Winter Term 2005/2006

SEMESTER THESIS

for

Tobias Rein (D-ITET)

Advisor: Rainer Baumann
Supervisor: Prof. B. Plattner

Issue Date: October 2005
Submission Date: March 2006

Student Webportal for Courses

Abstract

Course and group administration is plenty of work and a very painful process for all involved people. To simplify course administration and to relieve coordinators, we developed a powerful web-based course administration tool. It convinces by its ease of use, scalability and extendibility. Half the battle is, that students and administrators can make use of their common ETH Zurich account to login. Nevertheless, the application supports users without an ETH account too.

The web-tool offers particularly group inscription and attestation management. For group inscription, there are several procedures available as "First come first serve" and "Best match". "Best match" assigns students to groups based on their priority resulting in an optimal matching. The attestation management allows to handle attestations on a per lesson or assignment level.

The application has been validated and tested intensively in great detail. Hence, the application is ready to be used in summer-semester 2006.

Contents

1 Task Description

"Welcher Student kennt es nicht. Bei den Übungseinschreibungen herrscht Chaos: man bekommt nicht den gewünschten Platz, Absenzen wegen Krankheit oder Militär sind nur mühsam zu regeln, Musterlösungen sind nicht verfügbar und Fragen werden erst zuspäht beantwortet. Diesen Misstand kannst du nun beheben! Entwickle und realisiere in deiner Arbeit ein Webportal für die Studenten. Dies kann zum Beispiel mit PHP und MySQL erfolgen."

2 Introduction

Over the past years, ETH Zurich heavily extended its electronical portals. "einschreibung.ethtz.ch" and "edoz.ethz.ch" offer a large amount of services for students and lecturers. However, they are limited to course-level. The management of exercise groups and assignments is left to the coordinators them-self. Major problems are student assignment to groups based on priorities or management of attestations for practical and theoretical exercises. These issues are mostly done by hand. Papers are passed around from one assistant to another, rising troubles of coordination and of lost attestation lists.
There are several challenges handling these tasks within a web application. For example user management should not introduce now administrative work. Since ETH offers already a common login management for all students and employees, it is self-evident to make use of this database. Access for exchange students and hearer without an ETHZ login has to be offered. The application also has to support various courses with different requirements. Another challenge is the automation of assignments of students to groups which is much more difficult than one would expect.
There exist already many tools for group collaboration and lecture management. But none of them fulfills our requirements even approximately. They are built for different purposes and support hundreds of unnecessary functionalities, options and features. That makes them hard to handle. Furthermore, none of these applications supports user management based on the ETH database nor sophisticate group inscription functionalities or attestation management.
Therefore we designed, implemented and tested the "Student Webportal for Courses". We mainly focused on the specified requirements. Access is divided into three levels: administrator, course administrators and student. It offers user management based on the ETH database or a local repository. It supports a scalable design for courses with different properties. For the inscription mode, the tool currently provides two methods: "First come first serve" and "Best match". Students can inscribe in courses by them-self. A detailed management of the attestations is provided. Furthermore, the web-portal is registered by the Cooperate Design [3] of ETH Zurich. The application is multilingual, at the moment supporting English and German.

This thesis gives detailed information about the functionality and the implementation of the Student Webportal. The first section deals with the specifications and the database design of the application. The choice of the operating system and the software is explained in chapter 4. The section about implementation is divided in four parts. In the fourth part, the algorithms for inscription into groups are explained. Then we present the different manuals for the three types of application users. In section 7 details about the validation and testing can be found. Following, we present our plans about future versions of this application. A summary is given in the end of this paper.

3 Design & Specification

3.1 Specifications

This section deals with the precise specification of the Student Webportal. The list bellow shows the details of the specification and assumptions we made.

Users The access to the website is only for matriculated people. Therefore, the user has to login with his n.ehtz password. It is possible that some exchange students do not have a password. On this account the portal supports local users. They have the same access possibilities as normal users. The only difference is that their password is stored in the web-portal related database and not in the ETH LDAP database. To avoid complications their name always starts with "local.".
The users are divided in three groups: administrators, course administrators and users (students). An administrator has unlimited access to all parts of the website. A course administrator has access to manage a selected choice of courses and to handle local users.

Courses Each course has a name and takes place in a specified term. These two values have to be distinguished. Additionally, every course has some groups, which the students can inscribe in.

Groups Every group belongs to a course. A group has a unique name in the corresponding course and contains some lectures.

Inscription A student can inscribe in maximum one group per course. Course administrators cannot inscribe in a group of their own course. Principally, two modes for inscription exist: "First come first serve" and "Best match" (see section 5.4).

Attestation A student can get one attestation for each lesson.

3.2 Database Design

The backbone of the application is a mySQL database. If you are doing database design it is important to do it right at the first time. It is almost impossible to redesign the database completely in a progressive stadium of your work. It was one of the first things we have done for this thesis. With all the specifications we had, we made a first draft of the database design and finally we fixed the design (see appendix B). During the process, some small changes were necessary. We needed some more fields or the specified type for the variable had to be changed.

3.2.1 Tables

This section gives a brief explanation of every table in the database. Most of the fields do not need to be described explicitly. All the ID fields (Uid, Cid, Gid, Lid, Aid, Fid, UCid, UCGPid, LOGid, LANGid) are primary keys for the corresponding table. However, in every table exists another key, so that all the ID files are for redundancy. If you are inserting a new entry, it is not necessary to assign a value to these fields because they have an auto-increment mechanism. In the following a list of all tables in our database.

users This table assigns a user ID to every user. The field Ulogin is unique. The field Ulocal indicates whether the user is a local user (value 1) or not (value 0). Only local users have a password stored in the database. Upending is for the future use.

courses The data for a course are stored in this table. As previously mentioned Cname and Cterm are the key. If the value of CinscriptionMode is 1, the "First come first serve" mode (see section 5.4.1) is chosen. For the value 2, the "Best match" mode (see section 5.4.2) is active.

groups This table is similar to the table for courses. Every group has a name, which is together with the Cid a key. Cid has to be a foreign key referring to an existing ID in the course table.

lessons Lessons are attached to groups with the foreign key Gid. Two lessons of a group never start at the same time, so LfromTime, Ldate together with Gid is the key for this table.

files For future use. It is thought for exercises, solutions and other files related to a course. There is a mechanism that one can set the period when the file should be accessible for the students.

user2course This table fixes the inscriptions the students have done. It contains three foreign keys: one refers to the users table, one to the courses table and the last one to the groups table. There is no restriction that the Gid refers to a group, which belongs to the course with the given Cid. Therefore, this is a problem that has to be solved externally. Every user can be inscribed in only one group per course, so Uid, Cid and Gid is the key. This table has another functionality, it contains a field that says which users have permissions to administrate the course represented by the Cid. If the UCfunction is set to 1, the user is a course administrator. This field can also be used for future work. You might want to make some other specified users like assistants.

user2groupsPref This table is used for the inscription mode "Best match". It holds all the preferences and anti-preferences the students have done. UCid (referencing to the user2course table) is together with Gid (referencing to the groups table) the key. Possible values for the UCGPpref fields are: 1, standing for "this group is impossible for me", 2, for "this group is possible for me" and 3, representing "I prefer this group".

attestations A small table that shows all the attestations. Lid and UCid are the key. Both are foreign keys referring to the lesson and the user2course table respectively. If there is an entry in the table for a given Lid and a given UCid, the student has received the attestation for this selected Lesson.

administrators This one-filed table is just a collection of all the user ids of the administrators. This is the more modular solution than to save the administrators login in one of the PHP-files.

The following two tables are more or less part of the database design. They are standalone table for administrative purpose.

logger This table simply logs all of the actions the users are doing on the web-portal.

language The language table contains the bilingual texts. To keep an overview, the entries have two fields to specify where the text are used: LANGpage and LANGdesc. LANGpage says on which website this text is used and LANGdesc gives some more detailed information.

4 Platform & Environment

Every web application needs a server. Mostly, particular services are required and have to be installed on this machine. In this section, all the necessary packages for the web-portal are discussed. Before talking about software, a summary of the selected hardware is given including the operation system.

4.1 Hardware & operation system

A 2 GHz AMD Athlon (TM) XP 2800+ with two integrated 80GB RAID 1 hard disks is the core. To reserve machinery capacity these hard disks are running in clone mode. That means that both disks contain exactly the same, they are images of each other. If one is failure, no data will get lost.
The operating system running on the machine is Linux (version 2.4.27-2-386). The distribution we are using is a Debian 1:3.3.35-12.

4.2 Apache web-server

There is no doubt about using an Apache 2 web-server (version 2.0.54 Debian GNU/Linux). It is the most common web-server and its installation and setting are easy to handle.

4.3 PHP

Almost as obvious as using an Apache web-server is using PHP as scripting language. The option of using Perl or Python has been rejected. In my opinion, PHP is the most powerful language to create websites. Unfortunately, there is no PHP 5 available for Debian Linux at the present moment. Therefore, the running version is 4.3.10-16.

4.4 LDAP Database

The first issue we had to deal with was the question of authentication. We do not want to establish a new login procedure for the web-portal. It is more likely that the students can use the familiar n.ethz login name with the corresponding password. ETH Zurich holds an LDAP database to store all the student's information. By contacting the concerning person [1], we obtain access to the ETH LDAP server.
In fact, LDAP stands for a network protocol (Lightweight Directory Access Protocol) and not for a server type. However everyone is speaking of an LDAP server, if he means a directory server that can be accessed over TCP/IP with LDAP. It allows you to querying and modifying directory services. LDAP has derived from the Directory Access Protocol (DAP) and has been developed at the University of Michigan. The goal was to gain popularity with simpleness. Consequently, it is based on TCP/IP and includes only some of the DAP functionality.
The data-structure of a LDAP directory is a hierarchical tree. Every LDAP server administrates a certain sub tree, a partition. Every partition contains a set of directory entries. Each entry consists of a set of attributes having a distinguish name (dn) and one or more values. A schema defines these attributes. Unfortunately, not all schemata are consistent all over the world; hence there exists no worldwide database.
To enable LDAP on our system, we had to install a LDAP package (OpenLDAP v 1.130.2.11). With the key we received from the ETHZ LDAP administrator and this package we were able to establish a secure connection from our server to the LDAP database.

4.5 SSL

On the website the user is requested to enter the n.ehtz password. In case of conventional HTTP, this password would be transmitted in plain text to the server. So we decided to cipher the connection to the

user. For that reason, we installed an SSL package (OpenSSL/0.9.7e). The certificate we are using is self-signed, but for our purpose this is not a security risque.

4.6 MySQL Database

The most convenient solution to administrate all the users, courses and groups is to work with a database. MySQL 4.0.24 is the solution on our system.
For a powerful tool like a database, there has to be a powerful administration. To keep an overview on all the tables, we installed the phpMyAdmin [2] on our server as well. Sometimes, it was a live safer when I got lost in all these data. The running version is phpMyAdmin 2.6.2-Debian-3. The URL for this service is http://polar9.ethz.ch/phpmyadmin/.

5 Implementation

In this section, detailed explanations of the implementation are given. First, some information about the ETHZ Cooperate Design [3] are mentioned. Further, the functionality of an LDAP server is described. If you want to do extensions on the application, you should read the subsection about the code structure. Finally, the main algorithms used on the web-portal are explained.

5.1 Layout

An important part of every web application is the layout. A well-designed web page has to be user-friendly. It must be easy for the visitors to keep an overview on the sites. Because this project is part of ETH Zurich, we decided to profit from the Cooperate Design [3] of ETHZ. The Cooperate Design is a service of the Cooperate Communication Group [4] and was founded to provide a uniform look for the public relation of ETH Zurich. For websites, they support two ways of implementation. One is a web content management system (WCMS), the other one are PHP templates. For this project it is more convenient to use the templates because the whole page is very dynamic and based on a database.
The Cooperate Design specifies things you should carry on. These are written down in a documentation [5]. The main design divides the page in three vertical parts: head, content and foot. Depending on the specific site, the content section can obtain one, two or three columns. For the colors of your page you can choose out of 18 color patterns. Each pattern is a collection of five colors with different brightness. On the web-portal the pattern with number C140 is used. The background of the head is an

Figure 1: Color pattern C140 with its five colors

image. The Cooperate Communication Group offers a service that adapt an arbitrary picture to the colors of the chosen pattern. In the background of the Student Webportal is a screenshot of the advertising for an LCD television of Sony [6].

Figure 2: Basic Picture for the identity area.

5.2 LDAP authentication

The connection of the login on the web-portal to the n.ehtz account is done over an LDAP authentication. Actually, the functionality of the LDAP was very limited. Authentication (named bind), search queries and modifications on data were possible. Nowadays, more features like replication between different directories are implemented. The Student Webportal is using the bind and the search functions. It is possible to check the authentication without any search queries, since all the n.ethz users are allowed to bind to the ETHZ LDAP database (but they don not have permission to search in the database). Therefore, a bind-query to the LDAP server is sufficient to authenticate the user. If this query is successful, the user is valid and the entered password correct. Otherwise the authentication fails.

The search function is used in the application to set course administration rights for users who have not yet used the Student Webportal and are thus not stored in the internal database. With the special account we got from the ETHZ LDAP administrator, we are allowed to execute search queries on the LDAP server. If such a user needs to be registered in the mySQL database, the first step is to bind to

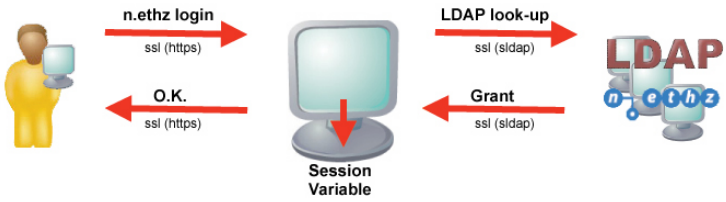


Figure 3: LDAP authentication

the LDAP server with our special account. Then, a search query on the distinguish name of the user is performed and the returned data are stored to the mySQL database. Now, the user has a unique user ID that is needed to allocate administrator rights.

5.3 Code structure

Following the hierarchical structure of the source code is presented. The template of the Cooperate Design (CD) already provides a strong defined structure. The CD recommends putting each sub-level of the web-application into an own directory. Additionally to the sub-level directories, there exist some folders containing information about the framework of the website.

5.3.1 administration

This folder contains all information about the administration for these websites. To have access to these files, you need to be either an administrator or a course administrator. To the sub-levels admincourses and localusers, both kinds of administrators have access. The other directories are not determined for course administrators. The directory contains many sub-directories, however more detailed information are unnecessary.

5.3.2 authentication

Normally, this page should never be called directly. Wherever a login is required, an automatic redirection to this side is made. For example, a student comes to the Course Portal and the first thing he does is clicking the link to the courses area in the main navigation. This page is not available for guests; hence the student is redirected to the authentication directory. After a successful login, the student is automatically forwarded to the course area site.
There is another functionality of this page. If a visitor uses the login box on the right of any page, the authentication directory is called with the login parameters. If the login is successful, the user is automatically redirected to the previous page where he did the login, so that the whole process is hidden for the user.

5.3.3 courses

This is the part for the students. It is reachable through the link in the main navigation in the head area. All the things a student can do in this section is described in section 6.3.

5.3.4 error

The error area is used for errors the Apache server throws or errors based on login permissions. It exists a single file handling all the different cases. Therefore, the web-server configuration has to be changed. The paths to the error files are set to /error/[error_number] (e.g. /error/403 for permission denied) except for that one of error number 404 (file not found). This one is set to /error/ and therefore the only existing

file. The procedure is the following: if an error occurs, the web-server forwards to the given file. But this file does not exist. Hence, a second error is thrown: file not found (404). This special error is forwarded to the index file in the error directory. Due to the web-server stores the path to the not found file, a script can readout the number of the error by using the environment variable "REQUEST_URI".

5.3.5 contact, imprint and info

These directories are for the sake of completeness. Contact and imprint are required for the Cooperate Design; info is a short introduction in how to use the Student Webportal.

5.3.6 include

This is a part of the framework. It contains the different sections of the page that the Cooperate Design specified. There is a file for the head (header.php) and one for the foot (footer.php). Additionally, there is a file for the pathline (pathline.php). This is the line below the head where the whole hierarchy is listed. On the right hand side on a level with the path-line is the service navigation (servicenav.php). In this menu the user can choose the language or print the current web page. There is a separate file for all the

Figure 4: The head, the pathline and the service navigation

header information of the HTML (head.php). This file includes all the style sheets and specifies some java script functions. The loginbox.php file is an extra file for this application and comes not with the template. It specifies the different login boxes on the right margin.

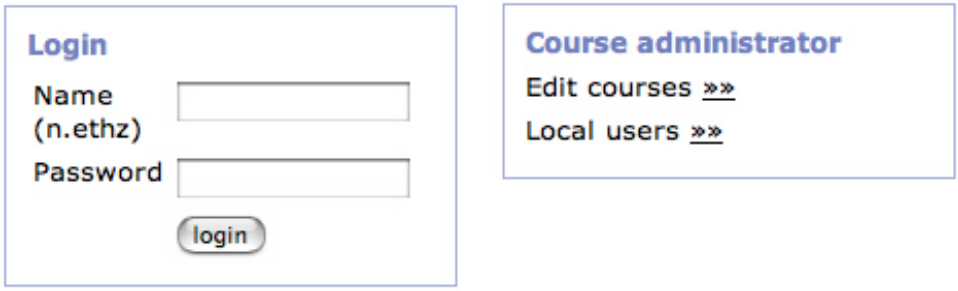


Figure 5: Different kind of login boxes

5.3.7 scripts

Several functions, which are used by the PHP-files, are stored in external files. These are placed in this directory. An extra file (scripts_en.js) is for java script functions. In the following a description of the files with their functionality is given.

database.php represents the intersection of the web-portal with the mySQL database. By including this file, a connection to the database is established. Several functions execute specific queries. The return values are either stored in PHP variables or in associative arrays. Typically, all names of public functions start with "sql_".

ldap.php contains two functions for connecting to the ETHZ LDAP server. One is for simple authentication (checks if the bind is successful) and the other one checks if a person exists in the database by using a search query.

session.php simply stores the session variables the user is transmitting by the POST method. Secondary, it stores the entries for the logbook with these values.

navigation.php is a smart implementation for the sub-navigation on the left hand side. First, a recursive function builds the menu with the sub-menus. A second recursive function prints this menu and highlights the active item. How the menu looks like is specified in a function called "add_submenu" in the bottom of the file.

encoding.php is useful in many respects. First, it encodes strings coming from and going to the mySQL database. This affects especially single and double quotes and some special German characters like "äöü". Moreover, there are a couple of functions handling dates and times. Often, these values are used in HTML forms. To reduce code complexity, a bunch of supporting functions have been instantiated. Some other functions dealing with en- and decoding are placed in this file.

bestmatch.php is the core of the "Best match" algorithm and contains all functions needed. For more details see section 5.4.2.

5.3.8 css, icons_site and images

These directories contain framework stuff of the Cooperate Design. The layout of the web page is based on style sheets. Hence, there exists a css folder collecting all the css-files. Images and icons are located in the images and the icons_site folders respectively.

5.4 Algorithm

The Student Webportal supports different algorithms to inscribe in groups. Up to now, two methods are offered. "First come first serve" privileges the students coming first. The "Best match" approach does not take care one the sequence the students assign to groups. It calculates the best matching allocation of students to groups based on their priority. More specified details are given in the following subsections.

5.4.1 First come first serve

This mode of inscription allows consecutively inscription to groups. The only restriction is that the groups have limited vacancies. The students can see the number of available places at the time of inscription. Once a group is filled up no more students can participate in this group.

5.4.2 Best match

The "Best match" algorithm is another option for the course inscription mode. If this method is chosen, the students can make their preferences during the period of inscription. The implementation of the algorithm allows three levels of priorities: preferred, possible and impossible. After the inscription period is over, the coordinator can start the algorithm which is looking for an optimal assignment of the students to groups. This problem is equivalent to a matching problem in a weighted complete bipartite graph [9, 10, 11]. The two partitions are the students on the one hand, and the groups on the other hand. The edges are weighted with the preferences the students have done (see figure 6).

After identifying this problem, we get in contact with Professor Steger [8] of the institute for Theoretical Computer Science at ETH Zurich. We found that there exist sophisticated solutions for this problem.

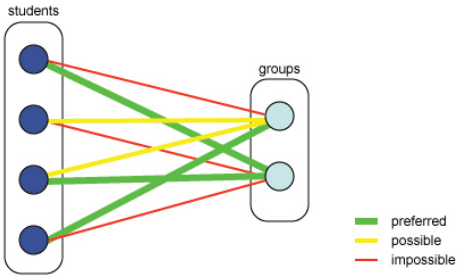


Figure 6: Weighted complete bipartite graph

Optimal realizations have a runtime complexity of $O(n * m + n^2 log(n))$. Scientific implementations with this complexity are only available in commercial libraries [12] for C/C++ and C#.
The Student Webportal has to handle hundreds of students. In this case, these algorithms may face the limitation of our computing power. Moreover, advanced tools are not freely available and hard to include in our framework. Therefore, we decided to implement an own algorithm. It guaranties that there is no better matching by exchanging two students between two groups. It does not guarantee that a permutation of three or more students can result in a slightly better matching. Having a runtime-complexity of $O(n^2)$, our algorithm is noticeable faster and can easily handle several hundred students. It is structured as following:

Loop $\forall$ students inscribed to the course

- Try to assign the student to a "preferred" group.
  If this is successful, continue with the next student.
- Try to change an already assigned student from a "preferred" group to another "preferred" group, so that you can assign the current student to a "preferred" group.
  If this is successful, continue with the next student.
- Try to assign the student to a "possible" group.
  If this is successful, continue with the next student.
- Try to change an already assigned student from any group to a "possible" group, so that you can assign the current student to a "preferred" group.
  If this is successful, continue with the next student.
- Try to change an already assigned student from a "preferred" group to another "preferred" group, so that you can assign the current student to a "possible" group.
  If this is successful, continue with the next student.
- Try to change an already assigned student from any group to a "possible" group, so that you can assign the current student to a "possible" group.
  If this is successful, continue with the next student.

  If all these points fail, do not assign the student to a group.

Loop $\forall$ students who are not assigned to a group yet

- Assign the student to the group with the most available places.

The algorithm works straightforward and is optimal with respect on permutations of two students. Since the algorithm is deterministic and processes all students in sequence, no deadlock could occur. In addition, the last loop guaranties that every student will be assigned to a group.
The result of the "Best match" algorithm from the Student Webportal is given in figure 7. Every column represents a group; every row stands for a student. The numbers represent the priorities: "preferred" (3), "possible" (2) and "impossible" (1). The boxes show the assignment of the students. To give a brief overview, the boxes are colored based on the preferences: green ("prefered"), yellow ("possible") and red ("impossible").

Group 5	Group 4	Group 1	Group 2	Group 3
1	1	3	2	2
3	3	1	1	2
1	3	1	2	3
3	3	2	3	3
1	1	2	3	3

Figure 7: Result of the "Best match" algorithm

6 Manuals

One of the big goal was to design a self-introducing website. The use without long instructions is of particular importance for a web application. The students do not want to read many pages to understand how to inscribe in courses and groups. The available web-portal might need some introductions for the administrators, but the student-area is kept simple and does not need long statements.

6.1 Administration area

People with administrator rights will see an administrator box on the right hand side after a successful login. The four possible areas an administrator can reach are listed in this box. Two of them are identical with the course administration areas: edit courses and local users. For more information see section 6.2. There is a restriction that might happen while accessing to the edit courses area. If an administrator is not a course administrator of any course, he will not have access to this area and an error is thrown by the server.
In the area called "administrate courses" new courses can be set up. The name and the term of a course can only be changed by an administrator. Moreover, in this area one can specify the course administrators. Finally, there is a logbook giving an overview on things, which happened on the Student Webportal.

6.2 Course administration area

All the course administrators can reach two additional areas by using the links in the box on the right. One is for editing local users. There, one can alter passwords and login names as well as the student data. It is also possible to create new local users. The second area for the course administrator is to edit his courses. For each course there exist four submenus, one to treat the attestations, a second one to edit the course data. In this submenu, it is possible to set the algorithm used for inscription. Moreover, the period of inscription can be defined. The third submenu is for managing all the groups contained in a course. A special feature is the so called "quick add assistant". If all the groups of a course are almost identical, this assistant can create in a few steps a number of groups with equivalent lessons. An additional sub-submenu allows configuring all the lessons and the group constraints. The fourth and last submenu is an overview on all students in a course. For more information about one of these students there is a link to a side showing all the details. If a student needs to change group, this can bin done there.

6.3 Course area

This area is for all the students using the web-portal. It is designed in two layers. First, a student needs to select a course. After this first step, different possibilities exist. Normally, the inscription period is in progress, so that the student can choose a group or specify his preferences. If the period of inscription has not started yet, the site is blocked. Another case is that the inscription period is over and the student is no more responsible to inscribe himself in a group. He has to contact one of the course administrators. After successfully signed in to a group, the student is able to check his attestations in this area.

7 Validation

To validate the Student Webportal an elaborate test plan has been developed. For feasibility, testing is divided into three phases: administrative set-ups, student usages, and administrative management. In the first phase, ten course-administrators set up courses with various properties. In the second phase, the web application has been opened for the students. Over 30 students registered for the different courses and exercise groups. In a third phase, management of the attestations and the functionality of the different inscription modes has been tested.
In all these periods no major failures or problems have been found. Only several typing errors have been discovered but they have been fixed.

8 Future Work

So far, the Student Webportal provides several features: course and group administration, handling of attestations, login with an n.ethz account and inscription with two different modes. These functionalities build a good foundation of the web-portal. During this thesis, we have already thought about many different extensions.
Support for file management could be included. This is thought for exercises and solutions. It should be possible to set the date and time when a given file is available for students, so that the coordinator can upload all files at the beginning of a semester and the web-portal automatically sets the files free at the given time. To hand in student exercise solutions would be nice too.
In addition, more inscription mode could be added. In some courses, the students have to inscribe in pairs or in larger groups. At the moment, this is not supported by the application. It is a challenge to implement an inscription for more students.
Furthermore, we thought about a communication channel. A forum could be the solution. Moreover, it would be nice if course-administrator had the possibility to send emails to all the students inscribed in his or her course.
Due to time limitation, these amazing extensions, as well as many others, could not have been included yet.

9 Summary & Conclusion

Course administration is a pain. Therefore we developed a "Student Webportal for Courses". It offers many functionalities:

- Course, group & lesson management
- Attestation management on a per lesson or assignment level.
- User management with a login based on n.ethz
- Inscription in courses and groups that can be done by the students them-self.

It has been profoundly tested and validated.

A ETHZ LDAP Database Structure

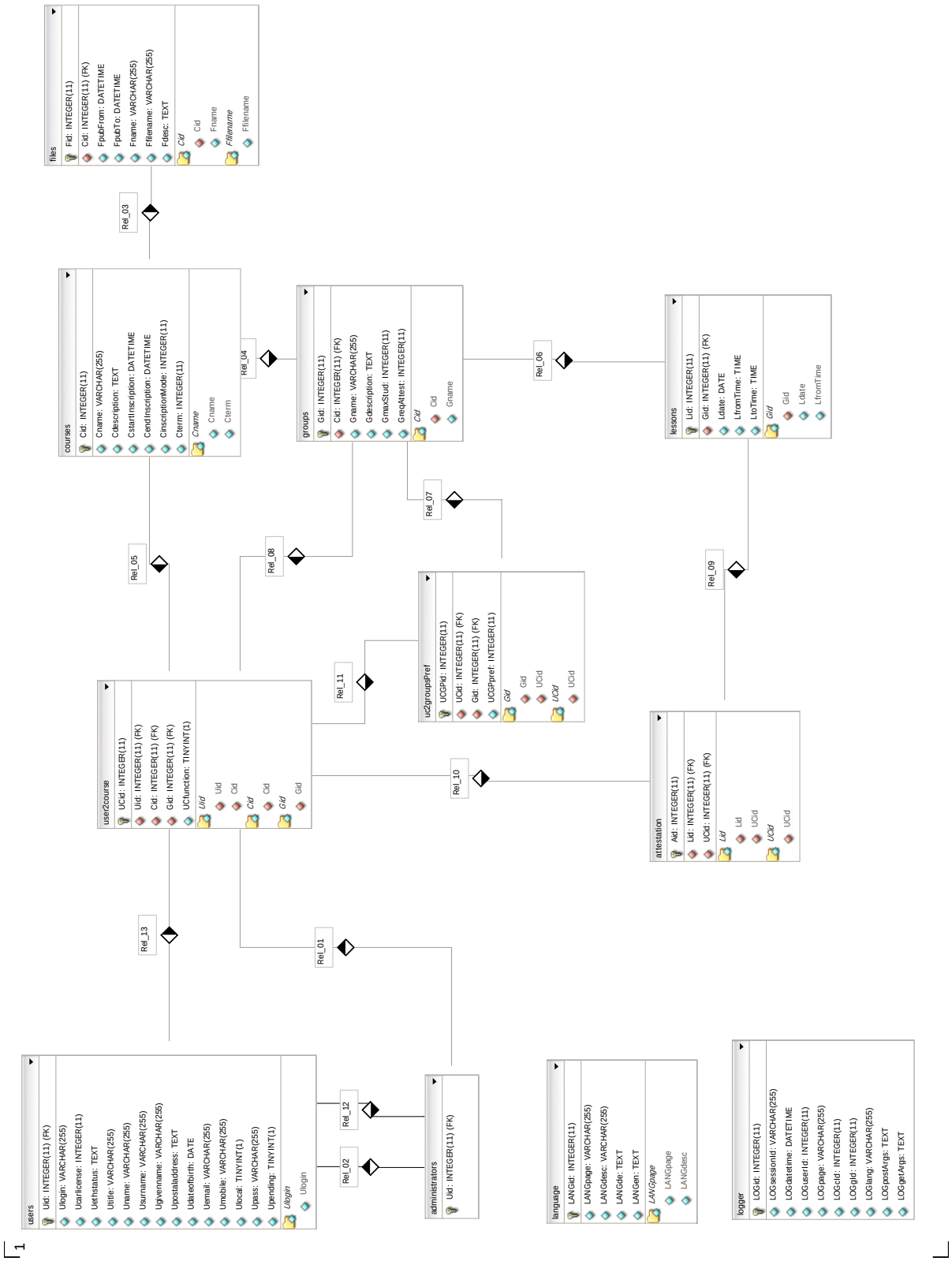
```
LDAP database structure:
------------------------
ou=id,ou=auth,o=ethz,c=ch  -o- ou=admins  -o- cn=nethz_master
                            |              |
                            |              o- cn=nethz_proxy
                            |              |
                            |              o- cn=wcms_master
                            |              |
                            |              o- cn=...
                            |
                            o- ou=nethz   -o- ou=users   -o- cn=...
                            |              |              |
                            |              |              o- cn=...
                            |              |
                            |              o- ou=groups  -o- cn=...
                            |              |              |
                            |              |              o- cn=...
                            |              |
                            |              o- ou=...
                            |
                            o- ou=wcms
                            |
                            o- ou=.....

User record: (ou=users,ou=nethz,ou=id,ou=auth,o=ethz,c=ch)
------------
    objectClass:
        # STRUCTURAL
            top
            person
            organizationalPerson
            inetOrgPerson
            swissEduPerson
        # AUXILIARY
            eduPerson
            posixAccount
            shadowAccount
            ethzOrgPerson

    attributes :
        # MUST
            cn (=uid)
            sn
            uid
            uidNumber
            gidNumber
            homeDirectory
        # MAY
```

```
nPID
nUID
PERSID
userPassword
loginShell /bin/tcsh
gecos
shadowLastChange -1 |
shadowMin        -1 |
shadowMax  -1 | required by
shadowWarning    -1 | Solaris login
shadowInactive   -1 |
shadowExpire     -1 |
shadowFlag       -1 |
*--- swissEduPerson ---
swissEduPersonUniqueID
givenName
swissEduPersonDateOfBirth
swissEduPersonGender
preferredLanguage
mail
homePostalAddress
postalAddress
homePhone
telephoneNumber
mobile
swissEduPersonHomeOrganization
swissEduPersonHomeOrganizationType
eduPersonAffiliation
swissEduPersonStudyBranch1
swissEduPersonStudyBranch2
swissEduPersonStudyBranch3
swissEduPersonStudyLevel
swissEduPersonStaffCategory
eduPersonOrgDN
eduPersonOrgUnitDN
eduPersonEntitlement
```

B MySQL Database Design



References

[1] Vladimir Nespor, ETH Zurich, Administrator of the LDAP server. vladislav.nespor@id.ethz.ch

[2] The phpMyAdmin Porject. http://www.phpmyadmin.net/

[3] Cooperate Design. http://www.cd.ethz.ch

[4] Cooperate Communications. http://www.cc.ethz.ch/

[5] Dokumentation zum Web-CD. http://www.cd.ethz.ch/services/web/dokumentation/

[6] SONY Bravia - The Advert. http://www.bravia-advert.com/

[7] PHP: Hypertext Preprocessor. http://www.php.net/

[8] Prof. Angelika Steger. ETH Zurich, Institute for Theoretical Computer Science.

[9] A Heuristic for Dijkstra's Algorithm with Many Targets and Its Use in Weighted Matching Algorithms. 2001; Holger Bast, Kurt Mehlhorn, Guido Schäfer. Springer-Verlag. London, UK

[10] Algorithmen für Matching-Märkte. 2005; Hui Jin. University Cottus

[11] Diskrete Optimierung. 2005; Maurice Cochand. ETH Zurich

[12] LEDA - Library of Efficient Data types and Algorithms (Version 5.1). 2005; Algorithmic Solutions Software GmbH

March 24, 2006