ENGINEERING TRIPOS PART IIA

Monday 20th April 2009 9.00 to 10.30

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.*

There are no attachments.

STATIONERY REQUIREMENTS
Single-sided script paper

SPECIAL REQUIREMENTS
Engineering Data Book
CUED approved calculator allowed

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

1 A simple software tool is required which allows text files to be retrieved based on string matching expressions conforming to the syntax shown in Fig. 1. For example, the expression:

```
'software engineering' . 'lectures'
```

would match all documents containing at least one occurrence of the string 'software engineering' and at least one occurrence of the string 'lectures'. The expression:

```
'software engineering' . ( 'lectures' + 'notes') . ! 'draft'
```

would match all documents containing at least one occurrence of the string 'software engineering', at least one occurrence of the string 'lectures' or the string 'notes' and no occurrences of the string 'draft'.

(a) Draw the parse tree for each of the two example expressions above. [25%]

(b) Draw a UML class diagram suitable for representing the parse tree of any string matching expression. Each component class representing a node of the parse tree should derive from a single abstract class which provides the following public interface method:

`IsaMatch(d:Document):Boolean`. [25%]

(c) Assuming that an `IsaMatch` method for strings is provided, add notes to your UML diagram to show how the `IsaMatch` method is implemented for terms and expressions. [25%]

(d) Explain how the syntax rules could be modified to give a higher precedence to the logical-and operator so that for example the expression `'a'+'b'.'c'+'d'` is equivalent to `'a'+('b'.'c')+'d'` and indicate how the UML class diagram would need to be modified to incorporate this extension. [25%]

(cont.

```
expression    =>    term [log-op expression]
term          =>    string | "(" expression ")" |  not-op term
log-op        =>    logical-or | logical-and
logical-or    =>    "+"
logical-and   =>    "."
not-op        =>    "!"
string        =>    'any sequence of characters'
```

Fig. 1

2 (a) Explain the purpose of the observer design pattern and list its advantages and disadvantages. [20%]

(b) Figure 2 shows a modified observer design pattern which allows an observer to view multiple subjects. If there are two observers both observing the same two subjects, draw a UML sequence diagram showing what happens when one of the observers calls the `SetState` method on one of the subjects. [20%]

(c) When an operation calls `SetState` on several subjects sharing the same observer, the design in Fig. 2 will result in many unnecessary updates. How can the design be made more efficient? [20%]

(d) In some cases a more complex mapping between subjects and observers will require a class to be interposed between subjects and observers. Figure 3 shows an example of this where it is assumed that subjects are nodes in a network such that when the state of a node is changed by an external agent calling the `ExtSetState` method, further state changes may propagate to other neighbouring nodes in the network via calls on their `SetState` methods. The `ChangeManager` ensures that when there is a burst of such state changes the `Update` method of each observer is called only after all state changes have taken place and it is called only once regardless of how many nodes it is observing. Define a suitable data structure for the subject-observer mapping and provide a pseudo-code implementation for the `Register`, `Notify` and `NotifyComplete` methods. [40%]

(cont.

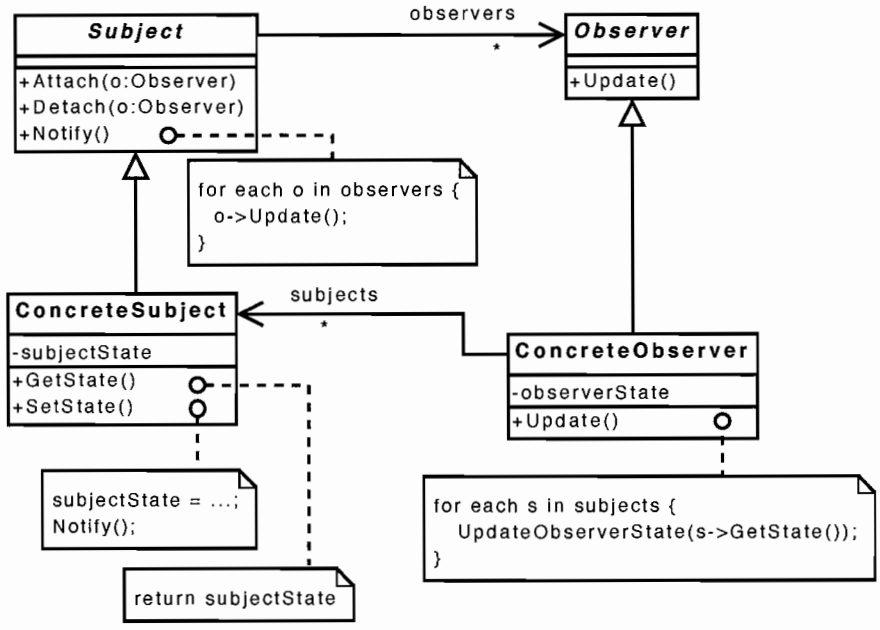

Fig. 2

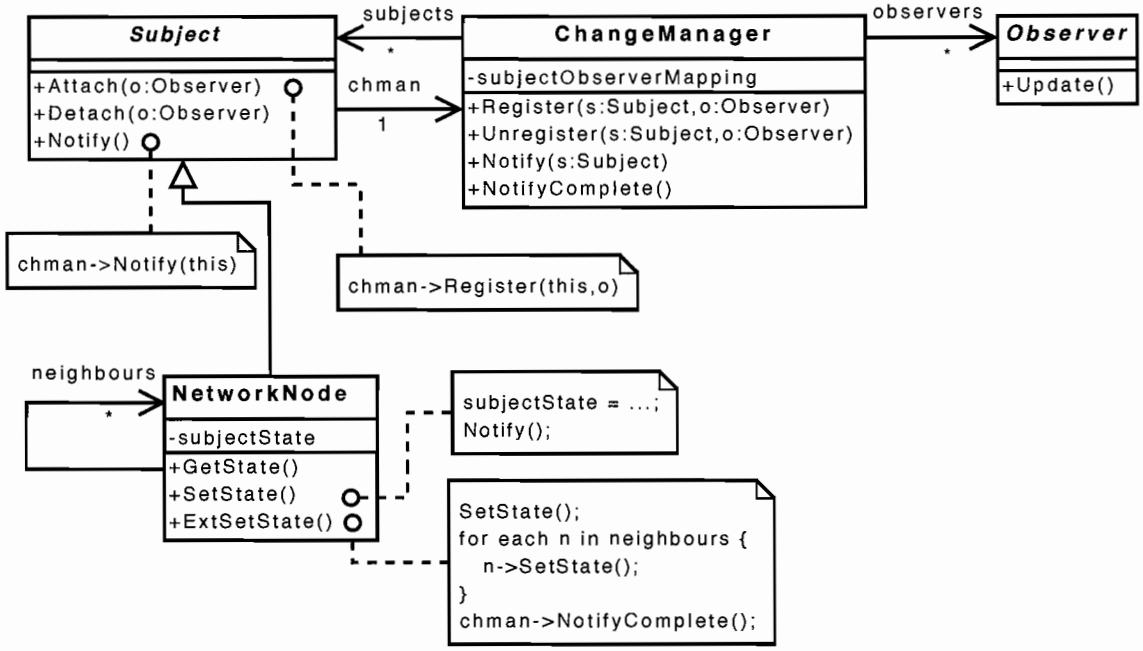

Fig. 3

(TURN OVER

3 A multi-threaded system utilises a set of N identical resources. When a thread wishes to use a resource it must first obtain a `ResourceKey` from a pool of keys by calling the method `KeyPool::GetKey` as shown in Fig. 4. When it is finished with the resource, it must call `KeyPool::ReturnKey` to return the key to the pool. However, the `KeyPool` class is not thread-safe and hence an additional wrapper-class `ProtectedKeyPool` is required to provide safe access to the resource pool.

(a) Explain what is meant by the term *thread-safe* and give some examples of what can go wrong when multiple threads attempt to use a class which is not thread-safe. [15%]

(b) As shown in Fig. 4, `ProtectedKeyPool` uses a semaphore with methods `Secure()` and `Release()` to provide the required thread-safety. Describe the function of a semaphore and explain briefly how it is implemented. [15%]

(c) `ProtectedKeyPool` also makes use of a signal with methods `Wait()` and `Send()`. Describe the function of a signal and explain its complementary relationship with the semaphore. [15%]

(d) Define suitable specifications for the methods `ProtectedKeyPool::GetKey` and `ProtectedKeyPool::ReturnKey` and provide implementations for them. [30%]

(e) After some time in use, it becomes clear that some threads require high priority access to the key pool. Modify your design to provide high priority access such that no waiting thread with low priority will be granted a key unless there are no threads waiting with high priority. [25%]

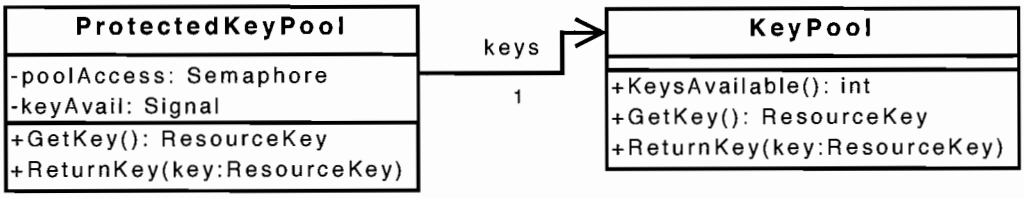

Fig. 4

4 (a) State the four ACID properties of transactions and explain why each is important. [25%]

(b) Figure 5 shows a sequence of sixteen actions scheduled for execution by four transactions T1, T2, T3 and T4 operating on four database accounts A, B, C and D. Each operation `Q.read` must acquire a shared lock Q.S on account Q, and each write operation `Q.write` must acquire an exclusive lock Q.X on account Q. Once acquired, all locks are held until the transaction either commits or aborts. Draw a resource allocation graph for this sequence of transactions and hence determine the first point at which deadlock occurs. [30%]

(c) Draw the corresponding *wait-for-graph* at the point of deadlock. [20%]

(d) Explain how the system can recover from the deadlock and discuss the criteria for choosing a *victim*. Which would be the best choice of victim in this case? [25%]

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

Fig. 5

END OF PAPER