



JISC Modular e-Administration of Teaching Final Report

Editor

Dr. Amyas Phillips, CARET, University of Cambridge¹

Contributing Authors

Anne Clarke, CARET, University of Cambridge

Dr. Laura James, CARET, University of Cambridge

Matthew Jones, CARET, University of Cambridge

Dr. Avi Naim, CARET, University of Cambridge

Dr. Amyas Phillips, CARET, University of Cambridge

Developers

Guy Chisholm, CARET, University of Cambridge

Anne Clarke, CARET, University of Cambridge

Matthew Jones, CARET, University of Cambridge

Dr. Avi Naim, CARET, University of Cambridge

Project Leads

Dr. Laura James, CARET, University of Cambridge

John Norman, Director of CARET, University of Cambridge

Dr. Rachel Padman, Department of Physics, University of Cambridge²

Prof. Richard Prager, University of Cambridge Engineering Department³

Contact

Dr Laura James, laura@caret.cam.ac.uk

¹ Centre for Applied Research in Educational Technologies, 16 Mill Lane, Cambridge CB21SB, UK

² Department of Physics, Cavendish Laboratory, J J Thomson Avenue Cambridge CB30HE, UK

³ University of Cambridge Engineering Department, Trumpington Street, Cambridge CB21PZ, UK

Table of Contents

Acknowledgements	3
Executive Summary	4
Background	5
Teaching Duties Module.....	5
Student Choices Module.....	5
Project Partners	6
Aims and Objectives.....	6
Methodology.....	6
Trunkless Development Model.....	6
Evaluation.....	7
Implementation.....	8
Teaching Duties Module.....	8
Student Choices Module.....	9
Outputs and Results	12
Summary	12
Teaching Duties Module	12
Student Choices Module.....	13
Evaluation	14
Outcomes	16
Conclusions.....	17
Appendices.....	17
References.....	17

Acknowledgements

The authors would like to thank JISC and the Institutional Innovation programme, which made this project possible.

The authors would also like to acknowledge the leadership contribution of Prof. Richard Prager, originator of the Teaching Office Database and the e-Administration of Teaching project, and Deputy Head of the Engineering Department (Teaching).

Dr. Rachel Padman, Academic Secretary to the Physics Teaching Committee and holder of numerous other positions throughout the University, has been of great assistance throughout the project in all matters concerning the Teaching Information System and its Student Choices module.

The contribution of staff time to the project by participating departments has been invaluable. The authors would like to thank the Departments of Physics, Engineering, Veterinary Medicine, Biochemistry and Chemistry, the Faculties of Divinity and English, and the Judge Business School.

In particular, we have also to thank our departmental contacts for their sustained interest and feedback: Helen Marshall (Physics), Jennifer Pollard (English), Carmen Neagoe (JBS), Rachel Tuley (Engineering), Dr. Sandra Fulton (Biochemistry), David Goode (Divinity), Katheryn Ayres (Vets) and Dr. James Keeler (Chemistry).

Executive Summary

Most universities have a central IT function. Despite this, or sometimes because of it, individual departments and academics are frequently motivated to develop 'home grown' IT systems, to fulfill unmet needs or to support particular departmental processes. These systems are often small and minimally complex, but highly fit for purpose, and likely to be of wider benefit.

This project took two such e-administration modules, originally developed (or part-developed) for internal use by the departments of Physics and Engineering at the University of Cambridge, and sought to extend their use to other departments, including those which would not have considered developing their own software internally.

The Engineering Department took the lead in developing and deploying a staff teaching duties allocation module generally known as the Teaching Office Database, or TODB. The Department of Physics took the lead in developing a module for recording and managing student's course options, known as the Student Choices module. The Centre for Applied Research in Educational Technologies (CARET) provided project co-ordination and development staff.

Student Choices attempted to follow a standard software development and deployment process, in which the needs of a variety of possible users are met (or traded off) within a single piece of software. Users would be expected to maintain the project beyond the development phase using common OSS practices, contributing patches to a central version or 'trunk'.

TODB followed an innovative 'trunkless' model. Instead of complicating the software by making it meet many peoples' needs at once, each user received a version customised for their specific needs. By this expedient the software is made simple and is easily related by users to their working practices, so that they are comfortable taking full ownership of it themselves. At the conclusion of this part of the project, TODB has been successfully customised and deployed to seven different departments.

Although the Student Choices module did not find users beyond the Physics department, its development has served as motivation and specification for the University's centrally-administered student information system, which has decided to offer similar functionality to the University as a whole. The inclusion of Student Choices in the e-Administration project also provided a valuable alternative perspective on sustaining 'organically grown' IT systems.

Background

Most universities have a central IT function. Despite this, or sometimes because of it, individual departments and academics are frequently motivated to develop 'home grown' IT systems, to fulfill unmet needs or to support particular departmental processes. These systems are often small and minimally complex, but highly fit for purpose, and likely to be of wider benefit.

This project took two such e-administration modules, originally developed (or part-developed) for internal use by the departments of Physics and Engineering at Cambridge, and conducted an experiment in extending their use to other departments, including those which would not have considered developing their own internally.

The Department of Engineering took the lead in developing a staff teaching duties allocation module generally known as the Teaching Office Database or TODB. The Department of Physics took the lead in developing a module for recording and managing student's course options, known as the Student Choices module.

Teaching Duties Module

Teaching loads need to be distributed in a way that is fair and reasonable. Sometimes, the distribution needs to be *seen* to be fair by staff. This web application supports this allocation of teaching duties and enables everyone to see what they and their colleagues are assigned to. Its advantages over the spreadsheets used by most departments are that it is web based and designed for this specific purpose.

The module centres around two main database tables: teaching staff and teaching duties. Duties are assigned to staff, and further associated with the delivery of course units. Each teaching duty may be weighted with a "point score" representing the nominal amount of work involved, which may be fixed or a function of the number of students taking the course. The system reports teaching load for each member of staff and each curriculum element.

Data are stored by academic year, and teaching points sums are presented for each year for which there are data, facilitating year-on-year comparison. Jobs may also be associated with information such as the subject group to which the job belongs, the job type (e.g. lecturing, administration, practicals) and the term in the academic year when it is performed, to enable further reporting.

The TODB does not make resource allocation decisions itself, but it facilitates the teaching allocation decision-making process by making appropriate information available in an appropriately-structured manner. It provides answers to questions about teaching allocations and teaching load, and can do so on demand during the teaching allocation process.

Student Choices Module

A number of degree programmes at Cambridge offer a broad choice of modules in 'Part I', which comprises the first one or two years of a degree and offers a basic undergraduate grounding in a field. Students will often receive teaching from multiple departments during their Part I. 'Part II' of an undergraduate degree at Cambridge features advanced study and research elements designed as training for academic research, and has traditionally been delivered within a single department, as have 'Part III' or Masters'-level courses.

Course structures in Parts II and III are increasingly changing to offer students the option of following routes into new interdisciplinary fields, or even to non-academic careers. Part III Physics for example shares courses with the departments of Chemistry, Materials, Earth Sciences, Astronomy, Engineering, Mathematics and the Judge Business School. This significantly increases complexity in two areas: 1) in arranging small-group teaching and/or classes to review the solutions to exercises;

and 2) in managing examination entries and monitoring them to be sure that all students are registered for the required number and combination of course units.

The department of Physics' Teaching Information System (TIS) currently handles the first part of this process, while the Engineering Department's course management system, COMET, currently provides facilities for managing the second. The intention of this module was to integrate these features in a way which could be offered to Physics and then other departments, enabling students to register and validate choices for courses up to a particular deadline each term, and administrators to make appropriate teaching arrangements.

Project Partners

The originating departments (Engineering and Physics) of these e-administration tools provided software serving as a jumping-off point for further development or as a specification-in-kind. Six further departments (Chemistry, the Judge Business School, Divinity, Materials Science, English and Veterinary Medicine) expressed an early interest in TODB systems. The Centre for Advanced Research in Educational Technologies (CARET), who have experience supporting the spread of new educational facilities, and of centralised support for educational technology and linking departmental initiatives to central services, provided a 'pool' of skilled developers, project management and other support.

Aims and Objectives

The goals of the project were

1. to develop modular and extensible web-based tools to facilitate the administration of teaching in higher education departments
2. to explore a 'grass roots' or 'organic' model of software development for this domain,

The project aimed to develop two existing e-administration tools, conceived and developed within departments for internal use organising staff teaching duties and student enrollments, into functional, easy-to-use and well adapted systems available to, and used by, other departments.

These tools were to adopt a modular architecture facilitating their adaptation to local circumstances, maintenance and extension at need by local support staff, and connection to other systems. They were also to be made available as open-source source code and documentation for adaptation and use both within Cambridge and at other institutions.

These aims have remained intact throughout the project. Although the teaching allocations tool proved far more successful than the student choices tool, the comparison provides some useful lessons.

Methodology

The project sought to explore ways of supporting 'grass roots' development, maximising its benefits through broad deployment, and sustaining it into the future.

Trunkless Development Model

The TODB project consciously set out to explore an unconventional development and distribution model, based on the hypothesis that although user-originated software solutions may lack polish they are often focussed, effective tools, easy to adapt because free of 'bloat', and worth sharing. The question is, what is an effective way of sharing these home-grown systems?

A traditional open-source approach, if the project gains users and contributors, entails new communication overheads, additional functional requirements, and abstraction in support of configuration options to support a more diverse user base. This is both necessary and efficient for big projects, but not all projects have a naturally large constituency, and a small project which goes down this route may become significantly more complex in order to realise relatively little benefit. Sharing without collaborating, on the other hand, maintains the project scope, but at the cost of being useful to fewer people. The TODB project developed a third, 'trunkless' approach, somewhere between these two.

The wider a project's user base, the bigger and more complex it tends to get, endangering the ability of its users to adapt and maintain it locally to their specific needs and ways of working, in which its original value lay. Instead of establishing a 'trunk' designed to satisfy all users but which inevitably constrains many, TODB adopted an approach of bespoke tailoring for every deployment. Each instance is unique, but because of this it can be free of complex abstractions and checks and simple enough that its users are readily able to maintain and adapt it locally. A many-trunked Banyan is a more appropriate tree-based metaphor for this model.

The practical features of this model can be summarised as follows:

- Original software is used as a base, following limited genericisation, for subsequent development and customisation. If this is not possible, it is used to provide a 'specification by example' from which to develop anew.
- The aim when genericising the originator software is to produce a common starting point for subsequent customisation, not a one size fits all software application able to address all variations at a single pass.
- Software is customised by project staff for each department, to accommodate its specific teaching administration processes and structures, based on one-on-one interaction between these same project staff and teaching administrators in each department.
- The software is designed to be easily comprehensible and adaptable, to accommodate specific departmental working processes and patterns, rather than imposing them. It is the tool of its users and does not arrogate to itself decisions or ways of working.
- The software is intended to be managed, maintained and possibly modified locally by the departments using it. The deployment process includes engagement with computer officers to empower them to support and possibly extend the software.
- Departmental computer officers are not expected to be developers, but to have basic, probably self-taught, PHP skills.
- "The source code is the configuration language", being extremely simple, procedural and standard. It does not make use of abstractions or frameworks, and does not require knowledge of libraries other than some of the standard PHP ones. Anyone making adjustments, be that computer officers, tech-savvy summer students, or short-term contract programmers can easily build a mental model to guide them.

Evaluation

The project plan proposes two top-level evaluation questions:

1. Do the modules meet the original needs of the departments? Did the project adapt to changing needs or changing understanding of requirements, and work appropriately within the institutional environment?

2. Has the project discovered useful information about the modular system of software development as opposed to the conventional process of centralised institutional IT? Has the project communicated the learning? Are other departments at Cambridge and beyond able to take advantage of the learning and modules?

Although the second has a meta-analytical component, both were addressed by asking project participants about their experience and opinions on the project. Results of an initial questionnaire were followed up in individual semi-structured interviews with departmental champions, conducted by a CARET evaluator not previously involved in the project. The results of this work were presented at a closing workshop for validation and further comment before incorporation in this report's conclusions.

To guide our evaluation, we broke the top-level questions into eleven smaller ones:

1. What motivations caused project partners to be interested in the software?
2. Are there needs the software still does not meet, or further modules which would be useful?
3. In what ways was the process of customisation valuable?
4. How difficult was it to produce 'generic' versions of software from departmental originals?
5. How much customisation was necessary for adoption by other users?
6. Were the modules successful, and what lessons can be learned from the differences?
7. How will the modules be maintained and developed in future?
8. How should they be made available elsewhere?
9. Under what circumstances is this model of software development for HEIs better than alternative models?
10. How do the costs and value of this model compare with alternative models?
11. Have the project's outputs and lessons been shared with the wider HEI community?

Implementation

The first phase of development was to study the originator systems and assess whether to use them as a jumping-off point for further development, or simply as specifications by example. It was apparent that TODB in its original form was a useful starting point, whereas that the student choices module was far less developed, being only partially implemented, across two separate environments, and thus requiring further specification.

Teaching Duties Module

The Engineering department as originator was naturally closely involved in development of this module, but partner departments, recruited before the project began, were equally engaged as the software was deployed to them. This is key to the sustainability of the 'trunkless' development model conceived by Prof. Richard Prager, who directed this half of the project.

As a lightweight grassroots enterprise the project took a 'deploy early, iterate rapidly' approach during its deployment to departments, following an initial phase of genericisation and limited refactoring.

To support adoption of project outputs, project partners were assisted with initial customisation by project staff, who, importantly, were able to both discuss particular needs and alternative models and make requested configuration changes and/or customisations. Project partners were also provided £5k funds to defray the costs of hosting and mitigate the risks of diverting staff time.

For TODB the genericisation process involved refactoring of the codebase for improved flexibility, documentation for subsequent users, elimination of Engineering department-specific features and,

importantly, collection of configuration parameters out of the code and into easily-located and edited configuration files. This process was necessarily a 'best guess' based on the development team's judgement and knowledge of practices in other departments, but served to provide a 'good enough' starting point for subsequent deployments.

TODB was taken to each of its partner departments in turn (rather than in parallel), each being assisted to adapt the tool to local practices. In order to facilitate the customisation process, initial deployments in each case were hosted by the development team and remotely accessed by the departmental users for testing and feedback. Once this was complete and the partners had got themselves in a position to host the tool locally, it was packaged and sent to them to deploy.

Every participant provided valuable feedback and had unique requests during deployment and customisation, and where these were judged to be of general benefit they were integrated back into the 'generic' version of the tool, and from there propagated out again to already-deployed instances. Thus the 'generic' codebase developed and benefitted from experience in the field even though nominally distributed according to a 'trunkless' model.

These upgrades were not disruptive to users, who received individual support from the project developers to deploy them. However, they did consume a considerable amount of developer time. Improvements to the generic codebase after project end will not necessarily be easy to distribute to existing installations, who may theoretically require 7 different versions of a patch. Unless users run into burning issues, it is unlikely that they will be interested in such updates.



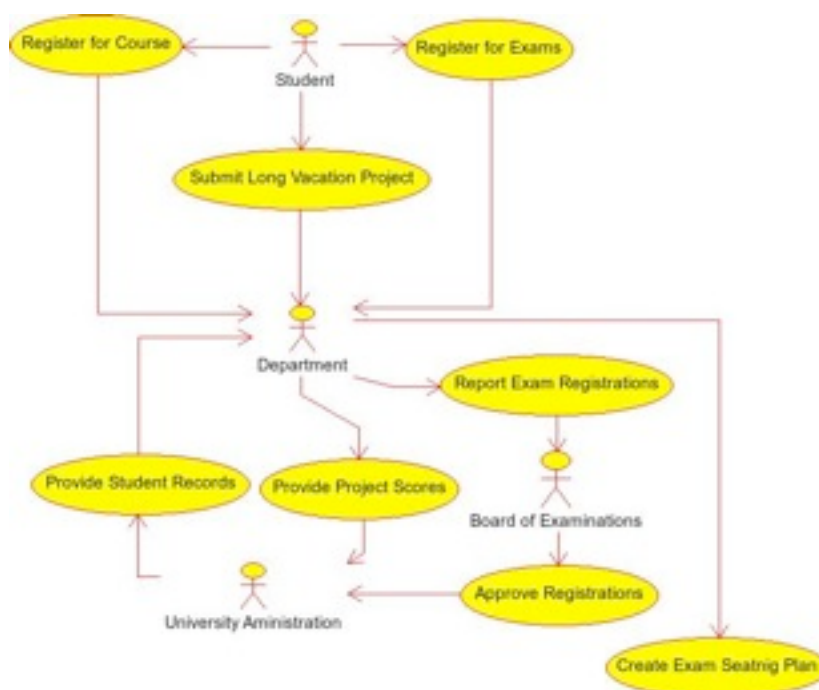
The Teaching Office Database system (TODB)

Student Choices Module

The project planned to take pre-existing elements of the Physics department's course and student management system, the Teaching Information System (TIS), plus parts of the Engineering department's equivalent 'COMET', and from them create a modular "student choices" web application, which would enable students to select their modules for future years of study in an informed way, and for administrators to know about their students' choices as soon as possible to facilitate planning.

The TIS is a web based tool which allows an administrator to easily view the assignments of students to courses, Parts, and Directors of Studies (College officers responsible for academic guidance and College-based teaching). Directors of Studies and lecturers are given the ability to communicate (via e-mail) with all of their students at the click of a button. Staff can quickly find out who is lecturing what, heads of class, examiners, supervisors, student representatives and membership of departmental committees. Students have a limited access to the system, centered around their personal information relating to their Part, course, projects and practicals.

In addition the TIS manages documents such as course handouts, which are available for students to view and download. Administrators also use it to advertise the departmental calendar, long vacation projects and student and staff surveys.



The Teaching Information System

We worked with Physics to determine an initial specification for Student Choices, as far as possible using the existing TIS as a specification by example, because it demonstrates functionality and interface design which academics and students are familiar with. A number of other departmental teaching officers were approached with this specification but none were sufficiently motivated by departmental pressures or potential benefits to engage more fully.

A prototype student choices module was used to confirm that we had correctly captured the desires and requirements of our stakeholders. Integrating feedback, we developed a final system in PHP and MySQL. This was designed to connect to the Microsoft Access database used by the TIS (a relic of that system's simpler origins, now something of a limitation) but also to be adaptable to run in stand-alone mode or connected to alternative student information systems.

This system was promoted to departmental teaching offices, both on its own and, subsequently, as part of TIS, at a variety of departments including History and Philosophy of Science, Criminology, Zoology, Biochemistry (representative of many biological sciences departments), Law, Geography and Maths.

After following up “back of box” software feature lists in user-friendly language with individual discussions exploring current pressures in the area of teaching information management and administration, we found that the need for this kind of system was quite limited. Some departments described scenarios where they have so few students that interposing software in the process would offer only negative benefits. In other departments, it was felt that existing, central university systems were perfectly adequate.

At this point it became clear to us that although apparently solving a common problem, departments other than Physics did not want such a system. A contributing factor to the under-appreciation of this risk was perhaps that, unlike TODB, Student Choices did not exist in a prototype form that could be demonstrated to potential partners, who consequently were faced with an unproven solution to a problem they weren't convinced they shared.

Departments did describe problems around managing **groups** more generally, for various activities, where groups could be anything from supervision groups, to “people who have an interest in this lecture”. Use cases around groups often involved communication (email this list, notify these people of a room change) or sharing (provide this file to this group).

Interpreting groups as a generalisation of a problem solved by TIS and COMET and common across numerous functions throughout the University, we have pursued the concept in parallel with the main thrust of this project, judging it to be of greater import to the University (and the community) than the specific offering of the Student Choices module. There is campus interest in group management, and some design conceptualisation was already in place, as a key component of next generation Sakai, on which Cambridge's VLE CamTools is based. The project team spent some time developing this into various use cases involving groups, as potentially a better (more sustainable) way of meeting campus needs than the genericised TIS-based software we had initially anticipated.

Ultimately, the Student Choices module has been deployed only in the Physics department. Without the wider viewpoint afforded by involvement of other departments, development of Student Choices despite the programmers' best efforts (e.g. running on an independent database) became implicitly dependent on the Physics department environment of systems and processes, specifically the Teaching Information System. Student Choices at present would need refactoring to stand alone, and in the process made harder for Physics to maintain, or else distributed as part of the TIS, which is not well documented and has not been placed in the public domain.

The process of deploying Student Choices to Physics has stimulated central University action in the problem area it addresses. Because of concerns over appropriate registration and reporting of student numbers to HEFCE, and an emerging policy that all student information should be held in the University's student information system, the project has caused the institutional providers of the SIS to acknowledge the problem as being within the SIS's domain, and start to work with our client department (Physics) on a solution. However, this action has also rendered our developed software redundant.

We will continue to monitor developments in this area CARET intends to continue to work with University stakeholders to monitor and support this delivery, and it may yet prove that our software is a necessary component of a final solution; but we would be happy with any outcome which solves the problem of student choices in Physics.

Outputs and Results

Summary

1. Robust, easily adaptable Teaching Duties module used by seven university departments, available with usability-tested documentation for both users and developers under OSS licence, for anyone wanting to use the software in future
2. Requirements definition for anyone wishing to implement similar functionality in future
3. Lessons on using an organic, trunkless development and sustainability model, for anyone using or considering a similar approach in future
4. Initiation and catalysis of central University provision of Student Choices functionality will ensure future needs in this area are answered effectively for all departments.

Teaching Duties Module

The system has been deployed to and evaluated by eight departments, who have either indicated their intention to continue using the software, or continue to evaluate the software with a view to adoption.

The **Engineering Department** developed the software originally as a response to their needs so it is practically certain that they will continue to use the software for the foreseeable future.

The **Judge Business School** has indicated that it will continue to use the TODB for the 2009-10 academic year, 'side-by-side' with their existing teaching allocation software, pending the outcome of a current triennial review of teaching policy. JBS and Engineering share some teaching duties so it is mutually beneficial for both departments use TODB, which can automatically share information between such instances.

The **Faculty of English** has installed a local copy of the TODB and has indicated a commitment to use the TODB in 2010-11 and to continue using it thereafter.

The **Faculty of Divinity** intend to use the TODB as their primary staff database and as a source for printable timetables from 2010-11 following evaluation in 2009-10, and expect to continue to use TODB for the foreseeable future.

The **Department of Veterinary Medicine** is, interestingly, not using TODB for its original purpose. Having conducted an evaluation they concluded that as a small department their teaching allocation problems are less of fairness and transparency than of dealing with the problems of often having only one person able to teach each course. However, they continue to use it for its timetable generation feature.

The **Department of Chemistry** has conducted an evaluation and comparing TODB to the previous manual approach recognise that the TODB could make it an easy and quick process. They have indicated that they plan to use TODB to simplify teaching allocations for the 2010-11 academic year.

The **Department of Physics** are currently evaluating whether the TODB meets their needs in the context of their Teaching Administration System, which although mostly concerned with student information might naturally be perceived as the place in which TODB-type functionality would be implemented.

The **Department of Biochemistry** were not originally partners in this project but actively requested the software, rather than merely accepting it for evaluation. The TODB is viewed as a solution to teaching allocation problems and has accordingly enjoyed much attention and enthusiasm. It is very

likely that they will continue to use the TODB in a sustained manner. Arrangements for hosting the TODB locally are currently underway.

The **Department of Earth Sciences** have come even later to the project but are keen: they may be the first to 'self-deploy' without the benefit of funded developer time.

Student Choices Module

Although ultimately student choices has not been deployed, its development has motivated and informed development of central institutional IT systems, both for its specific functions (within CamSIS) and around the general problems of groups and departmental variation vs. institutional consistency (within Sakai 3).

Instead of strictly local systems, CARET has developed thinking around a model where data is stored centrally in a robust, backed up setting, but where departments still retain control of the user interface, adding new functionality around their data. It is a commonplace of the Enterprise Services field that a high degree of customisation is compatible with linked systems provided the APIs and underlying meaning of data are maintained.

To some extent, TODB demonstrates this for ad-hoc alliances, for instance between Engineering and the Judge Business School, but a centrally-curated data store could be achieved using a CollectionSpace⁴-style layered model, with a "middle" layer constructed in a language which could be accessible to computer officers and technically-minded academics (such as javascript), enabling simple configuration of the system, including the user interface, without the need to engage with the more complex software and organisational tasks of institutional data storage and integrity. Departments would retain individual control of software and how they use it, within the limits of the data model.

Centralised data storage with APIs of this kind could fall within our next generation Sakai 3-based VLE. Sakai 3 is built on Apache Sling⁵, itself built on Java Content Repository⁶, which will potentially offer schemaless data capability and greater flexibility than, for example, the LAMP stack of the TODB. It also offers a potential future migration path into central services also based on JCR, enabling us to test the institutional software model of a lightweight customisable front end with central back end data store, as opposed to the TODB module style.

A general-purpose system to support activities around groups could be combined with this central storage model, and we consider that the Sakai⁷-based VLE may be a good home for the 'groups' work, and indeed an entire future TIS, using the "central data store with configurable UI" model. This work will continue beyond the end of this current project, at CARET and in the wider Sakai community.

Student Choices also provides something of a counter-example against which to contrast TODB's successful development and deployment of 'organically sourced' software, making a significant contribution to the lessons learned in that exercise.

⁴ <http://collectionspace.org/>

⁵ <http://sling.apache.org/site/index.html>

⁶ <http://jackrabbit.apache.org/>

⁷ <http://sakaiproject.org/>

Evaluation

Lead contacts / project champions in each department to which TODB was deployed were contacted towards the end of the project and asked both to reflect on their experiences in both a questionnaire and a semi-structured interview, conducted by an evaluator not previously involved with the project. Because of the small number of respondents, we do not attempt a quantitative analysis but use these data to highlight areas of particular interest. The summaries are organised by the corresponding evaluation questions. It was not possible to ask similar questions for Student Choices, but we have attempted to set the responses within this wider context.

1. What motivations caused project partners to be interested in the software?

Departments cited a number of factors including the emplacement of a new teaching officer looking to establish an effective way of working (1 department, plus Engineering for which a similar situation led to the initial creation of TODB), instructions from the head of department who had been convinced (1 department), an 'early adopter' teaching officer, unique functionality over and above existing systems (3 depts, particularly JBS). In all cases personal contacts were important in establishing the initial grounds and opportunity for evaluation.

Respondents also commented that the existence of a working system made it easier for them to engage.

"Mostly the module was a very good fit for JBS already" Carmen Neagoe, Teaching Office Administrator, Judge Business School, Cambridge

2. Are there needs the software still does not meet, or further modules which would be useful?

Four departments (Biochemistry, Divinity, English, Vets) expressed an interest in linking TODB to timetabling, and in one case, room booking applications in the form of the open-source MRBS⁸. In many ways this is a natural extension of the system, which in most deployments is the only IT system which knows about detailed teaching duty allocations. The benefits in non-duplication of work, simply by having timetabling and room booking together in one system, were felt by these departments be substantial.

This is an area CARET intends to investigate further, outside the scope of this present project.

Some timetable-entry and display features were implemented at the request of the Divinity Department, but some other departments offered these found them difficult to use. Usability and UI polish were a general concern for two departments, particularly the JBS who had a pre-existing, custom-developed Excel based system for comparison.

3. In what ways was the process of customisation valuable?

Biochemistry, the last department to receive a deployment, commented on the value of the experience and advice the developers were by that time able to share. The dual role of the developers as consultants was highlighted strongly as a positive and effective element of the project by two departments; English and Biochemistry.

"I'm grateful that [the project team] have been so involved with the individual faculties. As much as we're all a part of the same University, we all do things quite differently!" Jennifer Pollard, Computer Officer, English Department

No departments cited any 'red line' features whose absence was preventing them from adopting the system, but although the provision of custom features was a key part of the project, only JBS highlighted this as necessary for their adoption of the module, perhaps

⁸ <http://mrbs.sourceforge.net/>

because they had particularly demanding requirements (including connection to the Engineering TODB, and ways of handling 'contractual stint' with custom reporting and resolution features, such as credit, deficit, 'buy out' and 'roll over'), and an existing system to match and exceed. From the point of view of the English Department, though, the value lay as much in de-risking the project and improving the chances of a positive outcome.

"Knowing that the end product was going to be tailored to our specific needs and processes was a great relief, from my perspective." *Jen Pollard, Computer Officer, English Department*

4. How difficult was it to produce 'generic' versions of software from departmental originals?

This question is addressed in an accompanying report comparing approaches for supporting 'grassroots' development processes.

5. How much customisation was necessary for adoption by other users?

This question, which can only be answered for the TODB project, is addressed in an accompanying report describing the functional divergence between departments. Exact detail is available in the document "TODB modification journey".

6. Were the modules successful, and what lessons can be learned from the differences?

TODB was successful in terms of its initial aims, and while Student Choices was not successful in the way originally imagined, the outcome for Physics is satisfactory, and the exercise of contrasting and comparing with TODB has yielded valuable insights for ongoing development of institutional IT (discussed in more detail in the sections immediately preceding this).

7. How will the modules be maintained and developed in future?

Biochemistry, English and Divinity - all departments with a small IT team or just a single computer officer - were all confident of being able to maintain their software as necessary but had small expectation of developing new features themselves. JBS and Engineering saw this as a being possible, given sufficient impetus.

"It will all be down to how useful it is; which is how it should be - departments which find it useful will resource it. I don't think there will ever be enough central resource" *Prof. Richard Prager, Deputy Head of the Engineering Department (Teaching), Cambridge*

Participants understood that taking responsibility for their own TODB instances was a necessary part of the trunkless deployment process, and accepted it as the price of uniquely tailored software obtained in this way. They have subscribed to a mailing list for mutual support, and while CARET is also a member of the list it is understood that this is an informal arrangement. In an example of this support model in action, the Faculty of Divinity has assisted the Department of Veterinary Medicine in deploying TODB on an Apple Macintosh server. CARET will also publicise a contact email⁹ (via Google Code sites, see below) for new users to get help getting started with TODB, both within the University and externally.

8. How should they be made available elsewhere (if at all)?

No participants expressed any desire that the administrative processes of their departments be kept confidential, perhaps because these are relatively low-stakes and internal functions.

9. Under what circumstances is this model of software development for HEIs better than alternative models?

This question is addressed partly in the conclusion to this report, and in more detail in an accompanying report analysing our experiences with the TODB development model. Only one participant - the Faculty of Divinity computer officer - felt that a centralised system could be more effective.

⁹ todb-support@caret.cam.ac.uk

10. How do the costs and value of this model compare with alternatives?

A straw poll of participants in the evaluation workshop suggested they valued the software, support, customisation and training at just £1500, albeit with the major reservation in most cases that it would be highly unlikely that departments would even consider making such an investment. The question was framed in terms of how much investment they felt they could make a case for, to obtain similar outcomes. The ability of software as an instrument with which to achieve institutional change was considered a major intangible benefit, however - for example, the review of teaching associated with the introduction of a points system in the Faculty of English, or the rebalancing of teaching effort to avoid a concentration of resource on Part IIB (year 4) of the Engineering degree.

11. Have the project's outputs and lessons been shared with the wider HEI community?

The project team has distilled and shared the lessons of user-originated software development and benefits realisation, 'groups' thinking, and teaching duties allocation in this report, the project blog¹⁰, the Sakai Foundation wiki¹¹, and in three community meetings: [JISC Institutional Innovation Exchange](#) (Jan '10), [Modular e-Administration of Teaching Assembly](#) (Dec '09), JISC Institutional Approaches to Curriculum Design programme meeting (May '10). TODB 'generic' code has been made available under the GPL as a Google Code project¹², along with installation, developer, user and 'how to' documents under a Creative Commons attribution ('by') license. The documents have been reviewed for usability by a previously uninvolved member of CARET support staff.

Student Choices code has also been made available under the GPL as a Google Code project¹³.

Outcomes

The project has impacted institutional stakeholders including computer officers, teaching staff, teaching administrators, students, central IT, and subsequent projects.

In particular, departments adopting TODB have gained a transparent, effective way of managing and recognising teaching duties. Teaching administrators are freed of the 'politics' of the process, and teaching staff are more likely to have satisfactory teaching duties. Students benefit from more focused staff and the furtherance of a strategic goal which the University of Cambridge shares with other universities - to improve the breadth of undergraduate teaching by sharing courses between Departments.

TODB further represents a continuing experiment in 'simple software' development, producing a system sufficiently straightforward and easy to get to grips with that it can be locally maintained by non-developers in user departments. We have extracted the principles of simple software, and made them available alongside this report and on our blog.

The 'trunkless' concept offers a new sustainable development model, combining the economies of scale of a large project with the close fit to users' needs of a bespoke system, and in which non-developer end users maintain and modify their own software. It potentially provides a new approach to dissemination of valuable user-originated software in the HEI domain. The wide adoption of TODB

¹⁰ <http://modular-e-admin.blogspot.com/2010/02/institutional-innovation-exchange.html>

¹¹ <http://confluence.sakaiproject.org/display/GROUPS/Sakai+3+Groups+Home+Page>

¹² <http://code.google.com/p/todb/>

¹³ <http://code.google.com/p/studentchoices/>

suggests that departments genuinely perceive maintaining and running the software as being within their competence, but CARET will continue to evaluate the success of this model over the next few years.

TODB and Student Choices have both been exemplars of taking user-generated software and using it as a prototype for development and dissemination, providing requirements definition by example, for use in further work.

Student Choices is also an example of how experimental software development, merely by taking place, can catalyse institutional change in policy and process. The software developed may not be part of the eventual solution, but if the original need is met by other processes/systems, it is a good outcome.

Finally, in the course of the project we learnt that arts, social science and humanities departments often have strong technical skills. This shouldn't have been a surprise, but we feel it is worth recording.

Conclusions

One of the key learnings of this project has been that the software produced by academics themselves, whilst not always suitable immediately for greater take-up or scaled up production use, provides a strong and compelling demonstrator of desired functionality and can act as a powerful showcase of requirements. In our case, from these two examples (TODB and TIS) we are able to understand a variety of needs and desires, as the systems and interfaces we can view represent the culmination of initial design driven by need, plus some years of refinement as the relevant users gave feedback, and new needs were identified. This showcase reduces the need for user engagement and requirements gathering, which can be challenging in an environment where senior academics have other priorities and can be hard to engage. By taking existing and accepted functionality and interfaces, and migrating them to more scalable or differently maintained systems, we can retain a happy user base whilst meeting other institutional IT needs.

A second key area of learning concerns when and how to use an unconventional, 'trunkless' deployment model. It can not be said at this stage that the model is wholly validated, but we have shown that user-originated software can be successfully disseminated on a mass customisation basis, given certain preconditions, but that this requires input of resource above and beyond that required for ongoing local maintenance.

Appendices

This report has two appendices, in separate documents:

Appendix A - Overview of differences between departmental TODBs

Appendix B - e-Administration Requirements Specification

References

This report is accompanied by two companion documents:

Principles of Simple Software Development

The Trunkless Development Model