Allocating optional modules to University of York students using constrained optimisation

Alexander Muller

November 17, 2011

This is the report for a Bachelor of Science final-year project in Computer Science and Mathematics at the University of York. The project was supervised by Dr James Cussens, Senior Lecturer in the Artifical Intelligence Group, Department of Computer Science.

This report is 3,500 words, as counted by running `detex <report.tex> | wc -w`. It is 18 pages long.

From their second year onwards, most students at the University of York can choose between two or more optional modules to tailor their academic career, in the hope it will be more relevant, interesting and useful to them.

Optional module allocation in most departments is currently handled using a paper form which must be returned to departmental administrators. This project aims to design and implement a piece of web-based software that can be used by departments and students to allocate modules more fairly and with less administrative overhead.

Allocating modules "fairly" is a problem of understanding how staff and students view fair allocation, and translating that into a constrained optimisation problem.

The web application will be piloted by the Archaeology and History departments in Spring 2012 and, if successful, will be offered to all departments and maintained centrally by the University of York.

This report discusses the choices made around the technology used, the development methodology and details relating to the allocation algorithm. The software will be evaluated according to criteria set out by the project steering group, and the resulting evaluation is also discussed.

Contents

1. Statement of ethics **4**

2. Introduction **4**
2.1. The current state of module allocation 4
2.2. Web applications at the University of York 4

3. Research **5**
3.1. Development methodology 5
3.2. Database design 5
3.3. Maintainability and the future of the software 6
3.4. Sensitive information and the Data Protection Act 6
3.5. User testing 7
3.6. Usability 8

4. Development, implementation and testing **8**
4.1. Database structure 8
4.1.1. Entity-relationship diagram 8
4.2. User research 9
4.2.1. With students 9
4.2.2. With staff 10
4.3. Web application frameworks 10
4.3.1. Framework language 10
4.3.2. Summary 11
4.4. Design decisions and visual appearance 11
4.5. Accessibility 11
4.6. Interaction with other University software 11
4.7. Allocation algorithm 11
4.8. Testing 11
4.9. Pilot by the Archaeology and History departments 14

5. Further work **14**

6. Conclusions **14**

A. External links **15**

B. Participant consent form **16**

1. Statement of ethics

Those people volunteering to help with the project (interviewed during the research stage) will at no point be put in a position of physical danger. Consent will be obtained from all volunteers prior to their interview, and volunteers' personal information will not be published or shared. A copy of the consent form signed by all volunteers is given in Appendix B.

As far as the project author is aware, there are no immediate ethical issues relating to the creation of the module allocation software.

The project steering group has noted that as a student, the author must not be given access to any sensitive personal information. This includes, but is not limited to, student names, email addresses and degree course information. Development and testing of the software will be carried out with data that it is in a similar form to real data held by the University. The University's Data Protection Officer was consulted during the project, and their input is discussed in Section 3.4.

2. Introduction

This project is the result of requests by departments at the University of York for more flexibility in the way they offer optional modules to undergraduate students. The project is sponsored by University Teaching Committee and is overseen in that regard by Laura Crossley of the Academic Support Office.

As well as being marked as an undergraduate project, the software will be evaluated independently by Ms Crossley. If it is judged to provide sufficient advantages over the current methods of module allocation, it may be recommended that responsibility for the software is handed to IT Services.

A project steering group was formed consisting of representatives from each of the pilot departments (Archaeology and History), administrative staff responsible for IT and timetabling, and the project author and supervisor. At the initial meetings, this group set out the scope for the project...

2.1. The current state of module allocation

Universities worldwide want to give their students the most interesting and useful education possible, and one way this can be accomplished is by offering flexibility in modules students can take.

At the University of York, module allocation is something dealt with at the departmental level. Some departments make use of centrally offered software (eVision and SITS), though several departments feel the software is not flexible enough and continue to use paper-based forms that must be filled out and returned to the departmental office. A smaller number of departments, such as Computer Science, have written their own module choice and allocation software.

At some universities in the United Kingdom (for example, Warwick and Leeds), enrolement is completed online, on a first-come, first-served basis.

2.2. Web applications at the University of York

The University is constantly improving the quality of web software available to students and staff.

For the last few years, a single sign-on (SSO) solution has been provided centrally, using Shibboleth. This improves both user experience and security, as users are encouraged to only ever enter their University username and password into one screen, where the web address will always begin `https://shib.york.ac.uk/`.

In 2011, a new "student portal" was released, allowing students to view personalised information relevant to them in one place. The application was created by IT Services and was written in Java.

For the beginning of the 2011–12 academic year, new timetabling software is in use to give members of the University access to their complete timetable in one location—something which has not been possible before.

In June 2012, the University will move all students to Google's Apps for Education product for email and calendaring, one of the aims of which is to improve the user experience dramatically over the current webmail software.

3. Research

As web application development is an area that requires solving problems in many different areas of Computer Science, the research completed in this phase of the project was wide-ranging.

3.1. Development methodology

Zhang and Chung [1] note that prototyping can be used to reinforce client confidence as well as making better use of the time allocated to development and implementation.

Bochicchio and Paiano [2] note that creating prototypes of web applications has several advantages. The most relevant advantage for this project is that a mockup can be used to get feedback from non-technical stakeholders such as the project manager and the departmental contacts.

Prototypes can be defined as being low or high fidelity depending on how much they are designed to resemble the final application. Low fidelity prototypes can be creating using a marker pen and sheets of blank paper, whereas higher fidelity prototypes may be coded to appear in a web browser and allow the user to interact as they might with the final system.

Iterative development (also sometimes referred to as "rapid application development") is a process by which the product is gradually improved through trial and error. This differs from processes such as the waterfall method, where testing follows implementation, which follows design, which follows gathering of requirements. The key advantage of following an iterative development process is that it allows far more flexibility than other methodologies; it is common in software development that the requirements may change or be refined over the duration of the project, and the development process should be able to adapt as necessary [3].

Kuniavsky points out that an iterative development process is especially suitable for web applications, as prototypes can be created quickly. A low fidelity prototype of a web application (which might consist of sketched wireframes) can be created in a matter of minutes, while slightly higher fidelity prototypes such as a simplified application front-end can be running in a web browser within a day.

An iterative development process for this project might involve background research with users (in the form of interviews), the creation of a prototype, refining the user experience through more interviews, and repeating the "prototype → interview → refine user experience → interview" cycle.

3.2. Database design

A database management system (DBMS) is a piece of software that manages the database, including providing the ability to add or edit records stored in the database. The DBMS will be one of the more mature products used in the creation of this system, with many having been available since the 1990s.

Relational databases are a common feature of web applications. We note that the University of York already deploys MySQL and Oracle (both relational model) database systems for its web applications and management information system (MIS).

When designing a database, Johnson [4] gives several questions that he believes must be answered before a database can be created:

- What are the entities that need to be stored by the database?
- What are the relationships between these entities?
- What constraints are there on the database?
- What kind of queries will be written against this database?

All the questions above are relevant during the design of the database. While the first three questions relate to the structure of the database, the final question is especially important regarding database performance.

His method involves drawing an entity-relationship diagram (as the name implies, this answers the first two questions in the form of a graph) and then translating the E-R diagram into the database schema, which includes deciding which fields will become primary and foreign keys. Any constraints (such as minimum or maximum lengths of strings) can then be added depending on the DBMS product being used.

3.3. Maintainability and the future of the software

As the software created for this project will have to be maintained by the University's IT Services if it is evaluated as successful, care must be taken to ensure that the application is implemented in the most extensible and maintainable way possible.

Principles of good software engineering apply equally to web applications as to any other software project. There are several basic methods recommended by Green and Ledgard [5] for writing readable and maintainable code. Their recommendations are designed around the maintainer of the software having to do less work to understand the purpose of a given piece of code. The recommendations could be summarised as:

- Align parts of the code (e.g. equal signs) vertically when it makes sense to
- Write lines no longer than approximately 70 characters, and use line breaks if necessary
- Use simple English and short names for things that will be referenced frequently
- Add blank space around operators (e.g. `3 + 2 = 5` rather than `3+2=5`)
- Ident `if` statements to allow the reader to scan the code more easily
- Comment code when necessary, but comments are not a solution for bad code

They note that for any system that may have a long lifespan, every effort should be taken to improve "readability and maintainability". A project that will only be used for a short amount of time by the original author does not require as much attention to maintainability as one which will last several years and be maintained by several different programmers.

3.4. Sensitive information and the Data Protection Act

As noted in the Statement of ethics, we should discuss Charles' input here.

3.5. User testing

Jakob Nielsen is a web usability expert who has been publishing articles on his website, `http://www.useit.com/`, since 1995. In "Why You Only Need to Test with 5 Users", Nielsen asserts that usability tests should be run for all web projects, no matter how short the project timescale or limited the budget. This is especially relevant for a project such as the module allocation system, where the entire application must be developed in under six months and there is no budget allocated. Nielsen's advice is to run a single usability test with no more than five volunteers, and to run different tests if more participants can be recruited. His reasoning is that iterative design with testing after each iteration will uncover any problems unwittingly created during the development process. Finally, Nielsen points out that distinct groups of users need to be treated separately during user testing [6].

Nielsen Norman Group, a company founded by Nielsen with Don Norman in 1998, publishes reports on web usability. Among the 230 tips offered in one such report [7], Molich describes how to conduct user testing sessions. The bulk of his recommendations are around making the test participant feel comfortable during the session; this involves reassuring them that they are not being tested, telling them that they should simply perform the tasks as though they were at home and making the first task simple to allow the participant to gain confidence.

Cennydd Bowles and James Box work for a web design agency based in Brighton. In "Undercover User Experience Design", Bowles and Box describe various methods of usability testing with little time or budget. They give advice on asking questions in an unbiased way, so as not to influence the test. Like Nielsen, they advocate around five user tests, stating that even one is better than none. The authors suggest recording video (or, failing that, audio) of the interview as there will not be enough time to take notes during the session.

Bowles and Box put forward another method of eliciting information from users, namely the corridor test. This involves watching people use the current system for a very short amount of time and observing any usability issues they encounter. The primary advantage of this type of test is that it takes very little time or effort on both the part of the participant and the researcher [8].

In a 1982 paper titled "Pitfalls of user research, and some neglected areas" [9], J. M. Brittain sets out the different kinds of study that can be carried out during the research phase of any project. These are publishing a questionnaire or interviewing users, asking users for any input they have regarding a system or service, and observing users while they perform a task. One point made by Brittain is that user research is occasionally too narrow-focused - in his example, the library was focusing "upon the demands users make for documents" without necessarily considering how users read the documents once they are in possession of them. In the case of the web, one could argue that user research focuses too much on the specific task of interest and not on how users browse the web from day-to-day or what they generally use the web for.

User research (interviews with potential users or the distribution of questionnaires) should take place before the design phase in order to create a desirable product. Kuniavsky [3] describes a "family and friends usability test" that he claims provides immediate feedback on a prototype with minimal preparation and time required from the research participant.

His key points for conducting a usability study are:

1. Define your application's audience and the goals they want to achieve
2. Create scenarios that will help them accomplish their goals
3. Find test participants
4. Observe them while they play out the scenarios defined

Kuniavsky reinforces Molich's point that the most important consideration during a usability test is that the participant is comfortable; for example, that they understand it is not they who are being tested, but the application. He says that participants should be strongly encouraged to narrate their thought process as they use the application, as this provides useful insight to the researcher when they review the studies later.

3.6. Usability

In the past, web applications would `POST` data to the web server when the user clicked a submit button. Modern web applications, such as Google's Gmail email service, make use of JavaScript and fast Internet connections to automatically save data to the server while the user is working. Sandlund [10] explains that by automatically saving data at regular intervals, there is far less chance of data loss. However, it is important to include an explicit "Save" button on an application that auto-saves - one can imagine the confusion and uncertainty caused if the user is unaware that their data is being saved in the background.

```
<input type=submit>
```

During Sandlund's user testing, participants responded positively to the application automatically saving data. However, Sandlund decided not to include a submit button in his application, and he notes that some users found the lack of an explicit save button "a bit disturbing".

4. Development, implementation and testing

After researching the areas noted in the previous section, the software was researched, written and tested. This section describes how users helped influence the design of the software during interviews, a discussion on web frameworks, the visual appearance and other important factors for web projects.

4.1. Database structure

We start with a table of entities:

Student	Student ID (key), name, course
Allocation	Student ID (foreign key), module ID (foreign key)
Module	Module ID (key), department ID (foreign key), name, class size
Department	Department ID (key), name
Staff	Staff ID (key), department ID (foreign key)

As relational databases are so popular and are supported in many of the software frameworks we look at in Section 4.3, there is no advantage to choosing a database model apart from relational.

Web frameworks abstract away anything related to a specific database product, so the module allocation application can be written without regard for the final database product that IT Services will use to deploy the application.

4.1.1. Entity-relationship diagram

Figure 1 is an entity-relationship diagram of the module allocation system:

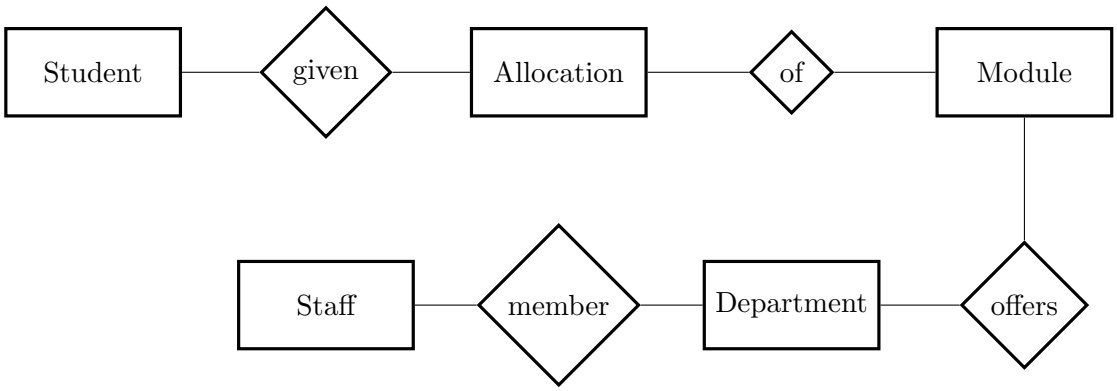

Figure 1: Entity-relationship diagram for the module allocation system.

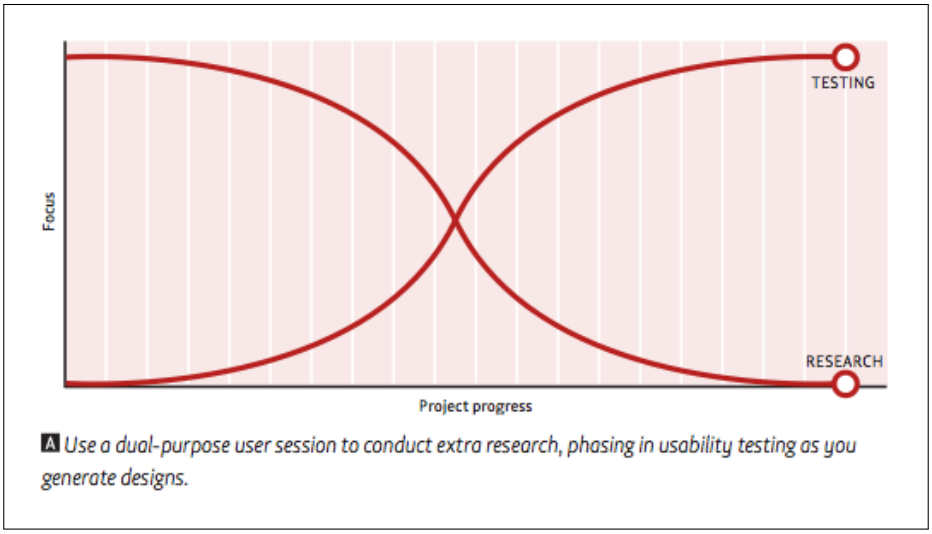

Figure 2: User session focus chart originally published in "Undercover User Experience Design", Bowles and Box [8].

4.2. User research

In the weeks before implementation started, students and staff were interviewed to understand more about their relationship with module allocation. The chart in Figure 2 was originally published by Bowles and Box and is incredibly relevant to this project.

The first user interviews (discussed in more detail in the following subsections) consisted mostly of research; for example, discussing how module allocation has worked in previous years. With this knowledge, a prototype could be created. The later user sessions focused more on testing the prototype with users to refine it and improve the user experience.

4.2.1. With students

In the meetings with students, the questions asked predominantly revolved around module allocation the University of York; it was important to understand how they feel about the current state of module allocation to ensure that the system improves the experience.

Students were asked to use a mockup of the application written in HyperText Markup Language (HTML) and the researcher observed to discover the main areas of difficulty around using web applications. Students were asked how and where they access to web, to get an understanding of the environments this application might be used in.

4.2.2. With staff

I also spoke to departmental administrators, who will be responsible for setting up the system. We discussed:

- The idea of "fairness"
- The user interface
- Administration

4.3. Web application frameworks

A framework is a certain amount of reusable code that helps application developers by reducing the complexity of common web operations. For example, any moderately complex application will need to write user input to a data store, and protecting against malicious input is an obvious concern when executing code in a database.

Many frameworks include functions that write to the database on behalf of the developer, and will automatically sanitise all input to prevent against attacks. As every application written should sanitise user input, it is this kind of repetitive action that frameworks help ease. A useful framework should increase the amount of time that a developer can spend building the unique parts of their application.

While it is possible to build a web application without using a framework, there is no advantage to rewriting basic operations for this project. If the system needed to operate at a scale or speed far beyond what these frameworks were capable of, it is likely that the software would have to be built from scratch to meet those requirements.

With the limited amount of time allocated to development and implementation, it is sensible to spend a small amount of time choosing a framework to allow more time to implement the system.

4.3.1. Framework language

The most important consideration when choosing a framework is the language it is written in, as this defines, among other things, how easy the software will be to maintain in the future.

There are three languages used for most web development at the University of York. *ColdFusion Markup Language* (CFML) is a language for creating web pages, the most popular commercial implementation of which is Adobe ColdFusion. ColdFusion is used widely across the University, including in applications such as the "People Directory" search tool (shown in Figure 4) and online account management facilities. *Java* was used in the creation of the new Student Portal. A *PHP* service is offered by the University for users to deploy their PHP applications.

```
<cfparam name="attributes.userid" type="string" default="unknown">
<h1>Hello, world</h1>
<cfoutput>
  <p>Welcome, #HTMLEditFormat(attributes.userid)#.</p>
</cfoutput>
```

In a presentation at The O'Reilly Open Source Convention (OSCON) in 2007 [11], Matt Raible compared several web frameworks written in Java; JavaServer Faces (JSF), Spring MVC, Stripes, Struts 2, Tapestry and Wicket.

CFML is unsuitable for this project as it is primarily a markup language - the allocation logic behind the application would have to be implemented in a programming language such as Java or Python. Using two languages to create this application would needlessly increase the complexity of any maintenance.

Outside the University of York, popular web programming languages include *Ruby* and *Python*.

While Ruby or Python are both good languages for web application development, neither is suitable for this project as they cannot be maintained by IT Services.

4.3.2. Summary

I chose a framework. It could be based on ColdFusion, Ruby or Python. Or something else entirely, even PHP or Java. But at the moment I'm not sure.

4.4. Design decisions and visual appearance

The application should be visually consistent with other University web software to instill trust in the user.

One area that usability gains can be made are in relation to how the application saves data. During one of the user research sessions, the participant explained that while choosing optional modules during her first year, she neglected to press that application's "OK" button and, unaware that she had not submitted choices, was unable to take her preferred modules.

As an explicit "Save" button should be retained to reduce surprise and allow the user to feel in control, the most sensible decision is to automatically save user input periodically and notify users that their choices are being saved. The submit button would save data in case of failure of the automatic save mechanism.

4.5. Accessibility

Perhaps talking about Dave Swallow, CS PhD.

4.6. Interaction with other University software

The system will be required to interact with SITS, the University's MIS.

4.7. Allocation algorithm

Let's discuss the algorithm here.

4.8. Testing

How will the software be tested?

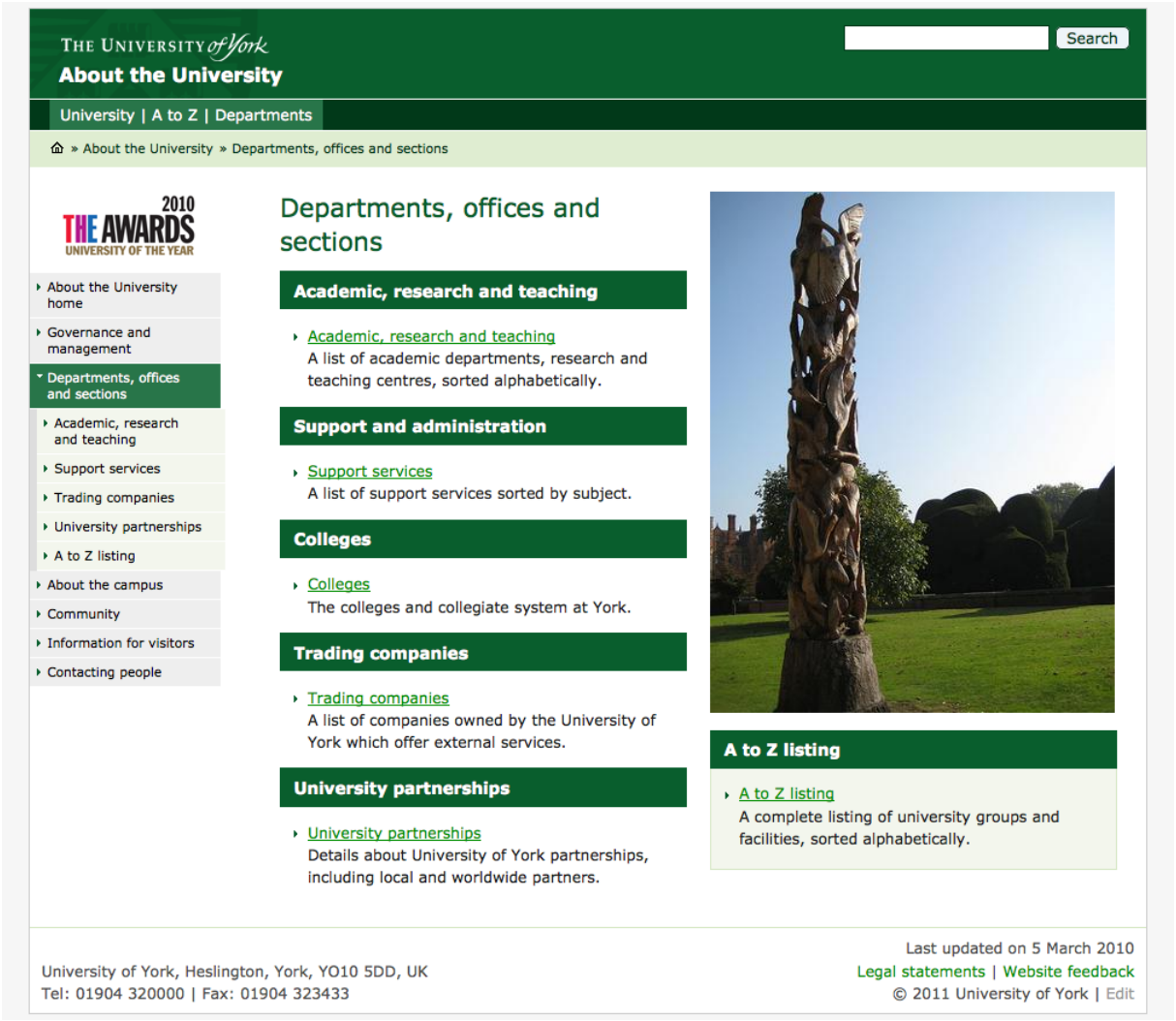

Figure 3: A general page on the University site, using University colours.

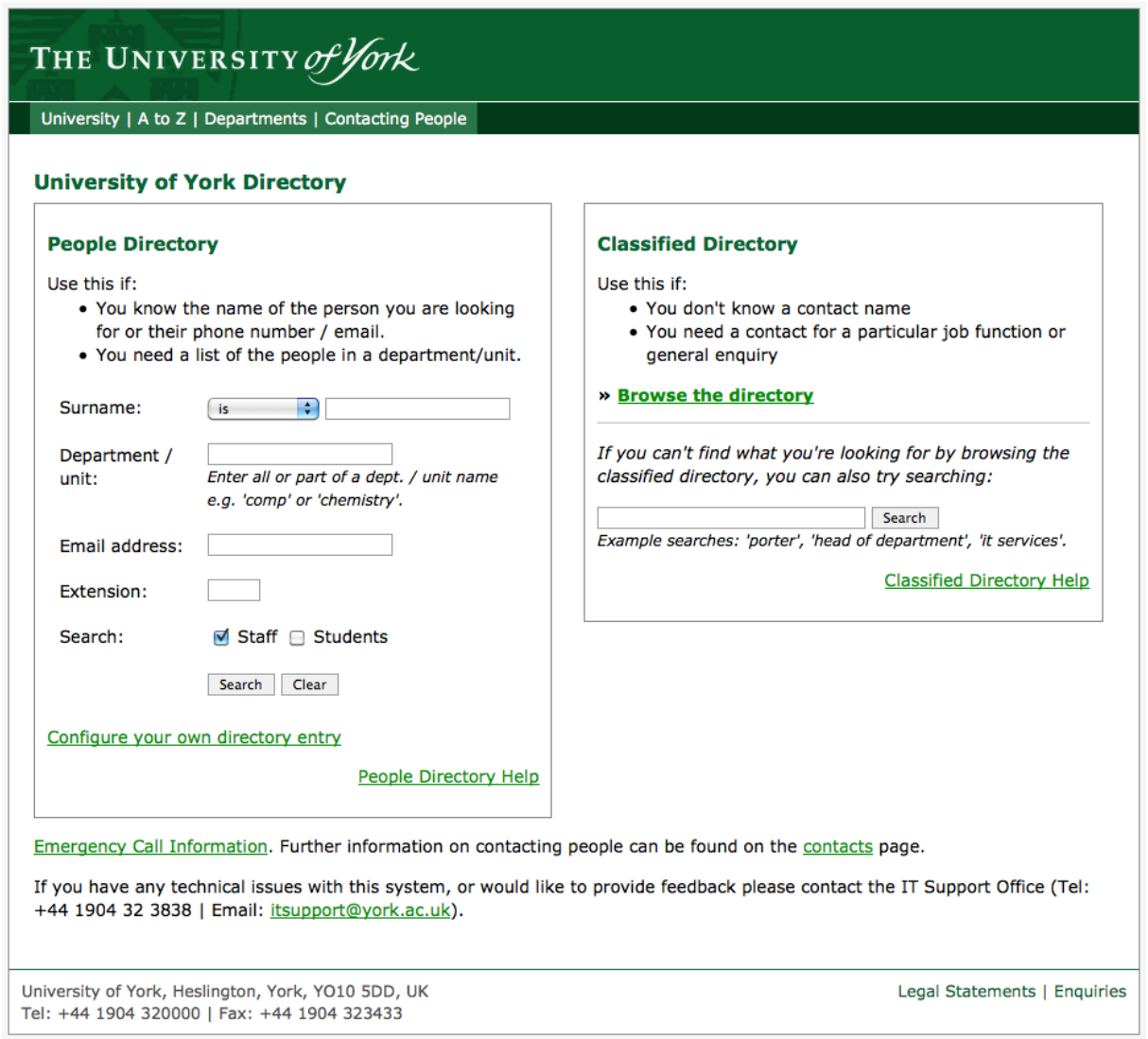

Figure 4: The University of York People Directory search tool.

4.9. Pilot by the Archaeology and History departments

How did the pilot go?

5. Further work

We should probably ask departments what they'd like to see in the future.

If I didn't have time to implement a weighting system (rather than a simple, plain ranking system) I could talk about that here.

6. Conclusions

It was a success. Well, hopefully.

A. External links

This appendix gives URLs to departments, offices or services mentioned throughout the document.

University Teaching Committee sponsored this project on behalf of the University of York: `http://www.york.ac.uk/about/organisation/governance/sub-committees/teaching-committee/`

IT Services hosted and maintained the application once it had been created: `http://www.york.ac.uk/it-services/`

I would like to make the code of this software available online, at `https://github.com/alexmuller/york-allocation`.

B. Participant consent form

All research participants (students and staff of the University of York) signed the consent form shown in Figure 5 immediately before their interview took place. This consent form is adapted from one made available by Alistair Edwards for Computer Science students to use during their projects. As of 5 November 2011, the original is available at `http://www-users.cs.york.ac.uk/~alistair/projects/consent.html`.

In each case, the top half of the form was retained by the project author and the second half was given to the participant in case they had any further questions about the interview.

```
# Allocating optional modules to University of York students
# BSc 3rd year project in the Department of Computer Science
# Participant consent, November 2011

Your participation in this interview is entirely voluntary; there will be no remuneration
for the time you spend helping.

All data gathered from this interview will be treated in a confidential fashion: it will be
archived securely and will be interpreted only for purposes of this student project.
Whenever your data are reported (in the project report or elsewhere), no personal
information will accompany the data.

There are no known risks to participation in this experiment, and you may withdraw at any
point. Please feel free to ask Alex if you have any questions.

If you are willing to participate, please sign this consent form.


Name:      _____________________

Signature: _____________________

Date:      _____________________


✂-------------------------------------------------------------------------------


# Allocating optional modules to University of York students
# BSc 3rd year project in the Department of Computer Science
# Participant consent, November 2011

Should you have questions about this interview at any point, feel free to contact either the
researcher or his supervisor.

## Researcher's contact details:

Alex Muller <am639@student.cs.york.ac.uk>
07525 370 388
30 Siward Street, York YO10 3LW

## Supervisor's contact details:

James Cussens <james.cussens@cs.york.ac.uk>
01904 325 371
RCH/326, The Hub, Deramore Lane, University of York, York YO10 5GE
```

Figure 5: Participant consent form.

Glossary

DBMS database management system. 5

HTML HyperText Markup Language. 10

IT Services is a support service that, "together with the Library and Archives, forms the Information Directorate" at the University of York. It is responsible for, among other things, computer hardware and software, infrastructure and web services at the University. 4, 11

MIS management information system. 6, 11, 17

OSCON The O'Reilly Open Source Convention. 11

`POST` is an HTTP request method that allows web browsers to send data to a web server as part of the request body. `POST` requests are typically used for submitting forms or uploading files on the web. 8

SITS is the University of York's MIS, a database used to store staff and student details, including information on which modules are available and which students are taking. 4, 11

SSO single sign-on. 4

References

[1] J. Zhang and J.-Y. Chung, "Mockup-driven fast-prototyping methodology for web application development," *Software: Practice and Experience*, vol. 33, no. 13, pp. 1251–1272, November 2003. [Online]. Available: http://dx.doi.org/10.1002/spe.547 [Accessed: 27 Oct 2011]

[2] M. Bochicchio and R. Paiano, "Prototyping web applications," in *Proceedings of the 2000 ACM symposium on Applied computing - Volume 2*, ser. SAC '00. New York, NY, USA: ACM, 2000, pp. 978–983. [Online]. Available: http://doi.acm.org/10.1145/338407.338712 [Accessed: 27 Oct 2011]

[3] M. Kuniavsky, *Observing the user experience: a practitioner's guide to user research.* Morgan Kaufmann, 2003.

[4] J. L. Johnson, *Database: Models, Languages, Design.* Oxford University Press, USA, 1997.

[5] R. Green and H. Ledgard, "Coding guidelines: Finding the art in the science," *Queue*, vol. 9, pp. 10:10–10:22, November 2011. [Online]. Available: http://doi.acm.org/10.1145/2063166.2063168 [Accessed: 4 Nov 2011]

[6] J. Nielsen, "Why you only need to test with 5 users," March 2000. [Online]. Available: http://www.useit.com/alertbox/20000319.html [Accessed: 8 Nov 2011]

[7] R. Molich and J. Nielsen, "230 tips and tricks for a better usability test," *Nielsen Norman Group Report*, June 2001. [Online]. Available: http://www.nngroup.com/reports/tips/usertest/ [Accessed: 11 Nov 2011]

[8] C. Bowles and J. Box, *Undercover User Experience Design.* New Riders, 2010.

[9] J. Brittain, "Pitfalls of user research, and some neglected areas," *Social science information studies*, vol. 2, no. 3, pp. 139–148, 1982. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0143623682900278 [Accessed: 11 Nov 2011]

[10] J. Sandlund, "Web 2.0 school software; designing for usability," 2009. [Online]. Available: http://publications.lib.chalmers.se/cpl/record/index.xsql?pubid=117379 [Accessed: 17 Nov 2011]

[11] M. Raible, "Comparing Java web frameworks," Presentation at OSCON, July 2007. [Online]. Available: http://raibledesigns.com/rd/entry/oscon_2007_comparing_java_web [Accessed: 7 Nov 2011]