Allocating optional modules to University of York students using constrained optimisation

Alexander Muller

November 8, 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 2,260 words, as counted by running `detex <report.tex> | wc -w`. It is 11 pages long.

The following line can be run from within TextMate (using `ctrl-r`) to output the word count:

```
detex  /proj/Project.tex | wc -w
```

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

No person volunteering to help with the project (those interviewed during the research stage) will be put in a position of 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 Consent will be obtained from all volunteers prior to their interview, and volunteers' personal information will not be published or shared.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. While it is possible the software may be repurposed for some completely different task, such an action is impossible to foresee.

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 of the software will be carried out with fake data in a similar form to the real data held by the University. The University's Data Protection Officer was consulted during the project, and their input is discussed in more detail later in this report.

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 responbility for the software is handed to the University's IT Services department.

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

1. At the University of York

2. Elsewhere around the world

Refer to documents provided by Laura

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, the university management information system), 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 other 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 applications available to students. In 2011, a new "student portal" was created, allowing students to view information relevant to them in one

place. 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. And in 2012, the university will start to use Google Apps for Education for email and calendaring, which will hopefully improve the user experience dramatically over the current webmail offering.

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.

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. DBMS products are mature, many having been available since the early 1990s.

Relational database models 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 MIS.

How should the database be structured? Good question [3].

We start with a table of entities:

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

We should draw an entity-relationship diagram.

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 evaluated as successful, care must be taken to ensure that the application is implemented in the most maintainable way possible.

There are several basic methods recommended by Green and Ledgard [4] for writing readable and maintainable code:

Use vertical alignment. Breaking lines when they're going to be long anyway. Use simple names for things. Less is more, use `i` for an iterator rather than `elementNumber`. The more often a variable is used, the shorter its name should be. Use white space to break up code. Using blank spaces around operators like the equals sign means the person reading has to do less work to understand the code.

Indent `if` statements. Comment code when necessary, but do not write bad code that you intend to fix by commenting.

Green and Ledgard argue that Unix code is unreadable, and "textbook code" is too deeply commented. We should strike a balance between the two.

They note that for any system that may have a long lifespan, every effort should be taken to improve "readability and maintainability".

3.4. Sensitive information and the Data Protection Act

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

3.5. Usability and 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 [5].

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. User research

In the weeks before implementation started, students are staff were interviewed to understand more about their relationship with module allocation.

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 an HTML (hypertext markup language) mockup of the application 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.

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

4.2. 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.

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

Rails guesses the model's attributes based on the database schema, whereas Django requires that the model's attributes be listed in the application, and the schema is then created.

```
class Allocation < ActiveRecord::Base
  belongs_to :student
  belongs_to :module
end
```

There's a good comparison of Django and Rails which is quite methodical. In it, the authors note that equivalent applications took $\frac{2}{3}$ the time to be implemented in Django than Rails [6].

There's also a good comparison of the three big players that pretty much rules out CakePHP as being rubbish [7].

4.3. Visual appearance

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

4.4. Accessibility

Perhaps talking about Dave Swallow, CS PhD.

4.5. Allocation algorithm

Let's discuss the algorithm here.

4.6. Testing

How will the software be tested?

4.7. Pilot by the Archaeology and History departments

How did the pilot go?

5. Further work

A chapter describing possible ways in which your work could be continued or developed. Be imaginative but realistic.

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

This is similar to the abstract. The difference is that you should assume here that the reader of the conclusions has read the rest of the report.

It was a success. Well, hopefully.

A. External links

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

`http://www.york.ac.uk/about/organisation/governance/sub-committees/teaching-committee/`

`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 1 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 1: Participant consent form.

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] J. L. Johnson, *Database: Models, Languages, Design.* Oxford University Press, USA, 1997. [Online]. Available: http://ukcatalogue.oup.com/product/9780195107838.do [Accessed: 28 Oct 2011]

[4] 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]

[5] 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]

[6] B. Askins and A. Green, "A Rails / Django comparison," *The Python Papers*, vol. 2, no. 2, pp. 44–53, 2007. [Online]. Available: http://ojs.pythonpapers.org/index.php/tpp/article/viewArticle/24 [Accessed: 22 Oct 2011]

[7] J. Plekhanova, "Evaluating web development frameworks: Django, Ruby on Rails and CakePHP," Fox School of Business, Temple University, Philadelphia, PA 19122, USA, Tech. Rep., September 2009, The Institute for Business and Information Technology Report. [Online]. Available: http://ibit.temple.edu/blog/2009/09/01/evaluating-web-development-frameworks-django-ruby-on-rails-and-cakephp/ [Accessed: 22 Oct 2011]