# Coursework commentaries 2016–17

## CO3320 Final project report

## General remarks

This report relates to the Final project report (FPR) submissions for CO3320 in the academic year 2016–17. Before reading this, you should read the accompanying report on the Preliminary project report (PPR) submissions. Much of what is written in the PPR report applies to the FPR submissions too, and will not be repeated here.

The comments below are organised into a number of categories:

- Organisation and presentation
- Project subject
- Literature review and background research
- Software design and development
- Experimental design
- Presentation of results
- Evaluation of results.

Each of these topics is discussed below. Note that most of these issues come up every year in the FPRs. If you are reading this report at an early stage in thinking about your Project and take time to ensure that each of these issues is addressed, then you should be in good shape for producing your FPR.

### Organisation and presentation

While there are no rigid limits on the length of a FPR, very generally speaking most reports come in at approximately 40–80 pages of main text (excluding appendices, reference list, etc.). Of course, this will depend on font size, line spacing and so on, but most students are able to describe their work adequately within that kind of range. This year, like every year, we saw a handful of reports that were much longer than this, some in the region of 150 pages of main text. It is almost never warranted to have such a long report; almost without exception, a report of over 100 pages is a sign that the student does not have a good sense of how to selectively present the important details in the main text, how to organise supplementary details in appendices, or how to refrain from including details of no direct relevance to the Project.

After deciding what information needs to be presented, care should be given to **how** it should be presented. Try to think from the perspective of a reader who knows nothing about your Project. Be sure to explain acronyms and abbreviations when you first use them; several reports were weak in this area.

Another notable weakness this year was that a significant number of reports did not include a self-evaluation at the end. This is a required part of the report, in which we expect to see the student reflecting on their experience of doing the Project and on what they have achieved.

You should follow the guidelines for the FPR structure set out in Section 6.2 of the **CO3320 Project** subject guide (pp.29–34).

## Project subject

This year it was very gratifying to see a small number of students who had identified an existing piece of work published in the academic literature as the starting point for their Project. While it is not necessary to do this, if you **are** able to do so, it can give you some confidence that the Project topic is suitable. One or two students even contacted the authors of the previous work to ask questions, request further data and so on. This is indicative of academic maturity.

## Literature review and background research

While many students provided good literature reviews, some struggled to find the appropriate level of detail. This section of the FPR should provide an overview of existing work and research of specific relevance to your Project topic. It should not be a general tutorial or text book account, but should only discuss those aspects of existing work and theory that are specifically relevant to your Project and will help you pursue your own work.

Projects that are mainly focused on software development should certainly include in their review a discussion of any existing software or apps available on the market. Several of this year's FPRs were weak in this area.

A good way to get a better idea of the level and breadth of detail required in the literature review is to look at some good Projects in the Project library on the virtual learning environment (VLE; see the end of this report for details on how to access this).

## Software design and development

While many students did a good job of describing the systematic process of software design and development, a significant number of reports were deficient in one or more aspects of this.

Some reports just presented the requirements for their software ('my software is going to do x, y, and z') without any discussion of or justification for why these features were necessary or desirable. Many students engaged in surveys or questionnaires with potential users in order to define and justify their software requirements, which was good to see. But whether or not such surveys are appropriate, there should be some clear discussion of why each feature is needed. Furthermore, especially for complex software Projects, it is usually wise to prioritise requirements and work on the most important ones first. That way, if development falls behind schedule in the Project, at least the most important aspects have been tackled.

A handful of students experienced problems relating to the fact that they were developing software for a client (e.g. either it was a work-related Project, or they were developing software to help a local company). Problems arose when the client changed their mind about what they wanted from the software midway through the Project. This problem (known as 'feature creep') can seriously impact the successful completion of a Project. The way to avoid this is to have a clear process of requirements gathering at the start of the Project and to use that to draw up an explicit list of requirements that is agreed between the student and the client at the start.

For any significant software development, the examiners expect to see a clear explanation of the software design, probably including UML diagrams, wireframes and so on. While most students provided this, others gave very few details of how their systems worked.

A few Project reports jumped straight from software design to results and analysis, with nothing said about the process of software development and testing. The examiners expect to see some description of these stages, covering, for example, what (if any) third-party libraries were used, what (if any) problems were encountered during development and how the code was tested to ensure it was working as expected.

## Experimental design

Many Projects involve some kind of experiments and the design of these must be carefully considered and discussed.

For Projects that involve some kind of machine learning technique (e.g. neural networks), careful thought must be given to the design of appropriate training and testing phases, including what subset of the available data will be used for training and testing. The dataset should also be examined prior to using it in experiments to ensure that it is as expected (e.g. that it does not contain any missing or nonsense values). Justification should be given for the choice of parameter settings for learning algorithms. This year, as in previous years, many Projects involving machine learning were rather weak on these aspects.

Elsewhere, in Projects that involved testing or surveys with human subjects, full details and justification should be given about how the subjects were recruited, the selection criteria (e.g. were you targeting a particular demographic?), how you decided the number of people that should be involved and so on. A disappointingly large number of Projects gave very little discussion of these kinds of details, including any ethical considerations.

## Presentation of results

Careful thought should be given to how the results of the Project would be presented. This year, as ever, we saw some Projects with very clear presentation of results, and others which left considerable room for improvement. These included reports in which numerical results from experiments were presented in page after page of numeric data tables. While there may be some justification for including these raw results in an appendix, it is not appropriate to include them in the main text, as the reader can learn very little from them. It is much better to present appropriate graphs showing the results in summary form, which the reader can understand much more easily.

Examples of Projects that did a good job in presenting results include those that included a video walk-through/demo of the developed software (uploaded as a video file along with the main FPR submission), and others that included a full user manual for their software.

What technique you choose will depend upon the nature of your Project, but you should be aiming to communicate your results to the reader as clearly and simply as possible.

## Evaluation of results

Having performed some experiments or having produced and tested a new piece of software, it is important to provide a convincing discussion of whether the results were good, and therefore whether the Project has been successful in achieving its aims. If the results are poor, it is important to offer an explanation.

This year we were gratified to see various Projects that displayed considerable thought and effort in these areas. For example, some Projects used **test** and **control** sets of users to measure the difference in performance of a task between users who used the developed software to help them, compared with those who did not.

Some other Projects used focus groups of people who had used the developed app, to gather honest feedback and suggestions for improvements.

There are many ways an evaluation can be done; it is an essential aspect of most Projects and should be thought about carefully when devising the Project plan.

In general, the 2016–17 FPRs spanned a wide range of standards, from the weak to the truly outstanding. The preceding comments have highlighted some of the common problems. Further advice on how to produce a good FPR can be obtained in the following ways.

- Read the **CO3320 Project** subject guide.
- Look at examples of good Projects in the Project library section on the VLE (https://computing.elearning.london.ac.uk/mod/data/view.php?id=6063).
- Discuss problems and questions with fellow students on the Discussion forum of the CO3320 page on the VLE.