Examiners' commentary 2018–2019

CO2226 Software engineering, algorithm design and analysis – Zone B

General remarks

The examination was set generally as a mixture of questions that test candidates' basic understanding of the material, and questions that require candidates to apply their knowledge and demonstrate deeper understanding. Candidates are reminded to read each question carefully and address all aspects of the question.

The paper is perhaps unusual in that it covers two separate (though loosely related areas): software engineering as well as algorithms.

This examination is therefore split into two parts. Part A assesses the software engineering and analysis material, Part B the algorithms material. Each part has three questions, of which two are to be answered (therefore four questions answered in total, two from A plus two from B).

Candidates should, however, be sure to number their answers clearly and correctly. There were some cases where it was unclear. Though examiners will do their best, if we cannot read or make sense of an answer, this makes it very hard to award marks.

Furthermore, if candidates wish an attempt at a question not to be marked, they should indicate this clearly (by crossing it out). Examiners are directed to mark only the first two questions answered in each part. There were a few instances of this.

All questions require a mixture of bookwork (e.g. explaining terms, giving definitions) and application (e.g. calculations, building UML models) with the majority being application oriented. So candidates should work on practising likely calculation/application tasks.

For Part A (software engineering), the key application skill is to produce a (usually UML) diagram from a scenario, the same scenario possibly being shared between questions. Candidates should ensure that they can do this well and quickly. We hope that the discussion below will help candidates to be mindful on where marks can be easily missed.

Some brief explanatory comments on answers involving (pseudo)code in Part B would be helpful for the examiners, as some answers were unclear.

Candidates would do well to show working on questions that involve calculation or the application of a process. This applies doubly in Part B where a mistake midway can lead to a wrong answer in many questions. Where this has happened often will be highlighted later.

If the candidate simply writes down what they think the answer is and it is wrong, then they will gain no marks. If we see working we can see what the candidate understands, and they may gain partial marks.

Candidates should also note that mark schemes often explicitly allocate marks to showing working and giving reasons for their answers (and questions may ask for this).

Comments on specific questions

Question 1

This involved a question where candidates were asked about the audience and scope of UML state diagrams. This was followed by the usual 'generate a software engineering model from a scenario' task in the domain of an online learning website (in this case, UML class diagrams).

a. Part (a) as noted above was bookwork. Candidates were generally able to explain what a state diagram was (although this was not really asked for). But when the question of audience and scope came up, the answers often became somewhat vague.

   A few short paragraphs would suffice for a good answer here. A few candidates either gave very terse answers or wrote essays, both to their detriment.

b. In part (b), the quality of class diagrams was variable. In general, candidates were able to capture the majority of the scenario well. There were a number of areas where candidates lost marks.

   - Some candidates produced diagrams with few classes (5–6 is definitely too few), and/or omitted key classes.
   - Sensible proposals were generally given for class attributes and methods, but not always.
   - More commonly, candidates failed to fully illustrate associations, composition and aggregation relationships between the objects, despite these being explicitly requested in the question.

Hint: the scenario is designed to give candidates a chance to show they can use all of the above.

Question 2

This question comprised a bookwork part about the concept of a composite state in a state diagram and how and why it can be useful. This was followed by the development of an activity diagram based on the same scenario as the previous question.

a. Part (a) was sometimes answered adequately, but often answers were vague. Examples were given in the stronger answers and sometimes lifted the quality of the written explanation. Again, a few paragraphs would suffice for a good answer.

b. Part (b) was generally answered well. Overall candidates were able to put most or all of the scenario on the diagram. Areas where candidates lost marks included (in descending order of seriousness):

   - Some candidates produced diagrams with missing swim lanes.
   - More commonly, candidates failed to fully capture concurrency and flow of control between actors.
   - In general, candidates should pay particular attention to identifying activities, decision points and fork/join nodes.

Hint: the scenario is designed to give candidates a chance to show they can use a broad range of model features.

Question 3

This question started with a bookwork question on the nature and application of extreme programming, followed by a scenario task (use case diagram, extending the scenario from 1(b).

a. Part (a) was not answered well. There was confusion about what the concept meant and as a result the applications given were often not

relevant. Given the popularity of "agile" methods at present, candidates would do well to understand this broad class of approaches. Again, a few paragraphs would suffice here.

b. Part (b) was generally answered well. Overall candidates were able to put most or all of the use cases on the diagram. In general candidates were able to capture much of the scenario. Common candidate errors included:

   - A few candidates gave use cases (as text) rather than use case diagrams. It is hard for examiners to give credit if the question is not answered as stated, so take time to read the question carefully.
   - Some candidates produced diagrams with missing actors (hint: sometimes time can be modelled as an actor).
   - More commonly, candidates failed to fully illustrate include, extend, and generalisation relationships between the use cases; despite these being explicitly requested in the question.

Hint: the scenario is designed to give candidates a chance to show they can use a broad range of model features.

Question 4

This question aimed to test candidates' understanding and application of data types and graphical representations. As usual, the question was a combination of tasks, plus some bookwork.

a. Part (a) was, surprisingly, generally answered vaguely, both in terms of distinguishing the two concepts (note: these concepts are important to understand as they are fundamental), and in their application.

b. Part (b) was generally answered well.

c. Part (c) in most cases was answered very well. Candidates should note that once an intermediate level cell in a Quadtree is all black/white there is no need to expand further.

d. Part (d) was typically answered weakly, maybe due to poor understanding of the underlying concept.

e. Part (e) was generally answered adequately in terms of the name of the representation, but less clearly as to how it can be extended.

f. Part (f) was generally answered vaguely despite being covered in the subject guide.

Question 5

This question aimed to test candidates' understanding of optimisation as well as computability/intractability concepts applied to an optimisation problem (the well-known bin packing problem). The question was a combination of tasks, plus some bookwork.

a. Part (a) was generally answered in a confused manner, indicating that candidates did not fully understand the approach and its trade-offs.

b. Part (b) was generally answered well.

c. Same as for part (b) above.

d. Part (d) in some cases was answered very poorly. Candidates sometimes did not demonstrate that they understood the difference between decision/optimisation and NP-complete/hard problems.

e. Part (e) was generally answered well.

f. Same as for part (e) above.

Question 6

This question aimed to test candidates' understanding and application of tree and heap data structures. As usual, the question was a combination of tasks, plus some bookwork.

a. Part (a) was generally answered well, though a few answers were confused despite the topic being covered in the subject guide.

b. Part (b) was mostly answered well.

c. Part (c) in most cases was answered very well. Candidates showed good understanding of these concepts.

d. Part (d) – almost all candidates did well at this part, indicating a good understanding of how binary search works. Pseudocode was not always given, however. Strong answers also gave a walkthrough.

e. Part (e) was generally answered well. Again, strong answers gave a clear explanation of the mapping between tree and array.

f. Same as for part (e) above.