005*, vol. 2005, no. 4, pp. 1 - 293

[10] Riggs W. and von Hippel, E. (1994) Incentives to innovate and the sources of innovation: the case of Scientific Instruments" *Research policy* 23 (4) pp 459-469

[11] Rogers E., (2003), *Diffusion of Innovations*, 5th Ed., New York, London The Free Press.

[12] SUNY http://le.suny.edu/sln/rpc/sln2tsr.pdf

[13] Wyles R (2004) 'Evaluation of Learning Management System software. Part II of LMS Evaluation'. *Open Source e-Learning Environment and Community Platform Project* http://eduforge.org/docman/view.php/7/17/Evaluation%20of%20LMS%20-%20Part%20II.pdf

Impressum
NewMinE - eLab
Università della
Svizzera italiana
Via G. Buffi 13
6904 Lugano
Switzerland
www.newmine.org
www.elearninglab.org