CORESOFT: A FRAMEWORK FOR STUDENT DATA

Mike Joy
Department of Computer Science
University of Warwick
Coventry CV4 7AL, UK
M.S.Joy@warwick.ac.uk

Nathan Griffiths
Department of Computer Science
University of Warwick
Coventry CV4 7AL, UK
N.E.Griffiths@warwick.ac.uk

Mary Stott
IT Services
University of Warwick
Coventry CV4 7AL, UK
M.B.Stott@warwick.ac.uk

Jon Harley
IT Services
University of Warwick
Coventry CV4 7AL, UK
J.W.Harley@warwick.ac.uk

Cathy Wattebot
Mathematics Institute
University of Warwick
Coventry CV4 7AL, UK
C.Wattebot@warwick.ac.uk

Derek Holt
Mathematics Institute
University of Warwick
Coventry CV4 7AL, UK
D.F.Holt@warwick.ac.uk

ABSTRACT

In this paper, we describe the Coresoft database, which provides an open framework for the provision of student data for software developed for teaching support. Coresoft arose in response to various University departments' practical need for student data in order to enhance their teaching methods by using computer-aided learning and assessment, along with providing web-based resources. The database contains data provided by the University's central administration, and data provided locally by academic and service departments, and addresses the problems inherent in combining data from such disparate sources. We finally describe the applications that are now being developed using CoreSoft to meet specific teaching needs.

Keywords

database schema, student records, coresoft, sql.

1. INTRODUCTION

University departments which host complex computing activities (e.g. general research and publishing, computational research, computer-based teaching, assessment and management of departmental resources) will have a variety of hardware and software standards into which database software must be integrated. To cater for a wide range of computing platforms, software vehicles such as Java [2] and standard Web languages such as Perl [13] and PHP [3] are preferred, and have facilitated the authoring of software tools used in the management of university courses. Many, if not most, of these tools will process data about students, courses and modules, and there is a requirement for accurate and timely data to be available. However, provision of student data has historically been managed by central university administration who have, for very good security and practical considerations, been reluctant to facilitate its distribution. This can create significant problems for individual departments in obtaining such data, and in ensuring that it is kept consistent with the centrally administered data (especially for departments lacking internal database skills). Although the work described in this paper addresses such problems at Warwick, the work is equally applicable to other universities facing similar problems, and our aim has been to ensure the work is as transferable as possible.

This paper documents an initiative at Warwick, the *Coresoft database*, which aims to make student data available to academics and department administrators in a simple and consistent format suitable for processing by locally written software packages.

3nd Annual Conference on the Teaching of Computing, Loughborough University

2. DEPARTMENT SOFTWARE

Many departments at Warwick have software written in, and for the use of members of, the department. The types of task for which they are used include:

- processing of student marks;
- management of student projects;
- management of seminar groups;
- examination grids;
- support of CAL software;
- support for admissions tutors;
- access to module registration data;
- mailing lists (both email and post);
- miscellaneous report generation;
- management of seminar groups.

2.1 Software Packages

At this point, it is useful to note that in common with other institutions Warwick has a central IT Services facility controlling IT resources across campus, including the network and communications infrastructure, management information systems, student records, and general software provision for both staff and students.

Additionally, individual departments (particularly the science departments) provide their own specialised software as appropriate.

We can identify a number of department packages that use student data:

- the Boss Online Submission System (Computer Science) [8];
- the ATAS Automated Assessment System (Physics);
- the Azulis CAL software (Computer Science and Physics) [6];
- student web portal (Law);
- general course management software (Mathematics, Engineering, Physics);
- membership and facilities management software (Sports Centre);
- course enrollment software (Academic Office).

Each of these has two data components:

- data provided by the central administration through the IT Services (such as names, ID numbers, addresses, and course and module registrations);
- data added locally (such as student marks and seminar group allocations).

These twin components give rise to two main problems. Firstly, some mechanism for obtaining and maintaining current data from the central administration must be established. Secondly, a schema is required to incorporate local departmental data into that provided by the central administration, in order to carry out local tasks, such as those listed above.

2.2 Case Study

Whilst different departments have their own preferences for the software they have adopted, the experience of the Mathematics Institute is not unusual, and illustrates the variety of technologies which have been employed.

The old Mathematics database consists of an evolved FileMaker Pro [4] database of more than ten years' standing, hosted on MacOS [7], and an Ingres [11] database on Solaris [9], linked together via ODBC [12]. In the near future it is envisaged creating Web interfaces to the Ingres database written in PHP or Java.

FileMaker is well adapted for design and use of forms, but is less flexible and less efficient than Ingres in calculation or adaptation to changing external input data from the centre. Both databases are used for look-up and reporting, with the Ingres database preferred for web interfacing and some report programming.

3. CENTRAL PROVISION OF DATA

Student data is held by the central administration on a proprietary student records system (SRS), and is accessed by authorised staff in the various service departments of the University, and also by some departmental administrators and secretaries. Interaction with the data is via a proprietary "tabbing" interface, which is available only on a subset of the Windows platforms. Direct access to the database tables is only authorised to a small number of IT Services personnel.

3.1 Database Structure

The tables which form the underlying database are not designed to be read by untrained persons, and have (for example) mnemonic names for both the tables and their fields which are difficult to remember. For example, the student table, called `SRS_STU`, contains fields `STU_HAEM` and `STU_CAD1` representing "home email address" and "correspondence address line 1" respectively. Clearly, even for a database literate person the use of such table names makes direct manipulation of the database through SQL unnecessarily difficult. Moreover, it complicates the process of developing third party software for departmental needs, should that software need to interact with the database (through standard protocols such as ODBC or JDBC).

Any database package implementation may involve "misuse" (that is, populating fields with data other than that intended), and the Warwick SRS is no exception. The extent to which this is practised varies, and only the IT Services' staff involved with the SRS would be aware of its precise extent.

Export of data from the underlying database is thus not a simple matter of copying the tables, since some processing is required for it to be comprehensible by an untrained person. Moreover, most of the data held is not relevant for department use, being either confidential (such as financial data) or used for statistical purposes (such as statistical returns to HEFCE), and it is appropriate to filter out such data.

3.2 Export of Data

Data from the SRS can be exported to other applications in one of three ways:

- through the menus provided by the SRS user interface;
- by direct online access to the underlying relational database;
- by offline access to the underlying database.

The first of these allows data to be exported in a variety of formats, but cannot be automated to be provided on a regular basis. Moreover, since interaction with the database is through a fixed interface, the type of data that can be extracted is largely predetermined and there is no facility for the execution of arbitrary queries, even where such queries can be guaranteed non-destructive (for example complex `SELECT FROM x JOIN y` statements). The second would not be permitted for security reasons. The third requires skilled intervention by a member of IT Services' staff, but can be automated.

Data is currently provided to the client departments by the export of data in two ways (both examples of offline access to the underlying database):

- CSV format in a text file, where this is straightforward (such as the Sports Centre, which requires little more than ID-Name pairs for University members, together with their status);
- through the use of Perl scripts which nightly populate a client application's database from the central database (this is the method used by the Computer Science Department).

Neither of these is entirely satisfactory, since any change in the data needed requires programming by an IT Services' member of staff. The download of data is further complicated by the differing requirements of the departments which multiply the number of downloads which must be coded by IT staff.

Despite these differing requirements, however, the type of data a particular department requires tends to be broadly the same as the others. Part of the objective of the Coresoft Project is to determine an appropriate "superset" of requirements that meets departments' individual needs while allowing IT staff to code a single set of export scripts.

3.3 Departmental Variations

Some persons active under the Coresoft scheme are not documented by the Central Administration, since they are not members of the University, but exist only in departmental data banks as Associates and Visitors. Their details need to be available when they act as tutors, assessors or postgraduate students for instance. These members may be peripatetic, provisional and temporary tutors or visiting students or people at a visiting stage of a likely appointment.

Additionally, University credit schemes can be complicated. For example, although each module has a "normal" credit value, this may vary depending on:

- the year of study in which it is offered (a module might perhaps be available to both 2nd and 3rd year undergraduates);
- the course which it is delivered as a component of;
- the year in which it is delivered (the credit value may be changed either temporarily or permanently);
- the "mode of assessment" of the student choosing the option (for example, differing proportions of credit coming from examination, coursework and essay or project).

4. Checking Course Regulations

This can be another very complex area in departmental administration. The aim of giving students freedom to choose subjects to suit their abilities and objectives without compromising the standard of degrees awarded can result in complex regulations.

It is necessary to check both for accidental choice of invalid module combinations and also for possible mischievous registrations (for instance, for the same module in successive years of study). Many of these checks are performed by auxiliary software. The exact details of these local conventions may not be held centrally on the SRS database.

5. The Coresoft Project

The aim of the Coresoft Project is to provide a database, accessible to academics and to departmental administrative staff, which would

- be partly populated by IT Services (data owned by IT Services, managed centrally and provided as "read-only");
- contain all data needed by academic departments, both currently and in the foreseeable future;
- be structured, and use naming conventions, which would cause it to be easy to use by database-literate (but otherwise untrained) staff.

This would have the following advantages:

- only one single database scheme would have to be populated automatically by IT Services;

- the software written by the different departments would in principle be portable to other departments, reducing duplicated effort in writing and maintaining the software.

It was also desirable that the structure of Coresoft should be portable to other institutions, in order to avoid duplication of effort.

5.1 Migrating to Coresoft

In order for the departmental packages to migrate to Coresoft the main changes are in the structure of the underlying database and in data entry mechanisms. Reporting scripts and web interfaces also have to be ported.

In the transition to the interdepartmental Coresoft database, reproduction of the functionality of data entry, reporting, data look-up, assessment calculations and checking of course regulation has to be very accurate, or else the whole administrative process can be put at risk. Current systems have developed in an evolutionary way, and perform exactly the tasks required, even though the mechanisms can be improved and updated.

5.2 Design Decisions

Table names and field names have been chosen to match easy understanding of their meanings. For example, table "member" contains information about University members, with fields such as "id", "first_forename", "family_name" and "email_alias".

Although database normalisation was desirable, this has only been achieved to a limited extent; tables with a high normal form are often not easy to use. For example, there are instances of fields that superficially seem calculable from other fields, but in practice are not. A good example is that of a person's initials - what are the initials of "Henry 5 d'Agin van de Court"? There is a trade-off between including initials in the table representing a person (and losing normalisation), and creating a separate table populated only by "unusual" initials.

The concepts embodied in the data are generalised sufficiently to be used in another institution. This is a potential problem, and much discussion of the actual meaning of concepts (such as "module", "course" and "enrollment", for instance) took place.

The database would be in two conceptually separate sections:

- Section A: read-only data provided centrally
- Section B: data provided in the departments

5.3 Implementation Decisions

The main deliverable of this project is a database *specification*, which should be implementable by any standard SQL database (or even a non-standard database such as mSQL [14]). However, most of the participants are using the MySQL [14] database as the engine of choice. Although MySQL is not a full implementation of the ANSI SQL standard [1], it follows the standard more accurately than many, and is sufficient for our purposes; it is fast, and is also free. We have therefore provided scripts for constructing the tables which have been tested using MySQL. By keeping the database structure simple the tables should be exportable to other databases with minimal change (and indeed have been tested with both PostGres and SQLServer).

We have also provided a Java [2] package containing "wrapper" classes and a JDBC [10] interface to a MySQL database, and we expect in due course to provide APIs in a variety of other languages.

6. ISSUES

As we were putting together the Coresoft specification, a number of interesting problems needed to be solved.

6.1 Nomenclature

Prior to this exercise, the department software had a fairly simple understanding of the concepts underpinning their data. For example, the following were regarded as fundamental:

- *Course* — a named degree/diploma programme of study leading to an award.
- *Module* — a named unit of study within a course.

However, the SRS database used a variety of terms with different meanings. Those that related to our initial understanding of "course" included:

- *Route* — a named programme of study (same as course above);
- *Course* — an administratively useful grouping of routes (e.g. all 3 year undergraduate Biology routes leading to the degree of BSc);
- *Programme* — a programme of study characterised by the name of the award (e.g. BSc, PhD, MBA).

6.2 Data Types

In order for the database specification to be portable, it is necessary for the data types to be generic. This has some unexpected practical consequences.

A String data type in SQL can either be of fixed length (`CHAR`), or of variable length (`VARCHAR`). The advantage of the former is storage efficiency if the relevant field size is fixed, whereas if the field size may vary then `VARCHAR` is necessary. Moreover, fixed length strings enable a greater degree of

automatic type checking both by the database and by the client software.

Whereas fields, such as ID-number, at Warwick are of fixed length (7 digits), this constraint may be different for other institutions. We were therefore minded to make *all* string fields of variable length in order to satisfy our portability criterion. However, some SQL databases restrict the number of variable length fields that a single database can contain.

The decision was therefore taken to use fixed length strings where this was clearly appropriate at Warwick, on the basis that this would maximise portability *within* our institution, and export of Coresoft to another institution would require minimal editing.

At present there is no standardisation even at institution level on lengths of names or lines of addresses. The provision allowed in Coresoft is generous and the task of adapting to possibly limited space in reports or forms is left to the interface software. We chose to follow the *de facto* standards for such data and an address is stored as four fields for the first lines of the address, plus one for postal/ZIP code and one for country. Whilst not a particularly elegant solution, it simplifies the import of data.

There needs to be room for extension of the Coresoft scheme for individual departments with special requirements, and we expect this to emerge as extra tables in due course. We do not envisage significant changes to the current tables.

The inputs to Coresoft do not currently allow for correct rendering of non-ASCII characters in data (this is a feature of the SRS database and for the time being will remain a feature of Coresoft).

6.3 Data Protection

The University is registered under the Data Protection Act 1998 [5], and the data held by Coresoft is covered by that registration.

Under the Act, students have a right of access to their personal data held in Coresoft. This is an issue which the participants are aware of.

7. ROLLOUT

The Coresoft database is in use. It already provides the data for the BOSS online submission system and for the Law School's student web portal, and will shortly be deployed for the Academic Office's software for managing course enrollment. Other packages, such as the ATAS and Azulis CAL package also use Coresoft through the APIs written for BOSS. The Mathematics Institute is actively evaluating the incorporation of Coresoft into its own packages.

Working with "real" data is always a hazardous exercise, and nightly downloads of data, and their porting to the Coresoft schema have provided us with a most interesting exercise, the result of which is that this task is now principally debugged. However, issues such as the inclusion and management of incorrect and inconsistent data are ongoing. We expect to be able to offer reliable uploads of data to participating departments from autumn 2002.

8. CONCLUSION

The participants at Warwick have agreed a preliminary specification of Coresoft; this can be viewed, together with a SQL creation script (tested with MySQL), at:

`www.dcs.warwick.ac.uk/coresoft`

The Coresoft project is essentially a pragmatic response to the database requirements of educators in one University, and we believe it has fulfilled its purpose, in that it has enabled University departments to develop computer-aided and web-based learning tools to enhance teaching.

Finally, the Coresoft project aimed to be applicable to other institutions, and authors would be very interested to know whether the model adopted is indeed portable to other institutions.

9. ACKNOWLEDGEMENTS

Other staff involved in this project have included Lee Earl, David Epstein, John Rawnsley and Jim Robinson.

10. REFERENCES

[1] ANSI, D*atabase Language SQL*, ANSI X3.135-1992 (1992).

[2] Arnold K. and Gosling J., *The Java(tm) Programming Language (Second Edition),* Addison-Wesley, Reading, MA (1998). http://java.sun.com/

[3] Choi W., Kent A. Lea C., Prasad G. and Ullman C., *Beginning PHP4*, Wrox Press, Birmingham (2000). http://www.php.net/

[4] Hester N., *FileMaker for Windows and Macintosh (2nd Edition),* Peachpit Press (2000).

[5] HMSO, *Data Protection Act 1998*, Stationary Office, London (1998).

[6] Joy M.S. and Luck M., Effective Electronic Marking for On-line Assessment, *ACM SIGCSE* **30**(3), 134-138 (1998)

[7] Langer M., *Mac OS 9*, Peachpit Press (1999).

[8] Luck M. and Joy M.S., A Secure On-line Submission System, *Software — Practice and Experience* **29**(8), 721-740 (1999).

[9] Mulligan J.P., *Solaris 8*, New Riders (2001). http://www.sun.com/

[10] Reese G., *Database Programming with JDBC and Java*, O'Reilly, (1997)

[11] Rothwell D*., INGRES and Relational Databases*, McGraw-Hill (1991).

[12] Sanders R*., Hands on ODBC 3.5 Developer's Guide*, McGraw-Hill (1998).

[13] Wall L., Christiansen T. and Orwant J., *Programming Perl (3rd Edition),* O'Reilly (2000). http://www.perl.com/

[14] Yarger R.J., Reese G. and King T*., MySQL and mSQL*, O'Reilly, (1999). http://www.mysql.com/