**Exception Handling**

We have an obligation to be professionally competent (else give rise to professional issues) with regards to CAMEL’s error managing system. We can either code defensively (which will only manage errors we can think of while coding) or use exception handling (which is our choice).

Exception handling has the benefit of catching all errors. However overuse and exception handling reliant code can become less readable and thus less maintainable after the fact.

Of course this also deeply affects our professional integrity. In order to address this we must best adhere to recommended practice; our code should only throw exceptions for errors that are abnormal, rare and if you will, exceptional.  
*SOURCES:*

* [*www.hanselman.com/blog/GoodExceptionManagementRulesOfThumb.aspx*](http://www.hanselman.com/blog/GoodExceptionManagementRulesOfThumb.aspx) *(accessed 03/02/16, created 30/08/06)*
* [*www.joelonsoftware.com/items/2003/10/13.html*](http://www.joelonsoftware.com/items/2003/10/13.html) *(accessed 03/02/16 created 13/10/03)*

**Analysing Student Answers (Lexical Comparison to Measure Collaboration and Engagement)**

Lexical analysis of student submitted answers can be reliable, consistent and appropriately indiscriminate and thus can provide a less socially objectionable system for plagiarism detection.

However the validity of information yielded from such a system is questionable. This is especially true within subjects of science, where answers of an explanatory or computational nature may appear very similar yet actually be suitably original. This brings into question the credibility of this system. Though since all such software carries the same uncertainty, it should not affect the professional integrity of CAMEL.

Any teachers that are misled to rely too heavily on such a system may treat a student unfairly if wrongly accused. As long as teachers are well informed that results are not definitive and should only be used as an indication of potential plagiarism, this should not be of much social concern. It is also prudent to remember that providing such a disclaimer and keeping other professionals properly informed is a professional obligation we must adhere to.

**Logging Student Answers**

Since this information is not classed as personal information, storage of student’s answers should not breach the Data Protection Act. The exception to this is the attachment of the student’s identity to the piece of work. In order to comply with the spirit and word of the law, as well as common ethical practice, access to this information should be restricted only to those who need it and the information itself should not be kept longer than necessary.

Students may wish to remain anonymous. Refusing this option could lead to socially based objection. It would also take control away from the student, hurting the system’s usability. This would of course in turn damage the professional success of CAMEL.

A functionality of anonymity must be implemented anyway, since marking cannot be implemented fairly without it. In fact, non-anonymous marking would conflict with protocol for most academic establishments within our target market. This would reduce CAMELs software quality from a professional aspect.

**Allowing Deadlines on Student Assignments to Prevent Late Submissions**  
 Since this may decide the fate of any given student’s final mark, the presence of any errors in this process could cause users significant issues.

Though this is a social consideration, it should be noted that an error like this could drive users away and hurt the professional integrity of CAMEL.

A potential work around could be a period of grace after a deadline. This could allow students time to rectify the problem or find an alternative. Unfortunately this method will not discriminate those who submitted late by their own fault. This is unfair to students who adhered to the deadline as they should have.

An alternative is the allowance of late submissions that are then flagged for individual consideration later.

**Breaking Down Core Code**  
 CAMEL is a project created and run mainly by few individuals. Breaking down, restructuring or refactoring this code will change it greatly. This change could make the code far less familiar and recognizable to those individuals. If these individuals cannot adapt, this could either inhibit CAMEL’s short to mid-term maintainability and growth or force its creators to depend on more expensive means of developing and maintaining it via more professional teams of experienced developers. Ethically this is an undesirable outcome. The individuals in question could be debriefed and somewhat trained to the nature of the new system as a means of rectifying this issue.

**Ease of Use – Disinclusion of documentation to teach users how to use CAMEL**

Intuitive systems often lack much documentation to guide a new user on how to use it. This will not give rise to any issues regarding the professionalism of the quality of CAMEL as software as long as it truly is intuitive and usable. Consideration must be given to all potential users, including minorities such as older less tech-savvy teachers/mature students, international students coming from a less technologically advanced background and those with accessibility needs such as impaired vision or dyslexia. This is not an exhaustive list of examples.

**Upgrade System to Python 3**

Remaining updated is an important feature of any system trying to survive in the software development world. Of course we have a professional responsibility to give CAMEL the best fighting chance.

Python 3 will eventually succeed Python 2, thus making conversion inevitable, so taking on this task would by some be considered an important part of providing our client with high quality service.

In fact, by the Code of Conduct published and enforced by the BCS (Chartered Institute for IT), part of our obligation of professional competence and integrity includes maintaining an awareness of technological developments on a continuous basis.

With that in consideration, we could not in good faith refuse this service to our client indefinitely.

However since the system is already so far developed with Python 2, perhaps a task like this should be prioritised to take place after bug fixing and major refactoring is completed.

*SOURCES:*

* [*www.bcs.org/category/6030*](http://www.bcs.org/category/6030) *(accessed on 03/02/16 updated 2016)*