ENGINEERING TRIPOS PART IIA

Thursday 8 May 2003 2.30 to 4.00

Module 3F6

SOFTWARE ENGINEERING AND DESIGN

*Answer not more than **three** questions.*

All questions carry the same number of marks.

*The **approximate** percentage of marks allocated to each part of a question is indicated in the right margin.*

You may not start to read the questions printed on the subsequent pages of this question paper until instructed that you may do so by the Invigilator

(TURN OVER

1 Figure 1 shows a UML class diagram describing some software designed to simulate the motion of aircraft around a busy airport. This software is designed to verify that aircraft maintain minimum distances as they descend, are placed into a holding pattern and land.

The simulation models the motion of several aeroplanes as they descend towards the airport. When an aircraft reaches an altitude of 1500 m it requests and is assigned a slot in the holding pattern by the airport. The aircraft then moves into the designated slot in the holding pattern and flies in that slot until the airport notifies the aircraft that it is time to land. The aircraft then leaves the holding pattern, follows a fixed trajectory towards the airport and lands.

(a) Describe the relationships between the classes shown in the diagram and identify any design patterns present. [25%]

(b) Draw a sequence diagram showing the events that take place when the `Airport` object calls the `move` function on an `Aeroplane` object which is descending and flies below 1500 m. [35%]

(c) The simulation is now to be extended to encompass two airports: the original and a nearby airport, which operates according to a different policy. It has both a different height at which it assigns slots in a holding pattern (2000 m) and a different holding pattern. Using good design principles, describe the changes that must be made to the system design and draw a UML class diagram which shows this new software design. You may omit any attributes, operations or pseudocode present in Figure 1 which remain unchanged. [40%]

(Cont.

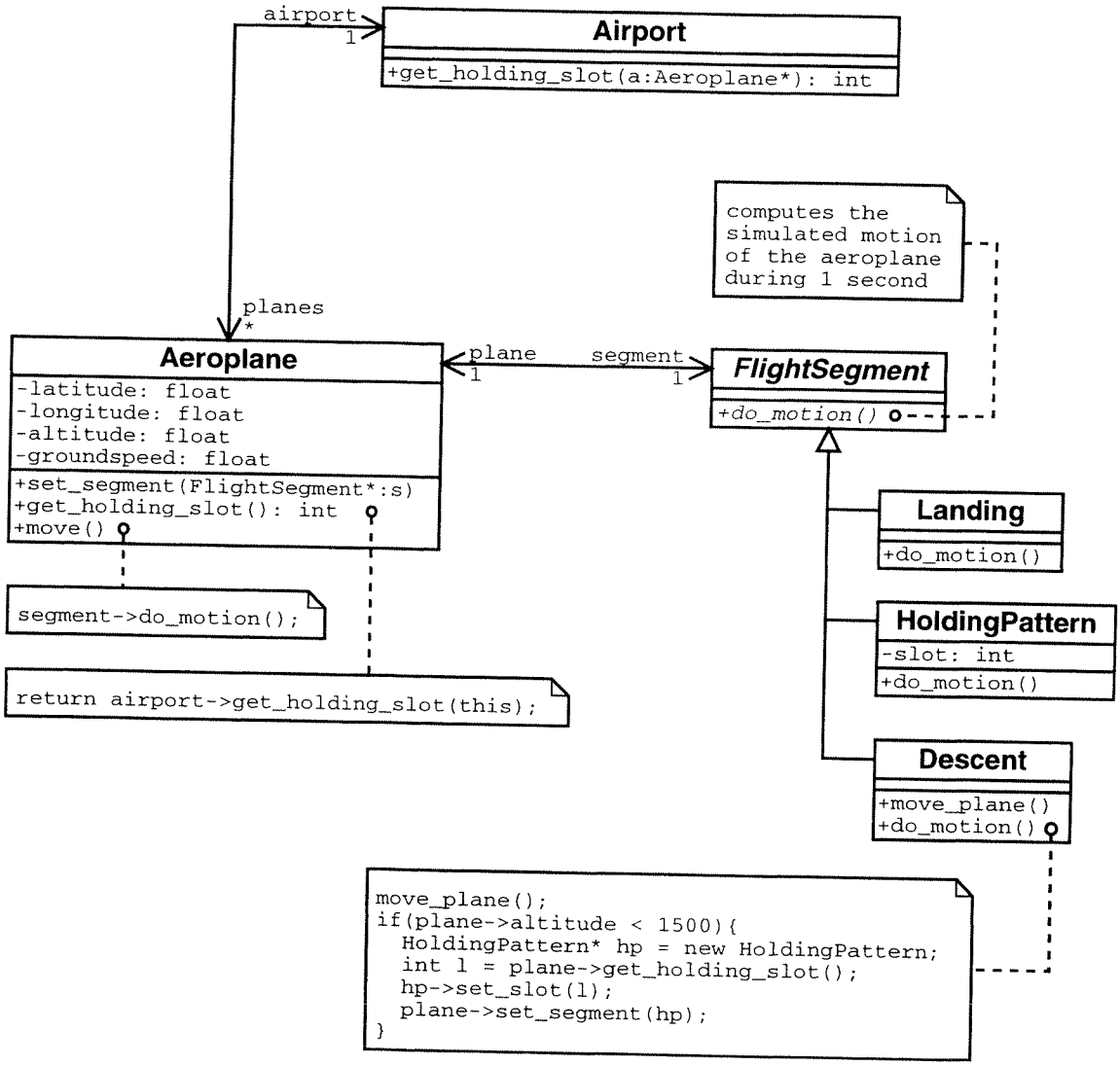
airport
1
Airport
+get_holding_slot(a:Aeroplane*): int
computes the simulated motion of the aeroplane during 1 second
planes
*
Aeroplane
-latitude: float
-longitude: float
-altitude: float
-groundspeed: float
+set_segment(FlightSegment*:s)
+get_holding_slot(): int
+move()
plane
1
segment
1
FlightSegment
+do_motion()
Landing
+do_motion()
HoldingPattern
-slot: int
+do_motion()
Descent
+move_plane()
+do_motion()
segment->do_motion();
return airport->get_holding_slot(this);
move_plane();
if(plane->altitude < 1500){
HoldingPattern* hp = new HoldingPattern;
int l = plane->get_holding_slot();
hp->set_slot(l);
plane->set_segment(hp);
}

Fig. 1

2 A large firm operates its finances by managing several internal accounts. Figure 2 shows a UML class diagram describing some accounting software that allows users to pay an invoice from one of these accounts or to transfer money between two accounts.

The software is to be modified to enable network access using CORBA. Thus the `ClientSoftware` will run on a user's local workstation, interacting with the user, while the `AccountingSystem` will run on a central server that the `ClientSoftware` can contact via the computer network.

(a) Explain the role of the proxy pattern in CORBA. [15%]

(b) Which classes in the diagram will require proxy classes in the client? [10%]

(c) Produce a CORBA `idl` file that defines the interface(s) required to build the distributed system. [25%]

(d) An addition to the software is required that will allow users to set a threshold on an account so that their `ClientSoftware` will be notified when the account balance drops below the threshold. Using good design principles, describe the changes that must be made to the system design to achieve this, by giving:

(i) a UML diagram showing the modified design;

(ii) a CORBA `idl` file for any new interfaces that are needed. [50%]

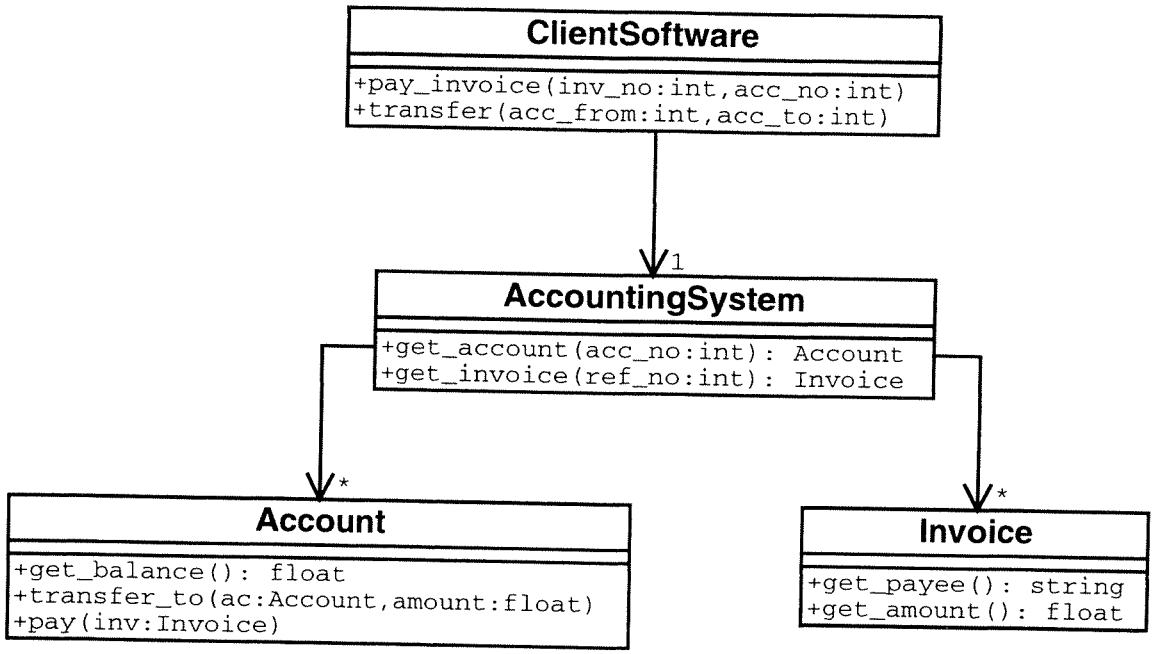
ClientSoftware
+pay_invoice(inv_no:int,acc_no:int)
+transfer(acc_from:int,acc_to:int)
1
AccountingSystem
+get_account(acc_no:int): Account
+get_invoice(ref_no:int): Invoice
*
Account
+get_balance(): float
+transfer_to(ac:Account,amount:float)
+pay(inv:Invoice)
*
Invoice
+get_payee(): string
+get_amount(): float

Fig. 2

3 A two-lane road has to cross a river on a narrow bridge, and a system is required to ensure that cars do not try to cross the bridge from opposite directions at the same time. In addition, the bridge is weak and can only support 3 vehicles at a time. Figure 3 shows part of the implementation for a monitor object which governs the behaviour of the traffic.

Each vehicle is modelled by one process. This process calls the monitor's `arrive` function when the vehicle arrives at the bridge, indicating its direction of travel. The `arrive` function blocks until it is safe to cross. Once this function unblocks and returns, the vehicle crosses the bridge and the monitor's `exit` function is called as it leaves.

(a) The `waitForDirection` data member in Fig. 3 is an example of a *condition variable*. What is a condition variable, and what happens to processes when the `cwait` and `csignal` functions are called passing the condition variable as an argument? [15%]

(b) Complete the implementation for the `exit` function so that the monitor of Fig. 3 behaves as specified, making use of `currentDirection` and the other data members as required. The current direction should swap once the last car comes off the bridge, in case any are waiting at the other end. [30%]

(c) The system could alternatively have been implemented using semaphores. From the point of view of *fault minimisation*, how is the monitor solution better? [15%]

(d) Describe two other design decisions or approaches to development which could help ensure that faults in the final program are minimised. [20%]

(e) Using the implementation completed in part (b), consider the case where a car arrives to cross in one direction when a long queue of traffic is already crossing in the other direction. Under what circumstances will this car be allowed to cross? How could the system be improved to make it fairer? [20%]

(Cont.

```
monitor bridge
{
    condition waitForDirection[2]; // Used to wait for your turn
    int numCrossing = 0;           // No. cars on the bridge
    int currentDirection = 0;      // Current direction is either 0 or 1

    void arrive(int direc)
    {
        if(numCrossing == 0)
            currentDirection = direc;
        else
        {
            while(currentDirection != direc || numCrossing >= 3)
                cwait(waitForDirection[direc]);
        }
        numCrossing++;
    }

    void exit(int direc)
    {
        // Complete this function in answer to
        // question 3 part (b)
    }
}
```

Fig. 3

4 The colleges of the University of Poldovia have commissioned you to design a new software system to manage their supervision reports. This system should allow supervisors to write reports on their students (to be read by their Tutor and Director of Studies) and also to claim payment for their supervisions.

(a) Identify 4 different *viewpoints* for this system, and 4 *services*. Express these in a structure chart. [30%]

(b) It is suggested that you buy commercial database software to provide the workings of the system, but produce the front end yourself by *evolutionary development*. Discuss the possible reasons for this decision. [15%]

(c) Where in the development process should the *stakeholders* be consulted? Consider the cases of the database system and the front end separately. [10%]

(d) Describe the difference between pessimistic and optimistic concurrency control in database systems. Which would be most appropriate for this supervision reporting system? [20%]

(e) The system needs to be robust to system failures. Describe how a crash recovery algorithm such as that described in lectures ensures that the ACID properties of the database are maintained (Atomicity, Consistency, Isolation, and Durability). [25%]

END OF PAPER