UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2001

BEng Honours Degree in Computing Part II
MEng Honours Degrees in Computing Part II
BSc Honours Degree in Mathematics and Computer Science Part II
MSci Honours Degree in Mathematics and Computer Science Part II
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute
This paper is also taken for the relevant examinations for the Associateship of the Royal College of Science

PAPER C220=MC220

SOFTWARE ENGINEERING - DESIGN I

Wednesday 2 May 2001, 14:00
Duration: 90 minutes
(Reading time 5 minutes)

Answer THREE questions

Paper contains 4 questions
Calculators not required

***Section A** (Use a separate answer book for this Section)*

1a i) Briefly describe the kind of information that dataflow diagrams model.

ii) Explain how data flow diagrams can be extended to show information normally reserved for dynamic models.

iii) What is a *Context* Diagram, and how does it differ from *Overview* or *Detailed* data flow diagrams?

b The following is the specification of Ahmad's handling of telephone bills. If Ahmad doesn't receive a bill in the post, then he continues to be cheerful. If the bill arrives and he has money in the bank, then Ahmad sends his payment and becomes depressed. If the bill arrives and Ahmad is out of the country, then he continues to be cheerful.

i) Draw a decision table describing Ahmad's handling of telephone bills as specified above.

ii) Is the specification consistent? Complete? Expand the decision table in part (i) to explain your answer.

iii) What kind of information do decision tables add to data flow models?

The two parts carry, respectively, 30% and 70% of the marks.

2a Explain the terms *clustering, orthogonality* and *broadcast communication* in the context of statecharts, indicating what they add to state transition diagrams and why they are useful.

b A web-based book evaluation system is to be developed to model a reader's response to book prices. Readers rate books as either *poor*, *OK* or *excellent* (by pressing the appropriate button). To begin with, a book's rating is *OK*. If a book's price is raised to over £30, then readers rate it as *poor*. If a book's price is dropped to less than £2, then it is rated by readers as *OK*. If the book gets good reviews in the press, then readers rate it as *excellent*, but only if the reviews are in a reputable newspaper. While the book's rating is *excellent*, it sells well, and while its ratings are *poor*, its sales drop.

i) Draw a state-transition diagram that models the state of books in the above system, indicating conditions and operations, where relevant.

ii) A book is published mainly in either hardback or softback, depending on its rating by readers. When a book's price is dropped to less than £2, it is published mainly in softback, but when ratings are reviewed and a book is rated as *poor*, it is published mainly as a hardback. By default, books are published in hardback. Extend the state transition diagram in part (i) into a statechart, taking into account the book's main publication cover (softback or hardback).

c State transition diagrams are well suited to describing use cases involving one object. Suggest and briefly explain an alternative notation for describing use cases involving multiple objects.

The three parts carry, respectively, 30%, 50%, 20% of the marks.

Section B *(Use a separate answer book for this Section)*

3a Consider the following schemas *Sample*, *SubOk* and *SubError* and the schema disjunction *Sub* = *SubOK* ∨ *SubError*. Write out how the schema *Sub* would appear if written in full, using schema inclusion and Δ notation.

```
┌──Sample ─────        ┌──SubOK ─────────        ┌──SubError ────
│ P: FN                │ Δ Sample                │ Ξ Sample
├─────────────         │ n?: N                   │ n?: N
└─────────────         │ report!:{OK, error}     │ report!:{OK, error}
                       ├─────────────────        ├──────────────────
                       │ n?∈ P                   │ n?∉ P
                       │ P'= P – {n?}            │ report!=error
                       │ report!=OK              └──────────────────
                       └─────────────────
```

b An automatic booking system for a cinema allows people to book tickets over the phone using a credit card. The overall state of the system can be either *closed* or *open*, depending whether the cinema is closed or open. The system allows a customer to book one or more tickets and to refund tickets. Tickets can be booked or refunded only when the system is open. To book ticket(s), a customer provides a credit card number of type **seqN** and the number of tickets, and receives in return the number of tickets booked and a confirmation message. As for the refund, the customer has to provide instead the number of tickets he/she wants to be refunded. At each booking, the system updates directly the cinema's bank account. The following schema *Cinema* defines the basic parameters of the system.

```
┌────── Cinema ──────────────────────────────────────────
│ numberofseats: nat        (capacity of the cinema)
│ price: nat                (price per ticket)
│ status: {closed, open}    (when the cinema is closed or open)
├────────────────────────────────────────────────────────
└────────────────────────────────────────────────────────
```

i) Write a state schema *SystemState*, for the overall state of the system, covering the two cases when the system is open and closed. In both cases, the schema includes *Cinema*, uses the variables *accountbalance* (the balance of the cinema's bank account), *seatsleft* (number of seats left). Include relevant constraints on the variables.

ii) Write operation schemas for the following operations. They all have to include the input *numbertickets?*, *cardnumber?*, and the outputs *ticketsbooked!* and *report!*. Marks will be awarded for correct use of Δ and Ξ.

BookedOk bookes a requested number of tickets and updates the cinema's account balance.

SeatsUnavailable gives no tickets because the cinema does not have enough seats available.

CinemaClosed gives no tickets because the cinema is closed.

iii) Write an operation schema for the operation *Refund*, which includes the input *ticketstorefund?* and *cardnumber?*.

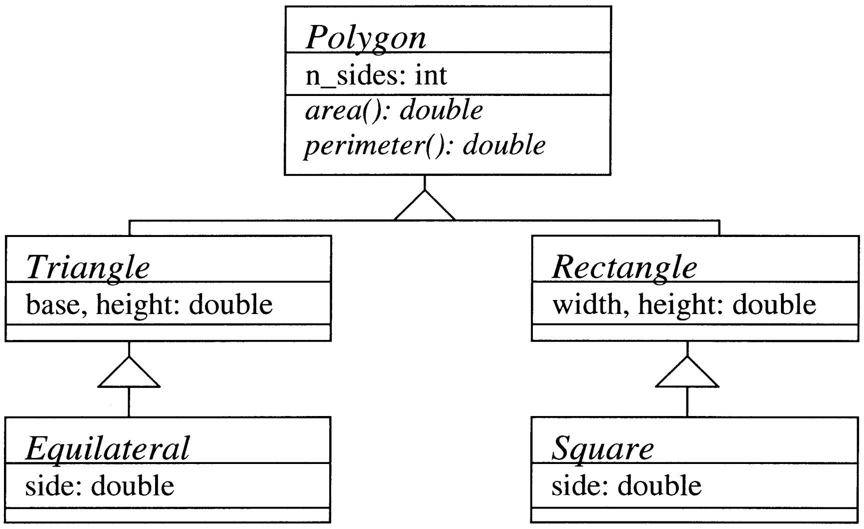
iv) Consider the operation *Booking* = *BookedOk* ∨ *SeatsUnavailable.* Is this operation total? Explain your answer.

The two parts carry, respectively, 30% and 70% of the marks.

Section C *(Use a separate answer book for this Section)*

4 A *polygon* is characterized by how many sides it has, as well as by its area and perimeter. Triangles and rectangles are two special classes of polygons. A *triangle* has 3 sides and is characterised by its base and its height. The area of a triangle can be obtained by evaluating the expression (base*height)/2. Equilateral triangles are special kinds of triangles characterised by having all sides of equal length. The perimeter of an equilateral triangle can be obtained by evaluating the expression (3*side). A *rectangle* has 4 sides and is characterized by its width and its height. The area and perimeter of a rectangle can be obtained by evaluating the expressions (width*height) and (width+height)*2, respectively. *Squares* are special kinds of rectangles characterised by having all sides of equal length. The perimeter of a square can be obtained by evaluating the expression (4*side).

The following OMT object model class diagram outlines the polygons we are interested in:



a Write Java classes that support the creation of polygons and the calculation of their area and perimeter as described above.

b Write a test function that

i) creates a triangle with base 9 and height 4 and prints out its perimeter and area, as well as the number of sides it has

ii) creates an equilateral triangle with side 5 and prints out its perimeter and area,

iii) creates a rectangular with width 5 and height 6 and prints out its perimeter and area, as well as the number of sides it has

iv) creates a square with side 5 and prints out its perimeter and area

The two parts carry, respectively, 70% and 30% of the marks.