Version RET/3

EGT2
ENGINEERING TRIPOS PART IIA

Tuesday 21 April 2015 2 to 3.30

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.*

*Write your candidate number **not** your name on the cover sheet.*

STATIONERY REQUIREMENTS
Single-sided script paper

SPECIAL REQUIREMENTS TO BE SUPPLIED FOR THIS EXAM
CUED approved calculator allowed
Engineering Data Book

10 minutes reading time is allowed for this paper.

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

1 (a) Describe the purpose of *Interfaces* in *Object Oriented Design*. Explain why implementing multiple *Interfaces* by a single class might be useful. [15%]

(b) The stock exchange company is building a new version of their trading system. The system's key component is a class `Market` that encapsulates business logic allowing traders to execute transactions such as buying and selling stocks. This class can also deliver notifications of such transactions to other classes. Additionally, it can provide price information for individual stocks.

The market information can be used and presented in multiple ways, one of which is a simple table view that lists each stock displaying its symbol (e.g. AAPL), current price and delta (the price change since the last update). This `Stock View` is updated when a transaction occurs in the market.

(i) Identify the principal *Classes* and their *Relationships* for the *Object Oriented* system that implements the `Stock View` functionality. Illustrate the design with the help of a *Class Diagram*. Consider the use of *Interfaces* to encapsulate different types of functionality provided by the *Class* `Market`. [40%]

(ii) Draw a *Sequence Diagram* for the scenario in which the `Stock View` is updated in response to a transaction executed in the market. [30%]

(iii) Discuss how *Agile Software Development* techniques can be used in the project described above. [15%]

2 (a) Describe the principles of good *User Interface* (UI) design and discuss the main considerations for the UI design of the software applications for smaller devices such as smart watches. [15%]

(b) A company specialising in manufacturing new appliances for smart homes would like to create an app that customers can download on their smart watch and use to remotely control their new smart kettle. The users should be able to set the kettle to boil at a specified time from anywhere in the house via their smart watch, and to receive notification when their hot water is ready. For optimal flavour, the kettle allows the users to set the right temperature for different types of hot drinks such as green tea, black tea or coffee.

(i) Design the UI for a software application for a smart watch with a small touchscreen that implements this functionality. Identify all main screens and interaction elements, explain their purpose and design constraints. [25%]

(ii) Following usability studies the company added a sensor that measures the amount of water in the kettle. The smart watch app should notify the users if they need to fill the kettle, depending on the number of cups they are planning to make. Extend the UI to enable this feature in the app. [15%]

(iii) Another popular feature request was to extend the app functionality to allow the users to notify other family members that the kettle is ready so that they can join and have a cup of tea or coffee together. Identify any additional screens required to support this feature. [15%]

(iv) Identify the most common *Use Case* for the smart kettle app and discuss the *User Experience*. If required, optimise the design of the app to allow the most common use case to be completed in the quickest possible way. [30%]

3 A government has designed a database to store information about its citizens' telephone calls. A simple version of the database is shown in Fig. 1. Each row contains all the information for one citizen indicated by their name (Name). For each phone call a record is kept of the telephone number from which the call was made, the telephone number to which the call was made, and the time-stamp at which the call began. E.g. for the first call a citizen makes these are denoted TelFr1, TelTo1 and TS1 respectively. Calls made from the same device have unique time-stamps.

The full database will contain millions of citizens and each citizen makes many calls.

Name	TelFr1	TelTo1	TS1	TelFr2	TelTo2	TS2	TelFr3	TelTo3	TS3
Alice	1781	4342	1	2781	5342	6	1781	5342	10
Bob	3917	6443	1	3917	7664	8			
Chris	4342	2781	6	5342	7664	15	5342	2781	20
Dave	6443	3917	12						
Eva	7664	3917	4	7664	1781	20	7664	3917	25

Fig. 1

(a) Evaluate the database design and suggest how to improve it. Draw an *Entity-Relationship Diagram* to illustrate your answer. [15%]

(b) Show the updated design of the tables including any new *Entities* and *Attributes* added in the answer to part (a). Identify the *Primary* and *Foreign keys* used. [15%]

(c) Design a query to return all of the telephone numbers that Alice has made calls from. Express your answer using relational algebra or SQL code and explain your solution. [10%]

(d) Design a query to return the names of the citizens who have called Alice. Express your answer using relational algebra or SQL code and explain your solution. [25%]

(e) The government wants to know the names of the citizens who were called by citizens who were themselves called by Alice. Design a suitable query and express your answer using relational algebra or SQL code. Explain your solution. [25%]

(f) The government wants to make many queries similar to the one in part (e). Describe features that could be added to the database to accelerate such queries and detail any potential disadvantages they may have. [10%]

4 (a) Explain the importance of the *serialisability* of a schedule in concurrency control. Define the *serialisation graph* and explain how it is used to determine whether a schedule is serialisable. [20%]

(b) Fig. 2 shows a sequence of nineteen actions scheduled for execution by four transactions T1, T2, T3 and T4 operating on four database accounts A, B, C and D.

action number	transaction	action		action number	transaction	action
1	T1	`A.read`		11	T3	`C.write`
2	T1	`B.read`		12	T3	`A.read`
3	T2	`B.read`		13	T4	`A.read`
4	T3	`D.read`		14	T4	`D.write`
5	T1	`B.write`		15	T2	`D.read`
6	T2	`C.write`		16	T1	`commit`
7	T1	`A.write`		17	T2	`commit`
8	T4	`B.read`		18	T3	`commit`
9	T1	`D.write`		19	T4	`commit`
10	T3	`C.read`				

Fig. 2

(i) Draw the *serialisation graph* for the schedule and determine whether it is serialisable. For this part of the question assume that no form of concurrency control is used. [15%]

(ii) Now assume that a concurrency control protocol is used in which each operation `Q.read` must acquire a read lock Q.R on account Q and each operation `Q.write` must acquire a write lock Q.W on account Q. Once acquired, all locks are held until the transaction commits or aborts. Draw a *resource allocation graph* for this sequence of transactions immediately after action number 15. [30%]

(iii) Draw the corresponding *wait-for-graph*. State which transactions will complete and the order in which they do so. Explain your reasoning. Comment on the efficacy of the concurrency control protocol for this schedule. [20%]

(c) What are the limitations of lock-based concurrency control methods? Outline an alternative concurrency control protocol and explain in which situations it is likely to outperform lock-based methods. [15%]

END OF PAPER

THIS PAGE IS BLANK