BCS HIGHER EDUCATION QUALIFICATIONS

BCS Level 5 Diploma in IT

Software Engineering 1

June 2015

EXAMINERS' REPORT

General Comments

This is a technical paper about Software Engineering. Questions seek to test candidates' knowledge and their ability to apply that knowledge. A poor answer is one that simply describes recalled information. A good answer will 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 risk management and ethics, and reflect on how development processes hinder or help the higher aims. Overall, the candidates' responses to the Software Engineering examination questions showed a relatively limited understanding of the subject. Most marks for individual questions are in the low range.

Below, a description of the two parts A and B, is completed with model answers. A more detailed description of how the cohorts managed to give an answer is also provided.

Section A

A1. The IT manager of a sport and fitness company asks you about Internet Security. The company has an e-commerce website hosted externally by an Internet Services Provider whose staff read orders and other customer inputs from the website prior to storing them in an electronic folder for the company to download. The company also uses a Virtual Private Network (VPN) link with the engineering consultancy that supplies designs for new bikes, and also uses email with other businesses in its supply chain.

Consider the above scenario and:
a) Identify and analyse THREE risks that cover the manufacturer’s use of the Internet.
(9 marks)
b) Rank the identified risks in priority order.
(4 marks)
c) Choose some mitigation action that is appropriate for each risk, giving your reasons.
(12 marks)

Answer Pointers

a) Three areas of risk in this scenario are:

.**Sales orders routed via the ISP** – can they be falsified or otherwise compromised?
.**The VPN – an internet ‘tunnel’** – can it create vulnerabilities to security breaches or hacker-access to corporate documents?
.**Email usage** – can staff use email freely? Can staff send corporate documents home i.e. outside the company?

b) The scenario depicts a **business-critical application,** here's a breakdown of the possible risks, ranked by their priorities:
.risks with **highest priority** order are related to the **security** of the transactions, and to the privacy of users and staff of the IT company
.**Mid-priority** risks relate to the **availability** and **reliability** of the online application
.**Low-priority** risks involve the **repairability** or **maintainability** of the service

c) Mitigation strategies:
.For the ISP routing of sales orders – review who at ISP has access to the company’s orders. Try to arrange automatic forwarding or automatic stire-and-forward without human intervention.
.For the VPN – implement a security policy including physical and IT controls, audits and reviews.
.For the email usage – monitor emails that leave the company so that any suspicious mailing can be retrieved and checked.

Examiner's Guidance Notes

This question requested the candidates to reflect on a scenario related to the security of an internet provider. The question was very popular and answered by a large number of students: in general the first question was well answered, while the answers to the second part showed that the candidates did not give a good explanation of why a certain order was chosen to rank the risks and related security.

A2.

A suggested process models for large software development is the Incremental and Iterative development.

a) Provide one definition for Incremental development and one for Iterative development, describing why they are different but complementary.

(9 marks)

b) Define TWO advantages of using increments and iterations in software development, as opposed to using a traditional mode of development.

(6 marks)

c) A software specification contains 3 requirements (R1, R2 and R3), and the creation of 4 components (C1, C2, C3 and C4). The requirements and components are linked as follows:

.R1 requires C1, C2
.R2 requires C1, C3
.R3 requires C3, C4

Using an incremental approach, and THREE iterations, describe how the components will be built, integrated and released in a product in each iteration.

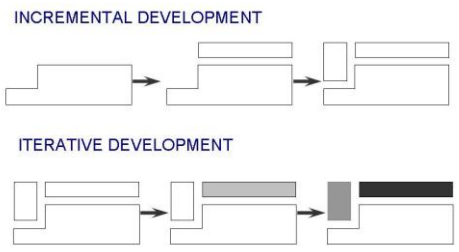
(10 marks)

Answer Pointers

---

a) Incremental development: starts with small functional subsystem and adds functionality with each new release

Iterative development: starts with full system, then changes functionality of each subsystem with each new release



Incremental produces new functionalities, while iterative refines existing functionalities

b) Some of the advantages are:

- .Shorter cycle time
- .System delivered in pieces
- .enables customers to have some functionality while the rest is being developed
- .Allows two systems functioning in parallel
  - .the production system (release n): currently being used
  - .the development system (release n+1): the next version

c)
SETUP:
Requirements: R1, R2, R3
Architecture: C1, C2, C3, C4
Planning: 3 iterations

Iteration1
R1, requires C1, C2
Develop and integrate C1, C2,
Deliver R1

Iteration2
R2, requires C1, C3
Develop C3, integrate C1, C2, C3
Deliver R1 + R2

Iteration3
R3, requires C3, C4
Develop C4, integrate C1, C2, C3, C4
Deliver R1 + R2 + R3

Examiner's Guidance Notes

This question required candidates to reflect on the practice of incremental and iterative practices. In general the first two parts of this question were well answered by all the candidates attempting it. The third part of the question was a practical example of such iterative and incremental practice, and was not well answered. In particular, there was some confusion in assigning the right requirements to the iterations.

A3.

a) In the context of software reuse, give an example of a reusable component.

(4 marks)

b) Discuss the practice of software reuse within the software life cycle and describe at least THREE of its benefits.

(9 marks)

c) Identify three possible risks that can occur when a system is built using reusable components and explain how these risks could be reduced.

(6 marks)

d) In the context of software reuse, explain why access to the source code may be desirable and in some cases necessary for the validation of the reusability of a component.
(6 marks)

Answer Pointers

a)
Software reuse involves the reuse of existing software systems or components of a system in the development of a new system. It enables a new software system to be built using software components that have already be tried and tested in other systems.

Three benefits are: *increased dependability* (tried and tested already), *increased productivity* (and time to market) as not developing the system from scratch, and *effective use of specialised knowledge in a domain*, e.g. mathematicians can be employed to develop a standard maths library which can be reused again and again and by non-experts.

b)

COTS = Component Off The Shelf or commercial-off-the-shelf; this refers to readily available software that is available in component form ready to reuse in the development of a new system. COTS may be commercial software, freeware or open source software. There are many “shopping cart” COTS available for reuse in e-commerce systems; they can be found by searching the web.

c)

1. No control over COTS future development – especially a risk if the source code of the component is not available – can be alleviated by obtaining the source code, only using Open Source components, also buying the company producing the COTS or making sure that a contract is made which ensure this is not the case.

2. Lack of support from COTS vendors – COTS vendors may be small companies and they may not be able to deliver the level of support needed by reusers, even if they have committed to supplying support, their business may change or they may even go out of business. Put in place an agreement if possible to cover these circumstances, e.g. the source is held by a third party and will be supplied if the company goes out of business. Also consider second sourcing important COTS.

3. A product using a number of COTS may suffer from COTS interoperability, i.e. the different COTS may not work together properly. This risk can be lessened by obtaining software documentation explaining the assumptions underlying the design and operation of the COTS software.

d)

The candidate should stress the importance of the black box accessibility to source code as a way to understand the inner working of the component to be reused. The validation of the component reusability will ensure that its functionality is the intended behavior of the reused component; such validation will involve the actual testing with other components and as stand-alone.

Examiner's Guidance Notes

This question required the candidates to produce various arguments related to the reuse and reusability of software components. The question was attempted by nearly all the students sitting the exam, and in general good answers were produced. The only part of the question that generated some confusion was the last one, which required a reflection on why the source code is needed in the integration of a module into an existing code base.

SECTION B

B1.

A client wishes to set up an internet shopping basket application. Requirements include the ability of a customer to register with a name, address, payment details and be assigned a unique customer identification. A customer should be allowed to order any item from the site. When an order is completed by a customer the items are either dispatched from stock or placed on back order from a supplier, in either case the customer is notified. Items dispatched from stock are debited to the customer, items on back order are debited when they are eventually dispatched.

a) Draw suitable UML class diagrams clearly showing a static view of the individual elements and their relationships. (15 marks)

b) Identify the dynamic organisation of objects and messages within the system and draw a UML sequence diagram. The diagram should clearly show the sequence of message flow. (10 marks)

ANSWER POINTERS

a) Candidates should show an ability to define basic UML terms in providing a static view of the requirements. Class diagrams are the most basic part of UML and candidates should provide at least an appreciation of putting simple requirements into the language of classes, with meaningful class names an identification of the elements of the requirements, such as customer name, ID, payment methods, order number, order ID and so on. A competent answer will include an appropriate use of the UML symbols and knowledge of linking the elements between classes. A good answer will indicate knowledge of super class, and sub classing and indicate aspects of inheritance between the identified classes.

b) This part of the question extends the most basic understanding expected in part a. The difference between static and dynamic representation in UML is shown in an interaction diagram. In particular the sequence diagram should show the UML symbols used.The identification of objects that act upon each other and the idea of a sequence in messages. A good answer will clearly show the correct ordering of a sequence and identify key points of interaction in order dispatch and back order decision points together with confirmation of customer debiting on completion.

EXAMINER'S GUIDANCE NOTES

This question achieved the worst performance in part B of the paper. Surprisingly some candidates achieved better marks on part b of this question. Part b was considered a better test of knowledge depth and it's not clear why some candidates had knowledge of process in UML and little (in many cases no knowledge) of the basics. It is clear from the performance in part a that candidates have a poor understanding of UML and its place in the software life cycle. An inability to use basic UML components is shown in the poor performance in part a. To be successful candidates need better preparation in understanding UML basics and some experience of interpretation from an application into UML terminology. Concentrating on the fundamentals such as static representations would have allowed many candidates to achieve a pass from part a alone.

B2.

a) Briefly outline the software testing methods used in:

1) 'White-box' testing.
2) 'Black-box' testing.

(8 marks)

b) Outline the difference between software verification and software validation

(5 marks)

c) Describe and give examples of the processes involved within integration testing

(12 marks)

ANSWER POINTERS

a) This question is testing the candidate’s knowledge of basic software testing techniques. The broad definition of white box as 'looking' at the software procedures, methods and structures (program logic) and black box as the testing of functional requirements should be given.

b) The candidate’s ability to distinguish between 'building the right product' (validation) and 'building the product right' (verification) is sought. Good answers will indicate the difference between meeting expectation (validation) and meeting the specification (verification)

c) As a verification testing strategy, integration testing can be seen as the phase where the previously tested components are tested as a whole This question is asking candidates to show their understanding of the strategies within integration testing such as top-down and bottom up also the regression test as a means of evaluating the effects of adding additional components to the software product.
A good answer will indicate an understanding of issues surrounding the choice between top down and bottom up strategies.

EXAMINER'S GUIDANCE NOTES

This question attracted an attempt by over 97% of candidates. Some candidates gave good answers to all sections of the question with two or three giving excellent attempts. Overall parts a and b were attempted with reasonably good marks (about one third of candidates) Most candidates found part c difficult and around two thirds found the top down and bottom up distinction difficult and had little appreciation of regression testing. These are areas that could have made a significant difference to the success rates as 5-6 marks from this part would have pulled a large proportion of failures to a pass level.

B3.

a) One of the many ways in which software can be reused is in the deployment of a complete software application as a 'commercial off the shelf package' (COTS) . Outline design considerations and possible problems in integrating a multiple COTS system. (10 marks)

b) Component based software engineering(CBSE) is a way of constructing a software system by reuse of software components. Describe the factors to be considered with development activities associated with:

1) Component qualification

2) Component adaptation

(8 marks)

c) Outline the key architectural requirements needed to satisfy component composition in a CBSE development context.

(7 marks)

ANSWER POINTERS

a) Candidates should show an appreciation of problems and issues in the reuse of application software such as a database or ERP systems. Answers should show an understanding of design considerations. These include appropriateness to the specific target use, redundancy in capabilities (too many options for intended use),data formats (target might require .pdf, but COT has .doc) and the need to evaluate the product to understand pre designed functionality. Possible problems include the lack of control, interoperability and support from suppliers. A good answer will show knowledge of particular issues surrounding the evolution of the application on changing platform versions.

b) This question tests the candidate’s understanding of CBSE .Component qualification is the assessment of a component in terms of fitness - usability and reliability. There are many factors to consider but three or four from API quality, exception handling, security ,run-time needs, interfaces, component messaging.

Component adaptation is concerned with 'wrapping' for integration. Answers should show an appreciation of the need to wrap a component to mitigate issues seen in qualification.

c) A candidate should show a knowledge of the key architectural requirements - Adler's architectural ingredients; exchange, automation, storage and object model. Good answers might include standards like COM or JAVABEANS.

EXAMINER'S GUIDANCE NOTES

This was the least popular question in part B. Generally part a was not answered well by many of the candidates, with only a couple of exceptions. Part b attracted some reasoned understanding of one or other of qualification or adaptation but for most candidates the question showed they had a lack of any real appreciation of the topic. Part c demonstrated that all of the candidates had difficulty with the idea of architectural requirements. Most were completely lost on this part. Performance on this question indicates both a lack of experience in package software development processes and toolkit resources in general. This is probably reflected in the few candidates opting to attempt this question.