BCS THE CHARTERED INSTITUTE FOR IT

BCS HIGHER EDUCATION QUALIFICATIONS
BCS Level 6 Professional Graduate Diploma in IT

SOFTWARE ENGINEERING 2

EXAMINERS REPORT

October 2015
Answer **any** THREE questions out of FIVE. All questions carry equal marks.

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

General Comments

The following issues appear significant :

1. Coverage of the syllabus. Candidates are advised to demonstrate more consideration to the less popular but relatively modern topics such as process improvement models and design patterns;
2. Subject awareness. A successful candidate should provide more breadth in responses given to all parts of the question by reading more widely publications within the profession as well as recommended text. There is evidence that many candidates attempted to take a common-sense approach to answering questions without evidence of comprehending fundamental concepts. This was particularly the case in topics such as requirements engineering.
3. Examination techniques. Some candidates could improve their time management to ensure that the majority of questions and their sub-parts are given sufficient time and attention;
4. Answers should be legible, well-structured and formatted.

It is important that candidate responses to questions are: comprehensible, exhibit breadth and depth in knowledge; and are appropriate to the questions and rubric of the paper itself.

Question A1

a) Explain what is meant by software re-factoring. Discuss the steps involved in refactoring a large software system and discuss why re-factoring little and often may be a more effective strategy for a software development team to adopt.

(10 marks)

b) A company wishes to convert a desktop application interacting with a local database to a web-based database application where the database will be on a remote server and users will access the application via a web browser. Discuss the high level architecture of both systems and indicate what steps you would undertake to achieve the necessary conversion.

(10 marks)

c) During the course of the project, the company decides that it wishes the new web-based application's interface to be made accessible to international users. How can this be achieved and what advantages would the company achieve by having an application that is accessible to international users?

(5 marks)

Answer Pointers
A good answer should cover the following points:

a) Re-factoring is the activity whereby a software system or component's internal code structure is changed to simplify and clarify the code without changing its external behaviour, i.e. its meaning. The overall aim of re-factoring is to make the source code more understandable, reusable and maintainable in the future.
The first step in re-factoring a large software system is to analyse the source code, removing dead code and eliminating cloned code by reusable components, breaking down large software components into smaller ones. After determining measures of coupling and cohesion, these can be used to guide restructuring of the code to lower coupling and increase cohesion. Re-factoring is usually accompanied by re-documentation in a large system where the documentation has been previously neglected. Re-factoring can also involve re-naming of program elements to clarify their purpose.
The re-factoring a large software system can involve considerable effort. The original developers may no longer be available and the rationales they used in developing the code may be lost. A large system is also likely to be complex and hard to understand and have been subject to much change over time by a number of people. If re-factoring is practiced throughout the development process as advocated by the agile approach, then these difficulties can be overcome. The developers know the code; they can deal with poor problematic code as soon as it is identified, by "bad smells".

b) Both the desktop application and the web-based application will have functionality that deals with user interaction, the business/process logic and interaction with the database.
The key difference between these is likely to be that these functions are not clearly separated in the desktop application whereas in the web-based application, these will need to be clearly separated into at least 3 tiers: the presentation layer (visible through the web browser), the web client; the business layer, the application server; and the data access layer, the database server.
So, converting the desktop application will require analysis of the source code to isolate functions associated with each tier and defining the interfaces between the tiers and the messages and data that needs to communicate between them. The conversion of the user interface (UI) will involve identifying web functions that replicate the desktop UI's functionality. The business/processing logic will be at the centre of the new application, so its interfaces to both the web-based UI and remote database server need to be developed. Attention needs to be paid to which, if any, processing functions are located in the web client; that is whether a thin or thick client is required. Here performance issues need to be considered.

c) Internationalization could involve offering the web interface in more than one natural language. It could also involve making use of more universally understood symbols in the interface, thus lowering the reliance on natural language.
One of key advantages of making an application accessible over the web is that it can be used by anyone who has access to web from anywhere in the world. Making the application more accessible to international users will obviously facilitate their usage of the application and allow the company to reach international markets.

Examiner's Guidance Notes:

This question attempted to assess the candidate's knowledge and awareness of software architecture and re-factoring. It was not very popular. Many answers were incorrect.
a) Most candidates explained sufficiently software re-factoring but some discussed reverse engineering and re-engineering instead. Only a few candidates identified the right steps involved in refactoring.
b) Many candidates did not discuss the high level software architecture of both systems (what was required), but e.g. provided a general discussion of 'cloud' computing, reverse engineering, re-engineering etc.
c) Most answers addressed various technical issues related to hardware and software technologies and only a small number of candidates discussed the issues related to the interface for international users. Business benefits were sufficiently discussed.

Question A2

a) Explain why it is necessary to maintain a software system over its lifetime. In your answer, you should outline the various types of maintenance that software systems require and give examples of each type.

(12 marks)

b) Discuss at least three factors which contribute to difficulties during software maintenance and give examples of each.

(9 marks)

c) It has been stated that it is equally important to maintain all the associated documentation of a system as well as its software over time. Discuss the reasons for this statement.

(4 marks)

Answer Pointers
A good answer should cover the following points:

a) Once a software system has been developed and deployed in use, it is enviable that various changes to the system will be required. It is the maintenance process that is concerned with effectively making changes to software during its lifetime. Unless software is maintained, it becomes progressively less useful.
Four main types of software maintenance have been identified in practice. They are as follows:
Corrective maintenance is carried out to correct errors (bugs) found in the software. Owing to incorrect assumptions made by a programmer, an inappropriate looping condition has been used and the program fails to process the last item in a list.
Adaptive maintenance is carried out to convert software from one software technology to another, e.g. from one programming language or operating system to another, in order to adapt the software to run in a new environment. The adaptation of old IBM mainframe FORTRAN program to run on Linux systems would be an example.
Perfective or evolutionary maintenance is carried out to enhance the system's functionality. After delivery, the customer may require additional functions, e.g. payment by PayPal to be added to an e-commerce website.
Preventative maintenance is carried out to improve a system's future maintainability by simplifying and clarifying its source code's structure without changing its meaning. An example here might be re-structuring the system to lower coupling between modules.

b) Three factors which contribute to difficulties during software maintenance are as follows:

1) Staff skills. Maintenance is often undertaken by junior staff who may lack expertise in the programming languages in which the system is written. It may also be difficult to hire staff with expertise in older languages and operating systems. For example, many older COBOL programmers were recalled to service to deal with the Year 2000 problem.
2) Program age and structure. Over time, the structure of software is likely to degrade unless positive efforts have been made to re-factor regularly. System documentation is often lost or out-of-date. Rigorous configuration management and version control may be lacking. For example, it may be that patches have been made to the executable versions of some modules without updating the corresponding source code and recompiling, so there is no longer a correspondence between the source code and executable versions.
3) Contractual/Team responsibility. The development team may have no responsibility for the subsequent maintenance of the software and little incentive to write maintainable software. In additional, the maintenance of a system may be the subject to a separate contract with a different company. For example, a system developed by subcontractors may be delivered with contracted functionality, but its underlying source code is difficult to understand and costly to maintain as time to must be allocated to understanding the system before changing it by the new staff.

c) Unless all the associated documentation of a system is maintained over its lifetime along with changes made to software, then it becomes less useful to both users of the system and maintainers of the system. Users misled by out-of-date documentation cannot make full use of system and may use it incorrectly leading to further problems. If maintainers do not have access to up-to-date system documentation, they will need to invest time and effort into gaining an understanding of the software and recreating the system documentation. This is an error prone situation and could lead to incorrect assumptions and erroneous changes being made to the software.

Examiner's Guidance Notes:

This question attempted to assess the candidate's knowledge and awareness of software maintenance and evolution. It was the most popular question. Many answers were incorrect.
a) Some candidates were unable to identify different types of maintenance. A few candidates provided irrelevant answers.
b) Only a small number of candidates managed to identify the main significant factors contributing to difficulties during software maintenance. Many candidates discussed less significant factors.
c) Most candidates provided adequate answers.

Section B

Question B1.

a) Present an outline of a software improvement framework with which you are familiar (for example CMMI), and discuss how such frameworks handle evaluation and improvement management.

(15 Marks)

The answer to this section should:

- Define the key terms of process, product, measurement and quality;
- Process attributes, and the improvement stages of Measure, Analyse and Change should be highlighted briefly;
- A brief history of the emergence of the chosen framework should be given. For example, work started at the Software Engineering Institute in the 1980s, resulting in Capability Maturity Model of the early 1990s, and the revised framework (CMMI) in 2001;
- Model components covering process areas (Process, Project, Engineering, Support management), Goals, and Practices
- Assessment (CMM) according to five levels: Initial, Repeatable, Defined, Managed, Optimising;
- Assessment in CMMI according to 6-point scale, staged or continuous assessment.

b) Discuss the extent to which the software industry of today is moving towards and achieving the very highest level of excellence, using such frameworks.

(10 Marks)

The answer to this section should:

- take the form of a discussion wherein evidence is presented concerning the adoption of such frameworks by companies;
- Show that the model (CMMI) is more popular outside of Europe, and strongly promoted as a necessity for US defence contractors;
- there are a wide range of models and products, but very little by way of standards;
- the industry is moving slowly towards greater professionalism and accountability. Whilst in many areas in the UK there is no compulsion to adopt such frameworks, increasingly consumer expectation of higher quality of service in products as well as support is an important driver.

Examiner's Guidance Notes:

This question attempted to assess the candidate's knowledge and awareness of software improvement models. It was the least popular (28%) on the exam paper, with only the 3rd highest pass mark of 45%. In many instances, there was some evidence of knowledge of CMM in particular. However, part b) was poorly attempted, and there is evidence that candidates demonstrated a lack of awareness of modern aspirations for such frameworks.

Question B2.

a) For each of the following reuse techniques, provide a clear definition and appropriate examples of usage:

i. Application system reuse
ii. Component-based development
iii. Design patterns

(15 Marks)

The answer to this section should do the following:

i. Define Application system reuse in terms of reusing entire application systems either by configuring a system for an environment (COTS) or by integrating two or more systems to create a new application. In such systems code wrappers play an important part in the integration and successful operation of such systems.
ii. Define Component-based development as an approach to software development that relies on reuse, where most components are viewed as stand-alone service providers. Using well-defined and publicized interfaces, such components are common amongst web technologies and services,
iii. Define Design patterns as a way of reusing abstract knowledge about a problem and its solution. The pattern should be sufficiently abstract to be reused in different settings. One of the most common examples is the Model-View Controller (MVC) consisting of a number of patterns that used for GUI design;

b) Explain why, despite the real and potential benefits of reuse, some companies still experience major difficulties with it.

(10 Marks)

The answer to this section should:

- Briefly summarise the benefits of reuse, namely increased productivity – available and accessible library of components means that both development and validation time should be reduced; and improved quality as components are tried and tested;
- However, some companies are challenged by issues such as: economies of scale such that the larger the pool of reusable components become, the more difficult it is to maintain and provide easy access and navigational capabilities, especially when their entries are continuously updated with multiple varieties of very similar functions, specific to individual users. Secondly, there is a tendency for many of their developers to rewrite components they believe to be inefficient, difficult to use, or missing a particular function.

Examiner's Guidance Notes:

This question attempted to assess the candidate's knowledge and awareness of software reuse concepts and developments. It was the 2nd most popular question (82%) on the exam paper, with the 2nd highest pass mark of 49%. In many instances, candidates demonstrated reasonable knowledge of different forms of reuse. However, there is evidence that knowledge of design patterns was lacking, and for part b) awareness of reuse in practice today was limited.

Question B3.

a) A company is looking to develop a new proprietary software application that can compete amongst current social media platforms. As Chief Analyst, give an outline of the different stages of requirements engineering, and discuss the tools and techniques that you would adopt to derive a complete and consistent requirements specification from the company.

(15 Marks)

The answer to this section should:

- Briefly explain requirements engineering as a process aimed at identifying the customer requirements and the constraints for its provision.
- The different stages include: Inception (line of questions aiming to understand the problem, the people and their expectations, the environment and its constraints), Elicitation (through meetings with stakeholders, customers, and developers), Elaboration (developing a refined technical model e.g. Use cases, of software functions, features, and constraints), Negotiation (conflict resolution or normalizing expectation and reality), Specification, Validation, Requirements management. The aim will be to establish a good foundation for the design and construction of the software.
- Prototyping tools will play a major role in every stage, esp in respect of apps or tools within the social media context, where the business case can change as quickly as the preferences of different community of users in terms of style, functionality, presentation, and usability.

b) Discuss why many systems continue to fall below user expectations, despite the established practice of software requirements engineering.

(10 Marks)

The answer to this section should:

- Highlight the general view that as there are no international standards that govern the way in which requirement are gathered and managed most people will have their own different way of working and achieving the objectives of requirement gathering. In particular, the speed of change may impact potential users, and developers in different ways and to different degrees. Thus, both parties are always likely to be out of sync with one another. However,
- Whilst, all can agree that the accurate capture and representation of requirements leads to successful projects and satisfied customers, only consistent and committed application of the principles of requirements engineering has the greater likelihood of achieving this. Requirements Engineering offers a professional, structured and logical approach to capturing and representing requirements, and is well supported by tools that, if followed consistently, will more often than not results in successful requirements management.

Examiner's Guidance Notes:

This question attempted to assess the candidate's knowledge and awareness of modern requirements engineering concepts and practice. This was the 3rd most popular question (64%) on the exam paper, with the lowest pass mark of 26%. In many instances, the evidence shows that candidates do not appear to recognise the topic as an engineering discipline and produced responses more akin to a traditional Certificate qualification in systems analysis.