Content management and architectural issues of a remote learning laboratory

Markus Wulff Patrick Lauer Torsten Braun

University of Bern, IAM
Neubrückstrasse 10
3012 Bern, Switzerland
{mwulff|lauer|braun}@iam.unibe.ch

Abstract

Internet-based distance learning requires an efficient development and management of learning material. It should be re-usable and it is necessary to keep the content in an open and portable format.

The infrastructure is expected to be available day and night. Users may access the experimentation setup from every Internet-connected work place at any time. Therefore, reliable hard- and software configurations must be used, which demand a low management overhead at the same time.

This article presents concepts and implementation that solve these issues for a remote hands-on networking laboratory. Concepts for reducing the management overhead, and increasing the availability of a remote laboratory through virtualisation are presented. Furthermore, a content formatting tool to support sustainability of the course material is introduced.

1. Introduction

Remote teaching and distance learning are of increasing importance for the education not only at universities. Online learning modules are an integral part of many courses and complement the traditional lectures. Such learning modules contain text, graphics, animations, and hands-on exercises via remote laboratory access.

The advantages of this blended learning approach are manifold. The students can access the online material from every place where they have an Internet connection and are not limited to fixed laboratory schedules. Furthermore, full online courses can be provided for students of remote learning universities or similar institutions. The students can read the relevant texts and do the laboratory exercises on-line, which provides a much higher flexibility. Even sharing courses with other institutions or providing access to other user communities is possible with an appropriate infrastructure.

In recent years, remote learning modules have been used to complement the lectures for the Bachelor and Master program at our institution. Most learning modules and infrastructure components have been developed during several e-learning projects [4, 17]. Due to a modular course architecture, adding new modules to the courses is fairly easy. The module repository can be completed step-by-step, and up-to-date research results can be incorporated.

In order to ensure sustainability of the course material, it must be updated from time to time and stored in an appropriate repository. Updates should be done only in this repository to prevent inconsistencies between the stored content and the content published in a Learning Management System (LMS). Furthermore, the text sources must be kept in a generic format, which is independent from the LMS used.

We do not host a LMS at our institute to keep the course content for the theory part—it has been outsourced. However, the laboratory infrastructure is installed and maintained in-house. The general course architecture is shown in Fig. 1. It has slightly changed during the years. The local access control system, which was used in earlier days, has been replaced by a Swiss-wide Authentication and Authorisation Infrastructure (AAI) [1, 13]. AAI is an attribute-based access control which enables a single sign-on system where the user is authenticated once at his or her home organisation. Then, checking authorisation of requested resource accesses is based on the user attributes the users home organisation provides. Thus, all users from any AAI enabled institution can register for an AAI enabled course by using their home organisation's account instead of getting a new user account locally.

With the reservation system, students can book time slots for each of the laboratory modules. If they want to access a module with a limited number of hardware devices, they get authorised by the reservation sys-

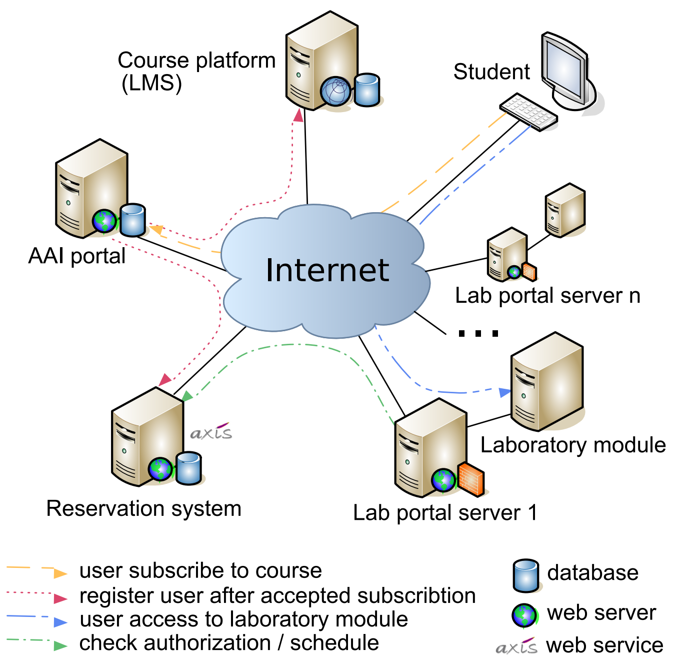

Figure 1. Current architecture of the e-learning laboratory infrastructure.

tem if the module was booked before by the respective user. The reservation for a certain module can be done over a Web based interface.

In the laboratory architecture, hands-on session modules are connected by a gateway that is called portal server. Portal servers connect the module specific hardware on one side and the user's computer on the other side. When the user wants to access the laboratory exercise, he has to authenticate himself at his AAI home organisation, which then sends attributes of the user to the portal of the respective resource. Based on this information the portal decides whether or not to grant access to the laboratory. Usually, there is one portal server per module but it is also possible to connect more than one module to a portal server. Any device that has an interface to a computer (for example a serial interface) can be connected to a portal server.

The hands-on experiments in our laboratory are not only used by local students. Other universities use the infrastructure as well for their education. This requires a high availability of the laboratory equipment. Not only the experimentation devices but also the portal servers and the reservation system must be up and running at any time. In contrast to these requirements, the administrative overhead running the exercise laboratory must be kept as low as possible.

In this paper, solutions for the issues of content management and laboratory architecture are discussed. In Section 2 a short overview about course content management is given. Furthermore, a tool for content formatting is introduced, which can convert learning module content into a specific format required by the LMS used. Section 3 then discusses architectural improvements of the laboratory infrastructure to solve the availability and management issues mentioned above. Finally, a short summary concludes the paper.

2. Online course content management

2.1. Overview and related activities

Several learning modules have been developed for our e-learning projects. One of the most important design goals of these modules is sustainability. In many cases the modules explain fundamentals of computer networks, operating systems or other computer science basics. Therefore, they are not subject to frequent updates and the content can be used for lectures over many years. Only smaller updates and corrections are performed if necessary.

The creation of such a learning module usually takes several weeks. Text must be written, graphics and animations must be created. It is obvious that this content should be stored in a Learning Management System (LMS) independent way to ensure that it can be used with any LMS. This especially applies if the LMS is hosted by a third party as it is not guaranteed that the current product is supported over the next years. In the case that a course must be migrated from one LMS version to another or, worse, to another product this will cause a lot of time consuming manual work.

The problem of storing e-learning in a reusable way has been already discovered several years ago. Different solutions have been proposed. The Sharable Content Object Reference Model (SCORM) [12] defines

a specific way of constructing Learning Management Systems and training content so that they work well with other SCORM conform systems. The common goals of different versions of SCORM are as follows: 1) packaging content and 2) exchanging data at runtime. The drawback of SCORM is that it is not supported by every LSM or the older versions of this standard only.
Many academic and other institutions are trying to make better use of networks and databases to efficiently and effectively achieve learning goals. One of the possible ways to go is to make learning resources accessible to educators and learners through learning object repositories (LOR). LORs are repositories, which organise reusable learning objects like courses, modules, but also images, videos and text documents in a clearly arranged way. To enable search engines to efficiently identify learning objects, a descriptive set of metadata is assigned to every object in the LOR. The goals here are re-usability of e-learning content, long-term archiving as well as to share teaching activities and make them visible to peers and to the public. Examples for LORs initiatives are the Switch LOR [9] in Switzerland or the eduSource project [8] in Canada.
If an appropriate LOR is provided in Switzerland we will consider using it. Not only on Swiss level such initiatives are on their way. A local LOR is also being developed at the University of Bern but this will depend on a specific LMS that we are not using today. This again shows the necessity of maintaining learning content in a platform independent way to be prepared for the frequent changes in today's e-learning landscape.

2.2. Content management

A satisfying solution for the exchange problem of learning objects between different content management platforms is not available until now. As stated in the Edutech LOR feasibility study [9], the interoperability is not guaranteed even if standards compliant content packages in SCORM format are exchanged. The integration of WebCT Vista LMS [3]—which is currently used for our courses—with any LOR causes problems because the API only allows to export bare content files without any course structure or other important metadata.
From today's perspective it is advisable to store our learning modules in a generic format and to set up a simple but efficient content management. The solutions to this content management issue should meet the following general conditions:

- The existing text sources can be integrated without significant modifications.
- New content can be created with tools that are available in our institute and people are familiar with (WYSIWYG HTML editors, word processors with HTML export capabilities etc.).
- It must be possible to create the content format required by a specific LMS on demand.
- The format of the learning content allows a later adaption to e-learning standards if required.

In order to be able to synchronise the access to the repository and to track modifications, the use of a version control system is advisable. Several solutions exist for this purpose. Subversion (SVN) [7] is one possibility here. It manages files and directories, and the changes made to them, over time. This allows to recover older versions of the repository data, or examine the history of how the data changed.

2.3. Content formatting

At an early stage of our learning content development it became clear that a tool for automatic content formatting is necessary. The goal was to reduce the effort needed to develop the learning content and to automatically create the pages for the LMS in an appropriate format. As a result the FFGF (file framework generator and formatter) [16] has been developed.
The FFGF tool is able to create the required document structure for a learning module including the header and footer for the pages. Furthermore, predefined templates can be provided for sections with general information, which are automatically included in the document. In a first step, the author has to create the table of contents for the module. This includes the definition of the time a learner should approximately spend for the sections. The time information is used to generate the module schedule to give the learner a clear overview over the time limits intended by the author. Figure 2 shows a sample schedule. In a second step, the FFGF tool creates the document structure by generating a separate file for each section and inserting the text from the templates provided. Now, the module author has to fill in the content into the section files (text, pictures, animations). This is done by using HTML as markup language. In the last step FFGF is run again and generates now the output, which can directly be uploaded to the LMS. The tool currently supports the format for WebCT CE and WebCT Vista.

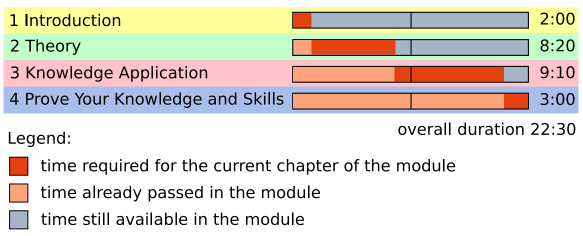

Figure 2. Course schedule example.

After several years of using FFGF we found some weak points in its architecture. The main problem is the fixed output format. It would require major changes in the program to adapt its output generator to another LMS document format. Another issue to be improved is the predefined document structure. This impedes subsequent changes to the document.

Due to the variety of mandatory changes to the current FFGF version, a complete redesign of the tool became necessary. The main improvements of the new programme version must be

- Easier adaptation to different LMS platforms
- Portable solution, usable by other institutions
- Better support for document management
- Simpler document syntax for easier document writing

Therefore, one of the main features of the new FFGF2 is its clear modular architecture. For the module author the most important feature is the automatic document structure detection and creation of the table of contents. It will no longer be necessary to define the module's table of contents in advance. The learning module content can be written into either a single file or split into several source files.

By keeping HTML for the markup of the text the writing of learning content can be done with a variety of available editors. The language is well known and provides all necessary features like different font styles, document structure and allows to include figures and animations. It is not the goal to provide yet another text/HTML editor nor an integrated development environment.

Figure 3 shows the architecture of FFGF2. The first layer represents the HTML document. For some sections, like generic introductions, templates are provided. The content of these sections is the same for every module and does not need to be rewritten every time. The HTML processing engine analyses the document structure and generates a XML [5] formatted document. At this point the templates are integrated into the document.

In the next layer the document is stored in XML format. This allows the content to be kept in a generic manner and simplifies the conversion to other formats. The XML format seems to be the most useful format regarding a later adaption to e-learning standards and the inclusion of learning object metadata.

The last component of FFGF2 is the output formatter. It has a generic interface combined with a plug-in mechanism. The respective plugins to create a certain output format can be attached here. Future changes in the output format do no longer require changes in the FFGF2 core. If a learning module must be formatted for a new LMS architecture or other document formats like PDF all necessary code is encapsulated in the output plugin.

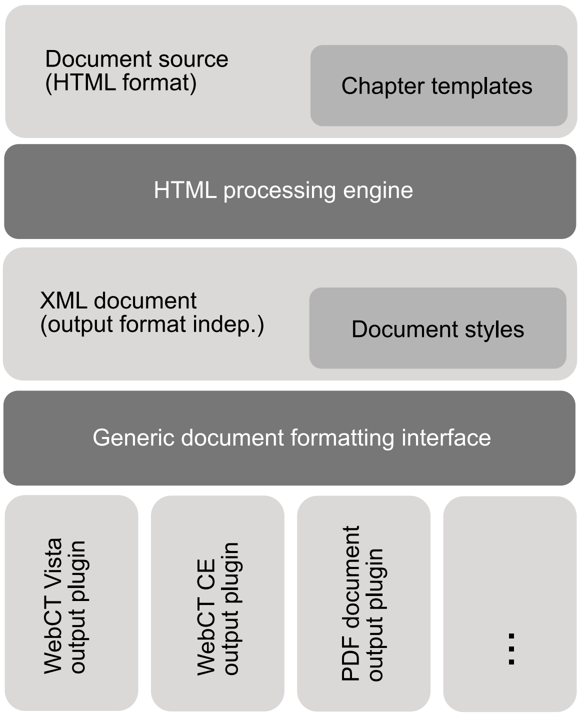

Figure 3. Architecture of the content formatting tool.

Once the document has been generated it is uploaded to a LMS like WebCT. A learner who has registered for the specific course is able to read it there, see the included pictures, watch the animations etc. The learning module document usually contains the description of the hands-on experiments as well. The laboratory exercises, however, are not stored on the LMS. They are located in our laboratory. A link in the document leads the reader to the laboratory computer for the respective experiment.

3. Laboratory infrastructure

As already described in the introduction, we currently have one machine with a central reservation system. Users can reserve timeslots for all instances of learning modules from this central location. Every learning context has its own category. Multiple portals can be part of one learning context so that multiple instances of the same learning module can be provided in parallel. Every portal is a website that offers access to the server "behind" it, usually through Java applets. Depending on the learning module this can be a shell on a machine through a SSH terminal emulator applet, a management console on a Cisco router, an editor applet for programming exercises or any other tool that can run in a Web browser.

3.1. Current architecture

In our laboratory setup, independent hardware for different learning modules is used [2, 19]. This has the advantage that changes on one module do not affect most of the others, but this comes at the price of administrative overhead and underutilised hardware. We had about 20 machines dedicated to the e-learning infrastructure, many of them completely idle most of the time—most activity is concentrated on the later part of the semester, so for about four months of a semester the machines are not used at all. Still the machines are powered on most of the time to provide access to all potential users.
Recently, multiple small hardware failures occurred in the laboratory. Because of the amount of machines the administration requires much time. This combination of underutilised hardware and administrative complexity forced us to reassess the current laboratory setup.

3.2. New laboratory setup

As a possible solution we started evaluating virtualisation methods to consolidate our hardware. This will free hardware for parallel installations as reserves in case of failures. Some other advantages are fast recovery as a virtualised machine can be regenerated from a backup within minutes, easy creation of new machines and, better utilisation of hardware. Because we are using Linux almost exclusively we had a large number of virtualisation options. Among the well-known tools are VMWare (commercial) [15], Xen [18], Virtualbox [14], Linux vserver [10], QEmu, OpenVZ and many more. Out of this rather large number of tools we took Xen, Virtualbox and Linux vserver into the closer evaluation. The features these solutions offer comply with the requirements of the targeted laboratory setup. Virtualbox is a full machine virtualiser/emulator. It can run arbitrary operating systems and has some interesting features like a "virtual" graphics card exported over VNC to allow remote access as if it were a local machine. The drawback is that the performance is quite moderate and it is not as convenient to manage. Xen offers "paravirtualisation" where the guest operating system gets adapted to the virtualisation environment. This limits the available operating systems for the guest systems, but as we were already very much focussed on Linux this was no drawback. The advantage is a comparatively good performance, so the amount of hardware needed should be smaller than with other virtualisers. Linux vserver is the most limited of the three, it uses one kernel for all virtualised instances. Within the virtual machines many functions are disabled for security reasons (for example access to the routing table, creation of device nodes). This limits the use of Linux vserver, but the advantage is that all VMs share resources like memory and the overhead is negligible.
Due to the different requirements of the portal and experimentation computers a hybrid approach seems to be advisable. The portal servers would be consolidated onto one Linux vserver instance. This makes sense as they do not use many resources. The Linux vservers offer encapsulation so that each portal can still be managed on its own. However, due to the structure of Linux vserver it now takes only about five minutes to create a new instance. The management is very easy as the host offers direct file-level access to the Linux vservers. This also allows easy backups and maintenance from the host. At the same time resource usage is low, compared to the other virtualisation systems. Each new vserver takes about 50 MB memory when running and idle. Consolidating all existing portals does not cause any performance issues for the single machine they are running on.

For the experimentation computers Linux vserver is not an option. Access to functions like routing tables or kernel modules is needed, so we had to find another solution for those. Xen offered the best compromise of flexibility and ease-of-use. We can consolidate up to 15 virtual machines onto one server; with other hardware we could potentially create even more virtual machines, but there are some limits like the available amount of main memory. If only one or two VMs are active this may be tolerable, but in the rare case that all virtual machines are in use they would all be very slow. Our current server hardware is a dual-processor machine with 2 GB RAM. It is about four times as fast as the previous dedicated machines. Two of those machines provide enough power to provide most of the previous 20 dedicated machines. Installing and managing Xen is not quite as easy as Linux vserver, but with some site-specific documentation and support scripts it is easy enough to work with it.

To facilitate the management we decided to add some monitoring. Using standard SNMP we are able to watch most relevant data points, for example processor load, memory usage, network traffic and whether a certain service is running. For data collection and representation we have evaluated a few monitoring applications, currently Cacti [6] is used and we are testing Nagios [11] in parallel.

With e-mail notifications the administrators can be warned whenever thresholds, like system load or disk space, are exceeded. This allows preemptive intervention instead of reactive administration when things have already failed. Still most management tasks are fully manual.

Figure 4 shows the conceptual design and separation of the virtual machines and physical servers. On the right side the Linux vserver host with the public portals can be seen, with the cold-standby as a shadow below it. On the left side the Xen hosts and their virtual guests are shown. Both Xen and Linux vserver hosts can be managed remotely, but only the latter needs to be exposed to the public Internet.

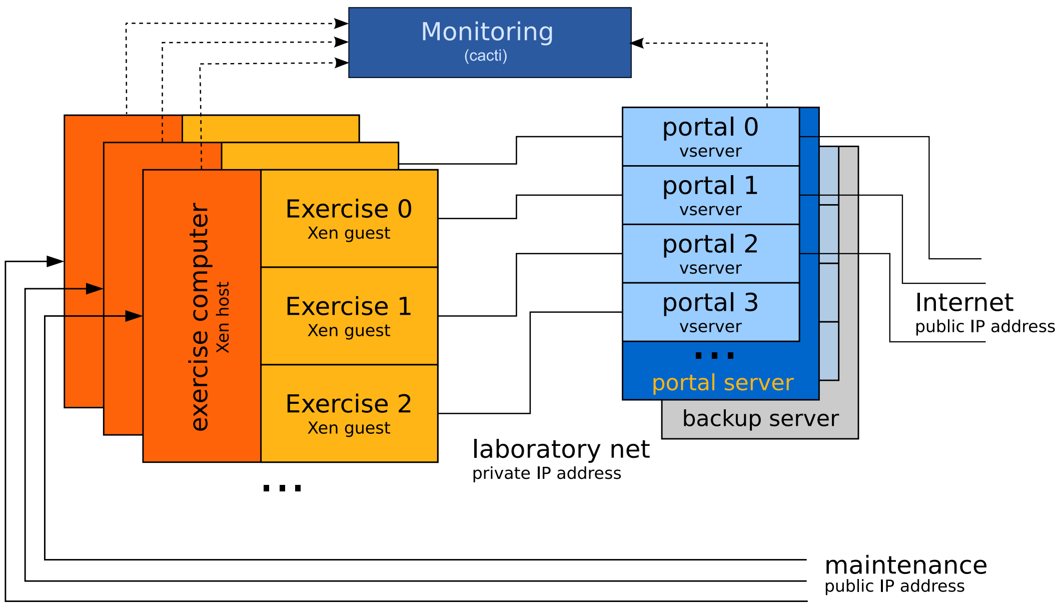

Figure 4. The proposed laboratory architecture.

Some of the existing machines cannot be integrated into the Xen virtualisation at the moment, mostly network- and hardware-specific devices. Still we are able to reduce the number of machines from around 20 to one portal server, two Xen machines and six non-virtualised machines. A backup of the portal and Xen machines is maintained on similar hardware to provide a "manual failover" in case one should fail. In this case the backup machine is booted and the relevant virtual machines are started with a script. This frees a lot of hardware for other purposes and significantly reduces administration time.

4. Conclusions

Two issues of our current e-learning infrastructure have been discussed. The new content formatting tool helps to maintain the sustainability of the course material by supporting an easier migration to new learning management platforms. A simplified document handling makes the creation of learning content more convenient for the learning module author and simplifies subsequent changes to the document structure.

By storing the learning modules in the XML format under a version control system the maintenance overhead is decreased. All changes are now made inside this repository and by using the formatting tool any course module can be retrieved on demand and converted into the desired format. This prevents inconsistencies if the same module is used in several courses.

The migration and virtualisation of our hardware has been a success for us. The necessary hardware has been reduced by more than 50% and the utilisation of hardware could be increased. At the same time we have a much better scalability—the single portal machine still has spare capacity to set up more portals if needed. We have not had any failures to test the failover capability, but we expect that our reaction time will also be much better than the "hours to days" reaction time of the old infrastructure.

The mix of virtualisation methods may be unusual, but it offers us optimal flexibility with only a small documentation overhead. With the help of documentation and support scripts it takes a few minutes to learn how to create a new virtual machine and then only a short time to setup and configure is required. Compared to the old infrastructure this is a huge improvement in productivity. At the same time we use less power and less space. While it is not applicable everywhere, virtualisation offers tools to consolidate underutilised machines and increases the availability at no extra cost apart from the migration itself. Backup and recovery become faster and easier.

References

[1] A. Baier, T. Bernoulli, T. Braun, C. Graf, and U. Ultes-Nitsche. Case study of the usage of an authentication and autorization infrastructure (aai) in an e-learning project. In *Information Security South Africa (ISSA 2006)*, Sandton, South Africa, July 2006.

[2] A. Berqia, A. Diop, and J. Harms. A virtual laboratory for practical exercises. In *Proceedings of International Conference on Engineering Education*, Manchester, UK, August 2002.

[3] Blackboard Inc. WebCT Vista (Blackboard). `http://www.blackboard.com/us/index.Bb`. last visited: December 2007.

[4] T. Braun, M.-A. Steinemann, and A. Weyland. VITELS – an e-learning course on computer networks and distributed systems. *SWITCH journal*, 2:32–35, November 2003.

[5] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau, and J. Cowan. Extensible Markup Language (XML) 1.1, W3C Recommendation. `http://www.w3.org/TR/xml11/`, August 2006.

[6] Cacti. `http://www.cacti.net`. last visited: January 21, 2008.

[7] B. Collins-Sussman, B. W. Fitzpatrick, and C. M. Pilato. *Version Control with Subversion*. O'Reilly Media, Inc., first edition, June 2004.

[8] eduSourceCanada. `http://www.edusource.ca/english/home_eng.html`, 2007.

[9] Edutech. Learning object repository – test results of feasibility study. `http://edutech.ch/lms/2006LOR/index.php`, 2006.

[10] Linux-vserver Virtualization Software. `http://linux-vserver.org/`. last visited: December 6, 2007.

[11] Nagios. `http://www.nagios.org`. last visited: November 30, 2007.

[12] SCORM – Sharable Content Object Reference Model. `http://www.scorm.com`. last visited: December 5, 2007.

[13] SWITCH (The Swiss Education and Research Network). AAI – Authentication and Authorization Infrastructure: System and Interface Specification, 2004.

[14] Virtualbox Virtualization Software. `http://virtualbox.org/`. last visited: December 6, 2007.

[15] Vmware Virtualization Software. `http://vmware.com/`. last visited: December 6, 2007.

[16] A. Weyland and T. Braun. OSLab Module Author Guide. Technical Report IAM-06-004, University of Bern, IAM, July 2006.

[17] M. Wulff and T. Braun. OSLab: An interactive operating system laboratory. *ERCIM News*, 71:46–47, 2007.

[18] Xen Virtualization Software. `http://xen.xensource.com/`. last visited: December 6, 2007.

[19] S. Zimmerli, M.-A. Steinemann, and T. Braun. Resource management portal for laboratories using real devices on the internet. *ACM Computer Communication Review*, 33(3):127–135, July 2003.