Examination Performance Feedback to COMP61532

Section A Answer all questions

A1: Explain the five fundamental concepts of a REA business model.

(5 marks)

Five concepts [1m each, 5m total]:
- Economic Resource is a thing that is scarce, and has utility for economic agents.
- Economic Agent is an individual or organization capable of having control over economic resources.
- Economic Event represents either an increment or a decrement in the value of economic resources that are under the control of the agent.
- Commitment is a promise or obligation of economic agents to perform an economic event in the future.
- Contract is a collection of increment and decrement commitments and terms.

A mix of answers was obtained for this question. Common errors were not knowing the concepts of resource, event and commitment.

A2: Explain the fundamental idea of a REA business model and two fundamental processes linked to this idea.

(5 marks)

REA idea:
If an enterprise wants to increase the total value of resources under its control, it usually has to decrease the value of some of its resources [1m]

Two fundamental business processes in REA model:
- Exchange is a process in which an enterprise receives economic resources from other economic agents, and it gives resources to other economic agents in return (giver-taker). [2m]
- Conversion is a process in which an enterprise uses or consumes resources in order to produce new or modify existing resources (consumer-producer). [2m]

Most students did well on this question.

A3: State two different situations in which design patterns are a tool for communication. Your answer should state who is doing the communicating and include a specific example of each situation. (4 marks)

Design patterns enable quick communication between experienced developers [1]
They also allow experienced developers to communicate best practice to inexperienced ones. [1] An example of the former is “do you think we should use a
Factory to create the Adaptors”? [1] An example of the latter is that you might draw a class diagram to show the Strategy pattern, and giving an example of its use. [1]

Mostly answered very well. Only a couple of people failed to read the question properly, substituting “what” for “who”.

A4: Explain how the Template Method design pattern works, stating clearly what is implemented where, and what is visible to classes outside the pattern (4 marks)

We have an inheritance structure, with an abstract class containing the outline of an algorithm (the template method), and subclasses which fill in the details. [1] The template method provides the public interface to the classes. [1] It provides slots in the template, in the form of abstract, protected methods which are implemented in the subclasses. [1] In addition, private (or protected) helper methods in the superclass are used when the implementation doesn’t vary. [1]

Again, mostly answered well.

A5 Under what circumstances is it appropriate to use the Visitor pattern? (2 marks)

Visitor inverts the normal OO structure and collects the whole of an algorithm in one class. This is appropriate when the object structure is stable but the algorithms to be applied to it are complex and liable to change. Otherwise it’s best to make each class responsible for its part of the algorithm in the normal OO way.

And again, apart from a few people who clearly didn’t know what the visitor pattern is.

Section B: Answer all questions

B1

b) Explain the idea behind the REA Exchange Process pattern and the central relationship in this pattern. (5 marks)

Basic idea behind this pattern [2m]:

An REA exchange process is a process that transfers some rights to the enterprise’s resources to other economic agents, and receives some rights to other resources in return.

The central relationship in this pattern:
• The exchange duality between the increment and decrement economic events [1m]
• Each exchange consists of at least one increment economic event [1m]
• Every increment event is related to at least one decrement economic event [1m]

Mostly answered well. Common errors were the omission of the first bulletin point above.

c) Explain the idea behind the REA Conversion Process pattern and the central relationship in this pattern.

Basic idea behind this pattern [2m]:

An REA conversion process is a process that uses or consumes the resources that are under the control of the enterprise, and produces new resources or changes some of the features of existing resources.

The central relationship in this pattern:
• Centred around the conversion duality (or conversion) relationship between the increment and decrement economic events [1m]
• Each conversion consists of at least one increment economic event & at least one decrement economic event [1m]
• Each event is related to exactly one economic agent by a provide relationship, and to exactly one economic agent by a receive relationship [1m]

About a half of the answers was correct. The common error was the problem with understanding the central relationship in this pattern.

Question B2

You are designing an e-commerce system for a bookstore. You want to apply some of the REA structural patterns you have learned from this course in your design. Your first step is to use the REA Exchange Process pattern to design the “purchase” relationship. Answer the following questions:

a) Show your design of the “purchase” relationship in a UML class diagram. Your diagram should show (1) all the entities and their interactions involved in this relationship; (2) the roles played by these entities in their interactions.

(7 marks)

Sample answer:
The “purchase” relationship is the inverse of the “sale” relationship, so the answer should have a similar form as the following diagram, but the REA entities in the answer should replace the entities in the diagram:

- Customer
- Bookstore (instead of Joe’s Pizzeria)
- Purchase (instead of Sale);
- Payment (instead of Cash Receipt)
- Money (instead of Cash)
- Book (instead of Pizza)

The relationships between these entities are the same as the diagram below.

The roles of these entities (denoted as << >>) are changed as follows.
- The Purchase should be an <<increment event>>;
- The Payment should be a <<decrement event>>;

Marks allocation:
- 7m if the answer is exactly the same as above (correct form, entities, relationships & roles); Otherwise, allocate/deduct marks as follows:
  - 3m for correct entities [0.5m for each entity]
  - 2m for correct relationships [1m for correct representation of the exchange relationship; 1m for other relationships]
  - 2m for correct roles [1m for showing <<increment>> on Purchase entity and <<decrement>> on Payment and 0.5m for other roles].

I was pleasantly surprised by students’ understanding of this important REA model and its concepts. Mostly answered well.
b) Explain which economic event is an increment event and which one is a decrement event, and why.

Sample answer:

The two economic events are Purchase and Payment [1m].

From the perspective of the Customer, the Purchase event is increment because the Bookstore transfers the ownership of the Book to the Customer [1m]; the Payment event is decrement because the Customer transfers the ownership of Money to Bookstore [1m].

Mostly answered well. A common error was not stating the perspective of the model. The sample answer above was produced from the perspective of a customer; the modelling concepts and relationships might be reversed if the perspective changed to the bookseller.

Section C: Answer ONE question from this section.

The questions relate to the design of an online assessment system.

C1

Most people chose this question.

a) “The server returns copies of the question paper created by a Factory which is a Singleton”. Explain this statement and how it relates to the use of design patterns for communication.

Instead of creating copies of QuestionPaper objects directly, this task is delegated to a separate factory class, which will create them. [1] This provides a separation of concerns and makes it easy to create exam papers flexibly, for instance we might randomly or adaptively select from a question bank, possible giving different papers to different students. [1] We only even need one instance of this factory class, hence it’s a singleton. [1] Design patterns provide a common language for so for instance here an important design decision is conveyed in a single sentence [1]

I accepted quite a range of answers here, as a number of good points were made. Some people failed to answer the last part.

b) Show the important elements of this class in Java syntax (4 marks)

public class QuestionPaperFactory {
private static QuestionPaperFactory instance; [1]

// Constructor is private so we can only create instances internally
private QuestionPaperFactory() { … } [1 bonus]

// Eager instantiation is also acceptable
public static QuestionPaperFactory getInstance() { [1]
if (instance == null) instance = new QuestionPaperFactory(); [1]
return instance;

// Methods to create question papers. [or similar, 1]
}

I was pleasantly surprised by how good most of the answers were to this.

c) Suppose the application is being extended so that it will accept question
papers written in a variety of different formats an convert those to the format
used by the exam software. Draw a UML class diagrams showing how the
Adapter pattern can be used to organise this part of the application. (4 marks)

of UML notation [1]

Note; in practice there will be many variations and this simplistic marking scheme will not
always work.

Actually it pretty much did, and there were a lot of good answers.

d) Given an example (not one given in the course) of another application in
which the adapter pattern could play a useful role.

Marks for any sensible example with the role of the adapter clearly explained. E.g.
adapting between different product description formats in a retail application.

Some examples recurred, suggesting that a number of people had done some sensible
background reading.

e) Explain how the Factory and Adapter patterns are related to GRASP
principles. For full marks you should include four different GRASP principles
in your answer. (4 marks)
High cohesion, e.g. a Factory creates objects and does nothing else.
Low coupling, e.g. Adapter decouples code within the application from external question formats
Both are Pure Fabrications – they are there to improve the design and are not related to domain classes
Adapter makes use of Polymorphism to provide Protected Variations
1 mark for each, or for other valid points

This was easier than I intended (I should have restricted the GRASP principle allowed)
Most people got full marks for this.

C2

Only two people attempted this, neither attempt was particularly successful.
This surprised me, as the Observer pattern is one of the best known, and a lot of people used it in the coursework. I’ll let the marking scheme speak for itself.

When an ABC exam is taking place, student activity is monitored by a tool, which shows a dynamically updated table of the students taking the exam, with information such as when the most recent backup occurred for each student, and whether any students have attempted to cheat by accessing applications or web pages other than the exam. The monitoring tool works by sending HTTP messages to the server, receiving data in response, so to a firewall it looks just like a web browser. In other words, the tool is polling the server at intervals for batches of information, rather than the server providing each piece of information as soon as it’s available.

a) In the first version of the monitoring tool, there was a 1-1 mapping between exams and invigilator tool instances. It was not possible for one exam to be monitored by many tools, or many exams to be monitored by one tool. Also, we expect in the future to have different types of tools for different users, e.g. invigilators in a room vs. managers overseeing the whole exam process. We might even have different types of servers, e.g. local vs. remote. Explain how the Observer design pattern can be used to remove these restrictions and provide the basis of future developments. (4 marks)

The monitoring tool is the Observer and the server the Observable. Hence the tool registers with the server to receive messages, and the server notifies the tool when an event occurs. This provides flexible communication between servers and tools. In addition, we can have a MonitoringInterface (an ObserverInterface) which different monitoring tools implement, and likewise for servers, giving the full Observer pattern.

b) Draw a UML diagram to illustrate the Observer pattern as applied to this situation. 6 marks
Diagram should show the above in correct UML notation. Observer/observable 2 marks, register/notify 2 marks, interfaces 2 marks. Max 3 marks if the generic Observer pattern is shown, rather than specifically related to the application.

c) In what way does the method of communication between the tool and the server described above require changes to the standard Observer pattern? 2 marks.

Normally, notification is per-event, and immediate (a method call). In this case the server must store events until the tool asks for them – the opposite way round from the standard pattern.

d) Suggest another way in which the Observer pattern could be used in the monitoring tool. 2 marks.

The obvious one is in the user interface – the Java event model. Could also be used to chain monitoring tools together.

e) Briefly suggest how two other design patterns might be used in the monitoring process. 2 marks

e.g. Strategy for different monitoring strategies, Proxy for indirect server access, Composite for dealing with the monitoring data hierarchically in a large organisation… marks for ay sensible suggestions.

f) Explain how the Observer pattern is related to GRASP principles. For full marks you should include four different GRASP principles in your answer.