UNIVERSITY *of* LIMERICK

OLLSCOIL LUIMNIGH

COLLEGE *of* INFORMATICS *and* ELECTRONICS

Department of Computer Science
and Information Systems

End-of-Semester Assessment Paper

Academic Year:	2006/2007	Semester:	Autumn
Module Title:	Systems Analysis	Module Code:	CS4125
Duration of Exam:	2.5 Hours	Percent of Total Marks:	55
Lecturer(s):	J.J. Collins	Paper marked out of :	100

Instructions to Candidates:

- Answer Q1, and any two other questions.

Q1 Answer ALL parts. Total marks awarded for this question: 40.

a) What are the characteristics of good software?

4 marks.

b) What benefits are derived by using the pattern of Separated Interfaces?
Illustrate the answer with a package diagram.

4 marks.

c) What is an extension point in a use case?
Illustrate your answer with a simple use case diagram.

4 marks.

d) What are the icons used to represent a boundary class, control class,
and entity class?

4 marks.

e) Draw a diagram that depicts a square belonging to one board only, and that each
square is uniquely specified by row and column attributes in the range {1..3}.

4 marks.

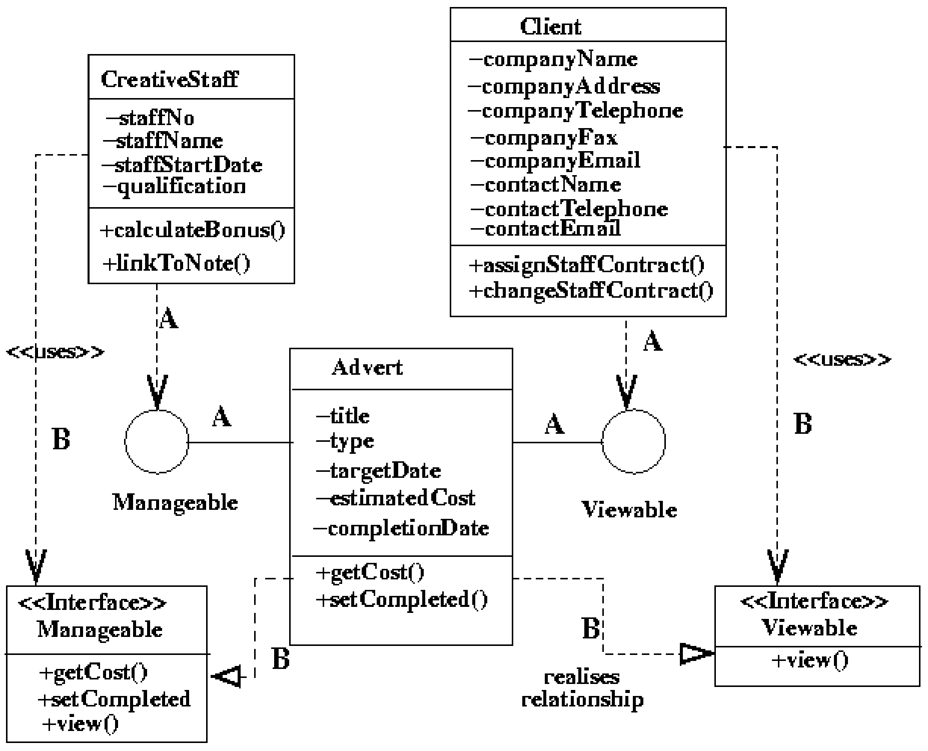
f) Illustrate initialisation in the MVC architecture using a sequence diagram.

4 marks

g) Draw an activity diagram using signals to capture the following series of events:
- Three days before the flight, my travel agent emails me with a list of required travel documents.
- If the list is not received by the three day deadline I cancel the flight.
- Otherwise:
    - Three hours before the flight, I order a taxi.
    - When the taxi arrives, I depart for the airport.

4 marks.

h) The two notations for specifying interfaces in UML 1.x is used in figure 1 to depict two interfaces Viewable and Manageable. Identify the error in Figure 1.



Interfaces for the Advert class

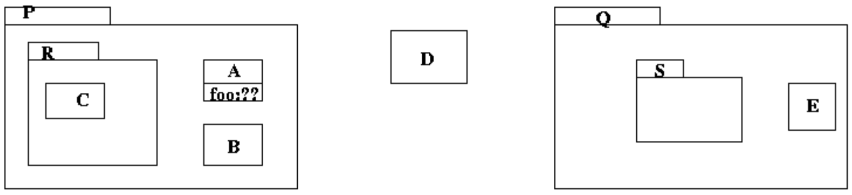
Figure 1.

4 marks.

i) List the features typically offered by a DBMS?

4 marks.

j) Given the package diagram in figure 2, what are the legal and illegal choices of class for attribute *foo* in class A?



Packages and visibility examples

Figure 2.

4 marks.

Q2 Answer ALL parts. Total marks awarded for this question: 30.

a) Describe the Select Factory architecture making reference to Select Architect and Select Models Neighbourhood. What are the three UML class stereotypes used in Select Component Architect, and what colours are used to represent each?

8 marks

b) One of the principles underpinning Design by Contract is the concept of behavioural subtyping – "demand no more: promise no less". Explain the meaning of the phrase "demand no more: promise no less", what is another name for this principle, and illustrate with a class diagram using a different example to that presented in lectures.

9 marks.

c) Draw a conceptual class diagram for a game of noughts and crosses, taking cognizance of the following informal specification.

- A board consists of squares
- Players place tokens on squares to denote ownership at a point in time.
- Players take turns at selecting moves.
- A rule book specifies allowed moves for a specified pattern of tokens.
- The rule book also specifies the starting configuration and end-of-game pattern.

10 marks.

d) Describe one benefits and two liabilities derived from application of the Model View Controller architectural pattern.

3 marks.

Q3 Answer ALL parts. Total marks awarded for this question: 30.

a) Describe the key features of the object-oriented paradigm. Illustrate the discussion with a diagram for at least two features.

8 marks.

b) Describe the concepts of coupling and cohesion. Discuss how these concepts can be applied to an object-oriented, and use diagrams where appropriate to ground the discussion.

9 marks.

c) A folder consists of a set of files and folders. Operations such as rename and delete apply to folders and files. Describe a design pattern that supports the requirement that both files and folders support a uniform interface, and illustrate the answer with a class diagram.

10 marks.

d) Specify the difference between a component and deployment diagram.

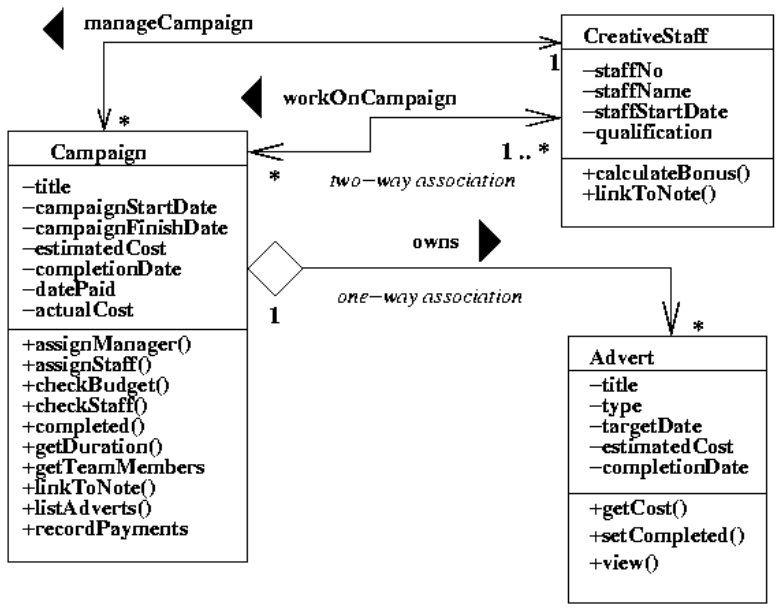
3 marks.

Q4 Answer ALL parts. Total marks awarded for this question: 30.

a) What problem is addressed by Gamma et al.'s Behavioural state pattern? Illustrate this pattern through the use of a class diagram.

8 marks.

b) Given the class diagram fragment in figure 3, using collection class design the one-way one-to-many association *owns* between *Campaign* and *Advert*.



Fragment of class diagram for Agate case study

Figure 3.

9 marks.

c) Draw a class diagram that depicts the design implicit in the code fragment Provided in Appendix A

10 marks.

d) Briefly describe the five system perspectives specified in Krutchen's '4+1 views'

3 marks.

Q5 Answer ALL parts. Total marks awarded for this question: 30.

a) List the activities that take place in system design.

8 marks.

b) Draw a collaboration diagram to illustrate the following dynamics: a word-processor (active object) creates a print file and then asynchronously requests a print spooler (active object) to print the job. The print spooler repeatedly reads a block from the print file and sends the block to the printer (active object) using a procedure call.

9 marks.

c) What are the main issues addressed by the Model View Controller architectural pattern? Illustrate your answer through the use of a class diagram.

10 marks.

d) Specify the differences between a state and activity diagram.

3 marks.

Appendix A

// Example taken from: Khalid A. Mughal and Rolf W. Rasmussen . A Programmers Guide to Java Certification Addision-Wesley, 2nd Edition, 2005

```
interface IStack {
   void   push(Object item);
   Object pop();
}

class StackImpl implements IStack {
   protected Object[] stackArray;
   protected int      tos;  // top of stack

   public StackImpl(int capacity) {
      stackArray = new Object[capacity];
      tos        = -1;
   }

   public void push(Object item)
      { stackArray[++tos] = item; }

   public Object pop() {
      Object objRef = stackArray[tos];
      stackArray[tos] = null;
      tos--;
      return objRef;
   }

   public Object peek() { return stackArray[tos]; }
}

interface ISafeStack extends IStack {
   boolean isEmpty();
   boolean isFull();
}

class SafeStackImpl extends StackImpl implements ISafeStack {
   public SafeStackImpl(int capacity) { super(capacity); }
   public boolean isEmpty() { return tos < 0; }
   public boolean isFull() { return tos >= stackArray.length-1; }
}

public class StackUser {

   public static void main(String[] args) {
      SafeStackImpl safeStackRef  = new SafeStackImpl(10);
      StackImpl     stackRef      = safeStackRef;
      ISafeStack    isafeStackRef = safeStackRef;
      IStack        istackRef     = safeStackRef;
      Object        objRef        = safeStackRef;

      safeStackRef.push("Dollars");
      stackRef.push("Kroner");
      System.out.println(isafeStackRef.pop());
      System.out.println(istackRef.pop());
      System.out.println(objRef.getClass());
   }
}
```