THE BCS PROFESSIONAL EXAMINATIONS
BCS Level 5 Diploma in IT

April 2008

EXAMINERS' REPORT

Software Engineering 1

General Comments

This is a technical paper about Software Engineering. Questions seek to test candidates' knowledge, and ability to apply that knowledge. It is a poor answer that simply describes recalled information. It is a good answer that can then show application of the recalled knowledge.

This exam encourages reflection, meaning critical review of what the topic means in the wider context of software engineering. Candidates are encouraged to read more widely about 'high end' goals of software engineering, such as ethics and people management, and reflect on how development processes hinder or help the higher aims.

Responsibility towards customers cropped up in several places in this exam, with a worryingly wide range of views. The ethical and quality norms of Software Engineering are plain; customers rarely know what they want, but it is our professional duty to help them towards an understanding of what we can do for what cost. Customers are not infallible, neither is their ignorance to be exploited. It is our professional duty to maintain continuous skill development to stay abreast of the rapid changes in our technology. The Society's guidance on ethical behaviour is expected to be known and understood by all aspirant members. This point needs close and serious attention by tutors and candidates alike.

Some candidates had not completed the front cover with the number of the question and part-questions attempted. This made it very difficult to find all the parts of an answer, then determine whether it was a continuation of a previous answer, and assess and record the mark for that answer as a whole.

Nonetheless, the candidates' responses to the Software Engineering examination questions showed a good understanding of the subject. Most marks for individual questions are average.

A good spread of attainment was evident. Some candidates were well prepared.

Question 1

a) Explain why configuration management is a necessary practice for software engineers. Your answer should give TWO examples, one relevant to software development and another relevant to maintenance as support for your explanation.

(10 marks)

b) The following is an outline specification for a small project. Discuss the criteria you would use to determine the life cycle model that this project should follow, and make a recommendation about selecting a suitable life cycle model.

(15 marks)

This project is for a company that specializes in health-and-safety training with chemical process companies. Their contracts include facilities maintenance, repair, supervision, risk assessment, and provision of health and safety training. The company's business model relies heavily on open and honest dealing in order to establish a culture of safety and trust when negotiating with potential customers.

This project is about creating a suitable web site for the company.

The project should analyse and create suitable images and text to promote the culture and capability of the company. It is highly likely that a database will be required to support the quality of web site that is required.

Answer Pointers

A good answer addresses these points.

a) CM is essential because software is very easy to change. A single keystroke is enough to invalidate months of testing.

b) During development, CM controls **build-state,** allowing different programmers to develop modules in parallel. CM also control **variants** for alternative, equivalent functionality such as different platforms, different operating systems, or different deployment characteristics. **Versions** of modules, representing incremental development, are also controlled. CM links test-specifications with groups of modules with requirements. It 'glues' the developments together in 'cognate' groups.

During **maintenance**, CM facilitates **perfective** by supplying coherent records of what, how tested, where and how built. CM facilitates **corrective** in the same way, allowing regression testing on the 'right' set of changed modules. CM facilitates **adaptive** change by allowing the addition of new variants and versions to the deployment product set. For some 4GL products, where each customer has a different version of the product, CM is essential for identifying the deployed version held by any customer who might make an error log thereby creating the problem 'which set of modules do we build to investigate this?'

b) **Issues:** The message for the website is ‘open, honest, safe’. The intended customer market is vertical (one sector) in a specialised domain (process engineering) with a strong quality culture (safety-conscious). A developer will need to work closely with the client in order to create the right images, text and presentation to address this market. Both will need to learn; the developer needs to learn about the target market, and the client needs to learn about the technology.

Life cycle model; These issues all indicate the need for an evolutionary or Extreme Programming LCM, where the client is closely involved in the evolving product.

RAD approaches likewise involve the client but this project does not use RAD tools for infrastructure or application development so RAD is inappropriate.

Sequential, build-test-deploy approach as a one-pass is not appropriate since it makes no allowance for the learning curve of the developer or the awareness curve of the client. Repeated design-build-test episodes are the essence of Evolutionary development.

Examiner’s Guidance Notes

Part (a) was answered very well, on the whole. Candidates are to be congratulated on a clear understanding of configuration control and the software entities therein.

Part (b) was not well done. In particular, the question contained many clues to help with the desired answer, such as ‘specialised engineering domain with a safety culture’. Many candidates offered recommendations without reference to any aspect of customer character. Some candidates described a life cycle model for developing a web site. This was not the point of the question. Other candidates correctly identified an iterative approach, but did not include the customer in the iterations!

Question 2

a) Describe the features and capabilities of a CASE tool with which you are familiar.

(10 marks)

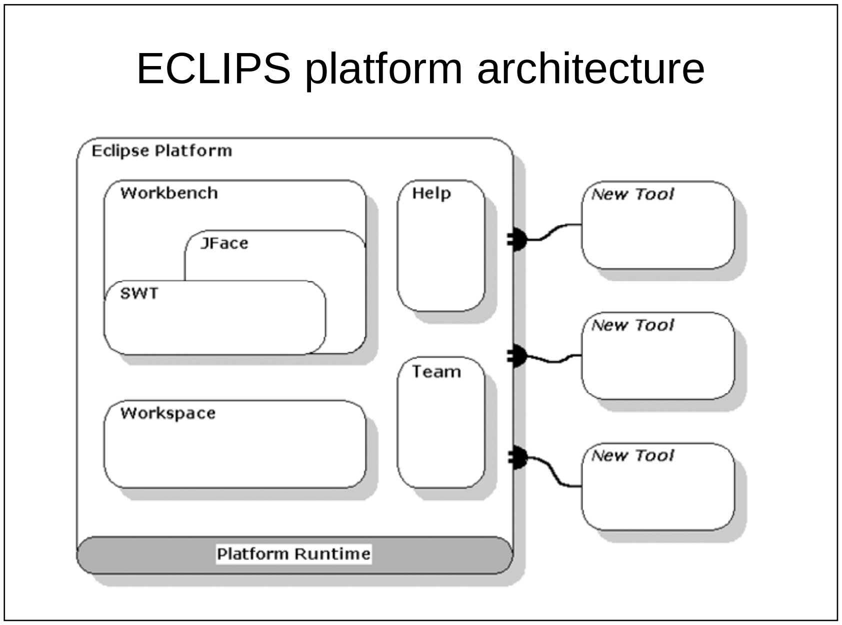
b) Give TWO examples where CASE tools have shifted the engineering focus of the development model in which they are used.

(15 marks)

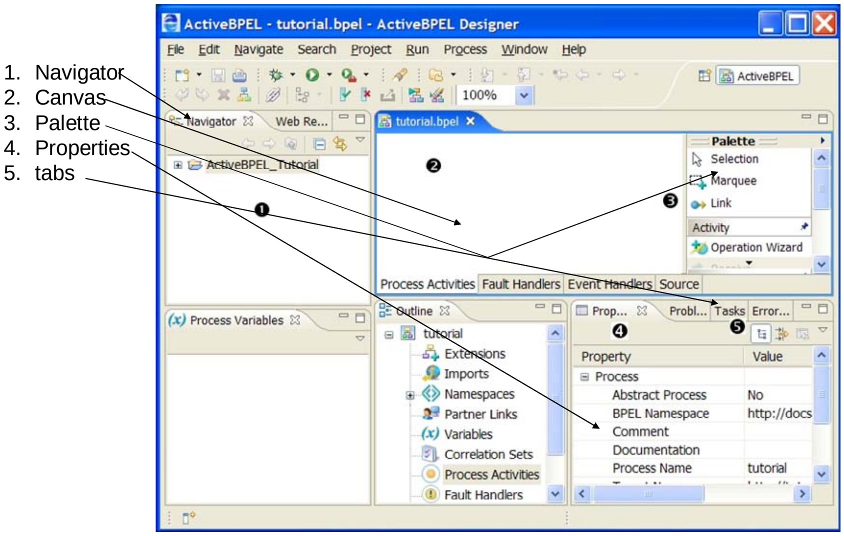
Answer Pointers

A good answer addresses these points.

a) The answer expected is a description from the Candidate of a CASE tool that he or she has used. A popular format is expected to be an 'Eclipse' format with an interface that allows the project to connect with the development processes, and the ability to attach specific toolsets for specific operations (e.g. internet access, testing, deployment etc.) See **Picture 1.** Alternatively, the candidate may describe the functionality experienced, citing something like **Picture 2** that identifies which bits of which environment are visible to the developer. Alternative descriptions at similar cognitive level are acceptable.



Picture 1 – CASE tool architecture



Property	Value
Process	
Abstract Process	No
BPEL Namespace	http://docs
Comment	
Documentation	
Process Name	tutorial

Picture 2: CASE tool screen layout specifics

b) A good answer will identify two areas, typically from the set

- shift from development to design and specification,
- consequential increase of emphasis on requirements capture and
- customer involvement.

Or similar commentary about the demotion of coding in favour of designing with CASE tools that generate the code, and LCMs that involve the customer the better to capture the requirements for successful (faster development with more accurate requirements and fewer phase-specific errors) design.

Examiner's Guidance Notes

For Part (a), this question sought expressions from candidates who had used and could discuss a relevant CASE tool. The model answer was described with a popular ECLIPSE model, but any CASE tool is acceptable as long as the candidate understands its capabilities and rationale.

For Part (b), candidates generally did not appreciate the reasons why software engineers use CASE tools. Some candidates agreed with the question ("yes, it shifts emphasis.") but did not give any reasons or further explanation. Many answers gave descriptions of functionality (again!) without realizing the consequences that impacted on the software development process.

Question 3

a) You are asked to manage a development project for a major company. You know that the company has a strong ethical stance. What THREE rules would you broadcast to your project team to ensure they behaved ethically on this project?

(10 marks)

b) Compare Function Points with Lines of Code as techniques to estimate the cost of a software project. In your opinion, which method is more reliable, and how would you make its reliability even better?

(15 marks)

Answer Pointers

A good answer addresses these points.
a) Any design method is acceptable if the candidate shows how it breaks the problem into parts, often with some checklist to guide the breakdown and limit its extent.

Reviews: the breakdown makes a problem tractable, designed in small and well-defined units. This makes review of internal intention (algorithm or data) and external interface (API) precise and exact.

Interoperability; the breakdown makes for well-defined interfaces, the easier to make interoperable.

Maintenance: modularity with low coupling is well-known for ease of maintenance. A decent design will have structured the development into its parts with, at worst, reasonable coupling so making the maintenance job that much easier.

b) The answer should show a clear understanding of ‘ethics’ – it can be expressed as ‘ends and means’ meaning targets and processes, or results and pressures on people to produce them. Or it could be the observation that there is always tension in a team when targets seem hard or frustrating to reach. Or it could mention bullying in the workplace, the ‘blame’ culture.

An ethical approach could be informal-personal, about trust in a manager, or keeping faith with promises made. Or it could be formal, such as codes of conduct for tolerance or treatment of all grades of personnel.

c) Team work has a range of roles and needs a range of skills for delivering a software product. The Productivity thing usually means ‘coding’ but could also mean ‘timekeeping’.

Strategies to combat slow coding could include reassignment to testing, or even some skills appraisal to find the strengths of the individual – for communication, documentation, testing or customer relations. Find the area of an individual’s best contribution, and promote it.

Examiner’s Guidance Notes:

Part (a) was not well done. It is clear that candidates do not appreciate what ethics means to a professional person, despite an ethical stance being required of every member of the British Computer Society. Candidates need better preparation in their awareness of what constitutes an ethical position. It is not ‘doing what the company wants’. It is, in part, a creative tension between market forces and personal concepts of right and wrong. Ethics does not tell you “to stay with the development until the end” but to be aware if the company’s demands exceed ethical boundaries such as personal or public safety.

In part (b), many candidates tried to avoid controversy by suggesting the Lines of Code be used in conjunction with Function Points. This question sought each candidate’s awareness of the common pitfalls of estimating, and the equally common methods to overcome them. No matter what method you prefer, getting a personal or company benchmark of previous history and productivity lends realism to what is otherwise a speculative procedure that remains one of the toughest software engineering problems to address.

Question 4

a) In the context of object oriented design and programming, explain the meaning of each of the following terms and give an example of each using either UML or any programming language of your choice:

i) Abstraction
ii) Encapsulation
iii) Modularisation
iv) Inheritance

(10 marks)

b) Consider the domain of libraries, identify five classes of objects and define attributes for each class and operations that apply to it. In the case of each class, provide an instantiation of an object belonging to it.

(15 marks)

Answer Pointers

A good answer addresses these points.
(a) In the context of object oriented design and programming, explain the meaning of each of the following terms and give an example of each using either UML or any programming language of your choice:
i) Abstraction is a simplification or generalisation of the software solution which allows one to work with concepts, such as objects and classes in terms that are familiar in the problem environment without regard to irrelevant low level details. E.g. any class diagram or code for a class and suitable commentary.

ii) Encapsulation means that all of the information about an object such as its data attributes, operations, internal objects if it is composite, set values, and other related information is packaged up under one name, the object name. E.g. any class/object diagram/code with associated encapsulated information and suitable commentary.

iii) Modularisation is means by which a software system is divided into separately identified components that when integrated together form the solution to the software requirements. A large program comprised of a single module cannot easily be grasped by a programmer; whereas, smaller individual modules are more understandable. Major classes within an OO system form its modules. E.g. any decomposition of a system into modules with suitable commentary,

iv) Inheritance is means by which the common attributes and operations of a class can be passed down to sub-classes and to instances of that class, i.e. specific objects. E.g. any examples showing classes inheriting from a super class with suitable commentary.

(b) Consider the domain of libraries, identify five classes of objects and define attributes for each class and operations that apply to it. In the case of each class, provide an instantiation of an object belonging to it.
Classes: books, borrowers, library staff, libraries, library cards, library catalogues etc
Book attributes: author(s), title, book type, etc
Book operations: add a book to catalogue, issue a book to borrower, recall a book, etc
Instance: the book, David Copperfield, by Charles Dickens, Penguin Classic edition.
Borrower attributes: name, address, type of borrower, books borrowed, etc
Operations: register as new borrower, obtain library card, check borrower status, borrow book, return books, etc
Instance: John Doe, a student borrower,
Library staff: name, employee type, pay, hours of work, place of employment, etc
Operations: record new employee, calculate pay for hours worked, calculate work schedule for staff
Instance: Mary Smith, head librarian
Library: name, location, hours of opening, type of library, etc
Operations: change opening hours, report status of library stock, report library borrowing records, report number of borrowers at library, etc.
Instance: The British Library
Library card: borrower name, identifier, photo, library name, bar code, etc
Operations: issue card, revoke card, update card contents, etc.
Instance: John Doe's library card for the British Library

Examiner's Guidance Notes

Generally students were able to give definitions of the terms as found in Software Engineering textbooks; however, the examples were sometimes confused or in the worst cases missing. There were some excellent answers for part b, but the majority were confused and made poor use of OO notation which was not a particular requirement for answering this part of the question.

Question 5

a) State three major factors that influence the cost of software development.

(3 marks)

b) Describe a cost estimation model, showing the formulation of the model and state any assumptions on which it is based.

(8 marks)

c) In the process of project planning, define the following terms:

i) Activity
ii) Milestone
iii) Critical Path

(6 marks)

d) Draw a simple Gantt chart for a project following the classic waterfall model for software development and state any assumptions on which your chart is based. On your chart you should identify all major milestones and activities.

(8 marks)

Answer Pointers

A good answer addresses these points.
a) State three major factors that influence the cost of software development.
Product attributes: cost, complexity
Hardware attributes: memory, runtime performance
Personnel attributes: capability, experience of staff
(any three of the above)
b) Describe a cost estimation model, showing the formulation of the model and state any assumptions on which it is based.
Assuming that students will describe some form of COCOMO, but other models are possible.

Basic COCOMO computes software development effort (and cost) as a function of program size expressed in estimated lines of code.

The relevant equations are as follows:
E = a sub b (KLOC) exp (b sub b)
And
D = c sub b (E) exp (d sub b)
E is Effort in Person-months
KLOC is Lines of Code in 1000s
D is Development months in chronological time

The coefficients a sub b, b sub b, c sub b, and d sub b are dependent on the type of project class: organic, semi-detached, or embedded and given by a table which distills data from a number of project studies.
c) In the process of project planning, define the following terms:
i) Activity is a unit of work representing a task to be undertaken within a project; each activity should have an estimated time associated with it.

ii) Milestone is a product of a software engineering activity or task that has been successfully reviewed. Milestones usually are identified at the end of each major phase of activities, e.g. the design of the system might be a milestone associated with the completion of the Design Phase and all of its activities.

iii) Critical Path is determined by analysis of the activity network which shows the order in which the identified activities must take place; it represents the chain of activities/tasks

d) Draw a simple Gantt chart for a project following the classic waterfall model of software development and state any assumptions on which your chart is based. On your chart you should identify all major milestones and activities.

Any chart showing at least Requirements, Specification, Design, Implementation, Testing, with suitable milestones and intermediate activities.

Examiner’s Guidance Notes

The factors influencing cost given in students' answers were rarely those anticipated in the model answer, but where clearly argued and justified, unanticipated answers were acceptable. Few students were able to describe a cost estimation model; and only a couple gave the formulation along with assumptions.

Many students were able to give definitions, but some were confused by the mention of “the process of project planning”. For example, they thought they needed to define what an project planning activity was.

In part d, most student demonstrated that they knew the phases of development and some key activities, but were not so clear on identifying milestones. However, few students actually drew a proper Gantt chart.

Question 6

a) Consider error messages found in any software application and suggest ways in which these might be improved.

(6 marks)

b) Three principles of user interface design are *consistency*, *minimal surprise*, and *recoverability*. Explain what is meant by each of these principles and give an illustrative example of each.

(9 marks)

c) What considerations are relevant when designing a user interface in order to make it more accessible for use by people with impaired vision?

(4 marks)

d) Discuss the business case for giving consideration to usability and accessibility when designing software applications for a worldwide market, e.g. web-based applications.

(6 marks)

Answer Pointers

A good answer addresses these points.
a) Consider error messages found in any software application with which you are familiar and suggest ways in which these might be improved.

At least two error messages and their improved forms.

b) Three principles of user interface design are consistency, minimal surprise, and recoverability. Explain what is meant by each of these principles and give an illustrative example of its application.

Consistency – the same user action gives the same response throughout the interaction.
E.g. any reasonable example taken from a user interface known to the student.
Example: Same keys for cut and paste used throughout

Minimal Surprise – the behaviour of the system surprises the user as little as possible; it follows well known metaphors with which the users are familiar. *Example: the Desktop metaphor used in Windows*

Recoverability – the user can easily “undo” actions done in error or by mistake and recover their original work or return to state before the error occurred.
Example: the “undo” command found in most editor applications.

c) What considerations are relevant when designing a user interface in order to ensure that it can be used by people with impaired vision?
Size of display; font types and sizes; colours used; sequence of material and content if screen reader is to be used; textual description of any images,

d) Discuss the business case for giving consideration to usability and accessibility when designing software applications for a worldwide market, e.g. web-based applications.

Universally used applications require internationalisation, to ensure that they are usable and accessible to all potential users in the world.
Any business case along the following lines:
“Conformance with the WCAG 1.0 (and other W3C) guidelines will enhance the market share and audience reach of your Web site by increasing its general usability. Adoption of WCAG 1.0 recommendations also demonstrates your commitment to social responsibility and equity of access to information and services. In addition, many of the WCAG 1.0 checkpoints will directly improve the performance of your Web services and reduce the maintenance effort required.” [W3C/WAI Education and Outreach Working Group’s draft: Business Benefits of Web Accessibility, 2001]

Examiner’s Guidance Notes

Few students understood part a; many thought it was about the need for eliminating errors through testing. Perhaps the question should have explicitly mentioned “user error messages” although a good answers need not have been restricted to user error messages. Few students gave examples of error messages from real applications.

In part b, many students failed to understand this part was about principles applied in the design of User interface; e.g. they discussed the need for an application to be able to recover from an error and backups.
In part c, students generally demonstrated some awareness of accessibility issues.

In part d, most answers totally ignored the “business case” and simply discussed usability and accessibility issues. Many mentioned the need for secure payment but without developing this clearly in their answer.