THE BCS PROFESSIONAL EXAMINATION
Professional Graduate Diploma

April 2007

EXAMINERS' REPORT

Software Engineering 2

General Comments

Most of the scripts produced by candidates were well structured and readable. However, the level of comprehension (of the question set as well as the subject matter) needs particular attention. A common practice by such students was that of writing all that had been memorised on a self-selected keyword topic without reference to the context given in the question.

The exam, at honours level, sought to test candidates' knowledge and ability to apply this knowledge according to the scenario presented. The role of the scenario is to create an opportunity for the candidate to show selection, prioritisation and critical judgment. These twin abilities, selection and application, characterise the level of this examination. Disappointingly, many answers were either to a question of the candidate's own making, or based on the recall of facts without reference to the scenario. Without evidence of critical judgement and relevant application, these answers were given low marks.

Question 3 and 5 proved to be two of the three most popular choices amongst candidates, but exhibited the lowest performance average overall. Question 2 was the third popular question; many answers could not differentiate between software process and software product. What is clear from the answers to these questions is that the breadth and currency of knowledge, regarding software engineering principles and practice, is very limited in some centres. Even candidates with employment in the profession often gave a very restricted view of a topic, revealing a specific knowledge but lacking breadth of understanding and application. Particular emphasis must be placed on the importance of candidates studying not only a recommended text, but reading more widely in terms of publications within the profession.

Similar to the previous year, some students continue to answer more questions than was required in the examination, and this practice reduced the time they have for each question and limited their opportunity to obtain the full range of marks for the questions attempted.

Finally, similar to the previous year, some student had not completed the front cover with the number of the question and part-questions attempted. This made it very difficult to find, then determine whether it was a continuation of a previous question, and assess and record the mark for that question as a whole.

Question 1

a) Compare three different approaches to project planning using these three metric styles: size, function and algorithmic. Make clear the principles in your answer. **(15 marks)**

b) Discuss the practical limitations of these metric styles. Discuss how they might be adapted for practical use. **(10 marks)**

Answer Pointers

This question assesses the candidate's understanding of metrics and their applications, and sought to test candidates' abilities to make critical judgments about application of metrics.

A good answer should take the following form:

a)

Size-related planning deals in KLOCS (1000s of line of code) or some other finished-product size index. Since planning comes before building, this is often achieved by analogy with pre-existing projects or systems. This is a product-centric method.

Function-related planning deals with some index (function points) that assigns numeric weights to different development elements such as transactions, files, complex calculations etc. At best, this is based on extensive systems analysis or collected in-house metrics, though at worst it uses 'measure by analogy'. This is a product-centric method.

Algorithmic planning (SLIM, CoCoMo, and others) mixes product and process algebraically to account for development variables such as experience of the team, technology maturity, and complexity of application. This is a dated, mixed process/product method, using parameters validated 20+ years ago on 3GL and 2GL projects. CoCoMo2 has current application, using alternative approaches at different times in the development cycle, with Object Points, KLOCS and modified cost drivers. It remains an algebraic model. This is a process-centric method.

b)

A good answer would draw on a defence of function-related planning because the algorithmic approaches (CoCoMo2 excepted) do not calibrate for modern development methods (RAD, Evolutionary, integrated CASE tools) and the size-related methods have limited practical relevance. For extra marks, the candidate will 'personalise' whatever is recommended by remarking that in-house metrics are likely to be more useful and accurate for calibration than 'industry standard' metrics.

Examiner's Guidance Notes

a) This question gave clear guidance, first to construct views of project planning based on three specific styles, then to analyse and derive commonalities. Many answers were restricted to descriptions of metrics and did not apply the ideas to project planning. Some answers asserted conclusions, e.g. "It is difficult", without stating what was difficult, or why.
b) This question invited candidates to show critical judgment by comparing characteristics of metrics with reality of software development. Many facts were stated without critical selection or application.

Question 2

a) Discuss the underlying principles of a standard for a software process and a standard for a software product. In your answer, you may refer to any of the international standards that exist. **(10 marks)**

b) You are asked to create a company standard for software configuration management. Giving your reasons, choose THREE elements of configuration management that you would use in your standard. **(15 marks)**

Answer Pointers

This question assesses the candidate's understanding of International Quality Standards by selection and application to process and product, and by selection and application to a specific software process.

A good answer should take the following form:

a) Size-related planning deals in KLOCS (1000s of line of code) or some other finished-product size index. Since planning comes before building, this is often achieved by analogy with pre-existing projects or systems. A product-centric method.

Function-related planning deals with some index (function points) that assigns numeric weights to different development elements such as transactions, files, complex calculations etc. At best, this is based on extensive systems analysis or collected in-house metrics, though at worst it uses 'measure by analogy'. This is a product-centric method.

Algorithmic planning (SLIM, CoCoMo, others..) mixes product and process algebraically to account for development variables such as experience of the team, technology maturity, and complexity of application. This is a dated, mixed process/product method, using parameters validated 20+ years ago on 3GL and 2GL projects.

b)
A good answer would draw on a defence of function-related planning because the algorithmic approaches do not calibrate modern development methods (RAD, Evolutionary, integrated CASE tools) and the size-related methods have limited practical relevance. For extra marks, the candidate will 'personalise' whatever is recommended by remarking that in-house metrics are likely to be more useful and accurate than 'industry standard' metrics.

Examiner's Guidance Notes

a) This question asked candidates to differentiate between qualities applied to software process and software product, and invited candidates to cite appropriate International Standards in support of their answers. Few answers made this differentiation, asserting that similar qualities applied to both process and product. This lack of critical selection lost marks by revealing that candidates could not differentiate between process and product. There was wide variation in candidates' interpretations of 'software process'. Interpretations about the development process, QA control processes and Life Cycle Models were acceptable if supported by reasons; those interpretations based on the documentation produced by a QA system were too far from the initial question to be creditable.

b) This question asked candidates to identify three stages of software configuration management, and describe how to define a descriptive standard for each stage with reasons for their selections. Again, although many descriptions were given, very few were supported by reasons. Descriptions by themselves are not honours-level answers.

Question 3

As a member of the development team for a new production control system, you have been allocated the task of designing the software module that picks items of a production line, inspects their quality of manufacture, and place these components into appropriate collection bins. The collection bins are labelled “PASS”, “REWORK”, “SCRAP”, and “UNDECIDED”.

a) Give a broad overview of both the Object Oriented and Structured Design approach to developing this software module, highlighting their similarities and differences using suitable diagrams and annotation based on the scenario described. **[15 marks)**

b) Write a detailed process specification for the system described above that exhibits good reusability characteristics. **(10 marks)**

Answer Pointers

This question assesses the candidate’s understanding of aspects of design through the selection and application of a design method to the problem outline given.

A good answer should take the following form:

a) Definitions of OO and Structured Design. E.g.,
An object oriented approach seeks to identify the objects, their description in terms of state and behaviour, and their interrelationships. The structured design seeks to identify the processes and their data entities.

Application Scenario. E.g.
(1) OO approach: objects include: Production Line; Item; Collection Bin; Picker. The latter could include behaviour (public methods) such as ***pick()***, ***inspect()***, and ***place****()*.

The Collection Bin should include a quality label that can be set with the value such as “PASS”, “REWORK” etc, and a count attribute of the number of items.

(2) Structured Design: Create a module with the logical call sequence of ***pick()***, ***inspect()***, and ***place****()* subroutines.

These subroutines work on data structures such as the *production line array*, the *item* element, and the *collection bin* array of file.

b) The process specification should be primarily in the form of a module specification identifying:

Input (call by reference) parameters: LIST of Items
Output (export only) parameters:
Process logic

E.g.,

MODULE PickAndPlace:
Input parameters: LIST of Items
Output parameters: Collection Bin TYPE
Process logic:
Item = pick(LIST of items);
Quality = ***inspect(Item)***;
Place(Item, Quality*)*
End PickAndPLace

Examiner's Guidance Notes

a) This question requires the student to apply and compare structured and OO design techniques for producing a possible solution to the problem described. The low marks for this question can be explained by the situation where the scripts submitted shows that many students have a good knowledge of one, but not of the other; and many more students were familiar OO concepts but unfamiliar with structured design principles. Thus, on this basis many students could only attain 50% of the marks available.

Typical solutions would make use of DFDs, ERDs, and OO class diagrams. There were a few candidates that appeared to have either ignored or misunderstood the scenario in the formulation of their answers and thus produced description of one or both design approaches. Other candidates focussed on OO modelling at the expense of structured design.

b) In this section, many candidates did not appear to understand what a process specification was, whilst others picked up on the word "reuse" and wrote essays on the benefits of reuse in OO. Very few (if any) marks were given for this type of answer.

It is hoped that in future candidates will ensure that they distribute the allotted time proportionately across the two sections. Thus, many marks were lost by candidates only addressing a single aspect of the question.

Question 4

a) Compare the types of testing that would be performed for the following three phases of the software development life cycle. Comment critically on the way they overlap with each other.

- requirements
- design
- validation **(12 marks)**

b) A mobile phone company is designing new software for automatic processing of upgrade requests on its website. You are assigned to design the tests. The application accepts data in the form:

Phone number = All numbers used by this company begin 07973 followed by 6 digits.
Account number = 6 digits not beginning with 0 or 1
Password = 6 alphanumeric characters
Selected upgrade model = 4-digit number
Payment plan = "monthly", "pay-as-you-go", "business"

The input conditions associated with each element are:

Data element	Input condition1	Input condition 2
Phone number	*Boolean* 07973 may be absent	*Range* 000100 and 800999
Account number	*Range* – 200000 to 999999	*Range* – 6-digit number
Password	*Boolean* may be absent	Length – 6 alpha-numerics
Selected upgrade model	*value* – 4-digits	
Payment plan	*set* – one of the types listed above.	

Derive test cases for each of the input domains where each test exercises the largest number of input attributes at the same time. **(13 marks)**

Answer Pointers

This question tested candidates' abilities to select testing techniques and apply them.

a)
For Requirements, expect some clear structuring, identification or labelling of requirements statements, plus a user review to assess completeness of both functional and non-functional needs.

For Design, expect a review to cross-reference requirements against design elements, plus some design-intensive review to assess technique competence for non-functional requirements (where appropriate, e.g. excluding hardware things like 24/7 reliability) and verification of methodology.

For validation, expect a detail run-through of requirements against available functionality, plus some sort of user demo to show look and feel plus other non-functional requirements.

The expected overlapping is the link between requirements, where functionality and performance are identified, with designing for these qualities and then accepting the finished product by testing for these qualities.

b)
A good answer would discuss either an Equivalence approach to the design of tests, or a Boundary-value approach.

Equivalence: a Range has 1x valid and 2x invalid outcomes; Value has 1x valid and 2x invalid; Set has 1x valid and 1x invalid; and Boolean has 1x valid and 1x invalid.

Boundary value testing: Range has 2x valid outcomes (top and bottom) and 2x invalid; Value has 2x valid (inside edges) and 2x invalid (outside edges); Set has 1x valid (in) and 1x invalid (out); and Boolean has 1x valid (true) and 1x invalid (false).

Examiner's Guidance Notes

This question was designed to encourage candidates to apply specific test strategies to complex data. Part (a) asked for critical judgement to be applied for testing two non-software items, requirements and design, then for testing the complete product (validation). Too many answers relied on trying to fit white box and black box testing on to non-software items, and on to user-focussed validation testing. Very many answers said, in effect, build it and find out. The point of testing in early lifecycle, as some answers discussed, is to prevent simple mistakes in planning documents from cascading into big errors in software items.

Part (b) offered a scenario to create opportunity, again, for critical selection of tests. The question pointed to efficiency, clearly signalling the answer sought.

Question 5

Software engineering, as a systematic, disciplined, and quantifiable approach to application development and maintenance, is considered by many today to have made significant progress in meeting the needs of users and the budgetary requirements of managers.

Write a report that clearly defines the concept of software engineering and provide a general discussion of the progress made in the industry in terms of the expectation, adoption, and successful application of engineering practice. **[25 marks)**

Answer Pointers

This question assesses the candidate's awareness of the fundamental progress made in producing "quality" application through the application of software engineering principles. A good answer should produce:

A structured report with appropriate headings, including introduction and conclusion;

Introduction providing an overview of the software crises re failure to meet user requirements, delivery on time, reduce application backlog;

Definition that highlight the importance to any engineering profession of possessing qualities of being systematic (methodical), disciplined (manageable), and quantifiable (measurable);

Body of report that discusses:

> the positive application of methods for requirements elicitation (prototyping), design (structured), implementation coding and maintenance (modular) that improves product quality;
>
> The management of software projects enhanced by the use of tools to measure and support processes (CASE tools).
>
> The limitations to all these are considered to be in areas such as the lack of standards, and education and professionalism within the industry.

Conclusion that craws attention to progress, but some issues of the software crises still remain in particular, the application backlog and project management.

Examiner's Guidance Notes

This question was designed to test candidates' knowledge of the progress made in software engineering as practice in terms of maturity of tools, accuracy of methods, and quality of products/processes in the realms of customer and client application environment.

Generally, many of the answers submitted recognised the origins of software engineering, were able to identify examples of the progress made, but were not structured in the form of a report and were unable to discuss the current weaknesses and issues of concerns within the industry. In particular, a significant group of candidates in some centres either did not understand the question, or simply homed in on two or more keywords and described the software development life cycle and the phase(s) to which the selected keywords belong.

It is hoped that in future candidates will have a better appreciation of the robust engineering standards of practice in terms of methods, tools, techniques, and quality to which software engineering (as a relatively young discipline) aspires.