THE FULLY DISTRIBUTED ARCHITECTURE OF VIRTUAL NETWORK LABORATORY

Petr Grygárek, Martin Milata, Jan Vavříček
Department of Computer Science, Faculty of Electrical Engineering and Computer Science
Technical University of Ostrava, Tř. 17. listopadu, 708 33 Ostrava, Czech Republic
Tel.: (+420) 597 323 263, Fax: (+420) 597 323 099
E-mail: petr.grygarek@vsb.cz, http://www.cs.vsb.cz/grygarek

5th Int. Conference on

Emerging e-learning Technologies and Applications

The High Tatras, Slovakia September 6-8, 2007

Abstract. In this article we present our implementation of an advanced task-based fully distributed virtual network laboratory management system which supports on-demand variable topologies built from laboratory devices located at multiple sites connected via Internet. Each site can act as an independent virtual labortory or share its' equipment with others. Available laboratory devices suitable for distributed virtual topology are searched dynamically when the task reservation is being requested, so that multiple topologies can be reserved in parallel. The distributed nature of the resulting system is completely hidden to the user. The architecture incorporates both hardware and software components and is completely based on open-source technologies.

Keywords: Virtual laboratory, Communication Technologies, Distance education.

1. INTRODUCTION

The practical work with networking devices is a necessary part of every networking course focused on developing real knowledge of computer network building and maintenance. For that reason, institutions providing education on computer networking field effort to build well-equipped networking laboratories and provide as much time as possible for students to access laboratory devices. Unfortunatelly, professional-level networking devices are often very expensive and cannot be made available to public access without supervision. On the other hand, the laboratory is most often idle during work-off hours, which makes investment to laboratory equipment inefficient.

For these reasons, we decided to implement a system called Virtlab, which allows to access laboratory equipment remotely via Internet. The architecture and experiences with our system will be described in the following article.

2. THE BRIEF HISTORY OF VIRTLAB PROJECT

Our work started three years ago. We decided to build the whole system using open-source technologies only, which promised to limit implementation cost considerably. The very first goal was to implement remote access to laboratory devices' consoles and appropriate web-based reservation system [1][3][4], as shown on figure 1. On the client side, we use standard Web browser with Java applets support to provide of terminal windows which simulate physical terminals connected to individual networking devices' consoles. Linux, PHP, MySQL, C and Java technologies were and still are used as a technical platform.

The original reservation system was designed so that we were able to offer particular tasks for reservations at individual fixed-length timeslots. Our primary goal was not just to make devices' consoles accessible remotely in reserved timeslot, but to provide a set of meaningful tasks prepared by experienced networking teachers, which will direct students to particular topic they can experiment with. The decision of establishment of task-oriented learning system resulted from our previous experience that without a such guidance, students commonly don't have an idea what they could try to do with the laboratory equipment.

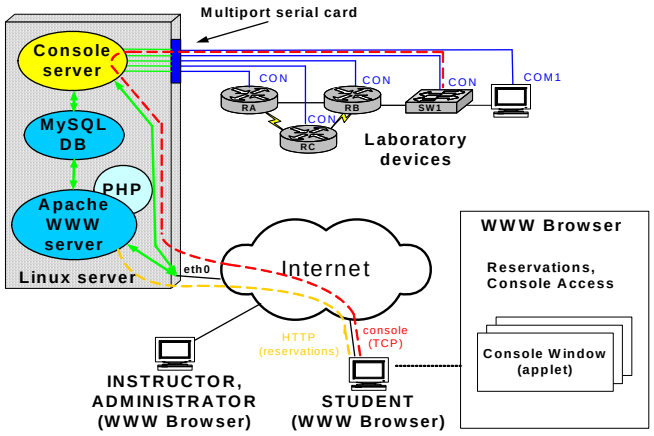

Fig. 1. The Basic Remote Access Architecture

From the beginning, we understood that it will be necessary to provide tasks with various topologies. Our first idea, that we will connect some topology manually, provide a set of continuous timeslots to access tasks on that topology and then change the topology to another, proved very unrealistic to manage. This is why we soon developed a concept of Virtual Crossconnect which allows us to connect required topologies automatically (fig. 2).

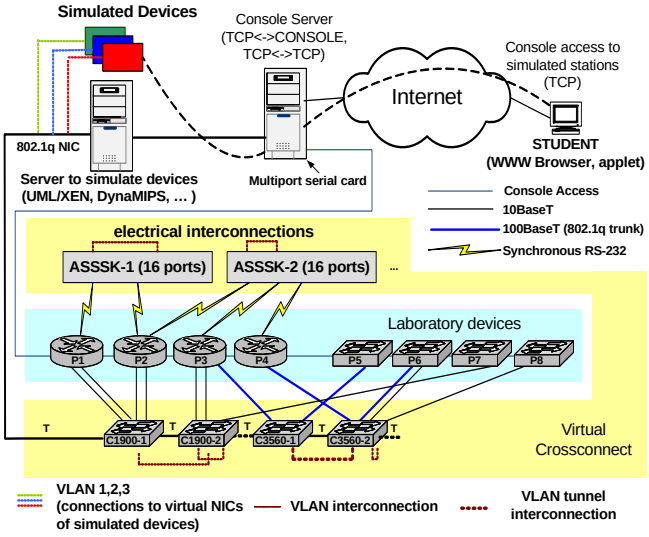

Fig. 2. The Virtual Crossconnect

The crossconnect is called virtual because it uses multiple crossconnect switching elements of various types to interconnect network devices' ports, but behaves like single entity from the point of view of the rest of the system. Very simple language was developed to describe required topologies. Those descriptions are stored together with specifications of individual tasks. Based on the topology description and description of the (fixed) interconnection between laboratory devices' interfaces and virtual crossconnect ports, we are able to generate configuration for every virtual crossconnect switching element and upload it to that element using either Ethernet or RS-232. The Virtual Crossconnect configuration upload is accomplished at the begginning of each reserved timeslot according to topology description required by the task to be scheduled at the ongoing timeslot (fig. 3).

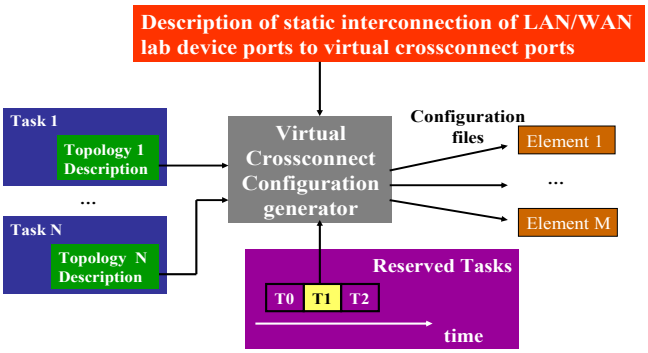

Fig. 3. Generation of Configurations for Virtual Crossconnect Switching Elements

We use VLAN-based approach and standard LAN switch to interconnect laboratory devices' Ethernet ports and our own hardware device called ASSSK-1 [5][2] to intereconnect serial ports (fig. 4). While developing the crossconnect for serial WAN ports, we focused on synchronous RS-232 interface, because it requires least number of signals to be crossconnected. ASSSK-1 behaves as DCE, i.e. provides clocking for all serial ports. The core of ASSSK-1 is composed of the analog switch array and ATMEL microprocessor which accepts commands from RS-232 controlling console and configures analog switch array accordingly. The command set to control the interconnections is very simple and inspired by IOS. The device is modular, so that other electrical interfaces may also be implemented if necessary.

Fig. 4. Our Implementation of Hardware-based Crossconnect for Serial WAN Ports

At the first implementation of ASSSK-1, we also tried to switch Ethernet ports, but it proved inefficient because of frequency limitation of available analog switch array circuits. Only 10BaseT ports can be switched by ASSSK-1. This is why we later decided to use VLAN-based interconnection using standard VLAN-aware 10/100/1000 Ethernet switch. The usage of VLANs is also advantageous for connecting of real networking devices with simulated ones, as shown on figure 2. Currently, we use XEN [9] to simulate stations and DynaMips [10] to simulate Cisco 7200-series routers. For interconnection of Ethernet trunk links, VLAN tunnelling technique [11] (also called QinQ sometimes) proved very useful. Using Cisco Catalyst 3500-series switch, we reached almost complete invisibility of the crossconnect switching element to laboratory devices, so that we can operate most of layer 2 service protocols like STP, CDP or LACP/PAgP between those devices transparently.

Because the implementation of Ethernet ports switching using LAN switch proved so efficient, we decided to simplify ASSSK-1 architecture so that it will only switch serial WAN ports. To make the device more replicable and decrease it's cost we also abandoned the modular approach and used FPGA technology to implement it with much more efficiency. The FPGA-based switching core is now fully digital. The prototype (called ASSSK-2 [8]) is currently under testing and all tests appear to be successful.

Another hardware device we are now working on is the FPGA-based multiport HDLC/PPP card for Linux-powered PC. Our aim is to experiment with WAN port switching using standard PC, which we expect to be both cheaper and more extensible. We started to work on tunnelling of serial links traffic over UDP, which allows us to create virtual WAN links over Internet, as will be explained later. The reason why we decided to develop our own HDLC/PPP card instead of buying the commercially available one is the low port density and high price of cards on the today's market.

3. MAJOR CHANGES IN THE ORIGINAL ARCHITECTURE

After the full implementation of virtual crossconnect, we realized the full power of it and we decided to change the overall philosophy of offering particular tasks at fixed-size timeslots with a schedule specified by Virtlab administrator. Our experience revealed that it was very difficult to guess how many timeslots should be each individual task made available for reservations. Very often nobody reserved timeslots in which an uninteresting task was offered, but on the other hand there was a contention for timeslots of another more interesting task. It led to suboptimal utilization of virtual laboratory.

The implementation of Virtual Crossconnect allowed us to redesign the architecture completely so that it is no longer necessary to create weekly task schedules and place them on the electronic noticeboard, where students can reserve particular timeslot with required task [6]. Today, we have almost finished the new version of the reservation system, which allows student not only to specify arbitrary timeslot, but also any task he or she wants to work on during the reserved time. Before the reserved timeslot, the virtual crossconnect simply interconnects the topology for any task that the student required for the timeslot.

The most important architecture change necessary to develop an efficient system of virtual laboratory usable by multiple students in parallel was the decoupling of logical network devices' identities specified in topology description for particular task from physical identities of laboratory devices actually used to interconnect the topology for a particular reservation. It allowed us to reserve the same task multiple times in parallel, provided that there are enough laboratory devices with required capabilities to act as logical network devices prescribed in the task's topology definition. In the previous architecture version, topology description were tightly bound to physical devices' identities so that it was not possible to reserve multiple tasks whose specifications overlapped in physical devices used, as shown in fig. 5.

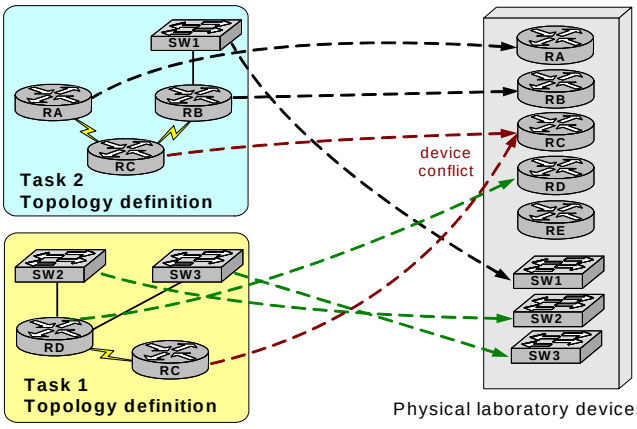

Fig. 5. Problem of Device Conflict in Case of Fixed Physical Devices Specification in the Task Description

By decoupling of logical identities of network devices in the task's topology description from physical ones, we only prescribe required features of individual network devices and their interconnection in the description of task's logical topology and map logical devices to physical devices dynamically at the reservation time (fig. 6). The mapping is based on the knowledge which physical devices are available at the time interval required by user for the reservation and on further limiting constraints, such as OS version or feature set supported.

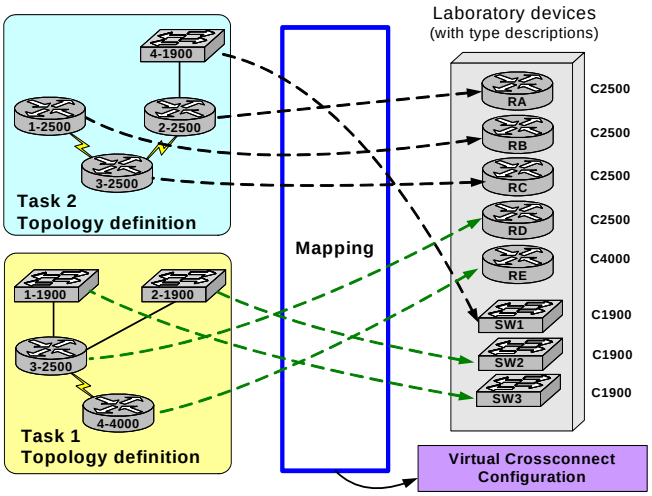

Fig. 6. Separation of Logical Task's Topology Description and Physical Devices Actually Used for Reservation

To implement the above mentioned principles, the original reservation system has been completely reworked. The new implementation of reservation and control system core was also created with regard to multilanguage environment, All menus and messages may be easily translated to any language utilizing UTF-8 charset and timezone. Each user may set it's own preferred language. We are now assessing means how to provide taks's specification in multiple languages and to present the specification to the user in the language according to his/her preferred language setting.

4. THE NEW DISTRIBUTED ARCHITECTURE

After a period of successful operation, we decided to extend our architecture out of scope of our institution. The primary goal was to develop a truly distributed virtual networking laboratory, which will allow sharing of laboratory equipment from multiple sites transparently and building of arbitrary topologies even between devices at different sites using Internet tunnels (fig. 7).

In the new distributed architecture, we still maintain the task-oriented reservation philosophy. The only extension which came almost for free was the implementation of a feature which allows advanced students to specify their own topology they want to interconnect in the reservation time if no exiting task filfills their current needs.

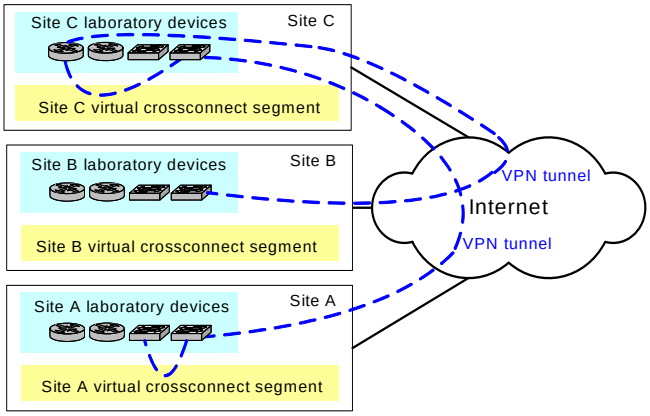

Fig. 7. Virtual Topologies using Internet Tunnels

We perceive the following main advantages of implementation of a distributed architecture:

- Individual sites may specialize on particular networking technology which they will make available to all distributed Virtlab participants. It is very desirable for costly devices, which would not be used efficiently by limited number of users of a single site.
- It is possible to build a large-extend simulated WAN environment and teach students how to operate it. This will allow students to develop competences required in real industrial environment operation.
- Other sites may lend their devices as a spare if some device at any site should fail.

The requirement of full laboratory device distribution brought two problems to solve. We need to be able to create interconnections of interfaces of network devices at different sites via Internet tunnels and we also need to generalize mechanism of console access to be able to access equipment at multiple sites seamlessly. All this stuff has to be completely hidden to user, so that the student just uses the devices without care in what site each device he or she uses is actually located. The virtual topology between devices of multiple sites has to behave the same way as a single-site topology.

The interconnection of laboratory devices' interfaces between different sites is implemented using extended version of Virtual Crossconnect. The original concept was generalized to the distributed crossconnect architecture, which we now call the Distributed Virtual Crosconnect (DVC). The Ethernet port interconnection is still VLAN-based, but frames can be now passed between multiple "segments" of the DVC using UDP and our own implementation of tunnelling server. Currently, only Ethernet ports (including trunks) may be inteconnected between sites using the DVC, but we are working on solutions which will allow us to tunnel HDLC or PPP frames over Internet tunnels also.

Concerning generalized concept of console access, our decision was to make users' terminal applet first connect to console server of the user's home site, which authenticates him or her using local authentication infrastructure. Only after successful authentication, local console server proxies the console connection to console server of site where the laboratory device is physically located. There is a trust between virtual laboratory components at individual sites (see section 6), so that the target site console server justs permits the connection to pass in. IPSec tunnels are implemented to secure communication between sites, so that external intruders are not allowed to get into the system from the outside.

Although the architecture is distributed in it's nature, the virtual laboratory at each site may be utilized without dependency on other sites. We completely avoided any centralized entities which could create a single point of failure. It was also necessary to take into account political issues, so that administrator of laboratory at every site is able to specify which devices he/she is willing to lend to users of particular other site at given time.

5. COMPONENTS OF DISTRIBUTED ARCHITECTURE

The basic components of our distributed virtual laboratory architecture, which are present at every site, are shown on figure 8. The figure also shows basic interaction between components, as will be shortly discused later.
At every site, interfaces of all laboratory devices are connected to the local segment of DVC, either physically or via trunk links (in case of simulated devices). The Tunnel Server extends the local VLANs used by DVC for laboratory devices' interface interconnection across Internet tunnels to other sites and also bridges VLANs if it is necessary to interconnect simulated devices connected to DVC using fixed VLANs. The Console server accepts connections to device consoles from the user's GUI (applet), authenticates them and forwards them either to consoles of local laboratory devices or to other sites, depending on the location of the target device. The local device's console may be reached either by RS-232 physical connection (Console Server incorporates multiple MOXA multiport serial cards) or via another TCP connection in case of simulated devices listening for console connection at preconfigured TCP port.

The Reservation Server serves in two ways. At first, it keeps tracks of local devices' reservations, requested by users of it's own and other sites so far. Second, the reservation server acts as a proxy for site's Control Server (also called Virtlab Server for historical reasons) to negotiate reservations of devices with Reservation Servers of other sites. If an user wants to make distributed topology reservation, he/she first connects to the Web GUI of Control Server of the site he/she belongs to. After the task to be reserved and and time interval is choosen, the Control Server attempts to map task's logical topology to the physical topology. The Control Server first requests the local Reservation Server to provide it a list of devices available at any site for the requested time interval.

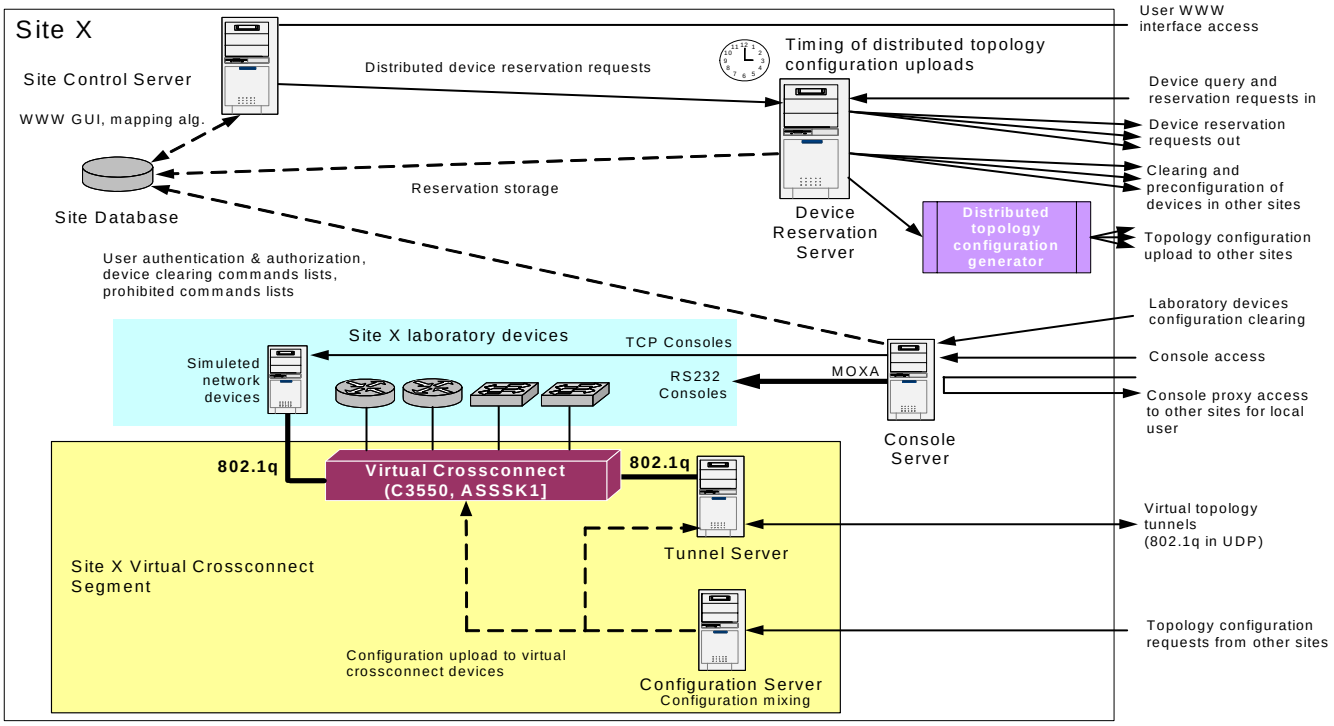

Fig. 8. Components of Distributed Architecture Present at Every Site and their Interactions

The Reservation Server queries Reservation Servers of other sites and those servers respond with a list of equipment which is free at required time and is allowed to be lent to the requesting site. After the Reservation Server returns the combined response to the Control Server, the Control Server starts a mapping algorithm to find suitable physical device for each logical device in the task's logical topology description. If the mapping is successfull, local Reservation Server is requested to negotiate the reservation of previously offered devices with Reservation Servers of other sites. If the reservation was successful, the resulting mapping is stored into the local database at the requesting site. The logical topology description is converted to physical topology description, which takes into account physical identities of selected devices and is passed to the local Configuration Activator, which is a part of Reservation Server. The responsibility for creation of configurations for switching elements of DVC and uploading of these configurations to all affected switching elements is given to the Configuration Activator. At the beginning of each timeslot reserved by some local user, the Configuration Activator gets the physical topology description from the database, launches script that generates configuration of DVC switching elements and uploads these configurations to DVC segments of it's own and other sites using their Configuration Servers.

The Configuration Server of each site accepts topology configuration requests from other sites, combines them together and with the current configurations of DVC switching element's at it's site and uploads new configuration to those switching elements. As we use IOS-powered Cisco Catalyst 3550 and ASSSK-1 with the similar configuration philosophy, our current switching elements combine previous and new configurations by themself, since the CLI IOS-style configuration is "incremental" by it's nature.

The last but not least component of the system is the Cleaning Server (not shown at fig. 8, which is somewhat simplified). This server is responsible for clearing of laboratory devices' configurations before they are made accessible to students at the beginning of reserved timeslot. We also plan to use the server to upload preconfigurations of individual laboratory devices before beginning of some tasks in the future.

The communication protocols between components were designed as text-based, HTTP-style request/reply transactions and are well documented. XML is used as data format whenever possible. This makes the whole system much more extensible and easier to debug.

6. THE SYSTEM SECURITY

Based on the 2-year experience with extending and maintenance of previous (i.e. non-distributed) version of Virtlab, we decided to implement security by a strictly layered approach. It proved inefficient to implement and maintain our own security mechanisms and it also resulted to poor readability and clarity of source code of the core Virtlab mechanisms, which increased a risk of implementation errors. This is why we decided to implement only pure functionality in the first distributed architecture implementation stage and rely on external security technologies whenever possible.

The security paradigm is based on two principles. The first one is that users are authenticated at their "home" site, where sets of roles assigned to individual users are maintained to authorize their activities. Either passwords stored in the local database or other external authentication systems operated at the site (such as LDAP or RADIUS) may be used. The other general idea is that components of Virtlab installations at individual sites completely trust themselves. IPSec tunnels will be created between individual sites so that no external intruder can neither interact with any server at any site nor to forge traffic between laboratory devices passing through Internet tunnel. Only well-secured HTTPS-based WWW user interface and console server at each site is exposted to the public Internet. Concerning measures against potential local intruders, source IP/MAC address filters are planned to be implemented at all hosting systems so that only allowed communications between system components will be permitted.

7. CONCLUSION

The implementation of distributed architecture presented in the article is now almost completed. We are now entering the early testing stage. During summer 2007 we plan to establish a piloting environment between two sites – VSB-Technical University of Ostrava and Karvina site of Silesian University in Opava. A part of laboratory equpment for piloting environment was provided by Czech Scientific and Educational Network (CESNET) as it's portion of expenses on the project number 213/2006. The current running Virtlab environment can be accessed at http://virtlab.cs.vsb.cz.

8. REFERENCES

[1] Grygárek, P., Seidl, D., Němec, P.: Enabling Access to Equipment of Computer Network Laboratory for Practical Trainging via the Internet. Proceedings of Technologies for E-Learning conference, FEL ČVUT Praha, 2005, ISBN 80-01-03274-4, pp. 43-52. [In Czech]

[2] Seidl, D., Grygárek, P.,: System for Automated Network Topology Management. Seminar on Opensources Solutions in Computer Network III. Silesian University Karviná, 2005. Presentation available at http://www.cs.vsb.cz/vl-wiki/images/1/1c/ASSSK-SLU.pdf. [April 2007] [In Czech]

[3] Grygárek, P., Seidl, D., Němec P.: Virtual Network Laboratory for CNAP. Annual Conference of Cisco Networking Academy Program, Brno 2005. Available at http://www.cs.vsb.cz/vl-wiki/images/a/ad/Virtlab-prezentace-CNAP-Brno.pdf. [April 2007]. [In Czech]

[4] Němec, P.: Virtual Network Laboratory. Master's Thesis, Faculty of Electrical Engineering and Computer Science, VŠB-TU Ostrava, 2005 [In Czech]

[5] Seidl, D.: System for Automatic Network Configuration Management. Master's Thesis, Faculty of Metallurgy and Materials Engineering, VŠB-TU Ostrava, 2005. [In Czech]

[6] Grygárek, P., Practical Experience with Implementiation of Virtual Computer Network Laboratory and Proposed Ways of its Further Development. Proceedings of Technologies for E-Learning conference, FEL ČVUT Praha, 2006, ISBN 80-01-03512-3, pp.58-68. [In Czech]

[7] Kubín, R.: Ensurance of Security and Implementation of New Features of Virtual Laboratory System. Master's Thesis, Faculty of Electrical Engineering and Computer Science, VŠB-TU Ostrava, 2006. [In Czech]

[8] Sedlář, P.: FPGA-Based Crossconnect for Serial Lines. Master's Thesis, prepared for publishing at Faculty of Electrical Engineering and Computer Science, VŠB-TU Ostrava, 2007. [In Czech]

[9] The Xen™ virtual machine monitor. Available at http://www.xensource.com/ [online, April 2007]

[10] Dynamips Cisco router emulator, Dynagen. Available at http://dynagen.org/ [online, April 2007]

[11] Cisco Systems: Configuring 802.1Q and Layer 2 Protocol Tunneling. Available at http://www.cisco.com/univercd/cc/td/doc/product/lan/c3550/12112cea/3550scg/swtunnel.pdf [online, April 2007]

THE AUTHOR(S)

Petr Grygárek (Ph.D, MSc.) is a professor-assistant at Department of Computer Science at VŠB-TU Ostrava. His professional interest is focused on computer networking, distributed systems and computer hardware. He is a coordinator and instructor of Regional Cisco Networking Academy and coordinator of Virtlab development group. He holds CCNP, CCNA, CCAI and Network Security courses teacher's certificate.

Martin Milata (MSc.) is a Ph.D. student at Department of Computer Science. His thesis is focused on problems of routing in mobile ad hoc networks. He is interested in computer networks and hardware. He studies the 2nd semester of CCNP-level Cisco Networking Academy Courses.

Jan Vavříček is a MSc. student at Department of Computer Science. His diploma thesis is focused on the distributed virtual laboratory presented in the article. He is an instructor of AutoCont Training Center. He currently studies the last semester of Cisco Networking Academy Courses at CCNA level.