UNIVERSITY OF LONDON **CO3314 ZB**

BSc Examination

COMPUTING AND INFORMATION SYSTEMS AND CREATIVE COMPUTING

Software Engineering Management

Thursday 9 May 2013 : 2.30 – 4.45 pm

Duration: 2 hours 15 minutes

There are five questions in this paper. Candidates should answer **THREE** questions. All questions carry equal marks and full marks can be obtained for complete answers to **THREE** questions.

Only your first three answers, in the order that they appear in your answer book, will be marked.

Questions involving a description or explanation should, wherever possible, be accompanied by an appropriate example.

A hand held calculator may be used when answering questions on this paper but it must not be pre-programmed or able to display graphics, text or algebraic equations. The make and type of machine must be stated clearly on the front cover of the answer book.

NOTE:

AN OUTLINE SPECIFICATION FOR A HYPOTHETICAL SYSTEM IS USED IN ALL OF THE QUESTIONS IN THIS PAPER AND IS APPENDED TO THIS PAPER AS PAGE 4.

READ IT CAREFULLY BEFORE ATTEMPTING TO ANSWER ANY QUESTIONS.

Question 1

Software life-cycle
Study the task description of the software system for underground train traffic monitoring, given in the appendix, and then answer all parts of this question.

a. Write a requirements specification for the train traffic monitoring software, and focus especially on the following:

(i) Identify the functions that have to show data on the rail network display. **12 marks**

(ii) Identify the functions that have to show data on the train status monitor. **6 marks**

b. Give the main stages of the real software life-cycle. **7 marks**

Question 2

Measurement, Cost estimation
Study the task description of the software system for underground train traffic monitoring, given in the appendix, and then answer all parts of this question.

a. Which are the levels of measurement in the software engineering process? **6 marks**

b. (i) Estimate approximately the total effort required to implement each function of train traffic monitoring software in thousands of delivered source code instructions (KDSI) according to the COCOMO model. **15 marks**

(ii)Briefly describe how such estimates can be used for design to cost. **4 marks**

Question 3

Software quality, Inspections
Study the task description of the software system for underground train traffic monitoring, given in the appendix, and then answer all parts of this question.

a. Explain which are the participants in the inspection team for software design. **9 marks**

b. Define some measures that can be used for inspection of the functions of the train traffic monitoring software according to the methodology of Fagan. **16 marks**

Question 4

Dependability measurement and Reliability Modelling

Study the task description of the software system for underground train traffic monitoring, given in the appendix, and then answer all parts of this question.

a. Discuss briefly the reliability of the train traffic monitoring software and find some unexpected events that may occur during the operation of the software and disrupt the proper functioning of the whole system. **12 marks**

b. Explain how do we calculate the time to next failure according to the early Jelinski-Moranda model after finding and removing *c* faults, assuming that the number of all faults is *n*, and *z* is the rate at which a fault is activated. **6 marks**

c. What lessons have been learned from the assumption made in early reliability models that all faults lead to failure at the same rate. **7 marks**

Question 5

Safety-critical software

Read the task specification of the software for underground train traffic monitoring, given in the appendix, and answer all parts of this question.

a. Identify which are the functions in the train traffic software that have a high probability of residual fault and may lead to critical failures during operation. **16 marks**

b. Discuss briefly how fault removal can be performed. **9 marks**

APPENDIX: TASK SPECIFICATION

Consider the task of developing a software system for underground automatic train traffic monitoring. The software should collect online information about train locations and send signals for collision avoidance. The users of this software are the dispatchers who are responsible for controlling and monitoring the traffic using computer displays. The display should be divided into two windows, called respectively a rail network display, and a train status monitor. The operations that the dispatcher is allowed to perform should be implemented in the software as options.

The first option is to show on the rail network display data about the main subsystems; these are the rail network, the schedule and the control module. The rail network subsystem includes positional data such as the train locations, the route lengths and the station co-ordinates. The schedule contains the timetable and the route load plan. The control model provides a red collision button which, when on, serves to indicate the necessity for collision preventing intervention. That is why the control module data should include the current train speed, the train acceleration and the status of the brakes. All these data should be kept in a global database.

The second option is to allow a dispatcher to review the status of each particular train by sending queries from the train status monitor. When the dispatcher sends a request by typing the corresponding train number, the system should show on the status monitor the current speed and the location.

The third option offered to a dispatcher is to send a signal for collision avoidance to a breakdown train. Serving a collision avoidance case should include: 1) checking the speed and location of all trains on the same route; 2) computing the distance to the train in front or the next station; 3) turning the movement indicator on the train status monitor into red to inform the dispatcher of a possible problem; 4) showing on the monitor the status data from the problematic speeding train; 5) waiting for the dispatcher to press a button to sending a signal for deceleration of the train; and 6) storing a collision report in the database.

The fourth option given to dispatchers is to print at the end of each working day reports about the traffic and the avoided collisions. The software should compute:- the average speed of each train, the arrival time of each train, the amount of energy used and the number of eventual collisions. If collisions have been avoided the software should print the speed before the avoided collision, the time of dispatcher response, the time when this has happened, and the closest station. The software should allow the dispatcher to print the report and save it on the database. The database should only be accessible by a dispatcher with a password.

END OF PAPER