

## Object Oriented Systems Development

There were 13 students registered, 2 were absent and 3 failed. The overall standard of answers was A unique case was of a student who secured almost full marks in the first two questions, and almost zero marks in the last two questions. One student was exceptionally brilliant in all answers and secured 91 marks. Overall the students were able to demonstrate the necessary understanding of the relevant competencies in the exam. More specifically, the competences tested were spread out into four questions, with sub-questions within. The mapping of the exam questions to the competencies is as follows:

- Question 1. ICAA4058A - Apply skills in object-oriented design, derive high level design from specification; Refine and document the design.
- Question 2. ICAA5049A - Develop high-level object-oriented class specifications including roles and responsibilities of classes.
- Question 3. ICAA5046A - Review and confirm requirements information and existing models, + ICAA5049A – state models and Iterations.
- Question 4. ICAA5050A – Develop detailed component specifications, analyse components. + ICAA5046A -resolve conflicts and build test model.

This exam coverage included quizzing the students on use cases and use case diagrams, class diagrams and the dynamic modelling aspect through sequence diagrams and state machine diagrams. Furthermore, students at this level were also expected to be familiar with the implementation models as well as the impact of OO on quality and testing. This was also assessed during the exam. Students were hardly able to draw these diagrams, and especially the class diagrams to an acceptable standard

The following comments are made about each question.

### Question 1

This question started by testing the students in terms of their knowledge and understanding of the use case diagrams and how the diagrams are different to the use cases themselves. These use case diagrams are of high importance to software requirements and design. This question also ensured that students demonstrate sufficiently good understanding of modern software development activities and, in particular, the object-oriented software development activities. Thus it was essential for students to know and understand UML notations and diagrams in design.

Subsequently, in this question, the students were also tested in terms of their ability to identify use cases and put them together in use case diagrams. The required diagram was a standard use case diagram and it must have had use cases, actors (shown outside the system boundary) as primarily

users of the system, possible relationships between use cases (such as <<include>> and <<extend>>) and notes to help clarify the diagram. In most cases, students were able to draw this basic diagram quite successfully. However, in most cases, students forgot to add notes and the use case to use case relationship did not appear to have been well understood. Furthermore, this question also tested students in their ability to document, in detail, a formal full use case. While the format for the use case can vary, it is essential for students to write it as formally and completely as possible. This is so because use case documentation forms the basis for the rest of the software development activities.

As a guideline, a good use case should include its Name, Actors, Pre- and Post-conditions, Main flow and alternate Flow.

**Question 2**

This question dealt with, in detail, identification of classes, their documentation and modelling with class diagrams. This is a crucial question, and fortunately most students were able to demonstrate their ability to understand and answer this question. Classes can be identified from use cases by using a well-known approach promoted by Grady Booch called 'Noun Analysis'. This requires the analysts to go through the use cases in detail and identify all the nouns in its description. Should there be any proper nouns, they should all be converted into singular common nouns, which provide the basis for 'starting' the class identifications. Later, as more use cases are identified and analyses, the list of classes can change by decomposition of classes into lower level classes or even merging of classes. Students were asked to correctly identify and list classes that they would have discovered after analysing the use case that they themselves had documented in the previous question. Students were also required to demonstrate in detail their ability to document classes. Classes in OO are documented in three main parts – name, attributes and behaviour/operations.

Finally, in the third part of the question, which is one of the most crucial in the paper, required students to draw a full detailed class diagram. 10 classes including both business (stereotyped as <<entity>>) and technical (<<table>>, <<boundary>>, <<control>> for example) were required. Furthermore, each class was meant to have two attributes and two operations. . This diagram, with its various class types, relationships and multiplicities is vital for the students of OO and hence should have been drawn with great depth. Interestingly, most students were able to make good attempts to draw the class diagram in this question.

**Question 3**

This question tested the Dynamic modelling aspect of the skill set required of students. This question included modelling with a sequence and a state machine diagram and also ensuring that these diagrams map with the static models drawn in the previous answer. A sequence diagram is a dynamic diagram that shows the manner in which messages are passed between various collaborating objects in order to satisfy a scenario. Therefore, these are example dynamic diagrams and are crucial to the development of a good system model. Students are expected to show a list of collaborating objects horizontally and timeline vertically. Furthermore, at design level, it is also essential to show all objects and messages on a sequence diagram with their mapping to classes and methods on class diagrams. The requirement in the question in terms of 5 objects and 10 messages would test the basic skills of a student. Most students were able to

create the sequence diagrams at an acceptable level. This question also had another dynamic diagram of the UML called the State Machine Diagram (SMD). Although this is a technical diagram, it can also be of immense help in requirements modelling. This question tested if the student could draw the diagram from a simple requirements modelling perspective. While some students got it right, many had a problem of understanding the difference between this and the activity diagram.

Finally, this question also tested the student's ability to understand the mapping between sequence and class diagrams. By drawing sequence diagrams, one aims to update the class diagrams. In this question, students were expected to draw a class diagram that would correspond to the given sequence diagram.

#### **Question 4**

This question checked the final competency of the students in terms of component modelling as well as quality assurance and testing. The question required the students to explain software components and demonstrate their understanding of mapping classes to component. Expected answer included description of component: "component diagrams are important implementation level diagrams. They show how the classes designed in the solution space are implemented and deployed through the modelling work in the architectural space. Components represent modular, cohesive and deployable 'chunks' of the system. Components encapsulate implementation and expose a set of interfaces to enable their usage." Furthermore, this question also checked the students ability to identify and draw package diagram that would show the components within sub-systems. In this package diagram, relationship between two packages, such as Register and Sales packages were to be shown. This relationship is a dependency relationship shown with a dashed arrow. Finally, the question also dealt with testing which requires planning, designing, writing and executing of the tests. A test case must have a good suite of test data. Typically, this is a set of 'valid' and 'invalid' data which should be appropriately dealt with in the system. Students were able to answer this question in general, but not specifically in terms of test cases.