THE BCS PROFESSIONAL EXAMINATION
Professional Graduate Diploma

April 2005

EXAMINERS' REPORT

Software Engineering

General

There is evidence of continued improvements in the scripts produced by many of the candidates this year in terms of the general standard of the written work, and the effort made to apply knowledge in the context of the problem situation.

The questions set have been generally demanding as befits honours level work, but effort has been made to guide candidates to formulate answers that exhibit the structure, length and depth of responses expected.

The paper as a whole would appear to be strongly focussed on conceptual aspects of the software engineering, life cycle models, and the early phases of requirements gathering, analysis, and modelling. It is noticeable that the vast majority of candidates attempted the two major questions in these areas (Q3 and Q5) compared to the others. The reasons are unknown, although it could be argued that the subject matter of these questions are the ones most student are likely to remember from their course.

As in previous years there seemed to be a significant minority answering more questions than was required in the examination, and thus reduced the time for each question and limited their opportunity to obtain the full range of marks for the questions attempted. Of the 30 candidates in this category, only 5 did one or the other of the two questions referred to above.

Finally, on some scripts, student had not completed the front cover with the number of the question and part-questions attempted. Further, the order of answers (and part-answers) was unpredictable in the sequence in which they occurred. This made it very difficult to find, then determine whether a written response was a continuation answer for a previous question, and therefore assess and record the mark for that question as a whole.

Question 1

1. *a)* The following is an outline specification of a project.

> *This project is about developing a web portal for hotel services. This means the consumables and services bought by hotel operators.*
>
> *The project is to research and prototype the appearance and look/feel of the web portal, populating it to operate as a 'virtual store', with inventory provided by others. Users will be suppliers who supply the inventory for the portal, and consumers will be hotel operators who browse and buy. In addition, the portal operator will need to attract customers, both to buy and to supply.*

Using the UML use-case diagrams, create scenarios that will aid requirements elicitation. Each actor and each way of acting should be given. Clearly label your scenarios with the names of the actors and the type of interaction that each actor is using. **(18 marks)**

b) Select and explain an area of requirements that, in your opinion, is not well handled by use-case diagrams. **(4 marks)**

c) Describe another method used by software engineers to elicit requirements. **(3 marks)**

This project is about developing a web portal for hotel services. This means the consumables and services bought by hotel operators.

The project is to research and prototype the appearance and look/feel of the web portal, populating it, finding 'levers of desire' to attract subscribers (both to buy and to advertise), and creating viable examples of ways to make incomes.

The portal will operate as a 'virtual store', with no inventory. Users will be consumers who browse and buy, and suppliers who supply the inventory for the portal. In addition, the portal operator will need to attract customers, both to buy and to supply.

Answer Pointers

a) A good answer identified 3 actors – Buyer, Supplier and Operator (6 marks) , and showed the Buyer to act in scenarios 'browsing' and 'buying' (4 marks), the Supplier to act in scenario 'advertise' (2 marks), and the Operator to act in scenarios 'monitor portal traffic', 'attract suppliers', and 'attract buyers' (6 marks). (18 marks)

b) Somerville Ch 6 discuses Requirements elicitation. He identifies Use –case scenarios as applicable, but lacking ability to describe areas of social and organisational need, so-called 'implicit' requirements that reflect what actually happens, rather than what is idealised formally to happen. (4 marks)

c) Prototyping is also a means of requirements elicitation. Here, an executable model of the system is demonstrated to end-users and customers in order to obtain informed feedback on preferred functionality. (3 marks)

Examiner's Guidance Notes

This was a practical problem that asked for knowledge and application of UML use-case diagrams. Most candidates who attempted section (a) received high marks. Alternative verbs, such as 'advertise' or 'broadcast' instead of 'attract', or 'maintain inventory' instead of 'monitor traffic', were equally acceptable because they indicated clearly what the candidate intended.

For part (b), candidates were equally divided between truly early-lifecycle techniques and late lifecycle techniques such as DFDs and ERDs. Clearly, DFDs and ERDs are used in detail design, not requirements capture. Part (c) produced the same spectrum of answers; with some candidates confusing design methodologies such as SSADM as an alternative to use-cases diagrams for requirements capture.

Question 2

2. *a)* "Software Configuration Management (SCM) is the management of change throughout the whole of the software life cycle."

Describe the basic steps involved when performing configuration management. **(5 marks)**

b) Suppose you are a software engineer in a company that develops software components for a software product. There are older versions of the product with a large number of customers. The new product was introduced in 2002, and has been updated every year since then.

You are given the goal of trying to reduce the time taken to respond to error reports. Identify suitable steps to create a process improvement programme for your SCM, and in particular consider goals, questions and metrics to derive the steps you identify. Give reasons for the choices made. **(20 marks)**

Answer Pointers

a) A good answer should present a set of steps such as:
Identification: number and label items e.g. test specs, module code, build lists
Version Control number and label version baselines
Change Control number and label change requests, impact analysis, number and label accepted requests.
Configuration Audit identify and label test specification with expected results, and the associated build list of modules in this particular configuration and whose functionality is tested by the test spec. Perform tests. Report results of test.

Configuration Reporting. Reports each describing a particular configuration-controlled status e.g test specs associated with particular build lists, module versions in specific build lists.
(5 marks)

b) A good answer picked up the hint of the GQM paradigm and discussed:
Goal – faster response to reported errors in s/w product (given in question) (2 marks)

Questions: How can erroneous product be identified?
How can response process be speeded-up?
How can consequential errors be avoided? (5 marks for two questions linked to the Goal)

Process improvements covers three areas; time, resources, and occurrences (frequency). The metrics cover these three areas. (6 marks for some plan of improvement.The model given here is an example.)

Ideas for Process Improvement programme include:
Measuring and publishing the metrics identified above, because making the measures visible will concentrate minds.
Allocating management responsibility for improving these measures, because responsibility focuses action.
Tool support for CM, because of the complexity of the job.
Training for CM staff, because of the complexity of the job. (9 marks for a minimum of three PI ideas, with rationale linked to the Goal and the questions.)

Examiner's Guidance Notes:
This question followed a popular 'define and analyse/synthesize' model. The initial part asked about the disciplines of configuration management. Many answers covered the basics of version and change control, but significant points that were consistently missed were impact analysis (part of change control) to test for unintended consequences of the proposed change, and configuration audit (testing) with configuration reporting, when the assemblage of parts ('the configuration) is verified as usable with definition of provenance (which tests, which compiler etc.)

Part (b) attracted many, varied answers but few that looked at the problem in a holistic way. Many candidates tried repeating the steps of configuration management without trying to relate them to the problem in hand. Good answers recognised the hint in the question, and addressed the goals of improvement, how they might be recognised as having been achieved (the questions), and how progress towards the goals might be measured (the metrics). Some candidates managed part of this, and received part-marks.

Question 3

3. Compare and contrast the following pairs of process models, giving particular attention to the tools, techniques, and life cycle phases of the project as progress is made towards a complete system:

a) Dynamic Systems Development Method and the Incremental Model **(10 marks)**
b) The Spiral model and Component-based development **(15 marks)**

Answer Pointers
This question assesses the candidate's knowledge of process models and general awareness of the similarity and differences between pairs of process models. Thus, a good answer should be well written, clearly structured, and cover the following:

a) **I. Their (DSDM and Incremental) similarity**. These include:
- increments: build, test, integrate, operate
- core business value prioritised
- linear sequential model

II. Their contrasting features. Thus for DSDM:
- More of a framework encompassing project and development techniques
- principle of 80% of the system produced in 20% of the time
- resources remain fixed, functionality variable
- time boxed (60 to 90 days), and concurrent teams
- configurability and reversibility at every stage
- modelling (business, data, process)
- application generation using 4GL

For the Incremental Model
- staggered linear sequences
- combines the elements of the waterfall model with prototyping
- each increment delivers an operational product
- product is finished when all requirements are met.

(10 Marks)

b) **I. Their (Spiral and Component-based) similarity.** These include:
- evolutionary software process model
- iterative approach

II. Their contrasting features. Thus, for Spiral

- series of incremental releases (from paper to firmware)
- 6 task regions or framework activities (client communication to client evaluation)
- task sets per region
- Concept development to product maintenance spiral
- technical risks considered at all stages

For Component-based
- an OO technical framework
- reuse via class librararies and repositories
- identify candidate classes
- Engineering construction and release task region

(15 Marks)

Examiner's Guidance Notes
This question was the second most popular question on the paper, primarily because it was based on different process models. It was clear from the scripts that students understood at least one process model in great detail, but only a very few had knowledge and awareness of all four models listed, especially component-based development. Therefore, many candidates were unable to attain marks for appropriately comparing and contrasting the latter approach.

Further, some centres did not appear to understand what is required when you are asked to "compare and contrast" the model pairs. Thus rather than focussing on such things as there similarity and differences, many of the answers consisted of detailed description of each model followed by a list of advantages and disadvantages.

Finally, many students understood the Spiral model, and could draw the diagrams for this and the DSDM approach, but there was a significant number that appeared to have no knowledge or awareness of the component-based development.

In conclusion, it was a more demanding question than the students might have imagined once they started to formulate their answers, and the detailed answers given by many to part a) or part b), would have left very little time for them to address the other with a similar depth of responses.

Question 4

4. Engineering is considered to be a discipline with guiding scientific principles and universally applicable methods for the construction of systems.

a) Discuss the guiding principles of software engineering in the context of constructing large programs and the mastering of complexity. **(15 marks)**

b) Evaluate the relative success of software engineering in adopting mathematical principles and producing universally applicable methods and tools. **(10 marks)**

Answer Pointers
This question assesses the candidate's understanding of software engineering principles within the wider context of the engineering discipline, and general awareness of progress made in this area so far.

A good answer should be well written and clearly structured.

a) In this instance the candidate should make a case for software engineering to be considered as an engineering discipline. Thus, the answer should cover such things as:

Constructing large programs require appropriate tools for:

- specification and testing of software;
- version control and component integration;
- documentation audit trail management;

Constructing large programs require appropriate and rigorous methodologies, that are both logical and systematic in their application to:

- analysis
- design
- implementation

In mastering complexity, decomposition into manageable units, and integration of robust units to form large programs, are important techniques.

Project management methods for meeting organisational and financial constraints;

(15 Marks)

b) The candidate should highlight some of the areas universal applicability of software engineering methods may be somewhat weak. Thus the answer should cover:

- Formal specification of software. Well-developed concept but limited training, and little agreement on standard or making it compulsory for all software related activities
- Software metrics. Lacks precision, questionable underpinning, and variability in results

(10 Marks)

Examiner's Guidance Notes

This question was the third most popular question on the paper, but possessing the lowest overall average mark. Many candidates for part a) would appear to have ignored the clause ".in the context of constructing large programs and the mastering of complexity" and focussed more on general project development life cycle principles. However, part b) was the most disappointing, as there did not appear to be a general appreciation or awareness of the relationship between engineering practice and the adoption of formal methods and software metrics generally.

Question 5

5. The manager of a local bank has authorised requirements engineering to be carried out by a team of IT consultants with the primary aim of producing the software specification to manage customer accounts. The manager's requirements are written in natural language and state that the system should be user-friendly and calculate interest daily for interest-bearing accounts.

a) Using the above scenario where appropriate, define each of the following pairs of terms making clear distinctions between each:
   ii) Stakeholder and system requirements
   iii) Functional and non-functional requirements
   iii) Throwaway and evolutionary prototyping **(15 marks)**

b) Using the local bank scenario outlined:
   i) briefly discuss the merits of using natural and structured language specification
   ii) specify the interface to a customer account using a simple program description language (PDL)

(10 marks)

Answer Pointers

This question assesses the candidate's knowledge of specific requirements gathering tools and documentation, and general awareness of instances where they are used and how they might be applied.

A good answer should be well written and clearly structured and cover the following:

a)

- Stakeholder and system requirements
  Stakeholder requirements include function and non-functional requirements expressed in a manner that is meaningful to users who don't have detailed technical knowledge;
  System requirements include complete and consistent specification of the whole system using an implementation model and useable as a basis for a development contract and the starting point for system design.

- Functional and non-functional requirements
  Functional requirements are the services a system is expected to provide;
  Non-functional requirements may define constraints on the system or relate to emergent system properties.

- Throwaway and evolutionary prototyping
  Throwaway. An evolutionary development process that experiments with poorly understood parts of a client's requirements and is discarded once the outline of the system development has been clearly identified and understood.

  Evolutionary prototyping progresses from initial experimentation with a customer requirement and progressively adding features proposed by the customer until the system is complete.

(15 Marks)

b) Natural language aids understanding and is flexible but can be ambiguous (e.g differentiating between mandatory and desirable requirements).

Structured language as a restricted form of natural language ensures a degree of uniformity and minimises ambiguity, but may not necessarily be understood by non-specialists.

```
INTERFACE manage_customer_accounts (AccountNumber)
BEGIN
    IF (InterestBearing(AccountNumber))
    THEN CalculateInterest(AccountNumber, DAILY)
    ELSE
        Manage(AccountNumber)
END.
```

(10 Marks)

Examiner's Guidance Notes

This question was the most popular of all the questions, but it had one of the lowest averages. This may be explained by the inability of many candidates to identify the distinctive feature of the pairs of terms listed in part a), especially stakeholder requirements as compared with system requirements. Thus, many candidates simply provided a detailed description of the requirements gathering and analysis phase.

Further answers to part b) was most disappointing as there did not appear to be a general appreciation or awareness of the use of structured languages in writing specifications, and any ability to apply such a language for describing the interface for the bank application. It is to be noted, a significant number of candidates spent time developing a GUI interface in a sort of Visual Basic style sheet to no avail.