New Course Registration Module for the University Study-Oriented System

Krzysztof Ciebiera, Janina Mincer-Daszkiewicz, Tomasz Waleń

Faculty of Mathematics, Computer Science, and Mechanics,
Warsaw University, Poland
ciebie@mimuw.edu.pl, jmd@mimuw.edu.pl, walen@mimuw.edu.pl

Abstract

The University Study-Oriented System (USOS) is an integrated student management information system for handling student affairs at Polish universities. Its development and deployment is coordinated and supported financially by the consortium of Polish higher education institutions. It is a huge database-oriented software application done partly in Oracle technology and partly using open source products (Internet modules).

Course registration is one of the most important functions of the system. The problem with the registration is that various faculties and higher education institutions use various registration strategies. The native module for web-based registration, USOSweb, was not able to fullfil needs of them all, so the new one was developed, based on fundamentally different registration rules.

The purpose of this paper is to describe both registration modules for course registration developed for USOS. The new module was deployed at all faculties of Warsaw University in the summer of 2003 and used during the academic year 2003/04 for registration for courses and exams which are offered to all students of the university.

Keywords: web-based course registration; tokens as virtual money; University Study-Oriented System (USOS); consortium of Polish higher education institutions

1 Introduction

In modern higher education institutions flexibility of study programs means in particular that students may freely choose a substantial part of their curricula. The software support is thus highly needed to help to make good choices and enable to pass the information on what has been selected. This information is later used for student assessment and checking whether a student met the requirements of a particular study program.

The important aspect of this process is the huge amount of data which needs to be gathered in the system. Doing by hand all the necessary registration activities would be a nightmare for every student administration office. The only economically acceptable solution is to share the work, ie. to let the students do it by themselves.

Student management information systems should therefore contain web-based modules for remote online registration. However designing such modules is not an easy task since various faculties and higher education institutions use various registration strategies.

In this paper we describe course registration modules developed for USOS — a student management information system for Polish higher education institutions. The native module, USOSweb, had already been used by a few universities for a couple of years. Generally, it supports registration for specialization courses offered by faculties to their students. The new module developed last summer is based on fundamentally different rules and supports registration for courses offered to all students of the university, like foreign language courses.

In the next section we shortly describe the architecture and main features of USOS and USOSweb. Section 3 is devoted to the main aspects of course registration and its implementation in USOS. In the following section requirements for the new registration module are outlined. More technical implementation aspects of the new module are described in section 5. The issues concerning deployment of the module in Warsaw University are considered in section 6 and some conclusions are drawn in the last section.

2 USOS and USOSweb

The University Study-Oriented System ([5, 8, 10]) is an integrated student management information system for handling student affairs at higher education institutions. Its development and deployment is coordinated and supported financially by the consortium of Polish universities [9]. It is currently being used at a few of those schools and is being deployed at the others.

The system supports all student-oriented activities such as handling students' and teachers' personal data, study programs, requirements of degree certificates, course catalog, course registration, class schedules, dormitories, tuition and financial aid.

USOS has the following architecture:

1. the main database is implemented in Oracle technology; it is accessed only by the administration staff via Oracle forms and reports. Oracle technology ensures effective handling of huge amounts of data accessed simultaneously from distributed client sites;
2. there are also a few Internet databases with massive numbers of clients accessing them via web browsers; this part of software is made in open-source technology. Internet databases are partial copies of the main Oracle database. In particular highly confidential personal data are not stored on the web, data are available mostly for reading, and modifiable data comprise only a small subset. All data transfers are encrypted. The Internet part of USOS is called USOSweb [6].

This architecture with a set of separate databases is more complex from the technological point of view than solutions with one data repository since independently updated databases have to be synchronized by special mechanisms. However, on the other hand such a solution substantially lowers overall costs since users of Internet modules do not need expensive Oracle licences, the distributed architecture ensures good performance even at rush hours and Internet databases do not endanger highly confidential data.

Since data may also be entered through the web interfaces to the Internet repositories (when students register for courses or academic teachers input grades to electronic examination forms) some tools for database synchronization are necessary.

In the prototype version of USOS a very simple way of transferring data by files was implemented. Then a more flexible and generic tool was developed, which connects remotely to both databases and automatically synchronizes tables contents. In both methods in case of conflicts (inconsistent changes performed simultaneously at both sites), the final decision is made by the human operator (like in CVS).

3 Course registration in USOS and USOSweb

Course registration is one of the most important functions of USOS and USOSweb since it is the basis for other vital activities of the system.

Four important aspects of the registration should be considered:

1. In modern higher education institutions flexibility of study programs means in particular that students may freely choose a substantial part of their curricula. The software support is thus highly needed to help to make good choices and enable to pass the information on what has been selected.

2. In our medium-size faculty every year about 1500 students register for about 300 courses. In Warsaw University in 26 didactic organizational units study more than 55 thousand students, the largest faculties have more than 6000 students. These numbers give some impression about the scale of the process. Doing all the necessary registration activities by hand would be a nightmare for every student administration office. The only acceptable solution is to share the work, ie. distribute it among students.

3. Registration can not be skipped since it is the basis for other more sophisticated activities like student assessment and checking whether a student met the requirements of a particular study program. All the information on which students take which courses HAS to be entered to the system anyway.

4. Course registration is just the first step in the registration procedure. At the next stage students choose particular student groups (classes, labs, lectoria, project teams etc.) which gather at various times and days of the week. This step has a significant impact on a student's weekly schedule and has to be carried out taking his/her preferences into account.

The native registration module of USOS supports the following registration strategy (it is explained on the example of the Faculty of Mathematics, Computer Science and Mechanics):

1. The course catalog for the next academic year is prepared and posted on the web.

2. The course schedule is announced.

3. The registration module of USOSweb is switched on. Students can register for courses, however registration is not yet a guarantee of acceptance.

4. When the registration module is switched off, the authorities get the lists of students asking for acceptance for particular courses and make decisions.

5. If the lower limit on the number of course participants is not reached, the course is cancelled.

6. If there is no limit on the number of students who can be accepted or the limit is not exceeded, all the students are allowed to take the course.

7. Otherwise students are accepted according to some predefined criteria. Usually these criteria are difficult to algorithmize so it is more reasonable to make and input the decisions by hand.

8. The results of the course registration are announced.

9. The group registration module is switched on. These students who were accepted for the course can now state their preferences concerning various groups of that course. Those preferences can be expressed in a couple of ways, e.g. a student may create some variants of his/her weekly schedule and/or list classes which should be chosen with high/low priority and/or list days and time periods which should be chosen with high/low priority (e.g. low priority given to Friday evenings, high priority given to classes starting after 11 am).

10. After a couple of days the module is switched off and the group registration engine assigns students to classes taking into account their preferences and trying to avoid time conflicts.

11. The results of the group registration are announced.

12. There is still another module which may be (optionally) used. This is called **stock-exchange**, although what is really being exchanged are places in particular classes. A student can submit a place in a particular class *for sale* and ask for a place in a different class. After a couple of days the stock-exchange engine is switched on and it automatically matches buyers with sellers (even in cycles longer than 2) exchanging students among groups.

13. This is usually not the end of the registration procedure. Some courses were cancelled, some students were not accepted for the courses they had chosen, other students might have had unresolved time conflicts and had to resign. It also happens that after 1-3 lectures a student decides to drop the course, however has to choose another one to earn enough credit points. Another stage of course registration may be open giving students the possibility to change their initial choices.

Let us summarize the most important features of this procedure:

1. The decision whether a particular student is accepted for a particular course is made off-line. This may be cumbersome but in the past all attempts to automate this process failed. It means that a student has to wait until the end of the course registration to get the results.

2. The registration is two-stage: first a student is accepted for a course, then for a particular student group of the course. During the second stage the student is guaranteed that he/she will be accepted to one of the groups, but not necessarily to the favored one. It may even happen that a student has to drop the course because the group schedule is unacceptable.

3. A student does not pay for registration, at least not directly. Day-time students pay only for failed courses, evening-time students pay a predefined tuition. In particular the system does not limit the number of courses a student can take. It is assumed that a student knows what he/she is doing, knows his/her time limits. If the registration is accepted, the student will be assessed for his/her achievement in that course. Failed courses mean automatically lower grade average.

Such registration procedure works reasonably well at the faculty level, where students take courses which are in line with their specialization. It is relatively easy for the faculty authorities to make decisions whom to accept and whom to reject. Students know where to complain in case they feel they are not treated fair. It is also easier to control the conformance to the regulations.

However, the curriculum of a typical study program involves also some courses which are not offered locally by the faculty to its students, but are delivered by some designated faculties to all students of the university. These are for example foreign language courses which are offered by the Center for Foreign Language Teaching, physical education classes which are offered by the Physical Education and Sports Center or general introductory lectures on philosophy which are provided by the Philosophy Faculty.

The model of registration for such courses is substantially different and needs separate software support.

4 Requirements for the new registration module

Some courses in Warsaw University are offered to all students of the university. Until last year students could find information about the offer only in the printed guides available in the university libraries. Quite often this information was outdated already at the time it was printed. They had to obtain courses' titles, codes, and schedules from such guide, than during the announced period go personally to one central registration point, wait in long queues and register. If any changes were necessary they could only be done in the same way.

Two years ago a small Access-based application was implemented to support the registration for foreign language courses. Registrations were stored in the database but only authorized operators were allowed to log in and input the data. The database was local so no information about students' progress in language courses was available electronically to the faculties. Queues in the registration point were still long during rush hours.

Registration for physical education classes was done only on paper, without any computer-based support.

The complains about this registration system could be heard from all parties: students, teachers, and administration officers. Users complained about:

1. long queues in registration points,

2. problems with getting up-to-date information on courses and course schedules,

3. problems with moving from one group to another after initial registration,

4. over-registration,

5. delayed due payments.

The last issue needs some explanation.

Each student of Warsaw University (with some exceptions) is entitled to 240 hours of foreign language courses free of charge. Also two first foreign language examinations are free. At some faculties a student is not required to attend foreign language courses. He/she is however expected to pass an exam at some level of fluency (for example B1 in English according to *Modern Languages: Learning, Teaching, Assessment. A Common European Framework of Reference*) and get necessary credit points to meet the requirements of his/her study program.

If a student wants (or needs) to take more language courses, he/she has to pay. The cost is determined for each academic year.

During thoroughful requirements analysis and many discussions with the rector's plenipotentiary for the organization of foreign language teaching and the head of Physical Education and Sports Center the following requirements for the new registration module were recognized:

1. Students should register directly for course groups on a first-come-first-served basis. This means that if only the group limit is not yet reached and a student is entitled for registration, he/she gets registered and this registration gets approved straight away (meaning that nobody can expell the student from the group).

2. Students should pay for the registration with **tokens**, which are units of virtual money. Each student gets some amount of tokens of various kinds, for example 240 *foreign language* tokens, 180 *physical training* tokens, 2 *foreign language exam* tokens. Extra tokens can be bought with real money, the price of one token of each kind is determined for every academic year. Each course should also have its price given in tokens, usually this price corresponds to the number of hours.

3. When a student registers, the system should charge him the appropriate amount of tokens of the proper kind. If more tokens are needed than the student possesses, he/she should be given the possibility to cancel or to approve the operation before the due payment is charged. The student should transfer money using his/her unique account number. He/she should be able to print the properly filled money transfer form with a bar code. After transferring money to the given account number, he/she should wait a day or two until the transfer is confirmed by the bank, then should log in again to check that the payment has been received (see [9]).

4. It should be possible to carry out registration in rounds. Each round should last for a given period of time. The system should automatically switch on and off at each round.

5. Tokens should serve one more purpose. It should be possible to use them to set a limit for the maximum number of groups a student can choose in one registration round. This limit could be used to prevent over-registration. If there was no limit, some students might register to many groups, just to postpone the final decision to the very last moment and unregister right before the registration is switched-off.

6. The new registration module should be integrated with USOS, since the necessary data should be obtained from the USOS data repository and further group processing and student assessment should proceed in USOS. In particular course grades should be stored in USOS and made automatically available to all faculties where students' achievements are checked against program requirements.

7. The system should support on-line web-based interfaces for various group of users: students, teachers, course providers, system administrators.

8. The system will be accessed by various users with different level of computer literacy. It should be fault-tolerant and resistant to incorrect data. It should allow for temporary blocking of critical functions while on-line registration is switched on, for example nobody should be allowed to change course fees while the registration is running. The administrator should have special tools for checking data consistency and for recovery from typical errors. In particular the system should log all critical operations for easier bug tracking.

The new registration module has been produced in-house by academic teachers and Ph.D. students of the Faculty of Mathematics, Computer Science and Mechanics of Warsaw University. Some of the team members were earlier involved in the design and development of USOS.

5 Technical aspects of the new registration module

The primary concern of the system designers was its performance. The system was expected to handle approximately 37 thousand students during a period of 1–2 weeks. It was also assumed that each student may want to visit the system many times: to get information, to register, possibly also to change the registration, to check the confirmation of money transfers.

From the architectural point of view the web part of the system comprises a separate database server (by *separate* we mean that it is neither USOS main database nor local USOSweb databases) and the independent web server. This solution not only helps to handle performance problems but also has a positive influence on system security. The web part was implemented using open source tools: MySQL [1] as the database engine and PHP [2] as the programming language.

Because of the limited capabilities of PHP, which does not support caching and has no built-in templates we decided to implement the presentation layer using Smarty Template Engine [4]. Smarty facilitates a manageable way to separate application logic and content from its presentation, making the system easily adaptable to the needs of different universities. Its unique feature is the caching of the parts of dynamically produced pages. It is reasonable to cache partial results of page generation when pages change occasionally. This was the case with the registration module, which contains a course catalog built before the registration was switched on and rarely updated afterwards. Caching also has a positive impact on the system performance.

For data synchronization between MySQL and Oracle we use Perl command-line driven application. *Synchroniser* connects to both databases, reads the data stored at both sites, and then performs proper actions to synchronize the tables. What is extremely important is that it can synchronize databases while both systems are working! For performance reasons some tables in the MySQL database have different structure then tables keeping the same data in the Oracle database. For example data originally stored as a record in a separate table (like teachers name, family name and title) is mapped into a single field stored in another table in MySQL.

For security reasons *Synchroniser* can only access Oracle data using a special set of views. Synchronization of most of the data is quite simple (ie. tables have the same structure and can be updated only in one of the databases). However information about student registrations can be modified at both sites (using either web interface or Oracle forms). In such cases *Synchroniser* decided which database has more revelant data and which of them should be updated. The program generates a log file with detailed information about performed actions.

It may happen that *Synchronizer* will not be able to synchronize data due to encountered critical inconsistencies, like exceeded group registration limits or the same tokens used twice. Such inconsistencies should not occur during the proper system usage (the Oracle interface is blocked for actions which might change registration data while the web registration is switched on), but ocassionally can happen. The system contains a special consistency checker which can be used by the administrator to check the stored data and get the report about possible inconsistencies in a readable form.

The web interface provides a simple search engine, which has been implemented using the full text scan feature of the MySQL server. Log analysis revealed that it had been widely used.

All the information about courses stored in the database can be retrieved in the LaTeX format and used to print a paper version of the course catalog.

The software was constructed in such a way that it is easily maintainable. Most of the system logic is hidden in procedures stored in the Oracle database. Oracle database views hide many technical details from the web part. It makes this web part less sensitive to changes in the database structure. This solution is also more secure since Oracle procedures are not amenable to malicious attacks from the net. In the whole project we used simple and widely known web technologies, which should also in the future lower the cost of system maintenance.

6 Deployment in Warsaw University

The project started in April 2003, the first registration was planned for September–October 2003 (foreign language courses and physical education classes), the next one for November–December 2003 (foreign language exams), so many activities had to be carried out in parallel. The deployment team consisted of the IT specialists from the Software Applications Office and was strongly supported by the authorities of the Faculty of Mathematics, Computer Science and Mechanics and the software development team. The deployment was a complex enterprise from the organizational point of view. Until then USOS was in use (usually in a limited range) on a couple of faculties of Warsaw University. The new application involved all faculties and the majority of students. The following activities had to be conducted in a short time:

1. entering into the system all students of Warsaw University entitled to foreign language courses and physical education (about 37 thousand). Faculties using USOS already had all the necessary data in the repository. Other data was either transferred from local database applications (if available), obtained from the student admission system database or entered manually. Some data was recovered from the old database of the Center for Foreign Language Teaching, in particular it helped to calculate the initial number of tokens granted to every student;

2. generating accounts for all these students and distributing the passwords. This was probably the most complex part of the whole process since the majority of students was on vacation and there was just a couple of days between the end of vacation and the beginning of the academic year for delivering the new passwords. Some faculties distributed passwords to students by e-mail, others used USOSweb, the other printed passwords on paper and distributed them manually;

3. building from scratch the on-line catalog of all foreign language courses, physical education classes, and language exams, comprising the complete schedule, descriptions etc. Altogether there were almost 700 course groups with more than 23 thousand places;

4. installing about 150 USOS clients spread between all Warsaw University faculties and train USOS users. Special trainings were offered to faculty administration officers, course providers, course lecturers, accountants handling financial affairs, authorities responsible for the whole process (faculties' deputy deans for student affairs, rector plenipotentiary responsible for the organization of foreign language teaching, the head of Physical Education and Sports Center).

Two Intel-based servers running Red Hat Linux were set up for the system: one (2 x Intel Xeon, 2GB RAM, matrix controller, 3 SCSI 36GB disks with RAID 5) as the database server and the other (2 x Intel Xeon, 512MB RAM, SCSI controller, 36GB disk) as the web server.

The system ([7]) was made available to the community in the beginning of September and the first round of the registration was switched on at midnight on October 2/3, 2003. The workload was heavy, especially during the first days of the registration (see figures 1 and 2; diagrams were prepared using Report Magic [3] — a freeware tool for generating statistical reports from log files created by web applications). During the first hour (00:00-01:00 am, October 3rd) the web server handled 41220 requests. The daily average was about 400000 pages, some of them requiring heavy calculations. Between September 29 and October 5 the system was referenced 1.7 million times by 20 thousand different sites. There were however no performance problems, even during rush hours.

The problems encountered were mostly of the organizational origin. Some faculties didn't manage to distribute passwords on time. In rare cases the distributed passwords were incorrect because of human errors. Many students had problems with the first logging in, since they confused small letters with capital, zero with capital letter O, one with small letter L. Passwords given to students by faculties allowed to log in only once, then

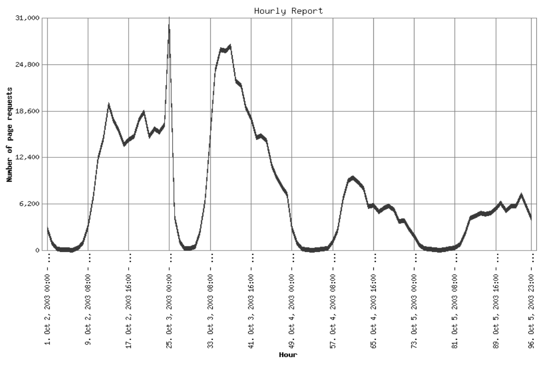

Figure 1: Number of page requests (hourly report)

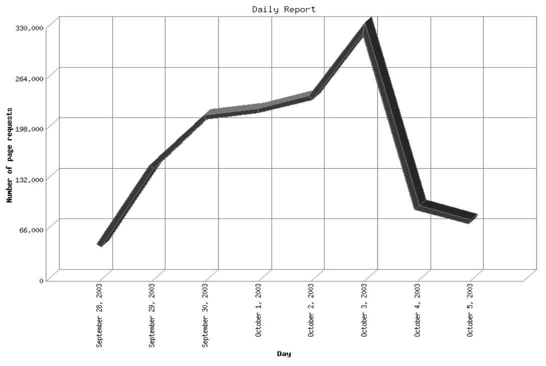

Figure 2: Number of page requests (daily report)

the new password had to be defined. Students somehow missed that step, and later tried to log in with the original password. There were also problems with missing students' personal data, money transferred to incorrect accounts etc.

The system developers read carefully e-mails send to the system help-desk and reacted to them promptly. Some problems of the users could be solved just by simplifying the system interface or attaching help screens.

Some problems were more of psychological nature. There were students and even administration officers who didn't trust in the success of the whole enterprise and tried to break the rules. There were students who didn't transfer money on time, probably believing that nobody will notice it. There were teachers who accepted students to their classes with the omission of the official registration arguing that *it worked that way a year ago*. There were even providers who wanted to change the initially announced schedules at the end of the registration period. However the consistent attitude of the authorities responsible for the deployment helped to solve such problems.

7 Summary

The new registration module was deployed at all faculties of Warsaw University in the summer of 2003 and used during the academic year 2003/04 for registration for courses and exams which were offered to all students of the university (until June 2004 it has been used for six registrations and is now being prepared for the next academic year). It has already been delivered to other universities gathered in the consortium.

The design and development of the registration software was not a complex task, although designers had to solve some sophisticated problems. The deployment was much more challenging, not only becasue of the scale of the whole enterprise but also because it needed changes in old procedures and enforcement of the new ones.

The important aspect of the whole process is that the new system helps to effectively manage finances of the course registration. Authorities get detailed financial reports, can better plan the budget and control the expences.

The system was financed by the consortium and deployed by local university task forces with almost no extra costs. Hardware was relatively cheap. The overall costs were insignificant as compared with the university budget.

The side effect of this enterprise was that it helped to popularize USOS among students, teachers and administration officers of Warsaw University.

References

[1] Home page of MySQL, `http://www.mysql.com/`

[2] Home page of PHP, `http://www.php.net/`

[3] Home page of Report Magic, `http://www.reportmagic.org/`

[4] Home page of Smarty, `http://smarty.php.net/`

[5] Home page of University Study-Oriented System, Warsaw, Poland, `http://usos.mimuw.edu.pl`

[6] Home page of USOSweb, Warsaw, Poland, `http://usosweb.mimuw.edu.pl`

[7] Home page of the new registration system for Warsaw University, `http://lektoraty.usos.uw.edu.pl/`

[8] J. Mincer-Daszkiewicz, *Student Management Information System for Polish Universities*, EUNIS'2002, The 8th International Conference of European University Information Systems, Porto, Portugal, June 19-22, 2002, pp. 271–281.

[9] J. Mincer-Daszkiewicz, *Deploying University Study-Oriented System at Polish Universities*, EUNIS'2003, The 9th International Conference of European University Information Systems, Amsterdam, the Netherlands, July 2–4, 2003, pp. 513–517.

[10] J. Mincer-Daszkiewicz, *Teaching Software Engineering by Developing Commodity Software*, EISTA'2003, International Conference on Educaton and Information Systems: Technologies and Applications, July 31–August 2, Orlando, Florida, USA, pp. 449-454.