BCS THE CHARTERED INSTITUTE FOR IT

BCS HIGHER EDUCATION QUALIFICATIONS
BCS Level 5 Diploma in IT

SOFTWARE ENGINEERING 1

Wednesday 30th March 2016 - Afternoon
Answer **any** FOUR questions out of SIX. All questions carry equal marks.
Time: TWO hours

Answer any Section A questions you attempt in Answer Book A
Answer any Section B questions you attempt in Answer Book B

The marks given in brackets are **indicative** of the weight given to each part of the question.

Calculators are **NOT** allowed in this examination.

Section A
Answer Section A questions in Answer Book A

A1. Maintaining software contributes significantly to the overall cost of a software product.

a) The cost of changing software during its maintenance phase can be reduced by investing in good Software Engineering techniques during its development phase. Identify **THREE** Software Engineering techniques you believe contribute to reducing the cost of maintenance and describe how each of these techniques achieves this.

(6 marks)

b) When a system requires significant and continual change, a decision must be made as to whether or not it is better to discard the old system and build a new one to replace it. Create **FIVE** questions you would ask to assist making that decision.

(10 marks)

c) A software development company has been experiencing significant issues regarding the maintenance of some of its existing systems.

You are a senior Software Engineer responsible for the evolution of a new software product. You have to decide how your Software Engineers will be split between the development team and the maintenance team of the new product.

Discuss and compare which factors would influence whether there should be separate teams for development and maintenance, one team covering both activities, or a small overlap between the teams.

(9 marks)

Answer Pointers

a) Software Engineering practices contributing to reducing the cost of maintenance may include:

Capturing a precise specification through validation reduces the possibility that the software does not deliver what the client expected.

Modularizing the design and code components (e.g. using object oriented development) and cross-referencing the components with the requirements (e.g. using UML's Robustness Diagram) will enable tracing requirement changes to affected components and to the tests that must be redone. Design, code and test artifacts can then be updated.

Configuration management will address policies, processes and tools for managing changing software systems, enabling the tracking of what changes and component versions have been incorporated into each system version.

b) Questions that will assist determine if it is better to discard the old system and build a new one to replace it:

Is the cost of maintenance too high?

Is the system reliability unacceptable?

Can the system no longer adapt to further change, and within a reasonable amount of time?

Is the system performance still beyond prescribed constraints?

Are system functions of limited usefulness?

Can other systems do the same job better, faster, or cheaper?

Is the cost of maintaining the hardware great enough to justify replacing it with cheaper, newer hardware?

c) Factors influencing whether there should be separate teams for development and maintenance, one team covering both activities or a small overlap between the teams:

The development team is familiar with the code, the design and the philosophy behind it, and the system's key functions that would contribute to the efficacy of maintenance.

If the developers know they are building something that they will maintain, they will build the system in a way that makes maintenance easier.

If developers know others will work from their documentation, developers tend to be more careful about documentation and programming standards.

Developers sometimes feel so confident in their understanding of the system that they may not keep the documentation up to date. This may lead to more people or resources to tackle the problem. It may well lead to a long interval from the time a problem is raised to the solution being implemented.

A maintenance team of new Software Engineers may be more objective than the original developers. A separate team may find it easier to distinguish how a system should work from how it does work.

If a new team of Software Engineers works with others who are familiar with the system's development and who are familiar with its documentation, the time for maintenance may be further reduced.

Examiner's Guidance Notes

This was the most popular question in Part A. The majority of candidates were able to articulate Software Engineering techniques they believed to contribute to reducing the cost of maintenance. They were less able to create five relevant questions to assist in deciding whether or not to discard the old system and build a new one to replace it. Most candidates showed some insight into the relationships between software development and software maintenance teams.

A2.

a) Agile Development is a process that values customer collaboration over contract negotiation. Discuss **THREE** issues a Software Engineer should be mindful of when adopting this approach during software development.

(9 marks)

b) Agile Development is a process that values responding to change over following a plan. Discuss **THREE** issues a Software Engineer should be mindful of when adopting this approach during software development.

(9 marks)

c) Compare and contrast the features of software development projects that would make them either suitable or not suitable for agile development.

(7 marks)

Answer Pointers

a) Customer collaboration over contract negotiation:

Customers don't clearly know what they want, change their opinion from time to time, may change their mind when they see the product.

Business people and developers must work together daily throughout the project. Customers need to be willing and able to spend time with the development team.

Details are often revealed during the development.

Difficulty prioritizing items on product backlog.

Developers and users should be able to maintain a constant pace indefinitely.

Project may suffer from scope creep.

b) Responding to change over following a plan:

Details are often revealed during the development, new ideas can be integrated into the product backlog which will require prioritizing.

Customers may reorganize priorities.

Scheduling becomes more challenging.

Refactoring may have to occur more frequently than previously planned.

Less predictable scope/costs.

Developer may use prototypes to aid requirement elicitation and customers may become confused between prototypes and the live system.

c) Features of software development projects that would make them either suitable or not suitable for agile development:

Suitable: Development of a small or medium sized product is being developed. Where there is a clear commitment from the customer to become involved in the development process. Where the requirements are unclear.

Not suitable: Development of large scale products. Safety-critical systems where a complete analysis of the system is essential. Large teams, geographically distributed. Team membership changes regularly.

Examiner's Guidance Notes

This was the least popular question in Part A. The majority of candidates were able to present some key aspects associated with agile development with a small representation demonstrating a broader appreciation of the issues involved. Similarly, many candidates were able to identify a few features of projects suitable/not-suitable for agile development but only a small representation provided a more detailed response.

A3. You have been asked to project manage the development of a small, interactive system for a village library. The system will provide functionality for adding new members (borrowers), updating members' details, lending books and returning books.

Risk type
Technology
People
Organizational
Tools
Requirements
Estimation

Discussions with the library staff have revealed the following facts: there is some confusion and uncertainty about the content and layout of the interface and also the requirements for adding new members and updating members' details; the library may soon be taken over by the main city library.

Discussions with your Software Engineers have revealed the following facts: the uncertainty about the requirements means that the total development time will be difficult to predict; some of the new team members are unfamiliar with testing tools; some key team members may be unavailable at critical times; the team are planning to use some reusable software components which to date have not been tested.

a) List **SIX** different types of risk. **(6 marks)**

b) Create a risk checklist for this project by providing **ONE** example, from the scenario above, for each type. **(9 marks)**

c) Develop a strategy for managing **FIVE** of the risks identified in part b). Your answer should be in the form of a table with **ONE** strategy for each risk. **(10 marks)**

Answer Pointers

a) Candidates should list the six different types of risk.

b) Risk check list, six examples from scenario, one for each risk type

Risk Type	Possible Risks
Technology	Reusable software may contain defects that mean they cannot be reused as planned.
People	Key development staff may be unavailable at critical times.
Organizational	A restructuring results in different management responsible for the project.
Tools	New team members are unfamiliar with testing tools.
Requirements	Client is unsure of some of the interface requirements.
Estimation	Development time is difficult to predict.

c) Examples of strategy should relate appropriately to FIVE of the risks

Risk	Strategy
Reusable software may contain defects that mean they cannot be reused as planned.	Replace potentially defective components with bought-in components of known reliability.
Key development staff may be unavailable at critical times.	Reorganize teams so that there is more overlap of work and people
A restructuring results in different client management are responsible for the project.	Schedule meetings with new management team to explain current project requirements, stakeholders, risks, development
New team members are unfamiliar with testing tools.	Provide training to bring new members up to speed.
Client is unsure of some of the interface requirements.	Adopt paper prototyping approach to guide client through requirement elicitation.
Development time is difficult to predict.	Adopt an agile development approach such as SCRUM

Examiner's Guidance Notes

This was the second most popular question in Section A. The majority of candidates were able to list different types of risks. They were, in the main, confident in associating examples within the scenario with the risk types. The majority of candidates were able to provide appropriate strategies for managing the risks identified.

Section B

B1.

a) Briefly describe the difference between a static and dynamic view of components in a UML model.

(4 marks)

b) Name any three UML diagrams that can be used to model dynamic behaviour and briefly describe the purpose of each one.

(6 marks)

c) You have been given some requirements for an internet based tool hire site. These include the ability of the system to register a new customer with details of name, address, telephone number and email. The system has to generate a unique id for each new customer. Once registered a customer can login and browse the tools catalogue and select the tool or tools they wish to hire. Tools are hired out in multiples of whole days. When a tool is hired out the catalogue displays its outstanding hire period, if a tool is not already out or booked it is displayed as immediately available. If a customer selects an available tool the order is fulfilled and the display is updated. If a customer selects a tool already hired, the system puts the customer on back order and updates the display with the additional hire period. When back orders become available for fulfilment customers are notified by email.

1) Draw a suitable Use Case Diagram, to show the tool hire site requirements.

(10 marks)

2) Draw an appropriate UML diagram, showing the flow of control by time sequence.

(5 marks)

Answer Pointers

1)

Candidates should recognise that a static view is concerned with structure of an application with classes, interfaces, objects and components being typical examples of UML static diagrams. Static diagrams map onto language implementation. The static view is commonly associated with the Class Diagram. A dynamic view should be described as the view which describe runtime behaviours of the application and components such as Use Cases, interaction diagrams and activity diagrams which provide a view of collaboration and sequence in the application.

2) Dynamic UML diagrams can be chosen from sequence, activity, collaboration, use case and statechart. The purpose should be explained briefly e.g. Use Case, used to gather requirements and get a user view and show the factors which influence how actors interact with requirements. An interaction diagram can be explained as exchange of messages between two or more components and could be shown as a sequence or collaboration. It would be acceptable to show a sequence and collaboration diagram as two distinct answers describing the differences between them.

3)

a) This question requires the candidate to interpret basic requirements and use the UML to provide a dynamic representation model. The proper identification of cases is expected and appropriate naming for cases and actors. A standard Use Case diagram is required here to show the basic requirements as set out in the description provided. A good answer will provide a system boundary, containing use cases from the requirements list such as a hiring order with extending relationships shown to back order and fulfilled order. Actors should be identified as in or outside of the boundary.

b) Here, the question extends the basic knowledge and understanding required from part a, it tests the candidate's ability to model dynamic flow of timing between components using a simple sequence diagram. Candidates will be expected to show an appreciation of Object Oriented concepts such as Objects and methods. The diagram should include identification of appropriate Objects (Customer, Order, Back order) and name methods from each object that are to be used in messaging between objects.

Examiner's Guidance Notes

This was the second most popular question in part B, with around 60% of candidates making an attempt with 44% of them achieving a pass. In 1. a) Most candidates achieved a pass mark or better with around a third of candidates showing no real knowledge of the difference between static and dynamic views. In b) many candidates could not correctly identify more than two UML diagrams to model dynamic behaviour. Most of the candidates achieved a pass mark by correctly naming appropriate diagrams but not giving a convincing description of the purpose. In 1 c) the overall response was disappointing, with many not able to provide a n adequate diagram. The number of attempts which showed very little evidence of knowledge about creating Use Case diagrams was evenly split with attempts that showed a rudimentary appreciation of diagram construction. Some 12% of candidates achieved a pass mark in c 1. For c 2, the distribution was similar to c1. It was interesting to note that quite a few candidates achieved a better percentage in c2 than they did in c1.

B2.

Software verification is part of a testing process which is concerned with ensuring that the product is being built correctly. .

a) In the context of traditional software development

1) Give an outline of the unit testing process for verification.

(5 marks)

2) Compare Top-Down and Bottom-Up integration testing in the verification process

(8 marks)

b) In the context of Object Oriented software development

1**)** Give a brief description of how unit testing would be carried out

(4 marks)

2**)** Describe the process of integration testing for software verification

(8 marks)

Answer Pointers

The question is testing a candidate's knowledge of verification as part of the software testing process. Since the process is quite different for traditional/classical software and Object Oriented software the question is also giving the candidate an opportunity to demonstrate knowledge of the different processes involved.

1)

a) The process of unit testing tests items from the smallest part of the software typically a module. Tests would include such things as interfaces (for I/O), data structures, control paths, execution routes, error checking and data flows.

b) Comparison of the two involves distinction between top down as an incremental approach with breadth first or depth first being described. Stubs as a technique should be acknowledged with stub substitution being a defining characteristic of this approach. Bottom up is from the smallest components upwards with subordinate program parts being gradually integrated. It is contrasted with top down by not requiring stubs. A good answer should indicate knowledge of potential pitfalls such as stub overload and missing errors.

2)

a) This question requires consideration of the particular issues of Object Oriented designs. The essential difference between traditional unit testing centres on the smallest unit being the class or object. Unit testing should be acknowledged as testing of the class and its methods. The techniques that can be used vary but could include fault-based or partition testing amongst others.

b) Integration testing poses a set of issues that cover Use cases through to thread based executions. Examples of either are acceptable, such as collaboration between classes on threads or partitioning techniques based on use cases. A good answer would include behavioural modelling in scenario testing.

Examiner's Guidance Notes

Overall this was the most popular question in part B. over 91% of candidates attempted this question with just over 31% achieving a pass mark. In part a. 1 over half of the candidates managed a pass mark or better. Only very few were able to give a good account of unit testing. Many had no clear idea of testing as part of verification. In part a. 2 over half of candidates achieved a pass mark. Candidates tended to show a better appreciation of the top-down, bottom up process in verification than they showed in unit testing. There were better, complete answers in this part of the question than in a.1. For part b. 1, once again unit testing, in this case within an OO context, seemed to pose problems. Around half of the candidates did achieve a pass mark, although the answers tended to show some insecurity in clearly distinguishing an OO approach and traditional approach to unit testing.

Part b.2 had the poorest pass rate in this question. Many candidates had no distinction between their basic understanding of traditional integration testing and the OO approach. Many candidates who demonstrated a reasonably good appreciation of integration testing in a traditional setting failed completely to offer even a rudimentary appreciation of verification testing in an OO context. In many cases a significant number of borderline candidates failed the question because of poor OO knowledge.

B3.

a) Computer Aided Software Engineering (CASE) tools can be classified according to their function. Briefly explain how the following CASE tools might be used.

1) Business process engineering tools **(2 marks)**

2**)** Project planning tools **(2 marks)**

3**)** Requirements tracking tools **(2 marks)**

4**)** Documentation tools **(2 marks)**

5**)** Database Management tools **(2 marks)**

b)

1**)** Integrated CASE tools seek to bring together a range of software engineering activities. Outline a set of requirements that an integrated CASE tool should have.

(5 marks)

2**)** Describe the role of an integrated CASE tool data repository and outline the typical contents of such a tool.

(10 marks)

Answer Pointers

1) This range of short questions gives the candidate an opportunity to show an appreciation of the functional range of individual CASE tools

a) The modelling of strategic information. Provides Meta model, representing business data objects (part of OOD). Movement of objects through the business
b) Tool that concentrate on project effort, project costs, project estimating, project duration and manpower planning numbers, task efforts.
c) Tracing where systems have been delivered not conforming to requirements, allowing tracing back to original RFP
d) W/P tools and documentation tools for logging all phases of the process e.g., Javadoc
e) Foundation of repository, project database, based on RDBMS or OODBMS

2)

This question allows the candidate to show an appreciation of what an integrated CASE tool might be expected to implement.   Requirement should provide a mechanism for sharing information between and across all software engineering tools. Enable changes to be tracked and managed. Keep track of versions and changes, could also include configuration management control. Facilitate access control for all tools used in the software engineering environment, providing process model specific configuration. Provide a common interface and provide a communication repository.

3)

A more detailed knowledge of the central role played by the repository of a CASE tool is sought in this answer. Items such as data integrity, information sharing, tool integration and documentation based on object descriptions. The contents of a repository can be chosen from examples such as; Application design – Naming conventions, integrity rules, data structures, system diagrams etc.'. Construction-source code, dependencies and change management information. Validation and Verification - Test plans, regression tests, test routines and result, quality measures. Also a possible range of similar contents from Project Management Information, System documentation information and Enterprise information. Good answers might include reference to problems of maintenance of the repository.

Examiner's Guidance Notes

This was the least popular question in part B with just over 54% of candidates attempting it, with a pass rate of approximately 31%. Nearly all candidates who attempted b3 made a good show on a (1...5) .The majority achieved pass marks with many at the higher end. Candidates showed a good appreciation of Case tools with the possible exception of Database management tools, where many confused the tools for management of a database with the use of a database in a Case context. Part b 1) showed that around half of candidates did not appreciate what an integrated Case tool should have, with quite a few candidates showing that they didn't fully understand what an integrated Case tool was. Part b 2) had approximately 20% of candidates showing any real understanding of the data repository with the remaining candidates showing scant knowledge, or providing a single example.