CITS4401 Software Requirements and Design

2020 SEMESTER 1 SAMPLE EXAMINATION

Time allowed: 135 minutes

Instructions

- The exam has 4 questions with a total value of 120 marks. *(1 mark per exam minute)*
- The time allowed to complete the exam is 135 minutes. *(2 hour exam plus 15 minutes to allow for any technology overheads)*
- This is a closed book ExamSoft exam.
- No handwritten or printed notes, books are allowed.
- No electronic devices are permitted during the examination.
- Calculators are not allowed for this exam.
- Students **may use blank paper** for rough working during the exam.
- But all answers must be typed in ExamSoft for marking.
- Any feedback for the examiner must be entered using the ExamSoft Notes feedback feature.

CITS4401 Unit Outcomes The exam is designed around the unit outcomes. Students are able to:

1. classify types of software requirements and designs;
2. apply requirements and design processes appropriate for a given scenario;
3. assess quality attributes of given requirements and designs;
4. utilise design patterns and idioms;
5. evaluate and document software design rationale; and
6. select a software architecture appropriate for a particular context.

Important notes about studying for the exam The questions in this sample exam are sourced from previous workshops and past papers. This is so students have access to sample solutions. Note that questions in the actual exam will *not* be copies of questions you have done before, so please do not attempt to study by rote learning questions from past papers.

In the CITS4401 exam you will be given short questions or a scenario to analyse (similar to the workshops) and asked to demonstrate that you can:

- distinguish the relevant from irrelevant facts in the scenario
- identify what topics we have covered which apply in this case, and how
- come up with a clear recommendation or answer
- justify that recommendation (logically, or via evidence covered in lectures etc.)

"Identifying relevant facts/topics" and "justifying answers" are key things we are looking for.

Use the marks awarded for each question to give you an idea of the level of detail required in your answer. The paper is marked out of 120: 1 mark per exam minute. That means a 15 mark question in the exam should take you about 15 minutes.

Often, the lecture and workshop content will provide a brief guide to understanding a concept (e.g. a design pattern), but for a *thorough* understanding (and more examples), you should review any relevant textbook chapters and make study notes on them.

Question 1 concerns the following scenario

The marketing team have been out to choose a project for a *personal productivity product for small computers that will sell at least one million copies at a retail price of at least $200.* They have identified a market opportunity: *MotivActiv* will give you friendly guidance and serious motivation to build healthy habits that help you reach your weight and fitness goals.

Question 1: Requirements Elicitation (30 marks)

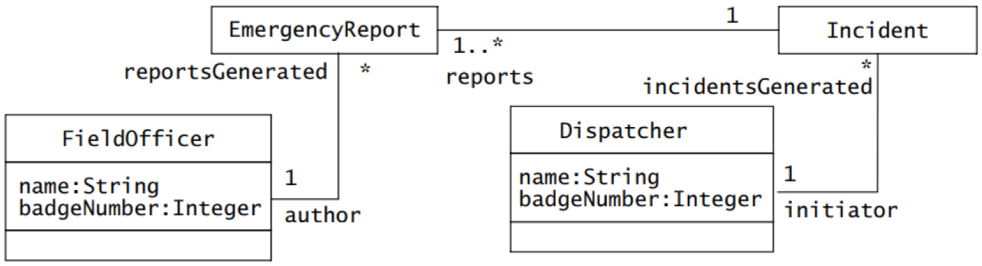
1. Identify three stakeholders for the *MotivActiv* system. For each, give a brief justification for why they are important. [12 marks]

2. Outline the boundaries for the software: which aspects of the system will be addressed by software, and which are outside your scope. [8 marks]

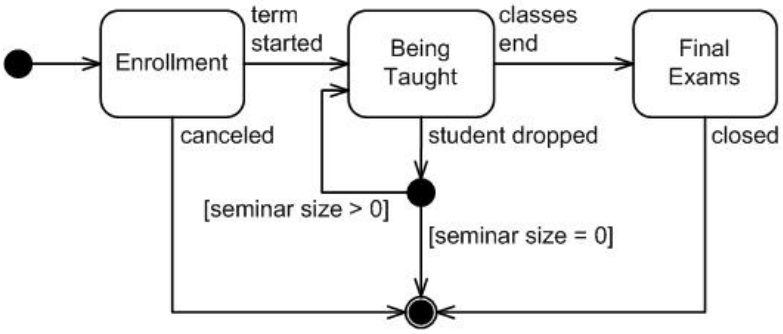
3. Outline a strategy for eliciting the requirements for the system. Justify your choice of methods. Explain how they complement each other. [10 marks]

Question 2: UML Diagrams (30 marks)

1. In plain English sentences describe the model that is described by the UML class diagram above. State carefully any assumptions you make and note any important information you believe is missing from this diagram. [15 marks]



2. In plain English sentences describe the scenario represented by the following UML state diagram. State carefully any assumptions you make and note any important information you believe is missing from this diagram. 15 marks



Questions 3 and 4 refer to the following scenario

Video Conversion Software. The system is a command-line application, able to read in video files in various formats (e.g. .avi, .mp4). It can then splits the audio and video into separate streams, and can perform various *transformations* on each. For instance, the video might be scaled larger or smaller, or cropped (bits are taken off the sides); the audio might have its volume increased or decreased. The system then joins the transformed video and audio tracks together, and outputs them in a format specified by the user.

Question 3: Design Architecture (30 marks)

Suggest an appropriate architecture for the video conversion software. Give dot-point reasons in favour of the architecture you have suggested. 30 marks

Question 4: Design Patterns (30 marks)

1. Define the term *anti-pattern.* Give an example of an anti-pattern for a system you have worked on (either in your university course-work, or outside university). 10 marks

2. Refer to the Video Conversion Software scenario outlined above.

   Having finished the command-line conversion software, you are now developing a graphical user interface (GUI) which re-uses components from that project.

   In your GUI, there is a video editing panel: users can drag video and audio files from their computer, drop them onto the video editing panel, and are then able to combine multiple audio and video files into a single, integrated movie. In your design, you have created a 'VideoData' interface to represent data from video files, and an 'AudioData' interface to represent data from audio files. (If a file has both video and audio data, that will result in one of each of these being created.) However, you run into a problem: the contents of a video or audio file can be extremely large, and holding all the objects in memory results in your software slowing to a crawl.

   Explain what design pattern (or patterns) you could apply to solve this problem. Give the name of the pattern, a brief explanation of why it is applicable in this case, and any new classes or interfaces you might need to create in order to implement it. 20 marks