BCS THE CHARTERED INSTITUTE FOR IT
BCS HIGHER EDUCATION QUALIFICATIONS
BCS Level 5 Diploma in IT
MARCH 2014

EXAMINERS' REPORT
Software Engineering 1

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.

Software Engineering 1 (Part A)

A1. *Software Engineering; the nature of software, theoretical models, the software crisis, the cost of maintenance, the cost of quality.*

OldStyle Computing is a company created in the 1970s, producing software and hardware for single processors and mainframes for a number of established clients. After a major downsizing of its business, the company has decided to shift its focus to more established software engineering techniques, and would require some understanding of how software engineering has changed in the last twenty years.

1) Explain how the concept of "abstraction" can produce added value to the company. **(5 marks)**

2) Descibe how software architecture can be leveraged by the company to advance the state of the art of its development approach. **(5 marks)**

3) Outline how metrics could be utilised the company to monitor its development business. **(5 marks)**

4) Apart from the use of abstraction, architecture or metrics, describe TWO other software engineering approaches that could be employed by the company in its new business. **(10 marks)**

Answer Pointers

This question is asking the candidates to focus on how software engineering has changed in the last 40 years, and how the paradigm of mono-user, mono-client, monolithic systems evolved in multi-user, prototype- and metrics-driven system development, and what concepts, techniques and methodologies were developed to sustain an increasingly small time to market. In particular, it asks candidates to comment on the Wasserman's discipline of software engineering, and the major items of innovation that were identified in the context of Software Engineering: Abstractions, Analysis and design methods and notations, User interface prototyping, Software architecture, Software process, Reuse, Measurement, Tools and integrated environments.

1) This section asks the candidates to comment on the concepts of abstraction, and how they provide a description of the problem, or the system, at some level of generalisation, which was also coupled with the concept of encapsulation, to provide an effective way to hide the details of the implementation and functioning of individual modules.

2) This section asks the candidates to join the previous section 1) with the concept of connectors and components, and the different approaches in which components can be encapsulated and abstracted: Modular, Data-oriented, Event-driven, Outside-in-design and Object-oriented decompositions

3) This third section asks the candidates to provide some insights in how metrics and measurements changed the approach in software engineering and development, by providing a means to describe quality goals in a quantitative way, using mathematics and statistics to design and evaluate process models, and a common approach and benchmarks to interpret those results

4) For this question, the candidates could describe the advances produced by the reuse mechanism, the introduction of more advanced software processes, the use of notations (UML above all) and the introduction of upper and lower CASE tools.

Examiner's Guidance Notes

This question was the least attempted overall, although it pitches a general question about the main characteristics of Software Engineering. The marks overall were quite poor, and it was clear that the open ended question, where students are asked to provide alternatives, points of view, or evaluation of techniques are still problematic for most students.

A2. Software Engineering key practices; the multidisciplinary nature of software design, team work, productivity, testing, product maintenance, and software product life cycle.

In the context of software product life cycle, different process models can be used by software organizations to develop end-user products.

1) Identify and describe THREE differences between the traditional Waterfall model, and the V model to software development. **(2x3 marks)**

2) Provide an example of a project that could benefit from the implementation of a Waterfall model, and identify the characteristics of its main development phases (requirements, design, implementation and testing). **(3+8 marks)**

3) Describe and outline the FOUR phases of a spiral model, and outline how the risk factor affects this model. **(8 marks)**

Answer Pointers

1)

A waterfall process differs substantially from a V model based on the type of interaction that end-users have with the system, and through the role of testing and prototyping.

In a waterfall process, requirements can be frozen, while the V model assumes that the requirements can be reworked upon

In a waterfall model, software development is considered as a manufacturing process, with no iterations in the activities.

The V model is a variation of waterfall, with a strong emphasis on testing, unit testing for low-level verification, integration testing for high-level verification, and acceptance testing for the validation of the requirements.

In either verification or validation, the left side of the V can be re-executed before testing on the right side is re-enacted

2) This question asks the candidate to identify a project type that could benefit a waterfall model: while the model does have critics, it still remains useful for certain types of projects and can, when properly implemented, produce significant cost and time savings. Whether you should use a Waterfall models depends on

how clear the customer's needs are, and
how much volatility one expects in those needs as the project progresses.

These are the phases of a Waterfall implementation:

- **Requirement Gathering and analysis**: requirements are thoroughly and accurately defined. All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc. The aim of the requirements phase is to produce a Functional Specification (detailing what the application will do). When using this methodology it is vital that all requirements are captured during the Requirements/design phase as it can be very expensive to re-visit requirements once implementation (coding) has begun. If the requirements phase is done badly (and this is often the case when the business confuses shoddy requirements with faster progress) the waterfall method delivers failure as the end result will only ever be as good as the specifications.

- **System Design**: The aim of the design phase is to produce a User Interface Specification (detailing how it will do it). The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. Only when the Functional Specification and the Interface Specification are signed off can the actual process of building the application commence (the implementation phase).

- **Implementation**: With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

- **Integration and Testing**: All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

3) The spiral model is depicted as in a spiral with circuits: each circuit represents an iteration around four major activities

a) **Plan**: in this activity a Post-mortem analysis of the previous loop is executed, and the next loop is planned
b) **Determine goals, alternatives and constraints**: in this activity the main goals of loop are determined, with alternatives and constraints
c) **Evaluate alternatives and risks**: in this activity, the risks of loop are identified, and analysed; also risk mitigation actions are defined
d) **Develop and test**: in this activity, the goals of loop are implemented

Examiner's Guidance Notes

This question was well attempted, and in general good marks were achieved by the students choosing it. It is a question on basic software processes, and how they could be employed by companies while performing their software development.

Around half of the students attempting the question could identify the main differences between processes, but the rest of the students could not idfentify the main parts of a spiral model, often misplacing it for a waterfall model. Also, there is some confusion in the identification of the V model, its characteristics and attributes. This is more problematic, since the question did not require an evaluation of the processes themselves, rather an explanation of the phases, in a descriptive way.

A3. Project Management; project estimating and project planning; management and maintenance of software products in the consumer marketplace, total cost of system ownership, software life-cycle cost modelling, project development cost modelling, project and product risk management.

The director of a company is trying to estimate the costs of the activities involved in various parts of a project:

1) Describe TWO advantages of the WBS technique, and TWO characteristics of the PBS visualisation. **(4 marks)**

2) Describe TWO attributes of a PERT chart, and explain how a PERT chart complements the WBS and PBS views. **(2+2 marks)**

3) Define and exemplify the following terms:

a) critical path

b) slack time

c) estimation error **(9 marks)**

4) Identify TWO reasons why the estimated length of an activity could radically differ from its actual duration. **(8 marks)**

Answer Pointers

1)

WBS is Work Breakdown Structure, and it depicts the Hierarchical decomposition of activities in subactivities. No temporal relationships are displayed, but a full coverage of the activities is possible in a limited space.

PBs is the Product Breakdown Structure, and it represents the Hierarchical decomposition of a product:

- Product
  - Requirement document
  - Design document
  - Module 1
    - Low level design
    - Source code

    - Module 2
        - Low level design
        - Source code
    - Test document

2)

Work breakdown structure depicts the project as a set of discrete pieces of work, while the PERT Activity graphs depict the dependencies among activities: the nodes represent project milestones, while the lines represent the activities involved. The PERT diagrams complete the PBS and the WBS views by adding the time constraints and the dependencies between activities, in turn allowing to evaluate the critical path.

- Time and Relationships
    - Gantt charts emphasize the time it takes to complete tasks
    - PERT charts emphasize relationships between tasks.
- Linear Vs. Interconnecting
    - Gantt chart is linear
    - PERT chart shows parallel or interconnecting networks of tasks
- Straightforward Vs. Complex
    - PERT charts are designed for only parts of projects
    - Gantt charts for straightforward projects without mid-stream changes.
- Project managers use both types
    - Gantt charts do not effectively illustrate the dependencies
    - PERT charts can be complicated and confusing.

3)

a) critical path is the Minimum amount of time it will take to complete a project. It reveals those activities that are most critical to completing the project on time

b) Slack time: the difference between the available time and the real time for that activity.

c) Estimation error is the difference between what has been estimated by a software estimation tool and the actual observed values of the activity.

4)

Key causes of such discrepancies could be

- Frequent request for change by users
- Overlooked tasks
- User's lack of understanding of the requirements
- Insufficient analysis when developing estimate
- Lack of coordination of system development, technical services, operations, data administration, and other functions during development
- Lack of an adequate method or guidelines for estimating

Examiner's Guidance Notes

This question is about project management and its importance in software projects. In general, the question did not require a specific software engineering background, but part 4) of the question required the students to understand whether or not a poor estimation of effort and resources could have an impact on the overall length of a project, and what could be the causes of such poor estimation.

Students performed better in the definition part: WBS and PBS should be well understood and applied to a software engineering context, as the critical path and the slack time for an activity. Again, when asked to reflect on the discrepancies that could be observed between the estimated and the observed values of an activity, students appear to have more problem in reflecting on these aspects.

Software Engineering 1 (Part B)

B1. *OO notation for describing software components and architecture*

1) Briefly describe the process of Object Oriented Development (OOD) within:

   a) Analysis

   b) Design

   c) Implementation **(6 marks)**

2) Throughout the various phases of the project life cycle, the UML can be used to graphically model processes. From a UML context explain the content and purpose of the following diagrams and state where in the software life cycle they are likely to be used

   a) Class Diagram

   b) Object Diagram

   c) Component diagram **(9 marks)**

3) A departmental library offers staff access to journals for reference. A member of staff can make a request for a journal. If it is available the request is satisfied, if it is not available the request is placed on a request list to await further processing.

   a) Provide a UML use case diagram to show this system.

   b) Provide a UML sequence diagram showing the message sequence for the request list. **(10 marks)**

Answer Pointers

1) Candidates should describe analysis being the point at which objects are identified and well described, also the importance of naming and the consistency in classification of objects. The importance of capturing the application domain fully should be mentioned. For design a good answer will describe the process of ensuring that the objects fully map to requirements, it is the point at which previous phase of identity is fulfilled in satisfying requirements. Implementation is the point at which the component and deployment objects are developed, this is the realisation of software in code and its running over a platform. Good answers will express the phases in standard OO terminology

2) Answers to this part should state the purpose of the UML concept relating to each of the entities The class diagram containing classes,interfaces,etc.' most commonly used in system construction and representing static aspects of the system. Object diagrams show instantiation of classes and share attributes of the class diagram, but represent a static view capable of being used in prototypes. Component diagrams consist of classes, interfaces and collaborations. Usage is at implementation.

Good answers will seek to explain the object terminology in terms of the place in the life cycle and the content of each entity reflecting this usage.

3) Candidates should be able to produce the simple use case diagram and should have an actor, use cases for the states and relationships(extends) etc.' The good answer will keep in perspective the role of use case in requirements so naming and descriptions have to be understandable , it should be clear and consistent in naming . The sequence diagram extends the technical understanding of UML to model control flow between objects using interaction. The sequence diagram models the order of message flow between objects. The sequence diagram will show the basic objects; staff actor,request,request satisfied, request fulfilled and should show the sequence of messages for request list object

Examiner's Guidance Notes

In section B this question attracted fewest candidates. Only around one third of candidates managed to correctly answer 60% or more of the material. Most candidates failed to provide a full answer to 3 a and 3 b In only a very few instances did candidates provide the correct diagram along with a correct interpretation. The answers showed that the majority of candidates do not have a clear understanding of either UML or Object terminology.

Answers to 2 showed a general appreciation of the three areas with c being less likely to be correct.

Answers to 2 were largely disappointing with a significant number failing to distinguish the context of the question being OO and answered as the standard software life cycle definitions.

About 40% of the candidates seemed to be completely lost over every part of this question and completely gave up subsequently attaining very low marks.

B2. *Validation, verification, and testing*

1) Give a definition, with examples of the Validation & Verification process within the software life cycle of a large project. **(10 marks)**
2) Describe the software program inspection process and state the advantages it is thought to have over testing. **(10 marks)**
3) Give one example of an automated static analysis tool. **(5 marks)**

Answer Pointers

1) A definition will make the distinction between validation as focused on the product meeting expectation as opposed to meeting the specification document and verification being more focused on meeting the written specification. Good answers would see it as 'fit for purpose' evaluation. A large project gives scope to address the fundamentals of the process throughout all of the possible management stages of the complete lifecycle

2)As part of a static view of the V&V process this gives candidates an opportunity to show knowledge of that process. A good answer will acknowledge it as part of the defect detection process and integral to quality management as a verification tool. There are several advantages documented to show that this technique has a more effective outcome in defect observation than an alternative testing emphasis. Candidates should note the advantages of using static methods don't require knowledge of program interactions, so solving a single error statically, doesn't end the search but the other (test) methods fix the bug whilst potentially hiding other linked errors(since there is no further exhaustive methodical process. Another possible advantage could be the ability to do piecewise defect detection without the need to run a program for output.

3) This could be any stand-alone static or integrated verification suite containing static analysis such SVT for static UML model verification, LINT for Unix based C code or mission critical static analysis tools such as LDRA. The candidate should make it obvious that although the tool is automated the process is statically driven

Examiner's Guidance Notes

Although in part B most candidates answered this question most failed it. Around half of the candidates failed to correctly answer even one third of the question. Most candidates answered 1 as though it was an opportunity to describe the waterfall paradigm. Very few were able to keep to the question focus on validation and verification. Not many used the notion of a large project to explore the roles. 2 produced a significant number of vague answers, with a few exceptions, where a minority fully understood the distinctions being sought. Only very few candidates managed to correctly discuss the advantages over testing. 3 Was surprisingly very poorly answered by the majority. The standard half line answer tended to be "C.O.T.S". Candidates clearly have little idea of automated tools. As in B1 some 40% of candidates were out of their depth and largely gave up, giving in many cases trivial answers and in some cases answers that didn't address the question. A very large number of candidates seemed to want to relate all of their answers

to aspects of database design and implementation, searching for tenuous links to an area they seemed much more comfortable with.

B3. *Software Engineering Tools and Environments*

1) Identify and describe the role that upper and lower CASE tools have in various phases of the software life cycle. (**12 marks)**

2) Explain what software reuse is and identify a range of benefits and problems associated with using this approach to development.

(8 marks)

3) Outline any one technique that supports software reuse which might be used in a CASE tool repository. **(5 marks)**

Answer Pointers

1) Candidates should show an understanding of lower tools being used towards the downstream implementation and upper tools being concerned with initial upstream activities such as requirements. The various phases should be identified and appropriate CASE tools e.g. PERT at project planning and compilers at code building. Good answers will be able to make clear the role of a particular tool across functional and activity classifications (although a definition of lower or upper strictly refers to activity based and a given tool addresses a specific task) it's expected that candidates will be familiar with named products. A good answer will indicate knowledge of tools that encompass the whole life cycle activity and might be regarded as 'workbenches'

2) A good answer will explain the approach to maximising efficiency in development resources

by reusing different software products from applications as a whole through to Objects and methods. A good answer will appreciate the benefits of reuse for dependability of pre-tested entities and incorporating expertise in the domain application through to reduction of development times. The cons of reuse will cover such areas as the need to maintain the repository the lack of domain knowledge, use and management of the library or repository, team members being familiar with the API's etc.'

3) Answers to this question will range from Design Patterns to Libraries. A good answer should explain how the technique might be managed and issues of updating, versioning etc.'

Examiner's Guidance Notes

This was the second most popular question in section B, with the highest pass rate. Around one half of the successful candidates answered two thirds or more of the question. Sadly, over one third of candidates correctly answered less than a third of the question. 1 tended to have a majority correctly identifying upper/lower, becoming slightly more vague as we cover the different stages. 2 Those who successfully answered tended to have a good grasp of reuse with pros and cons. 3 Most struggled to find a reasonable answer; most thought of libraries but couldn't explain further. 30% of candidates question were catastrophically poor and clearly had little idea of the question. In these cases it is difficult to offer guidance, since there is little evidence of direction.